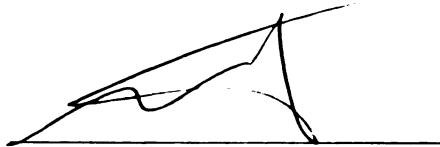


This is to certify that the
dissertation entitled
**Teaching for Fluency with Information Technology:
An Evaluative Study**

presented by
Mark George Urban-Lurain

has been accepted towards fulfillment
of the requirements for
Ph.D. degree in Educational Psychology



Major professor

Date 11/1/2000

**LIBRARY
Michigan State
University**

PLACE IN RETURN BOX to remove this checkout from your record.
TO AVOID FINES return on or before date due.
MAY BE RECALLED with earlier due date if requested.

DATE DUE	DATE DUE	DATE DUE
SEP 05 2003		
MAY 20 2007		
01 25 07		

TEA

2002

**TEACHING FOR FLUENCY WITH INFORMATION TECHNOLOGY:
AN EVALUATIVE STUDY**

VOLUME I

By

Mark George Urban-Lurain

A DISSERTATION

**Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of**

DOCTOR OF PHILOSOPHY

Department of Counseling, Educational Psychology and Special Education

2000

5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000

ABSTRACT

TEACHING FOR FLUENCY WITH INFORMATION TECHNOLOGY: AN EVALUATIVE STUDY

By

Mark George Urban-Lurain

Higher education faces the challenge of helping students learn to be Fluent with Information Technology (FIT) so they are prepared for a lifetime of constant technological change. This study evaluated one effort to prepare students to be FIT: an introduction to computing course for non-computer science majors at a large university. The major goal of the course is to help students learn to use computing technology to solve problems in a variety of domains without learning computer programming. To meet this goal, the course uses an inductive, problem-based curriculum with unique, modified mastery model, criterion referenced, performance-based assessments called bridge tasks (BTs.) This study describes the course curriculum and assessments in detail.

The impact on student outcomes of three major course components were analyzed: incoming student variables, instructional and classroom variables and computing concepts as measured by the BTs. For each component, discriminant analysis was used to predict final student course grades. The data ($n = 5068$) were divided into two sets. The first data set (70%) was used to derive the discriminant functions; the remaining data set (30%) was used to test the classification accuracy of the discriminant functions. If the functions performed

NO. 1
NO. 2
NO. 3
NO. 4
NO. 5
NO. 6
NO. 7
NO. 8
NO. 9
NO. 10
NO. 11
NO. 12
NO. 13
NO. 14
NO. 15
NO. 16
NO. 17
NO. 18
NO. 19
NO. 20
NO. 21
NO. 22
NO. 23
NO. 24
NO. 25
NO. 26
NO. 27
NO. 28
NO. 29
NO. 30
NO. 31
NO. 32
NO. 33
NO. 34
NO. 35
NO. 36
NO. 37
NO. 38
NO. 39
NO. 40
NO. 41
NO. 42
NO. 43
NO. 44
NO. 45
NO. 46
NO. 47
NO. 48
NO. 49
NO. 50
NO. 51
NO. 52
NO. 53
NO. 54
NO. 55
NO. 56
NO. 57
NO. 58
NO. 59
NO. 60
NO. 61
NO. 62
NO. 63
NO. 64
NO. 65
NO. 66
NO. 67
NO. 68
NO. 69
NO. 70
NO. 71
NO. 72
NO. 73
NO. 74
NO. 75
NO. 76
NO. 77
NO. 78
NO. 79
NO. 80
NO. 81
NO. 82
NO. 83
NO. 84
NO. 85
NO. 86
NO. 87
NO. 88
NO. 89
NO. 90
NO. 91
NO. 92
NO. 93
NO. 94
NO. 95
NO. 96
NO. 97
NO. 98
NO. 99
NO. 100

no better than chance, they would correctly classify 12.5% of the cases. The incoming student variable model correctly classified 32.8% of the test data ($p < .000$). For the second model, adding instructional variables to the incoming variables increased classification accuracy to 42.4% ($p < .000$). For the third model, adding the computing concepts to the model increased the classification accuracy to 56.5% ($p < .000$). The computing skills variable loadings on the discriminant functions clustered on the underlying computing concepts, predicting students' abilities to apply the concepts to higher-level problem solving.

This study has implications in at least three areas. First, it provides evidence that it is possible to teach students to be FIT without learning computer programming. Second, this study has implications for the use of technology in instruction. It shows that, by using database technology, it is possible to create, deliver and evaluate complex, performance-based assessments on a large scale, with high inter-rater reliability, in a cost-effective manner. Using this technology, it is also possible to model student learning and to identify student conceptual difficulties, then use this information to intervene and improve learning. Finally, this study has institutional implications for improving the instructional design of courses and curricula. The data collected and feedback processes used in the course can be used to document instructional outcomes for accreditation purposes. The study also has implications for distance education by demonstrating that it is possible to deploy instruction and performance-based assessments at the large scale needed to make distance education fiscally viable.

Copyright by
Mark George Urban-Lurain
2000

To Jan, for your love and support.

com
teco
tran
as m
Wan
tear
abo
tear
tesc
tour
Tea
Com
tear
Pho
date
Com
Tear
Sier

ACKNOWLEDGMENTS

I would like to thank Dr. Yong Zhao, who agreed to chair my dissertation committee, for guiding me through the labyrinth of the dissertation process, keeping me focused on the big picture when I was lost in minutiae. I also want to thank the members of my committee: Dr. Joe Byers, who was especially helpful as my advisor and provided wise counsel throughout my studies; Dr. Donald Weinshank, my colleague and friend, from whom I have learned so much about teaching with – and about – technology; and Dr. Stephen Yelon, whose ideas about instructional design were instrumental in the creation of the course that is the basis for this study.

I would also like to thank the other members of the design team responsible for the creation of the Computing Concepts and Competencies course: Gary McCuaig, with whom I worked for many years at Instructional Television and who contributed so much to this course; Tom Danieli, who, as a Computer Science graduate student helped to get the project underway before he moved on to Perot Systems; and Ryan McFall, who joined the project as a Ph.D. candidate in Computer Science and Engineering, and without whose database expertise the course could never have been implemented.

I also wish to thank the members of the Computing Concepts and Competencies team. Especially Bernie Holmes for his cool, calm and collected management of our complex computing systems and Mary Gebbia for overseeing the massive administration of a course of this size. I wish I had room

to ackn

ass sta

fabri

effec

l

course

reped

becom

throug

each o

.

produc

frustra

contra

thank

compu

to acknowledge by name the hundreds of graduate and undergraduate teaching assistants and grading staff who actually make this course run. Working with this fabulous team has been a rewarding and learning experience. Without their efforts, I would not have had the luxury to undertake this research.

I also want to acknowledge the thousands of students who have taken this course. In many ways, you were all participants in this research. You have helped me remain focused on the purpose of the project: to help students become fluent with information technology. I have seen your joys and pains throughout the course and hope that it has been a worthwhile experience for each of you.

I also must acknowledge the creators of Microsoft Office. Without this product with which I produced the final manuscript, I might have forgotten how frustrating computing can be.

Finally, I would like to thank my family for their support throughout this protracted process. In particular, I would like to express my deepest love and thanks to my wife Jan, who spent many lonely hours while I toiled away in the computing hut. I'll be home soon, hon.

ACKN

TABLE

LIST C

LIST C

LIST C

Chac

Prim

W

W

W

Ho

gg

The

Tr

Tr

Tr

Res

Data

Chac

Wha

Wha

H.

.

C

Tr

A

M

F

S

S

S

D

Tr

TABLE OF CONTENTS

ACKNOWLEDGMENTS	vi
TABLE OF CONTENTS	viii
LIST OF TABLES	xiii
LIST OF FIGURES	xviii
LIST OF ABBREVIATIONS	xx
Chapter 1 Introduction	1
Primary Questions	3
What Constitutes FITness?	3
What is an Appropriate Curriculum and How Do We Determine It?	4
What is the Best Instructional Design to Meet These Goals?	5
How do we assess student outcomes to determine if we are meeting the goals?	6
The CS0 Course at Michigan State University	6
The Curriculum	8
The Instructional Design	9
The Assessments	10
Research Questions	11
Data Sources	11
Chapter 2 Literature Review	14
What Constitutes FITness?	14
What is an Appropriate Curriculum and How Do We Determine It?	16
History of Computer Science Instruction for Non-Majors	17
Traditional Computer Science Curriculum	17
Curriculum for Non-Computer Science Majors	18
Trends in CS0 Courses	20
Applications	20
Management	22
Multimedia	23
Programming	24
Simulations	27
Social	28
Survey	29
Discussion of Trends in CS0 Courses	30
Transfer from Programming to Other Domains	35

Cognition and Programming.....	37
Expert Versus Novice Knowledge	38
The Importance of Conceptual Understanding	40
What is the Best Instructional Design to Meet These Goals?	44
Motivation	45
Computers and Cognition.....	46
How do we assess student outcomes to determine if we are meeting the goals?	48
Summary.....	52
Chapter 3 Methods.....	55
Research Setting.....	55
Program Design.....	58
Design Inputs	58
Instructional Goals.....	59
Instructional Design	60
Inductive, spiral curriculum	60
Multiple “tracks”	61
Problem-based, collaborative learning	62
Performance-based assessment	64
Program Implementation	67
Instruction	68
Classroom structure	68
Instruction and assessment schedule.....	70
Assessment.....	73
Creating bridge tasks.....	74
Delivering bridge tasks	77
Evaluation criteria	78
Evaluating bridge tasks	81
Final project.....	82
Data And Hypotheses.....	84
Evaluation Framework.....	84
Incoming Students.....	85
Instructional Variables	86
Assessments	88
Outcomes	89
Analysis Framework	91
Discriminant Analysis	94
Data Sources.....	99
Incoming Student Data.....	99
Instructional System Data.....	100
Assessment Data	101
Outcomes Data.....	101
Hypotheses	101
Individual Incoming Student Differences Hypotheses.....	102
Instructional System Hypotheses	103

Assessment Hypotheses.....	104
Outcome Hypotheses.....	104
Chapter 4 Results.....	106
Student outcomes.....	107
Incoming Student Variables.....	108
Demographic Variables.....	108
Grade Point Averages (GPA).....	109
ACT Scores.....	111
Students' Majors.....	112
Class Standing.....	114
Student Age.....	115
Gender and Ethnic Classification.....	116
Incoming Student Computing Experience.....	117
General Computing Experience.....	117
Computer Application Experience.....	121
Algebra and Computer Programming Experience.....	126
Survey Factor Analysis.....	128
Incoming Data Model.....	130
Discriminant Function Derivation.....	130
Discriminant Function Validation.....	136
Interpretation of the Discriminant Functions.....	141
Summary.....	151
Instructional Variables.....	152
Teaching Assistants.....	152
Teaching Assistant Experience.....	152
Student Ratings of Teaching Assistants.....	153
Student Participation.....	155
Attendance.....	155
Number of Bridge Tasks Taken.....	157
Student Preparation.....	158
Interactions Between Students and Course Instructors.....	159
Instructional Data Model.....	163
Discriminant Function Derivation.....	163
Discriminant Function Validation.....	169
Interpretation of the Discriminant Functions.....	173
Summary.....	180
Computing Concepts and Skill Variables.....	181
Bridge Tasks.....	181
Computing Concepts and Skills Model.....	186
Discriminant Function Derivation.....	187
Discriminant Function Validation.....	195
Interpretation of the Discriminant Functions.....	199
Summary.....	209
Student Evaluations.....	209
Student Course Evaluations.....	210

Student Evaluations of Their Teaching Assistants	214
Student Evaluations of Their Assistant Teaching Assistants	217
Analysis of the Student Evaluations	220
The Course “Fairness” Factor.....	220
The “Preparation and Participation” Factor.....	223
The “Course Resources” Factor	225
The Teaching Assistant SIRS.....	227
The Assistant Teaching Assistant SIRS	229
Summary of the Student Evaluations	231
Student Course Evaluations	231
Student Ratings of the Teaching Assistants	235
Evaluation of Hypotheses.....	236
Individual Incoming Student Differences Hypotheses.....	236
Instructional System Hypotheses	248
Assessment Hypotheses	250
Outcome Hypotheses	252
Summary	253
Chapter 5 Discussion	255
Major Findings.....	255
Incoming Students and Generalizability of Findings	255
What Constitutes FITness?	256
Instructional Findings.....	260
Authentic, Performance-Based Assessments	262
Implications	264
How to Teach FITness	264
Single CS0 Course or Computing Across the Curriculum?	266
Technology in Instruction.....	269
Institutional Implications	271
Future Research Directions.....	275
Appendix A Sample Lesson Plan for Day 6.....	280
Appendix B Sample Bridge Tasks from Fall, 1999	307
Sample 1.0 BT From Fall 1999.....	308
Sample 1.5 BT From Fall 1999.....	310
Sample 2.0 BT From Fall 1999.....	312
Sample 2.5 BT From Fall 1999.....	315
Sample 3.0 Track A BT From Fall 1999	321
Sample 3.0 Track C BT From Fall 1999	328
Sample 3.0 Track D BT From Fall 1998	332
Appendix C Evaluation Criteria for Sample Bridge Tasks.....	336
Sample 1.0 BT Criteria	336
Sample 1.5 BT Criteria	339

Sample 2.0 BT Criteria	342
Sample 2.5 BT Criteria	346
Sample 3.0 Track A BT Criteria.....	352
Sample 3.0 Track C BT Criteria.....	358
Sample 3.0 Track D BT Criteria.....	362
Appendix D Student Course Survey and SIRS Questions.....	368
Course Survey Questions.....	369
Lead TA SIRS	371
Assistant TA SIRS	372
References	373

Tab

Tab

Tab

Tab

Tab

Tab

Tab

Tab

Tab

Tab

Tab

Tab

Tab

Tab

Tab

Tab

Tab

Tab

Tab

Tab

LIST OF TABLES

Table 1 Course Syllabus for Fall, 1999.....	71
Table 2 Number Of Students by Semester.....	109
Table 3 Correlations Between Course Grade, Semester GPA and Cumulative GPA by Class Level	110
Table 4 ACT Scores	111
Table 5 Number of Students from the Majors with Highest Enrollments	113
Table 6 Percentage of Students Enrolled Each Semester by Class Standing..	115
Table 7 Ethnic Classifications.....	116
Table 8 Length of Time Incoming Students Report Working with Computers ..	118
Table 9 How Students Report Learning to Type.....	118
Table 10 Student Self-Reported Understanding of the Meaning of the Term “Windows”	119
Table 11 Student Self-Reported Understanding of the Meaning of the Term “User Interface”	120
Table 12 Student Self-Reported Understanding of the Meaning of the Term “Hypertext”.....	121
Table 13 Student Self-Reported Experience with E-mail.....	122
Table 14 Student Self-Reported Experience with the World Wide Web	123
Table 15 Student Self-Reported Experience with Word Processing.....	123
Table 16 Student Self-Reported Experience with Spreadsheets.....	124
Table 17 Student Self-Reported Experience with Electronic Databases	125

Table 18

Table 19

Table 20

Table 21

Table 22

Table 23

Table 24

Table 25

Table 26

Table 27

Table 28

Table 29

Table 30

Table 31

Table 32

Table 33

Table 34

Table 35

Table 36

Table 37

Table 38

Table 39

Table 18 When Students Report Taking College-Level Algebra	126
Table 19 Amount of Student Computer Programming Experience	127
Table 20 Final Set of Survey Variables and Factor Loadings	129
Table 21 Distribution of Cases by Final Course Grade	131
Table 22 Tests of Equality of Group Means	132
Table 23 Stepwise Variable Inclusion and Removal Summary	134
Table 24 Significance of the Discriminant Functions	136
Table 25 Incoming Data Classification Results for Cases Used to Derive the Discriminant Functions	138
Table 26 Incoming Data Classification Results for Unselected Cases	140
Table 27 Correlations Between Independent Variables and Standardized Canonical Discriminant Functions	143
Table 28 Correlations Between Independent Variables and Rotated Functions	145
Table 29 F-Statistics Among Each Pair of Group Centroid Means	147
Table 30 Unstandardized Functions at Group Centroids	148
Table 31 Teaching Assistant Experience By Semester	153
Table 32 Factor Loadings for Average SIRS Ratings of TAs	154
Table 33 Student Self-Reports of Doing Homework and Attending Help Room	159
Table 34 Correlations with Student Interaction and Factors	162
Table 35 Tests of Equality of Group Means	165
Table 36 Stepwise Variable Inclusion and Removal Summary	167

Table

Table
D

Table
C

Table

Table

Table

Table

Table

Table

Table

Table
C

Table
C

Table

Table

Table

Table

Table

Table 37 Significance of the Discriminant Functions	169
Table 38 Incoming and Classroom Data Classification Results for Cases Used to Derive the Discriminant Functions.....	171
Table 39 Incoming and Classroom Data Classification Results for Unselected Cases	172
Table 40 Correlations Between Independent Variables and Rotated Functions	174
Table 41 F- Statistics Among Each Pair of Group Centroid Means.....	177
Table 42 Unstandardized Functions at Group Means	178
Table 43 Bridge Task Skill and Concept Difficulty	185
Table 44 Tests of Equality of Group Means	188
Table 45 Stepwise Variable Inclusion and Removal Summary	190
Table 46 Significance of the Discriminant Functions	195
Table 47 Incoming Data, Classroom Data and Computing Concepts Classification Results for Cases Used to Derive the Discriminant Functions	197
Table 48 Incoming Data, Classroom Data and Computing Concepts Classification Results for Unselected Cases	198
Table 49 Correlations Between Independent Variables and Rotated Functions	200
Table 50 F- Statistics Among Each Pair of Group Centroid Means.....	205
Table 51 Unstandardized Functions at Group Means	206
Table 52 Student Responses to Course Survey Questions.....	211
Table 53 Correlations Between Survey Questions and Rotated Factors.....	213

Table 5-

Table 5-

Table 5-

Table 5-

Table 5-

Table 5-

Table 6-

Table 6-

Pa

Table 6-

Table 6-

Table 6-

Re

Table 6-

Table 6-

Table 6-

C

Table 6-

Table 6-

Table 6-

Table 6-

Table 6-

Table 6-

Table 54 Student Instructional Rating System Responses for Their Lead TAs	215
Table 55 Lead Teaching Assistant Grades Assigned by Students	215
Table 56 Lead Teaching Assistant SIRS Factor Loadings	216
Table 57 Student Instructional Rating System Responses for Assistant TAs...	218
Table 58 Assistant Teaching Assistant Grades Assigned by Students	218
Table 59 Assistant Teaching Assistant SIRS Factor Loadings.....	219
Table 60 Summary of Regression Model for the Course “Fairness” Factor.....	222
Table 61 Summary of Regression Model for the Course “Preparation and Participation” Factor	224
Table 62 Summary of Regression Model for the “Course Resources” Factor ..	226
Table 63 Summary of Regression Model for the Teaching Assistant Ratings..	228
Table 64 Summary of Regression Model for the Assistant Teaching Assistant Ratings	230
Table 65 Final Grades by Ethnic Classification	240
Table 66 Cumulative GPA and ACT Scores by Ethnic Classification	242
Table 67 Experience Using Computers for Communication by Ethnic Classification	243
Table 68 Incoming Knowledge of Computing Terms by Ethnic Classification ..	244
Table 69 Number of BT Attempts by Ethnic Classification	245
Table 70 Attendance Rates by Ethnic Classification	245
Table 71 How Often Students Report Doing Homework by Ethnic Classification	246

Table 7
C:

Table 7

Table 7

Table 7

Table 7

Table 7

Table 7

Table 7

Table 8

Table 72 How Often Students Report Attending Help Room by Ethnic Classification	247
Table 73 Instructor – Student E-mail by Ethnic Classification	248
Table 74 Sample 1.0 Bridge Task Evaluation Criteria	337
Table 75 Sample 1.5 Bridge Task Evaluation Criteria	339
Table 76 Sample 2.0 Bridge Task Evaluation Criteria	342
Table 77 Sample 2.5 Bridge Task Evaluation Criteria	346
Table 78 Sample 3.0 A Bridge Task Evaluation Criteria.....	352
Table 79 Sample 3.0 C Bridge Task Evaluation Criteria.....	358
Table 80 Sample 3.0 D Bridge Task Evaluation Criteria.....	362

Figure 1 D

Figure 2 D

Figure 3 C

Figure 4 F

Figure 5 L

Figure 6 E

Figure 7 E

Figure 8 A

Figure 9 T

Figure 10

Figure 11

Figure 12

Figure 13

Figure 14

Figure 15

Figure 16

Figure 17 T

Figure 18 C

Figure 19 C

LIST OF FIGURES

Figure 1 Distribution of articles by category.....	31
Figure 2 Distribution of CS0 article types by year.....	32
Figure 3 Concept map of relationship among skills, schema and concepts	43
Figure 4 Program Design and Implementation	57
Figure 5 Logical structure of Bridge Tasks	76
Figure 6 Evaluation Criteria	79
Figure 7 Evaluation of the implementation	85
Figure 8 Assistant TA exercise feedback questions	90
Figure 9 Two group discriminant analysis	95
Figure 10 Distribution of final course grades	107
Figure 11 Group centroids for function 1	149
Figure 12 Group centroids for functions 2 through 7	150
Figure 13 Class attendance distribution	156
Figure 14 Number of bridge task attempts	157
Figure 15 Number of E-mail messages sent to students by semester	160
Figure 16 Group centroids for functions 1 through 7	179
Figure 17 Bridge task repeat rates	182
Figure 18 Group centroids for function 1	207
Figure 19 Group centroids for functions 2 through 7	208

Figure 20 Concept map of computing skill schematic structure..... 258

1
2
3
4
5

ACM

ACT...

ATA...

BT.....

CQI...

CS.....

CSE

CSO

CS1...

CS2...

CUM...

FIT...

GPA

MSU

SIGC

SRS

TA.....

ZPD...

LIST OF ABBREVIATIONS

ACM	Association for Computing Machinery
ACT ...originally: American College Testing Program, now pronounced "A" "C" "T"	
ATA	Assistant Teaching Assistant
BT	Bridge Task
CQI	Continuous Quality Improvement
CS	Computer Science
CSE	Computer Science and Engineering
CS0	Computer Science 0 (zero) course
CS1	Computer Science 1 (one) course
CS2	Computer Science 2 (two) course
CUMGPA.....	Cumulative Grade Point Average
FIT	Fluent with Information Technology
GPA.....	Grade Point Average
MSU	Michigan State University
SIGCSE.....	Special Interest Group for Computer Science Education
SIRS.....	Student Instructor Rating System
TA.....	Teaching Assistant
ZPD	Zone of Proximal Development

they m

to s to

is no ic

now in

over 5

worke

not us

prima

of tec

1999

Chevr

Depot

import

iterate

know

effect

need t

trac

encou

CHAPTER 1 INTRODUCTION

As college students prepare to be the “information workers” of tomorrow, they must be able to use a variety of rapidly changing computing systems and tools to solve an ever-expanding range of problems across disciplines. This need is no longer restricted to students in technical disciplines; information technology now infuses all aspects of life. In business, computers have been responsible for over 5% of the current increase in worker productivity in the U.S. On average, workers who use computers on their jobs earn 15% more than workers who do not use computers (U. S. Department of Labor, 1996). Technology has been the primary driving force behind the longest bull market in U.S. history. The impact of technology is not a temporary “blip” in the stock market. On November 1, 1999, the Dow Jones Industrial Average removed long standing blue chip stocks Chevron, Goodyear, Sears, and Union Carbide, replacing them with Home Depot, Intel, Microsoft, and SBC Communications, companies that reflect the importance of information technology to the U. S. economy.

Information technology is also changing what it means to be educated or literate. The role of teacher is changing from transmitter of a particular canon of knowledge to helping students learn to access and evaluate information effectively and critically (American Association of School Librarians, 1999). The need to prepare students to use information technology has spread from the traditional technical disciplines, such as the sciences and engineering, to encompass every academic discipline. As a result, higher education faces the

maiden

RIT, or

Techno

univers

adder

college

Proced

effect

helping

Prepar

Curric

meet t

meet

develo

major

stude

the st

source

challenge of helping students learn to be Fluent with Information Technology (FIT) over a lifetime of constant technological change (Committee on Information Technology Literacy, 1999). Michigan State University – a large, land-grant university – is a microcosm of these trends. Interviews with the chairs of 67 academic units at MSU indicate that employers in every discipline expect the college graduates they hire to know not only the important concepts and principles of their domains, but to be able to use information technology effectively. Employers count on recent college graduates to play a key role in helping other employees learn to use information technology.

There are several questions that need to be answered if we are going to prepare students to be FIT. What constitutes FITness? What is an appropriate curriculum and how do we determine it? What is the best instructional design to meet these goals? How do we assess student outcomes to determine if we are meeting the goals? Michigan State University addressed these questions by developing an introductory computer science course for non-computer science majors. This dissertation is an evaluative study of how well this course prepares students to be FIT.

The rest of this chapter sketches an overview of the above questions and the structure of the course. It then outlines the research questions and data sources used to evaluate how well the course answers these questions.

The
informatic
informatic
deeper le
knowledg
and to lea
three type
computer
concepts
intellectua
situations
skills will o
while the c
restricted
undergrad
applicatio
to solve pr
this definit
educational
is not so ci

Primary Questions

What Constitutes FITness?

The report Being Fluent with Information Technology (Committee on Information Technology Literacy, 1999) introduces the term Fluency with Information Technology (FIT). Persons who are FIT move beyond “training” to a deeper level of conceptual understanding that allows them to apply their knowledge of information technology to solving new problems in new domains and to learn to use new software as it becomes available. FITness requires three types of knowledge: (a) contemporary skills, the ability to use various computer applications; (b) foundational concepts, the basic principles and concepts of computing that form the basis of computer science; and (c) intellectual capabilities, the ability to apply information technology in particular situations and use this technology to solve new problems. The contemporary skills will change quickly over time, with the advances of computer software, while the underlying concepts are more stable. Intellectual capabilities are not restricted to a single course but should be developed throughout the undergraduate curriculum. Therefore, students should learn both skills with applications and computing concepts and principles so they could use computers to solve problems in a variety of disciplines after they leave the course. While this definition of FITness is one that many faculties in computer science and educational psychology can support, the best curriculum to achieve these goals is not so clear.

disc p

rap d

recom

Curre

recom

sub e

*know

comp

with

(Kof

Some

des

not f

scie

from

cont

Wha

ACU

bet

What is an Appropriate Curriculum and How Do We Determine It?

Because computer science (CS) is a young and quickly changing discipline, the structure and content of the curricula and courses is evolving rapidly. The Association for Computing Machinery (ACM) published its first recommendations for CS undergraduate recommendations in 1968 (ACM Curriculum Committee on Computer Science, 1968). The most recent ACM report (Tucker & ACM/IEEE-CS Joint Curriculum Task Force, 1991) specifies the subject matter that should be included in a CS curriculum as a series of "knowledge units" and is generally accepted as the basis for the undergraduate computer science curricula in most accredited institutions.

ACM also has recommendations about the contents of specific courses within the undergraduate CS curriculum. These courses were labeled CS1 (Koffman, Miller, & Wardle, 1984) and CS2 (Koffman, Miller, & Wardle, 1985). Some writers refer to introductory service courses for non-CS majors with the designation CS0 (e.g., Goldweber, Barr, & Leska, 1994). However, the ACM has not formally addressed the CS0 service course intended for the non-computer science students. Therefore, introductory service courses have had no advocates from within the computer science community to provide guidelines on the nature, content or structure of these courses. As a result, there is disagreement over what to teach and why it should be taught in much of the ACM CS0 literature.

A review of all 54 articles about CS0 courses that have appeared in the ACM Special Interest Group on Computer Science Education (SIGCSE) literature between 1979 and 1998 suggests that this literature may be broadly classified

along tw

should w

else (usu

teach wh

W

concept

which co

concept

reflected

T

concept

major fie

confiden

Maintain

and tran

who cte

tradition

students

points

new pro

along two dimensions. The first dimension is "what" should be taught; that is, should we teach CS0 students to program or should we teach them something else (usually application software)? The second dimension is "why" we should teach whatever we are teaching.

While these authors agree that it is important for students to have a conceptual understanding of information technology, there is no consensus about which concepts are critical and what curriculum best helps students learn the concepts. One of the questions this study explores is the curricular perspective reflected by the course design.

What is the Best Instructional Design to Meet These Goals?

The desired outcome is for students to learn the underlying computing concepts and principles so they can transfer them to solving problems in their major fields of study. Furthermore, students should acquire the ability and confidence to learn to use new software and to solve new problems on their own. Maintaining student motivation is important to ensure a high degree of retention and transfer. However, it is often difficult to stimulate interest among non-majors who often see the course only as a "requirement." Combine these factors with traditional assessments such as quizzes and multiple choice exams and students' primary motivation often becomes a quest for the extrinsic rewards of "points." Under these conditions, there is little retention and even less transfer to new problems.

How do w

Ass

require st

1983). Co

require th

they will e

& Rosche

Ma

outcomes

course ma

des red le

grade on r

Levine, 19

accommo

large univ

single sem

(Osin & Le

Mid

semester,

majors. To

Continuous

lements

How do we assess student outcomes to determine if we are meeting the goals?

Assessments must be consistent with the instructional objectives and require students to demonstrate competence in a variety of situations (Merrill, 1983). Computing lends itself to authentic performance-based assessments that require the students to apply their knowledge to solve problems similar to those they will encounter in courses in their majors or in the workplace (Smith, diSessa, & Roschelle, 1993, p. 149).

Mastery learning is one strategy for coupling assessment with desired outcomes. In traditional mastery learning, students continue to work on the course materials until they demonstrate mastery of specified materials at the desired level. They do not take a fixed set of examinations in order to receive a grade on the basis of single-attempt assessments (Block, Efthim, & Burns, 1989; Levine, 1985). Instead of the instructor setting the pace, mastery learning can accommodate individual student variation (Lee & Pruitt, 1984). However, in a large university curriculum, students are expected to complete courses within a single semester, so there is usually little opportunity to use true mastery learning (Osin & Lesgold, 1996).

The CS0 Course at Michigan State University

Michigan State University's CS0 course is a large (1800 students per semester), introductory computer science course for non-computer science majors. To meet these enrollment demands, the design team applied Continuous Quality Improvement (CQI) principles to the design, development, implementation, management, and evaluation of the course (Kaufman & Zahn,

1993). The

apartmen

the cond

non-maj

O

course s

of the co

not requ

various p

process.

what inst

to evolve

rearrang

feedback

the cours

D.
Engineer

M.
Engineer
Psycholo

G.
Michigan

R.
Engineer

T.
Engineer

1993). This included extensive needs assessment interviews with 67 client apartments and extended discussion with other computer science faculty about the concepts and principles that needed to be part of an introductory course for non-majors¹.

One of the earliest decisions the design team made was to design a course structure that would be very modular, allowing for changes in some or all of the content while minimizing the disruption on parts of the instruction that did not require change. An important part of the instructional framework is that the various parts are tightly coupled with feedback loops built into all phases of the process. These feedback loops provide the data that is used to help determine what instructional and assessment changes are needed. This allows the course to evolve with the rapidly changing computing environments by changing and rearranging course content within this modular instructional framework. The feedback loops also provide a rich data set for extensive evaluation of how well the course prepares students to be FIT.

¹ The design team consisted of:

Don Weinshank, Professor, Department of Computer Science and Engineering, Michigan State University.

Mark Urban-Lurain, Instructor, Department of Computer Science and Engineering; Ph.D. candidate, Department of Counseling and Educational Psychology, Michigan State University.

Gary McCuaig, Instructional Designer/Producer, Instructional Television, Michigan State University.

Ryan McFall, Ph.D. candidate, Department of Computer Science and Engineering, Michigan State University.

Tom Danieli, then a M.S. candidate, Department of Computer Science and Engineering, Michigan State University.

conce

to be f

report

learn

progr

train

level

induc

comp

class

Reig

conce

conce

(Ten

instr

conce

Previ

on the

Succe

The Curriculum

The design team decided that the curriculum had to focus on computing concepts and principles. However, the client departments needed their students to be fluent with a variety of ever-changing computing applications. They reported that students who had taken the previous CS0 course in which they learned computer programming were unable to transfer what they learned about programming to the use of a variety of application software. However, simply "training" students on application software would not prepare them for the higher level problem-solving aspects of FITness.

Rather than taking a deductive approach, the design team took an inductive approach. The course uses a spiral curriculum to introduce students to computing concepts by having them solve a series of problems that epitomize classes of problems for which various computing skills are the solutions (Reigeluth & Stein, 1983). Students thus build from procedural skills towards conceptual understanding, rather than first trying to learn decontextualized concepts and then attempting to use those concepts to solve problems (Tennyson & Cocchiarella, 1986).

As students grapple with what may appear to be unrelated problems, the instructor ties them together, showing how each is an example of particular concepts or principles. Subsequent instruction relates the new problems to the previously learned concepts and principles. Students can thereby "triangulate" on these concepts and principles, refining their schemas as they solve successively more abstract problems. Ultimately, the students' conceptual

understanding becomes rich enough to support independent problem solving beyond that which is possible with solely procedural skills.

The Instructional Design

The course was designed to enhance student motivation. First, the course offers several different “tracks,” with each track having focal problems from a variety of domains to pique student curiosity. Students can select the track most appropriate to their interests or major, increasing the relevance of the materials. Students who are interested in fields in which data analysis is crucial may take the track that concentrates on problems requiring the collection and analysis of data. Students interested in disciplines that emphasize writing might take a track that concentrates on creating Web sites and preparing reports and presentations. Regardless of the focal problems in the track, the purpose of each track is to help students learn the underlying computing concepts and principles that are common to all computers and software, such as data representation and manipulation.

Second, rather than an individualistic, competitive structure, the course is based on a collaborative learning model. One of the characteristics of such instruction is that specifically addresses student expectancy of success. Exercises are designed to encourage students to help each other succeed and engender feelings of competence, rather than to stratify and categorize students in a competitive manner (Johnson, Johnson, & Smith, 1991, p. 2:9).

consi

with

creat

caus

sche

perfo

conce

probi

exten

stude

view o

Attr

comp

princ

bas s

resea

being

creat

The Assessments

To accommodate individual student differences, keep assessments consistent with the goals of encouraging student problem-solving, and work within the institutional constraints of a fixed-credit semester, the design team created a modified mastery model assessment called *bridge tasks* (BT). The course progresses at the pace specified in the instructional design. At regularly scheduled intervals, students take a bridge task. Bridge tasks are individualized, performance-based assessments that require students to synthesize the concepts and competencies they have learned to solve a variety of computing problems.

A key factor in the bridge tasks is that each one contains one or more *extension tasks* that are designed to evaluate transfer. Extension tasks test the students' ability to apply the concepts and principles they have learned to solve new classes of problems about which they have not received direct instruction. Although learning a set of skills for using particular software may help students complete routine tasks, they must understand the underlying concepts or principles to complete these extension tasks.

Bridge tasks are criterion referenced and evaluated on a mastery pass/fail basis. If a student fails a bridge task, he or she continues in the class but must repeat the failed bridge task until he or she has successfully passed it before being allowed to take subsequent bridge tasks to increase his/her course grade.

There are several advantages to this assessment model: 1) it provides a greater opportunity for formative feedback than traditional multiple choice

exa

por

tas

gr

the

mea

this

shi

Inter

thes

clie

acc

clie

ges

rep

mea

clie

examinations; 2) the student's motivation shifts from the extrinsic accumulation of points, to the intrinsic goal of mastering of the concepts so they can complete tasks similar to those they will encounter in their subsequent courses and after graduation; and 3) the course grade indicates what concepts and competencies the student has mastered. Thus, client departments have a more reliable measure of student abilities when planning subsequent instruction that requires this course as a prerequisite.

Research Questions

Recall that FITness requires three types of knowledge: (a) contemporary skills; (b) foundational concepts; and (c) intellectual capabilities (Committee on Information Technology Literacy, 1999). However, the best curriculum to achieve these goals is not so clear. How does this definition of FITness meet the MSU client department needs for their students? How well does the curriculum address questions of FITness? How well does the curriculum meet the MSU client department needs?

Since the goals of FITness reach across all disciplines, it is important to design instruction to maximize retention and transfer. How well does the course prepare students to solve new problems? How well do the assessments measure transfer?

Data Sources

The course design provides three sets of data to answer these questions: **stu**dent data, instructor and teaching assistant data and external data from

student

On the

income

records

saved

about

respon

about

the nu

compi

some

to the

each

effect

some

the co

the co

insto

reco

about

students and the client departments. There are several types of student data. On the first day of each semester, students complete a survey to determine their incoming computing experience. Each student's daily class attendance is recorded. Each student's email correspondence with the course instructor is saved. Data from each student's bridge tasks include fine-grained information about the performance on each criterion of each BT.

The course has a large number of teaching assistants (TAs) who are responsible for the daily classroom instruction. There is extensive data both about and from each teaching assistant. Records are retained for each TA about the number of sections and semesters of teaching experience. Students complete instructor ratings for their TAs at both midterm and at the end of the semester. The teaching assistants complete on-line forms that provide feedback to the course instructors on individual daily exercises and overall feedback on each day's class. The course instructors use this data to evaluate the effectiveness of each classroom exercise and refine the instruction each semester.

The course instructors maintain contact with students to assess how well the course is meeting the design objectives. Each instructor teaches sections of the course to assess the effectiveness of the lesson plans. The course instructors also meet with students from all sections during office hours and record the nature of the students' problems in the course database.

At the end of each semester, students complete an extensive survey about the course and about their teaching assistants. Client departments provide

feedback through semi-annual meetings of an oversight committee composed of the academic deans from each of the client colleges.

These rich sets of data were incorporated as part of the instructional design with two purposes. First, to provide feedback for the ongoing revisions of the instructional design and delivery. Second, to allow evaluation of how well the course meets the goal of preparing students to be FIT. This study examines questions of how well this approach meets the goals of FITness.

acc

Tech

edu

we

it?

info

instr

sho

we a

the

the

be c

ass

W

form

rius

out

De

CHAPTER 2 LITERATURE REVIEW

The primary questions identified in the introduction cut a wide swath across the educational literature. First, what constitutes Fluency with Information Technology (FIT)? This question has cultural, economic, political and educational facets. Once we select a perspective from which to define FITness, we can then ask “What is an appropriate curriculum and how do we determine it?” The educational psychology and computer science education literature can inform curricular decisions and the answers to the question “What is the best instructional design to meet these goals?” Finally, the assessment literature should inform the question “How do we assess student outcomes to determine if we are meeting the goals?” Each of these questions can be – and has been – the subject for copious research. For this study, the literature was reviewed from the theoretical perspectives adopted by the design team.

What Constitutes FITness?

Questions about preparing students to use information technology must be considered from a cultural perspective to understand what may be implicit assumptions in the criteria that we eventually select. As Bowers points out, “When we think that expertise in the area of computers involves only a technical form of knowledge for using and improving computers, we are, in fact, under the influence of the conceptual guidance system of our culture. In terms of the cultural bias built into our way of thinking, new ideas and technologies are understood as progressive by their very nature” (Bowers, 1988, p. 2). Bowers

goes on to argue that these cultural biases lead to a “conduit” view of language rather than a view of language as a “dynamic process that shapes our thoughts as we use it to communicate with others” (p. 41). This conduit view of language can lead to the perspective that information technology is simply a way of expanding the language conduit. Furthermore, in the United States, these cultural paradigms lead to an overwhelmingly economic perspective on answers to what constitutes FITness (Besser, 1993).

Since the early 1980’s, reports on the needs and crises in education have explicitly addressed the need to prepare students to be part of a computer-literate workforce. A Nation at Risk frames the “risk” in the context of a workforce that may not be prepared to compete in a global economy that is driven by technology (Gardner & and others, 1983). More recently, the 1999 National Academy of Sciences report Being Fluent with Information Technology (Committee on Information Technology Literacy, 1999) asked “what should everyone know about information technology in order to use it more effectively now and in the future?” The report claims that the term “computer literacy” has come to be associated with superficial “training” on computer applications rather than concentrating on deeper conceptual understanding that will prepare students to cope with rapidly changing information technology. The report therefore introduces the term Fluency with Information Technology (FIT) to replace the term “computer literacy.” Persons who are FIT: (a) are able to cope with unexpected difficulties due to immature technology; (b) have a deeper understanding that allows them to customize off-the-shelf applications to meet

th

tr

to

ca

co

co

fo

th

inf

ne

-f

de

fa

de

un

re

so

their needs; (c) have abilities beyond the rudimentary use of the tools that makes the tools more useful to them; (d) have a better understanding that allows them to exploit new developments with information technology and comprehend their capabilities more quickly; and (e) have a better understanding that allows them to cope when things go wrong or new applications become available.

According to the report, FITness requires three types of knowledge: (a) contemporary skills, the ability to use various computer applications; (b) foundational concepts, the basic principles and concepts of computing that form the basis of computer science; and (c) intellectual capabilities, the ability to apply information technology in particular situations and use this technology to solve new problems. Regardless of the cultural biases that may underlie these types of knowledge, they do provide a useful framework in which to make curricular decisions.

What is an Appropriate Curriculum and How Do We Determine It?

In higher education, the task of teaching about computers has traditionally fallen to the computer science departments. Since most course curriculum decisions are made by faculty at the department or college level, it is important to understand what content computer science faculties deem to be important. This next section reviews the history of computer science instruction for non-computer science majors in higher education.

Tr

wh

Tr

es

un

So

th

Co

re

cu

W

a

fo

J

n

P

f

S

t

c

c

History of Computer Science Instruction for Non-Majors

Traditional Computer Science Curriculum

There is general agreement among computer science professionals as to what should constitute the curriculum for a degree in computer science (CS). The Association for Computing Machinery (ACM), the oldest professional computer science organization, published the first recommendations for CS undergraduate recommendations (ACM Curriculum Committee on Computer Science, 1968). These recommendations were updated a decade later to reflect the advances in computing technologies and the maturing of the discipline (ACM Curriculum Committee on Computer Science, 1979). ACM also made recommendations about the contents of specific courses within the CS curriculum. The first two courses in a CS curriculum were designated CS1 (Koffman et al., 1984) and CS2 (Koffman et al., 1985). Because of the rapid advances in computing technology, ACM again revised their recommendations for the computer science curriculum for majors in 1991 (Tucker & ACM/IEEE-CS Joint Curriculum Task Force, 1991).

The purpose of an introductory CS1 course usually includes an introduction to the field of computer science, problem solving, algorithms and programming to prepare students for subsequent courses in the curriculum (Barrett, 1996). The conditions of instruction vary by institution, but generally the students in CS1 courses have relatively homogeneous math and science backgrounds and are motivated to learn the material in order to pursue further courses in computer science. The desired outcome of CS1 courses is for

students to have the background they need to move into a CS2 course and to be prepared for the rest of the computer science curriculum. There are a number of instructional methods that are appropriate for meeting these outcomes, but in most CS1 courses, students spend a substantial amount of time designing and writing computer programs to learn the fundamentals of problem decomposition, algorithm development and programming language syntax. This curriculum is intended to provide students with a foundation upon which their subsequent courses can build a deep structural understanding of computing concepts and principles.

Curriculum for Non-Computer Science Majors

With the advent of microcomputers, introductory computer science courses specifically intended for non-computer science and non-science/engineering majors flourished at most colleges and universities. Initially, the primary content of these courses was an introduction to computer programming, since it was still necessary to program in order to use microcomputers. Microcomputers changed a number of the conditions for these courses by allowing stratification of the curriculum. While computer science students learned Pascal with an emphasis on algorithms and data structures, engineering and science students often learned FORTRAN, with an emphasis on solving computational problems. At the same time, students who had not previously taken computer science courses began to take introductory courses that focused on programming in BASIC (e.g., Weinshank, Urban-Lurain, & Danieli, 1990; Weinshank, Urban-Lurain, Danieli, & McCuaig, 1995; Weinshank,

Urban-Lurain, & Olds, 1988). However, an underlying assumption of these courses was still that for students to understand computing concepts, they had to understand how to program computers.

While the ACM has been actively involved in revising recommendations for CS curricula and courses intended for undergraduate computer science courses, they have not addressed the curricula of CS service courses intended for the non-computer science students. This means that there are no professional organizations providing guidelines for the nature, content or structure of the non-major courses. This does not mean that faculty in computer science departments have ignored these courses. For example, some writers refer to introductory service courses for non-CS majors as CS0 following the CS1 and CS2 designations (e.g., Goldweber et al., 1994). However, there is no set of accreditation guidelines to which computer science departments or faculty turn when creating these courses as they do when creating curricula and courses for majors. One resource for faculty creating CS0 courses is the ACM Special Interest Group on Computer Science Education (SIGCSE).

Within ACM, SIGCSE provides a forum for university faculties who are involved in teaching computer science. They hold annual conferences and publish regular bulletins and conference proceedings (ACM Special Interest Group on Computer Science Education, 1998). Until 1997, ACM SIGCSE was associated with the National Educational Computing Conference but ended that association because the foci of the two organizations were diverging, with NECC concentrating on computing in K-12 and SIGCSE focusing on higher education.

Trends in CS0 Courses

Since computer science faculties have had the responsibility for addressing the needs of non-major students, this section analyzes the trends in courses for non-majors. All of the National Educational Computing Conference Proceedings from 1979 through 1981, and all of the SIGCSE Bulletins and SIGCSE Conference Proceedings from 1982 through 1998 (Vol. 1) were reviewed. All articles in which the primary focus was CS0 courses were included. Generally, these articles conform to the SIGCSE "regular paper" format, describing classroom experiences, new curricular initiatives, new teaching techniques, or an educational research project related to particular CS0 courses at the authors' institutions. That is, they represent what departments are actually doing in this area as opposed to what computer science faculty believe should be done.

Each of the 54 articles was classified into one of seven categories based on its primary emphasis. In alphabetical order, the categories are: Applications, Management, Multimedia, Programming, Simulations, Social and Survey.

Applications

These are courses in which the primary instructional focus is on learning about computers through the use of a variety of application software. For example, Wetmore (1980) identifies three different classes of students who need computer education: computer science students, other science students and everybody else. He reports that from 1969 until 1976, they taught all students batch programming but for the non-majors they "were not successful in

convincing the students that the computer was a useful tool, however, as most of them never used it during the rest of their academic careers" (p. 140). As a result, they taught non-majors word processing, using the EDT line editor on a time-sharing system. This is one of the earliest reports of teaching word processing rather than programming in a CS0 course.

By the mid-1980's, the proliferation of microcomputers prompted many authors to question the need to teach programming to non-majors. Bailey (1987) argues that programming is "archaic" (p. 503) and that students can learn to use computers to solve "real" problems and design algorithms by using spreadsheets or databases, allowing them to focus on higher level problem solving while avoiding the frustration and minutiae of programming.

While many authors identify problems trying to teach programming to non-majors, others point out that simply replacing programming with applications "training" that focuses on keystrokes does not prepare students to be independent problem solvers (Goldweber et al., 1994). Some authors report teaching non-majors by replacing programming with applications and grappling with assessing what students have learned. Lee and Pruitt (1997) used homework, rather than exams, to assess students' learning. Townsend (1998) identifies problems evaluating students with the traditional software applications approach, either by objective tests that only test keystrokes or by projects in which it is difficult to verify that the work is that of the student. She added first-hand observation of student work on their projects as part of her evaluation process.

The complete list of articles categorized as having primarily an application emphasis includes (Bailey, 1987; Curl & Hussin, 1993; Dyck, Black, & Fenton, 1987; Goldweber et al., 1994; Kolesar & Allan, 1995; Kolodny & Ott, 1981; Lee & Wu, 1997; Martin, 1986; Peterson, 1987; Rabung, 1994; Townsend, 1998; Wetmore, 1980).

Management

In recent years, the demand for CS0 courses has significantly increased enrollments and course sizes at many institutions. As a result, some articles focus on the logistics and management of courses of this size. Canup and Shackelford (1998) discuss several pieces of software they implemented to support the large service courses. They use software to help them manage groups of students, distribute, collect, grade and provide students with feedback on assignments, and provide some individualization for students in a large course. Kay (1998) discusses some of the logistic problems faculty must address as enrollment in CS0 courses expands to hundreds of students per semester. He identifies three problems associated with increased course size: (a) increased faculty workload, which includes more time to prepare and evaluate assignments and managing the greater diversity of student abilities; (b) student anonymity, leading to reduced student motivation and engagement and increased attrition; and (c) instructional inconsistency, if large courses are broken into multiple sections with Teaching Assistants, instruction and evaluation can vary across the sections. Kay discusses several strategies that faculty can employ to address these problems.

Nulden (1998) contends that most instructional uses of computers have focused either on presenting information or on using computers for student practice. He used a Newton® to keep notes about student and group work and used this system to provide rapid feedback to his students. His premise was that assessment in higher education should focus more on process and less on product, and that mobile computing can support this. This system allowed continuous assessments to be integrated into the course. While the author suggests using such a system in large-enrollment courses, he had only tested it in small classroom settings.

Multimedia

The primary instructional focus of these courses is on the creation of multimedia content. These are fundamentally programming courses. However, rather than studying procedural programming languages, students learn how to program to create multimedia content. King and Barr (1997) emphasized computing concepts by teaching scripting languages so that art students could create multimedia content. Spooner and Skolnick (1997) used hypermedia to allow engineering and science students to explore case studies in their domains. Gurwitz (1998) used HTML as a "gentle" introduction to programming and reports that students were more motivated to learn the material because of their interest in the World Wide Web.

Programming

The primary focus of these courses is on learning to program in a variety of languages including Basic, Scheme, Pascal, and others. The earliest CS0 literature advocates programming since at that time there were few other ways to use computers. However, even the earliest articles acknowledge that students taking CS0 courses have different interests, needs and motivations than do majors. Clark (1979) advocated a developmental metaphor for defining computer literacy. "Instead of understanding how computers work, computer literacy is simply the ability to exchange information with computers at a level appropriate to the problem the user wants to solve" (p. 107). Clark proposes a five level developmental model of computer literacy. Level one involves communicating with a computing system. At level 2 users can use programs that require decisions about program function but still require no decisions about computer function (e.g., which analyses to select when using SPSS). Levels 3 through 5 require the ability to program.

Rodriguez and Anger (1981) offer a pragmatic definition of computer literacy as a justification for teaching all students to program.

[Computer literacy is] the ability to read and write computer programs of moderate complexity in some high-level language and the ability to choose an appropriate computer to carry out a particular implementation. In short it means having sufficient knowledge for living and working with computers in the present computerized age. (p. 66)

Thus, the rationale for teaching programming to CS0 students was that programming constitutes the minimum "sufficient knowledge" that literate members of society required at that point in time. This requirement was pragmatic; it did not imbue learning to program with any cognitive attributes other than equipping students with the knowledge needed to use computers to accomplish tasks in other domains. However, as microcomputer software advanced rapidly in the 1980's, the rationale for programming began to change.

Halaris and Sloan (1985) approach the issues of computer literacy from a liberal arts perspective. They define four levels of computing knowledge: (a) computing awareness, including an introduction to computing, understanding the applications of computing, and the roles of data and computing; (b) computing literacy, including "hands on" computing experiences, experience with computer-based problem solving and computer-based information services, and some introduction to computer programming; (c) computing fluency, which requires advanced computer-based problem solving using structured analysis, design and programming; (d) computing expertise, the level of knowledge exhibited by computer professionals. They discuss the "controversy" over programming, since by 1985 computer applications allowed people to complete tasks far more complex than could be accomplished only by programming just a few years earlier. While they admit that most non-majors will never program, they assert that "since computing is based upon programming and programs, all individuals should understand the concept and process of programming" (p. 324). Thus, the

definition of literacy moved from the pragmatism of Rodriguez and Anger (1981) to the more abstract requirement for understanding deeper computing concepts.

By the mid-1990's, authors who advocated programming acknowledged that students would not need to program to use computers, but most still claimed that programming was necessary to understand computing. Biermann et al. (1994) propose four levels of abstraction that literate persons should understand: (a) language translators, (b) machine architecture at the assembly language level, (c) the switching circuit implementation of computer architecture, and (d) the VLSI circuitry from which the computer components are constructed. Their goal "is to enable students to understand the mechanisms of computation throughout the hierarchy beginning with a higher level language (Pascal in this case) continuing through translation and execution phases down to the electrons in the semiconductor chip" (p. 295). Clearly, the definition of literacy was becoming more abstract than the pragmatism of the 1980's.

Konstam and Howland (1994) used the Scheme language in introductory courses for liberal arts students. They used the analogy of programming as a "language" in the human sense, with different dialects, syntax, semantics, etc. They claim non-majors should learn to program for two reasons: (a) to learn about structure of languages, in particular, natural languages; and (b) to teach about data and procedural abstractions. This moves beyond the definition of literacy as a deep understanding of computing to an assumption that there is a deep structural relationship between natural and computing languages. This implies that there is some underlying similarity between how computing

languages are implemented in computer hardware and how human language works.

As computing technology advanced and faculty continued to teach programming in CS0 courses, many encountered problems with student motivation to learn programming; students did not see how programming would actually help them use computers. Herrmann and Popyack (1994; 1995) address the impact of students' perceptions of the relevance of learning to program on motivation. They structured a course that focuses on using programming to solve data analysis problems developed in conjunction with faculty from students' major departments. They based their instructional design on situated cognition research, incorporating authentic learning environments and tying the programming concepts to data analysis to enhance student motivation.

The complete list of articles categorized as having primarily a programming emphasis includes (Arnou, 1991; Arnou, 1994; Baruch, 1986; Beidler, Cassel, Lidtke, & Owens, 1985; Biermann et al., 1994; Cherry, 1986; Clark, 1979; Dalbey, 1991; Ellison, 1980; Halaris & Sloan, 1985; Herrmann & Popyack, 1994; Herrmann & Popyack, 1995; Konstam & Howland, 1994; Levine, Woolf, & Filoramo, 1984 ; McFall & Stegink, 1997; Popyack & Herrmann, 1993; Price, Archer, & Moressi, 1988; Ricardo et al., 1986; Rodriguez & Anger, 1981; Ryder, 1984; Schimming, 1980; Spresser, 1985; Tu & Johnson, 1990).

Simulations

The emphasis in these courses is on learning about computers by creating or using simulations. Woodson (1982) created a Personalized Instruction System

with self-paced modules for a computer-managed course designed to teach introductory computer literacy. The simulations allowed students to explore the operation of computing systems. They also used computers to facilitate course administration by using E-mail for student-instructor interaction and computer-administered quizzes for assessment. Taffe (1991) had students using simulation software to build models and simulations of phenomena such as deer populations, small town growth and environmental pollution.

Social

The course emphasis in these articles is on the social impact of computers, their roles in a changing society and the resulting literacy requirements. Baron (1984) asks how educators make decisions in any discipline about appropriate content for a particular audience. She proposes four categories that can be used to make these decisions: (a) acculturation, the fact that computing is becoming more ubiquitous; (b) economic considerations, students who understand computing will be able to obtain better jobs; (c) social mechanisms, the need to understand computing to function in a changing society; and (d) mental discipline, the idea that learning in one discipline results in general improvement in mental functioning. She notes that even though Thorndike refuted "mental muscle" theories as early as 1901, discussions of computer literacy often invoke mental discipline as one justification for learning programming.

Another aspect of the social impact of computers is that there are ethical and legal issues that did not exist before the advent of computing. Turk and

Wiley (1997) note that issues of intellectual property, software piracy, viruses, hackers, targeted marketing, online privacy, and data collection affect the lives of everyone in society. They argue that computer literacy courses must address these issues if students are to be adequately prepared to deal with them.

Survey

This category includes both courses that are intended to be an introduction or overview to the discipline of computer science and "computer appreciation" courses. Schneider (1986) notes that in many scientific disciplines (e.g., physics) courses for non-majors focus on an introduction to the discipline. Such courses outline the key concepts and principles, and the nature of the problems studied by professionals in the discipline. He notes that many people outside of computer science misunderstand what computer scientists do (e.g., they think that computer science is programming.) The course he proposed emphasizes the theoretical constructs of computer science (hardware, operating systems, algorithms, programming languages, etc.) but spends little time teaching students the details of learning to write computer programs. Parker and Schneider, (1987) elaborate on this proposal. Interestingly, in 1987, they predicted that there will soon be declining demand for CS0 courses in higher education because they expect that students will learn this material in high school, a prediction that has not yet come true.

Ourusoff (1986) advocates teaching a "computational view" of nature, focusing on how computer science contributes to building models of natural, biological and social phenomena. Kay (1993) also focuses on the concepts and

current

points

require

need

emph

Feins

Schn

that th

the ar

havin

half o

categ

current issues in computer science in an honors course for non-majors. He points out that many non-majors will eventually be in positions where they will be required to make purchase, strategic or policy decisions about computing, so will need a conceptual understanding in order to make the best decisions.

The complete list of articles categorized as having primarily a survey emphasis includes (Allen, Porter, Nanney, & Abernethy, 1990; Biermann, 1990; Feinstein & Langan, 1985; Joyce, 1998; Kay, 1993; Ourusoff, 1986; Parker & Schneider, 1987; Schneider, 1986; Sellars, 1988).

Discussion of Trends in CS0 Courses

The overall distribution of articles by category is shown in Figure 1. Note that three categories, Programming, Applications and Survey constitute 80% of the articles. Programming is the largest category – with 42% of the articles having programming as their primary focus – while applications account for 21%, half of the number of programming articles. However, while the programming category is the largest, the trend over time has away from programming.

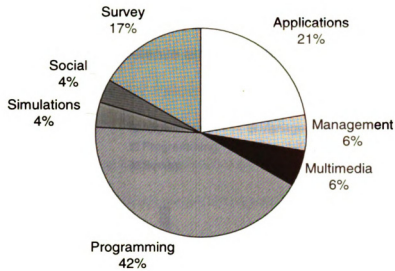
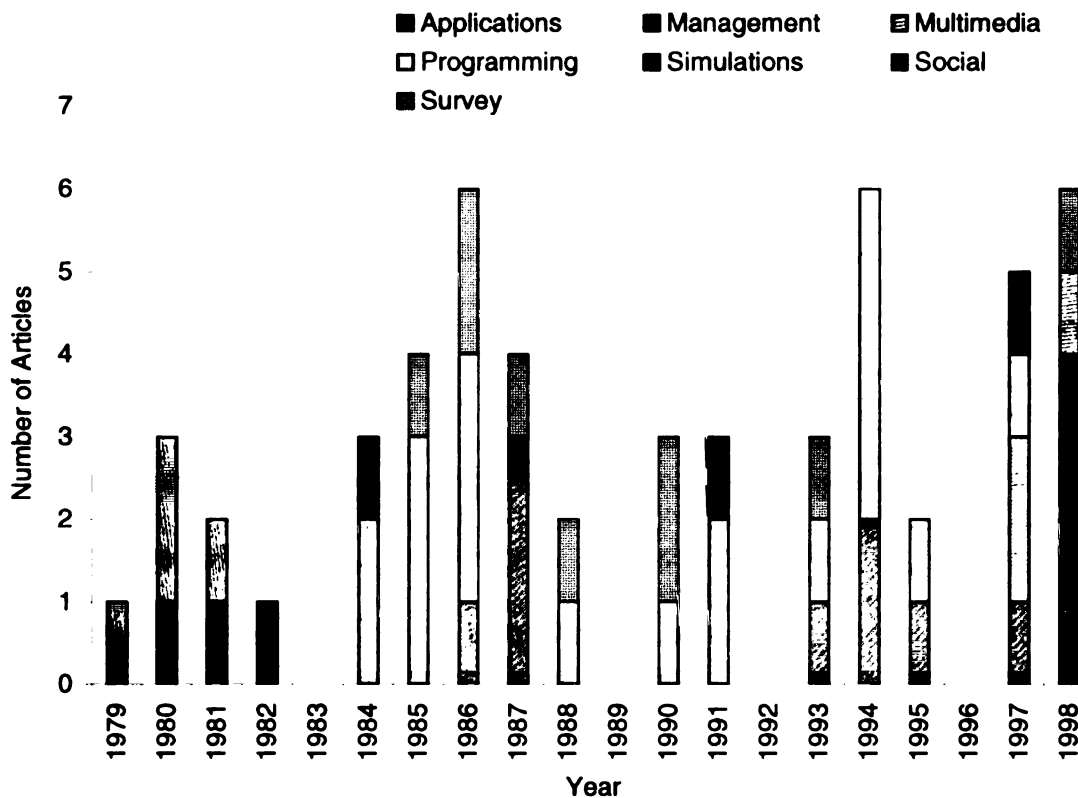


Figure 1 Distribution of articles by category

The distribution of the various types of articles over the years is shown in Figure 2. Programming has been the major focus of these courses until very recently. In the decade between 1979 and 1988, 52% of the articles had a programming focus. However, the number of courses with programming as the primary emphasis has decreased recently. From 1990 to 1998, 36% of the articles had a programming focus. Note that four of the 10 articles from that period were published in 1994. In 1998, there were no articles about CS0 courses in which the primary focus was programming. In that year, the emphasis was on management issues: coping with the rising demands of large student enrollments in these courses. This shift may be due to institutional, rather than departmental, requirements. There are institutional demands to prepare

increasing numbers of ever more heterogeneous students to use computing technology effectively. Furthermore, many colleges and universities are now requiring technological competence among students in a wide variety of



disciplines.

Figure 2 Distribution of CS0 article types by year

Just as there are no ACM CS0 guidelines, most computer science departments do not include the CS0 course in their curriculum concerns. Such courses are often taught on a rotation basis, with each faculty member doing whatever s/he believes to be appropriate when it is her/his turn to teach the course. However, with the explosive expansion of information technology in all

facets of society, the role of technological expertise and understanding as part of being a literate person cuts across most disciplines in higher education.

Computer science faculties are increasingly being called upon to be the experts who define "computer literacy." Recall that computer science faculty create most of these courses and that the conventional wisdom among CS faculty has been that learning programming is the first and most fundamental way to learn about computers. However, the role of programming in CS0 courses is controversial. Soloway (1993) asked several well-known computer scientists to discuss the need for "everyone" to learn programming. Soloway maintains that teaching programming to non-CS majors is no longer necessary because modern software eliminates the requirement to know programming to be a sophisticated computer user. His colleagues disagree and assert that learning to program does indeed have general educational benefits and argue that learning to program should be part of "computer literacy" for the general population.

The Committee on Information Technology Literacy (1999) discusses the debate over the role programming in understanding information technology. The authors – most of whom are computer scientists – assert that algorithmic thinking is valuable for educated people and provide three reasons for learning programming concepts as part of FITness:

1. Exploiting information technology systems. The report argues that educated people need to be able to customize the software to use these tools to their fullest and that such customization requires programming. However, one has to ask how likely it is that persons with one or two computing courses will

attempt to build complex systems using a large, object-oriented, event-driven programming environment such as Microsoft's Visual Basic for Applications. Such an undertaking requires not only good programming skills but also software engineering and systems analysis expertise rarely taught in introductory courses (including CS1 and CS2 courses.)

2. Gaining knowledge assumed for capabilities. The report claims that programming knowledge enhances the ability to engage in sustained reasoning and manage complexity and that this knowledge better enables people to troubleshoot or debug a computing system. Again, the practical exigencies of such situations make it unlikely that learning to program in a CS0 course will prepare students to do such troubleshooting. For example, consider the complexity of debugging new software or operating system upgrades that result in erratic system operation or crashes. An in-depth understanding of programming constructs does not make this task any less daunting when the advice from technical support is to reinstall the offending application and/or operating system.

3. Application to non-information technology problems. Finally, the report claims that learning to program enhances other abstract reasoning capabilities. It cites Papert's (1980) claims that a deep understanding of programming can support the development of new ways of thinking in domains outside of computer science. Papert was teaching LOGO to elementary students and claimed to have found evidence that learning LOGO enhanced general problem solving

abilities in students. Although his claims initially generated excitement, subsequent research has failed to find improvement in general problem solving.

Transfer from Programming to Other Domains

Claims that learning computer programming transfers to other domains and improves general problem solving are widespread. Mayer, Dyck and Vilberg (1986) reviewed several studies that examined the relationship between learning to program and general problem solving abilities. They point out that the few studies that support the claim that learning to program enhances general thinking abilities are either based on anecdotal or personal introspection data, both of which are unreliable. They conclude that "there is no convincing evidence that learning a [sic] program enhances students' general intellectual ability, or that programming is any more successful than Latin for teaching 'proper habits of mind' " (p. 609).

The particular conditions in which students learn to program may have a significant impact on how much transfer occurs. Littlefield et al. (1988) thought that there could be problems with some of the studies of transfer from programming to general problem solving because most students receive instruction that focuses only on learning to program. They claim that many studies that look for transfer assume that general problem solving skills will develop and transfer incidentally, regardless of the type of instruction. Littlefield et al. investigated the impact of teaching 5th grade students LOGO by having the teacher provide structured lessons that focused on the various elements of the language. Their hypothesis was that this more structured instruction would

enhance transfer. They found that students in the more structured environment learned LOGO better than students who explored LOGO with little explicit direction from the teachers. However, they found no differences between the experimental and control groups on tests of general problem solving abilities.

There are many possible types of transfer that might occur from programming. Transfer to general problem solving would constitute far transfer. However, such transfer is much more difficult to engender, and measure, than near transfer to more constrained domains (Kurland, Pea, Clement, & Mawby, 1989). Howe, Ross, Johnson, Plane and Inglis (1989) examined a more modest goal of incorporating LOGO as part of mathematics instruction, asking if students' mathematics abilities would improve. They report that after two years of using LOGO as part of the math curriculum, the students' teachers rated the experimental group as able to reason and argue mathematical concepts better than the control group. However, the authors acknowledge that there were confounding variables such as the experimental group receiving close personal attention and taking extra time to study math that could be likely explanations for their results. Olson, Catrambone and Soloway (1987) examined a still more constrained claim: that learning to program may provide students with skills that transfer to solving algebra word problems. They selected word problems because (a) such problems are difficult for students to learn and (b) word problems are procedural, hence they should be a "nearer" transfer from programming. However, they found no transfer to algebra word problems for the programming group.

In another test of far transfer, Kurland, Pea, Clement and Mawby (1989) studied the impact of learning programming on the general problem solving skills of high school students who had studied programming for two years. The authors found that not only did the students general problem solving not improve, the students actually had little understanding of programming at the end of two years of study. The authors conclude that the pedagogy of teaching programming needs further study before it is possible to begin making claims that learning to program transfers to other domains.

Cognition and Programming

In an effort to provide a framework in which to consider the relationship between programming and problems solving, Linn and Dalbey (1989) describe an "ideal chain of cognitive accomplishments" (p. 57) that students should obtain from learning computer programming. This chain has three components: (a) single language features, (b) designing skills, and (c) general problem solving skills. To test this framework, Linn and Dalbey examined students from 17 different middle school programming courses. They conclude that the students who receive exemplary instruction that focuses on higher-level problem solving move further along the chain of cognitive accomplishments than students who receive typical instruction that focused on the syntax of the language. However, regardless of the type of teaching, learning programming per se had no impact on students' general problem solving abilities. Discussing the lack of transfer, Linn and Dalbey conclude

It is certainly unreasonable to expect progress in general problem solving from a first course in programming. If the proposed chain of cognitive accomplishments is an accurate depiction of how such learning might occur, considerable proficiency in programming will first be necessary. Students are probably more likely to develop problem solving skills from learning several programming languages and developing some robust and general templates for programming. (p. 78)

Linn and Dalbey do not find any evidence that students derive general cognitive benefits from the limited programming experience of an introductory course because there are differences between the deep structural knowledge of expert programmers and the more fragile knowledge of novices.

Expert Versus Novice Knowledge

Regardless of the domain, experts notice features and meaningful patterns of information not noticed by novices. Experts have a large repertoire of knowledge from which to draw, and their knowledge is "conditionalized" (Simon, 1980). That is, experts not only have a broad range of experience and knowledge but they understand the appropriate conditions under which to apply particular principles. For example, it is this type of conditionalized knowledge that allows chess experts to consider only a subset of moves, rather than performing an exhaustive search. This principle also holds true in the domain of programming. Lewis and Olson (1987) point out that expert programmers have several sets of "operations" (collections of sequential statements that perform

common programming functions) that they can apply to classes of common programming problems when they encounter them. Novice programmers approach each of these classes of problems as if they are unique and must grapple with them as new problems rather than being able to apply known solutions to a recognized classes of problems.

Holt, Boehm-Davis and Schultz (1987) compared expert programmers' cognitive representations of software with those of college student programmers. Both groups were asked to modify programs written by other programmers and then to describe their mental models of the programs. The experts' mental models were influenced primarily by the difficulty of the modifications they had to perform on the programs. This group categorized the programs based on the similarities and differences in the types of modifications, invoking sequences of actions that form sets of operations (Lewis & Olson, 1987). In contrast, the novice programmers' mental models focused on the structure and content of the programs. This group tended to categorize the programs based on the similarities to other such programs they had encountered (e.g., programs that use search trees, linked lists, etc.), rather than on the modifications requested.

Although the idea that learning to program "enhances general problem solving" persists, as we have seen, this claim has little support in the research literature. As Baron (1984) points out, Thorndike dismissed the "mental muscle" concept of learning as early as 1901. Other writers (Michaels & O'Connor, 1990; Rogoff & Lave, 1984) refute the idea that learning to solve problems in a specific context – such as the constrained domain of a programming language – transfers

to

tha

tha

ma

Exp

con

don

con

to t

imp

pro

exp

don

of s

kn

two

indi

Whic

rap

Que

to other domains. It is a fallacy to claim that programming is "mental calisthenics" that builds strong minds.

So why did the computer scientists who authored the FIT report still assert that learning to program would have general educational benefits? The authors may have been extrapolating from their expert experiences with computing. Expert knowledge allows experts (e.g., computer scientists) to see the connections between problems in one domain and problem solving in other domains. In such situations, they are doing what all learners do as they construct knowledge: they are attempting to connect that which they do not know to that which they do know. However, concluding that programming causes improved problem solving in other domains is the logical fallacy of *post hoc, ergo propter hoc*. It is the deep knowledge of expert programmers, not the mere exposure to programming, which facilitates transfer to problem solving in other domains. This type of transfer requires expert knowledge that results from years of study (Bransford, Brown, & Cocking, 1999). It is precisely this expert knowledge base that the novice not only lacks but also cannot acquire in one or two courses.

The Importance of Conceptual Understanding

It is only through a conceptual understanding of technology that individuals can transfer their knowledge about technology from the conditions in which they learned about technology to new situations, allowing them to adapt to rapidly changing information technology. However, learning theory calls into question the idea that conceptual understanding requires learning to program.

Programming was originally necessary to be FIT because all interactions with the computer required programming (e.g., Rodriguez & Anger, 1981). Interaction with information technology has now moved to higher levels of abstraction. For example, where collecting and analyzing data used to require writing computer programs, today we use spreadsheets and databases to do these tasks with much greater ease and sophistication. So, if students do not program, how are they to learn and understand computing concepts? How can they learn more than a set of isolated skills that will be obsolete as soon as they leave the class?

For students to adapt to new computing systems and solve new problems, they must have a deeper understanding than is generally acquired by training with specific software packages. Students need to construct mental models or schemas of the computing systems they are using. One way to help students construct mental models is by using an inductive, rather than a deductive, approach to instruction. Course content should be organized to introduce students to computing concepts by having them solve a series of problems that epitomize classes of problems for which various computing skills are the solutions (Reigeluth & Stein, 1983). One way to accomplish this with an inductive approach is by structuring the course with a spiral curriculum (Bruner, 1960) in which students are presented with increasingly challenging problems to solve using a variety of software packages. They thus build from procedural skills towards conceptual understanding, rather than first trying to learn decontextualized concepts and then attempting to use those concepts to solve problems.

en

typ

ap

ho

ins

prn

ref

pe

kn

rep

As Tennyson and Cocchiarella note, people learn concepts “as contextual entities (correlational structures), with common attributes that are the most typical, or average, members of a class” (1986, p 45). As students grapple with apparently unrelated problems, the instructor must tie them together, showing how each is an example of particular concepts or principles. Subsequent instruction must relate the new problems to the previously learned concepts and principles. Students can thereby “triangulate” on these concepts and principles, refining their schemas as they solve successively more abstract problems. This pedagogy is consistent with a constructivist perspective on how novice knowledge evolves into expert knowledge (Smith et al., 1993, p 148). Figure 3 represents the process.

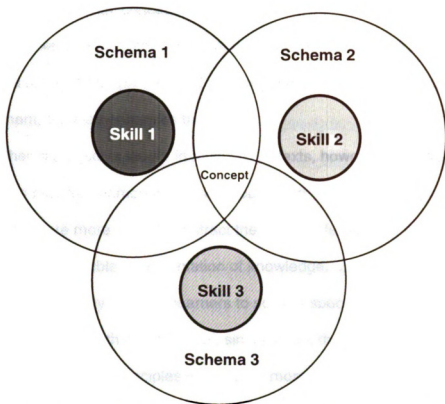


Figure 3 Concept map of relationship among skills, schema and concepts

Students learn Skill 1 (represented by the smaller circle) and build a mental schema representing a generalization of this particular skill (indicated by the larger circle), Schema 1. This schema may consist of accurate representations of the computing concepts along with misconceptions. Students then learn subsequent skills and build corresponding schemas. The intersection of these schemas triangulates on a more accurate representation of the concept (Anderson, 1984; Shuell, 1986). As learners understand the concepts underlying an increasing number of discrete skills, they reduce the cognitive load required to represent the knowledge, as compared with the requirements of storing an increasing number of discrete, unrelated skills (Chandler & Sweller, 1991).

Ultimately, the students' conceptual understanding becomes rich enough to support independent problem solving beyond that which is possible with solely procedural skills. This approach not only engenders improved conceptual development, but also enhances transfer.

When a subject is taught in multiple contexts, however, and includes examples that demonstrate wide application of what is being taught, people are more likely to abstract the relevant features of concepts and to develop a flexible representation of knowledge. ... One way to deal with lack of flexibility is to ask learners to solve a specific case and then provide them with an additional, similar case; the goal is to help them abstract general principles that lead to more flexible transfer. (Gick & Holyoak, 1983, p. 50)

Although this approach entails additional instructional expense over instructional models that focus merely on skills, the goals of retention and transfer justify the additional effort (Foshay, 1991).

What is the Best Instructional Design to Meet These Goals?

Our desired outcome is for students to learn the underlying computing concepts and principles so they can transfer them to solving problems in their major fields of study. Furthermore, students should acquire the ability and confidence to be able to learn to use new software and to solve new problems on their own. This means that the instructional design must address motivation and cognition issues that are appropriate for the CS0 student population.

Motivation

When designing instruction, student motivation is important to ensure a high degree of retention and transfer. Keller (1983) notes four dimensions of motivation: (a) interest, whether the learners' curiosity is aroused; (b) relevance, whether the learner perceives the instruction meets personal needs or goals; (c) expectancy, the degree to which the learner's perceived likelihood of success is under his or her control; and (d) satisfaction, the learner's intrinsic motivations and reactions to extrinsic rewards. Similarly, Yelon notes that students are best motivated to learn something new when they see it as relevant (1996, p 8).

In courses for CS majors, we presume a high degree of interest, relevance and sense of control. In contrast, with non-majors, it is difficult to stimulate interest among students who often see the course only as a "requirement." Further, many non-majors have enormous anxiety about using computers and fear that this is an area in which they have little control (McInerney, McInerney, & Sinclair, 1994; Ropp, 1997). When these factors are combined with traditional assessment measures such as quizzes and multiple choice exams, students' primary motivation often becomes the short term goal of completing the assignments in a quest for the extrinsic rewards of "points." Under these conditions, there is little retention and even less transfer to new problems. Students must see the course content as relevant to their interests, the instruction must account for the diverse range of student incoming experiences with computing technology, and the assessments must motivate the students to focus on higher level problem-solving, rather than extrinsic rewards.

Computers and Cognition

Much of the educational psychology literature on learning in recent years claims that learners construct their knowledge by interacting with their environment and other people. There are a number of constructivist schools of thought. Some focus primarily on the individual learner (e.g., Beilin, 1985; Berliner, 1992; Cahhan, 1992; Greeno, Collins, & Resnick, 1996). Others focus primarily on the social nature of knowledge construction (e.g., Gergen, 1994; Rogoff, 1994; St. Julien, 1994; Vygotsky, 1978). In either case, the consensus is that learning is not the mere transmission of knowledge from the teacher to the student but requires that students actively construct their knowledge. The more social perspectives emphasize the importance of linguistic interactions among individuals as a way of negotiating and constructing meaning. Winograd and Flores (1986) note that

Computers do not exist, in the sense of things possessing objective features and functions, outside of language. They are created in the conversations human beings engage in when they cope with and anticipate breakdown. ... Computers are not only designed in language but are themselves equipment for language. They will not just reflect our understanding of language, but will at the same time create new possibilities for the speaking and listening that we do – for creating ourselves in language. (pp. 78-79)

If our goal is to have students develop a conceptual understanding of computing technology, the instructional design must support active student

inquiry. We can adopt one of the two constructivist perspectives. An individual perspective focuses on the action of the individual learner in the construction of her/his knowledge (e.g., Piaget, 1977). The design team initially approached the instructional design from an individual, cognitive perspective. For a time, they considered an instructional design with a heavy emphasis on Intelligent Tutoring Systems (ITS) as the basis for instruction. However, an extensive review of the ITS literature (Urban-Lurain, 1996) showed that ITS were not likely to be fruitful. As Rosenberg (1987) points out, most ITS do not base their representations on any cognitive theory and almost all enforce one standardized style of learning.

Adopting a linguistic perspective when designing computing instruction moves us away from the idea of teaching students to think like a computer towards helping students learn to think with computers. Vygotsky defines the Zone of Proximal Development (ZPD) as "the distance between the actual developmental level as determined by independent problem solving and the level of potential development as determined through problem solving under adult guidance or in collaboration with more capable peers" (Vygotsky, 1978, p. 86). Designing instruction to emphasize collaborative learning allows students to experience the ZPD with each other in the context of using computers to solve problems. Grappling with problems in a collaborative setting brings intersubjectivity to the foreground as students interact and negotiate meaning during the problem-solving process. As peers work together to solve problems, they co-construct solutions and knowledge which they then individually internalize (Tomasello, Kruger, & Ratner, 1993). Collaborative learning also

offers several benefits such as improved achievement, enhanced critical thinking competencies, improved attitudes towards the subject area, and reduced student anxiety when learning new material (Johnson et al., 1991).

How do we assess student outcomes to determine if we are meeting the goals?

It is critical to consider assessments as an integral part of instructional design. While cooperative learning can improve student learning over individualized, competitive classrooms, assessments in a cooperative classroom can provide a new set of challenges. As Bloom, Hastings, and Madaus (1971) note, assessments are most frequently used to classify and stratify students. If assessments are norm-referenced and competitive, students will view each other as competitors, sabotaging our efforts to encourage cooperation in the classroom. To overcome this problem, instructors who adopt a cooperative learning instructional design sometimes assess students based on their performances on group projects or assessments. However, there can be serious equity problems when assessing students as a group, depending on the distribution of abilities across the group members. Groups that have high-ability students tend to have better interactions and perform better than groups that do not have such students (Webb, Nemer, Chizhik, & Sugrue, 1999). This suggests that it is important to assess and identify incoming abilities before assigning students to groups upon which their performance will be assessed, a task that is not only difficult but also expensive.

Another possibility is to use individual, criterion-referenced assessments in a cooperative classroom. If students are not competing against each other, they

are much more willing to help each other strive towards the performance goals set by those criteria (Johnson et al., 1991). Criterion-referenced assessments are also an integral part of mastery learning. In traditional mastery learning, students continue to work on the course materials until they demonstrate mastery of specified materials at the desired level. They do not take a fixed set of examinations in order to receive a grade on the basis of single-attempt assessments (Block et al., 1989; Levine, 1985). Instead of the instructor setting the pace, mastery learning can accommodate individual student variation (Lee & Pruitt, 1984). Some students may require a year to master a subject, while other students may master it in a few months. However, in a large university curriculum, students are expected to complete courses within a single semester, so there is usually little opportunity to use true mastery learning (Osin & Lesgold, 1996). Designing instruction and assessment with mastery learning as the goal within a fixed semester system is a formidable challenge.

Computing lends itself to assessments that require the students to apply their knowledge to solve problems similar to those they will encounter in courses in their majors or in the workplace (Smith et al., 1993, p 149). Performance assessment is a broad term for assessments that require students to construct responses, rather than select responses from multiple choices. Khattri, Reeve and Kane (1998) propose several characteristics of performance assessments systems that can vary. The first is the assessment task, which ranges from short, time-constrained examinations to longer portfolio work. The second characteristic is the scoring method used to evaluate the performance and can

include general guidelines or structured frameworks intended to ensure inter-rater reliability and validity. Combining the assessment task and the scoring method provides taxonomy that Khattri et al., use to define two categories of performance assessments. Task-centered performance assessments evaluate particular skills and competencies. These are fairly constrained and generally have well-defined scoring rubrics, but the underlying principles behind the tasks may not be clear to the students. Construct-centered performance assessments emphasize general skills, but do not have clear-cut scoring guidelines and are more difficult to use for summative evaluation.

Validity is a concern for performance assessments and has several components. Content validity is the most obvious. Does the assessment measure the content that we intend to measure? Beyond that, even if the assessment measures the content, is the measure meaningful? That is, does it represent “real world” applications of the content? A second aspect of validity is generalizability. Generalizability pertains to the relationship between the scores on the particular assessment and a student's ability to perform in other similar circumstances (e.g., transfer.) Finally, fairness is an aspect of performance assessments that has only begun to receive attention. Do different outcomes reflect differences that are not related to the students' abilities or knowledge of the content we are trying to assess?

Closely related to validity are concerns about reliability. There are two types of reliability that must be addressed in performance assessments. The first, inter-rater reliability is the area in which the most work has been done. It is

relatively straightforward to provide scoring rubrics to raters and calibrate scoring from different raters using statistical methods (e.g., Raymond & Viswesvaran, 1993). The bigger challenge is inter-task reliability. With a large number of complex performance measurements, student performance can be affected by not only rater variability, but also by variability in the sampling of the tasks. Shavelson, Baxter and Gao (1993) found that, in typical mathematics and science performance assessments, ten or more tasks per student are required to obtain high generalizability ratings. This means that we must have many opportunities to assess students with performance assessments; we cannot rely on one or two high-stakes performance evaluations.

Finally, if an instructional goal is to promote conceptual understanding, the assessments must be designed to evaluate the students' higher cognitive activities. Baxter and Glaser (1997) suggest a framework for evaluating the cognitive complexities of science performance assessments. They note that there are four types of cognitive activity that can be used to distinguish between experts and novices in many areas of scientific inquiry. The first is problem representation. Experts form mental models of problems before they attempt to solve them and use these representations to guide their actions. On the other hand, novices may focus on surface features of the problem that look familiar. The second type is the type of solution strategies that students apply to the problem. Experts have a rich set of strategies that they apply in the context of their mental models. Novices follow rote procedures or continue to repeat mistakes in the face of failure because they have no other alternatives to try. A

third area is self-monitoring. Experts are aware of the procedures that they are following and how successful these procedures are at converging on a solution to the problem. Novices are less inclined to engage in self-regulatory behavior. Finally, experts can explain the concepts and principles that form the basis for their solutions. Novices often simply describe their actions, rather than concepts or principles that motivated the actions. Baxter and Glaser recommend that test developers use this cognitive framework to help clarify performance objectives and determine how well the objectives fit with the actual performance scores.

Summary

This chapter reviewed the literature from the perspective adopted by the course design team. The Committee on Information Technology Literacy coined the term Fluency with Information Technology (FIT) to describe a deeper conceptual understanding of information technology than has been traditionally associated with “computer literacy.” In most colleges and universities, the task of defining and teaching the course content of the introductory course for non-CS majors (CS0) generally falls to computer science faculties. The CS0 course has traditionally included programming. However, the trend in these courses between 1979 and 1998 has been away from teaching programming. There has been much controversy over the role of programming for non-CS majors. Over the years, the justification for teaching programming in the CS0 course has moved from the practical – programming used to be necessary to use computers at all – to the contention that learning to program is needed for conceptual

understanding or to enhance general problem solving skills. However, the literature on transfer does not support these contentions.

Conceptual understanding is important if we are to teach for transfer. However, the best instructional design to promote transfer depends upon the domain and the intended audience. Maintaining student motivation is important in any instructional design. For a course that is intended for non-CS majors, the content and problems must be presented in a context that the students perceive as relevant in order to promote student interest. Collaborative learning is an effective way to promote active student learning and maintain student engagement and motivation.

Assessments must be designed that determine if students are FIT. However, a collaborative classroom structure requires criterion-referenced, rather than norm-referenced assessments. If students are competing for norm-referenced grades, they will not participate fully in collaborative exercises because they will perceive doing so as detrimental to their grade.

Finally, because FITness requires the ability to apply concepts to solving problems, it is best measured with performance-based assessments that require students demonstrate their conceptual understanding in the context of solving representative problems from domains in which they will need to use information technology. However, performance-based assessments are labor intensive to create, administer and grade. Validity and reliability are critical issues that must be addressed if performance-based assessments are to be used in a large enrollment course.

The next chapter reviews how the design team addressed these challenges by outlining the design of the course. It then presents the research questions and hypothesis and outlines the analyses to evaluate them.

CHAPTER 3 METHODS

Recall that there are several questions that need to be answered if we are going to prepare students to be Fluent with Information Technology (FIT.) What constitutes FITness? What is an appropriate curriculum and how do we determine it? What is the best instructional design to meet these goals? How do we assess student outcomes to determine if we are meeting the goals? This chapter reviews how Michigan State University has addressed these questions by tracing the design and implementation of Michigan State University's CSO course. It traces the process that the design team followed to determine their operational definition of FITness and the curriculum they designed to meet the needs of their students. It then outlines the instructional design that they created to meet these needs and the unique assessment system they created to measure student outcomes. The course implementation includes a large number of data sources that are used as part of the ongoing operation of the course and provide a rich set of data for evaluating the coherence and success of the instructional system. There are three sources of data: student data, data from the instructional staff (course faculty and teaching assistants), and feedback from the client departments. Using these data, a number of hypotheses may be tested to evaluate the success of the instructional design and implementation.

Research Setting

The design team applied Continuous Quality Improvement (CQI) principles to the design and implementation of the course (Kaufman & Zahn,

1993). Figure 4 provides an overview of the process. The *program design* took place during 1996 and 1997. It was an iterative process consisting of collecting inputs, articulating instructional goals, and developing the instructional design. The arrows in the program design portion of the figure show this iteration. From the resulting design, the team implemented the instructional system; the course was first offered for approximately 200 students during summer, 1997. Currently, the course has enrollments of approximately 1800 students per semester during fall and spring semesters and approximately 200 students during summer semesters.

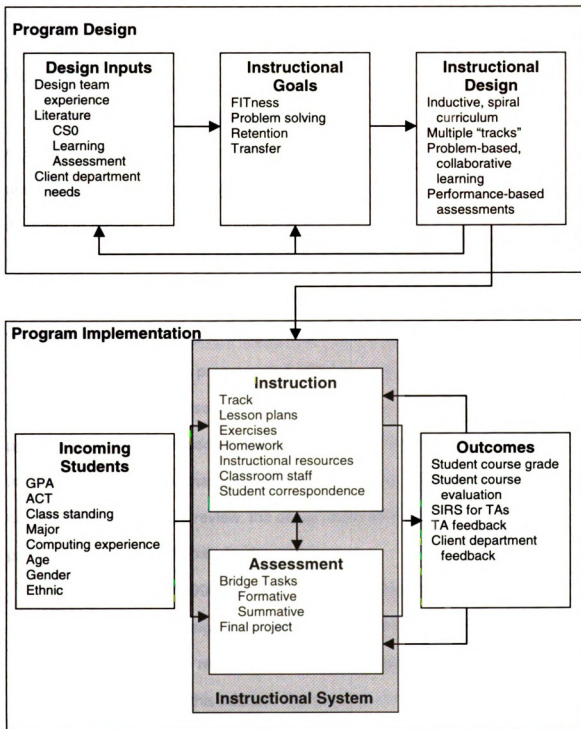


Figure 4 Program Design and Implementation

Program Design

Design Inputs

The design process began in the *design inputs* box of Figure 4. The course was going to replace two other courses, so the design team began by assessing the strengths and weaknesses of the courses that the new course was intended to replace. The first course was a “computer literacy” course in which the primary focus was on the history of computing, the social impacts of computing and learning a variety of computer applications. The second course was an introduction to programming course in which the primary focus was on learning the fundamentals programming with some exposure to computer applications. Neither course was satisfactory in terms of retention, transfer and problem solving. Before addressing these problems, the design team reviewed the literature outlined in Chapter 2. Based on their evaluation of the two existing courses and the literature review, the design team developed an outline of the preliminary instructional goals for the new course.

In the summer of 1996, after developing this preliminary set of instructional goals, the design team conducted a series of hour-long interviews with the chairs and faculty representatives of the 67 client departments whose students would be taking this course. The purpose of these interviews was to identify the computing concepts and skills the client departments considered important for their majors in their future courses and careers.

The interviews with the client departments indicated that they were concerned about the previous courses. They found that students did not retain

or transfer what they had learned to the new software or computing environments they needed to use in their majors. Furthermore, the overwhelming consensus among the client departments was that their students did not need to know or use computer programming in their subsequent courses or careers. The competencies that the client departments did identify included using computers to solve problems in a wide variety of domains and the ability to adapt to new computing systems and use them to solve a wide range of problems. Another theme that emerged from the interviews was that, regardless of the discipline, employers expect the college graduates they hire to be capable of helping their existing employees learn to use information technology.

Instructional Goals

Based upon the analysis of the interviews and the literature along with the experience the design team had teaching CS0 courses, the design team refined the *instructional goals* for the course. The competencies that the client departments did identify were consistent with many of the principles of FITness. They wanted students to be able to use computers to solve a variety of different problems across a variety of domains throughout the curriculum and into their careers. This meant that the instruction would have to promote retention. Finally, since computing systems and software are changing rapidly, the ability to transfer the concepts to new computing systems and problems was a significant objective. Although understanding computing concepts is critical to meet these goals, the traditional approach of teaching programming had been shown to be

no longer viable. This meant that the design team had to adopt a new approach to teaching computing concepts.

Instructional Design

The next step was creating the *instructional design*. In the previous CS0 course, the design team had created a highly integrated instructional design using television, textbooks, and computer simulations (Weinshank, Urban-Lurain, Danieli, & McCuaig, 1992; Weinshank et al., 1995). The focus of that course – like most CS0 courses up to that time – was programming. Because of the rapidly changing nature of computing, that course, which used a state-of-the-art programming language when it was designed, was outdated within four years. Furthermore, reaching the instructional goals of FITness, problem solving, retention, and transfer without learning programming required a different instructional design.

Inductive, spiral curriculum

As we saw in the literature in Chapter 2, many CS0 courses take a deductive approach that is an extension of the curriculum for CS majors. They present abstract computing concepts to students with the goals of having students a) learn the concepts, b) learn to identify a variety of disparate computing problems, c) link the problems to the underlying concepts and d) apply the concepts to the solution of the problems. This is a long chain of inference that has rarely been successful in CS0 courses. Students come away with little

u

p

C

m

de

no

wi

cu

sy

st

co

ch

set

The

str

the

and

Bey

Son

pres

understanding of the computing concepts and scant ability to solve new problems.

The design team decided to take the inductive approach discussed in Chapter 2 and shown in Figure 3. Critical to the success of this approach is maintaining the importance of computing concepts. Although having students develop a set of computing skills is a necessary condition for this approach, it is not sufficient. Training students on a set of “keystrokes” in a variety of software will not meet the instructional goals. Instead, the design team structured the curriculum to present increasingly challenging problems that require students to synthesize the apparently discrete skills they are learning. In this process, students begin to comprehend the utility and importance of understanding the computing concepts for the practical requirements of adapting to new and changing software.

Multiple “tracks”

After analyzing the client department needs, the design team identified a set of “core” competencies that appeared to be common across the university. These included a) the basic functions of an operating system; b) hierarchical file structures; c) computer networking (E-mail, the Web, distributed file systems); d) the ability search for and make sense of information from a variety of sources; and e) the ability to use word processing software to create research reports. Beyond the core competencies, various departments had different requirements. Some wanted their students to be able to create more elaborate reports and presentations; some wanted their students to be able to perform sophisticated

de

bu

de

of

te

co

in

pa

ch

sp

ca

co

an

CO

SO

re:

pr

pr

de

of

data analysis; and some wanted their students to use computers for financial and business applications.

Given the different applications required by the various departments, the design team decided that a single, monolithic course would not meet the needs of all of the clients. While the underlying computing concepts that the design team identified are similar across domains (e.g., using parameters to allow the computer to take different actions depending on the values of parameters) the instantiation of these concepts varies across applications. For example, Web pages can be used to demonstrate parameters by exploring the effects of changing parameters in the HTML tags for Java applets. Students can use spreadsheets to see the effects of changing function parameters on the resulting calculations. Multiple tracks allow students to learn the underlying concepts in a context that is useful to their majors. This approach maintains student interest and motivation, a key factor in learning (Keller, 1983; Yelon, 1996).

Problem-based, collaborative learning

In the previous courses, students attended lectures about computing concepts one week and then tried to apply those concepts to actual problem solving in a computer lab the subsequent week. Many students had difficulty retaining and connecting the de-contextualized concepts of the lectures to the problems they were being asked to solve a week later. Since one goal was preparing students to use computers for problem solving in their domains, the design team decided that problem-based learning had to be a major component of the instructional design. Students had to actively use computers to solve

P

t

c

c

c

c

i

a

c

c

r

c

e

t

k

w

r

f

t

r

p

(s

problems rather than listening to lectures. This meant that all instruction had to take place in a computer laboratory so students could be working with computers continuously during class time.

After reviewing the collaborative learning literature and the client departments' needs for students to work with and teach computing to others, the design team decided on a collaborative learning model. Because this is an introductory course for which there are no prerequisites, incoming students have a wide range of computing knowledge and experience. This presents a number of instructional challenges such as a) ascertaining and keeping pace with changing incoming student experience; b) maintaining motivation among the more experienced students; and c) ensuring that novice students do not become discouraged or fall behind. To accommodate the variance in student knowledge, each day the students are randomly assigned to groups of two to four so that they work with different peers every day. One day, a student may be the more knowledgeable other in the Zone of Proximal Development (Vygotsky, 1978) and will provide scaffolding for the less knowledgeable students in the group. The next day that student may be the less knowledgeable member of the group and have to articulate to the other group members what is unclear or difficult about the particular problem.

Each class consists of a series of focal problems, the solution of which requires students to learn and practice new skills. Each problem builds on previous skills and concepts, extending the range of the students' capabilities (see Figure 3.) Generally, students spend some time discussing possible

solutio

The TA

the pro

solving

the ins

design

referen

solving

surrog

authen

consis

within

modifi

curricu

schedu

solutions to the problem in small groups and then attempt to solve the problem. The TA then leads a debriefing where the groups report on their solutions and the problems they encountered. This helps the students focus on their problem solving and how it relates to the underlying concepts.

Performance-based assessment

Assessment is a critical component of the instructional design. Because the instructional goals included retention and transfer, and the instructional design uses collaborative learning, the assessments needed to be criterion-referenced (Johnson et al., 1991). In addition, to evaluate students' problem solving skills, the design team knew it was important that the assessments not be surrogate measures such as multiple choice tests but rather they wanted to use authentic, performance-based assessments.

To accommodate individual student differences, keep assessments consistent with the goals of encouraging student problem solving, and work within the institutional constraints of a fixed-credit semester, the course uses a modified mastery, performance-based assessment model. In this model, the curriculum progresses at the pace specified in the syllabus. At regularly scheduled intervals, students take a *Bridge Task* (BT) that requires them to

synthesize the concepts and competencies to that point in the course². The students must use their homework, in-class assignments and materials provided to them as part of the bridge task to solve the problem (Urban-Lurain & Weinsbank, 1999b).

To test transfer, the bridge tasks contain one or more “extension tasks” that require students to apply the concepts and principles they have learned to solve new problems they have not previously encountered. Although learning a set of skills for using particular software may help the students complete routine tasks, they must understand the underlying concepts or principles to complete the extension tasks.

Bridge tasks are evaluated on a mastery level pass/fail basis. If a student demonstrates sufficient mastery on the first bridge task, he or she “locks in” a grade of 1.0 in the course. If a student fails a bridge task, he or she must repeat the failed bridge task until passing before taking the next bridge task. For each subsequent bridge task passed, the student’s course grade is incremented by 0.5

² The design team coined the term *Bridge Task* to convey many different ideas. First, students are concerned about how their grades are determined and bring years of experience and expectations about assessments to the course. Bridge tasks are not regular norm-referenced “tests” with which most students are familiar; they are different than any assessments they have previously experienced. A unique term helps break the students’ preconceptions. Second, the assessments test transfer. Students must use the material from the class and their conceptual understanding as a “bridge” to solving new problems. Third, the assessments are based on a mastery model. Students must cross each “bridge” in order before coming to the next “bridge.” Finally – following on the physical bridge metaphor – some students have a hard time changing perspectives on assessments; they are used to collecting “points” as an extrinsic reward. To those students, the instructors can seem to be “trolls under the bridge” (Asbjörnsen & Moe, 1859/1969, p. 184-5) when they are required to demonstrate mastery by repeating any bridge tasks they do not pass.

until he o

bridge ta

increase

In

material

level. T

the basi

Instead

individu

curricul

so the

Studen

is a m

and a

the fin

still be

and p

reliab

al., 1

tasks

the t

stud

until he or she has passed the 3.0 bridge task. Once the student passes the 3.0 bridge task, he or she completes an integrative semester project that may increase his or her course grade to 3.5 or 4.0.

In traditional mastery learning, students continue to work on the course materials until they demonstrate mastery of specified materials at the desired level. They do not take a fixed set of examinations in order to receive a grade on the basis of single-attempt assessments (Block et al., 1989; Levine, 1985). Instead of the instructor setting the pace, mastery learning can accommodate individual student variation (Lee & Pruitt, 1984). However, in a large university curriculum, students are expected to complete courses within a single semester, so the design team had to adapt the course to use a modified-mastery model. Students proceed through the course at the pace outlined in the syllabus. There is a maximum of twelve opportunities to take bridge tasks during the semester and a total of five bridge tasks that students must pass to be eligible to complete the final project. Therefore, students may take each bridge task two times and still be able to complete them all. Allowing students to repeat failed bridge tasks and providing twelve testing opportunities addresses the problems of inter-task reliability and generalizability in performance-based assessments (Shavelson et al., 1993).

There are several advantages to this assessment model. First, bridge tasks are criterion-referenced, not norm-referenced. While students complete the bridge tasks individually, they are not evaluated on a competitive basis; students' grades are not dependent on doing better or worse than their peers.

This is a requirement of a collaborative learning model that fosters cooperation and peer-teaching (Johnson et al., 1991). Second, bridge tasks provide formative feedback, resulting in a greater opportunity for learning than traditional multiple choice examinations. The students' motivation shifts from accumulating points to mastering the concepts so they can complete tasks similar to those they will encounter in their subsequent courses and after graduation. Third, bridge tasks provide summative feedback; the course grade actually indicates which concepts and competencies a student has mastered. In courses that use norm-referenced assessment, a grade of 2.0 often means that a student has accumulated the mean number of points from a variety of exams and homework assignments. The grade does not indicate what knowledge the student does or does not have. The intent of this model is that the course grade reflects the concepts and competencies on which a student has demonstrated mastery.

Program Implementation

Program implementation began in the summer semester of 1997. As indicated by the arrows in Figure 4, *instruction* and *assessment* are closely coupled. The design team determined the sequence of the concepts and competencies to be taught. From that sequence, they determined which concepts and competencies each bridge task would assess. This then dictated the sequence of the instruction. Feedback from the *outcomes* is then used to refine the bridge tasks and instruction each semester. This tightly coupled system provides a structure in which the content and assessment can evolve

quickly in

from the

Instruct:

1800 s

compu

require

section

is a "le

under

design

assign

plan,

minu

com

gen

stud

trac

The

quickly in response to changes in the *incoming student* population and feedback from the *outcomes*.

Instruction

Institutional demand for the course necessitates an enrollment capacity of 1800 students per semester. The design goal of meeting each class in a computer laboratory coupled with the capacity of the computer laboratories required 60 sections of 30 students. Resource constraints dictated that each section would be met by teaching assistants (TAs) rather than instructors. There is a "lead TA," usually a graduate student, and an "assistant TA," usually an undergraduate student. To ensure instructional consistency across sections, the design team created detailed lesson plans, exercises and homework assignments for each day's instruction. (See Appendix A for an example lesson plan.)

Classroom structure

Each section of the course meets twice per week for one hour and 50 minutes per meeting. Each day begins with the students "signing in" using a computer program that records their attendance. These data are used to generate random groups in which the students work that day so that only those students who are present are assigned to groups. The attendance data are tracked in the database but students are not graded on class attendance.

Each class consists of the series of problems on which the students work. The design team structured the problems so that they draw upon the concepts

from the readings for that day. Each exercise begins with the lead teaching assistant setting up the problem and leading discussions about how the concepts apply to the problem. The instructors created a series of PowerPoint slides for each day's instruction that are keyed to the lesson plan. These resources assure consistency across sections.

The students next work for 5 to 30 minutes to solve the problem. During this time, both teaching assistants circulate among the students to facilitate their problem solving. This is an important portion of the class since interaction with the students as they struggle with the problems is critical to the students' learning. The teaching assistants generally do not answer students' questions by telling them how to do a particular task. Rather, the teaching assistants ask leading questions such as "Where did you look in the help system?" or "What does your partner have to say about that?" The purpose is to facilitate the interaction among the students and encourage their metacognition about how computing systems work.

After the students complete the exercise, the lead TA then guides a discussion of the problem, calling on students to review their solutions and asking questions to help the students to reflect on the relative merits of their solutions.

Homework assignments generally extend the material learned in class and set up material for use in the subsequent class. Homework is not collected and graded in the traditional sense. Rather, the homework is used in the next day's classroom assignments and is used to complete the bridge tasks.

At the conclusion of each class, the teaching assistants complete feedback forms on the course Web site about each of the exercises and the overall class. The instructors use this feedback when they revise the instruction.

Instruction and assessment schedule

The sequences of instruction and bridge tasks are closely coupled. The course syllabus for fall semester, 1999 is shown in Table 1.

Table 1

Course Syllabus for Fall, 1999

Class Day	Topic
1	General introduction to the course; introduction to the World Wide Web; survey of prior computing experience
2	Introduction to cooperative learning; Introduction to electronic mail; Basic window skills
3	On-line Help; networked file systems; MSU's AFS file system
4	Networked filesystems; Locating, copying files; disk quotas
5	Bridge Task 1.0
6	Introduction to WWW authoring; HTML; creating Web pages.
7	More WWW authoring; Links; Images
8	Searching bibliographic databases; Boolean operators.
9	Bridge Task 1.5
10	Word Processing; entering, editing and manipulating text.
11	More Word Processing; formatting commands.
12	Word processing: Styles
13	Word processing: Large documents; document-wide formatting
14	Bridge Task 2.0
15	Spreadsheets: Introduction, dynamically changing data; MS-Excel environment.

(table continues)

Table 1 (cont'd)

Class Day	Topic		
16	Spreadsheets: Formulas, copying data, functions.		
17	Computer Hardware and Software Specifications		
18	Spreadsheets: Sorting; creating and modifying charts.		
19	Bridge Task 2.5 TRACKS DIVERGE		
20	Designing a Web Site	Introduction to data analysis; data importing.	Introduction to Financial analysis
21	Using Tables to control page layout	Logical functions; data transformations.	Modeling; financial analysis; Absolute Cell references
22	Using Applets on Web Pages	Data queries; autofilter.	Advanced spread-sheets: Modeling; financial analysis
23	Adding new pages. Completing the Web site	Pivot tables; graphing.	Logical functions, nesting functions, charting
24	Bridge Task 3.0	Bridge Task 3.0	Bridge Task 3.0
25	Object embedding and linking.	Object embedding and linking.	Object embedding and linking.
26 – 28	Final Project	Final Project	Final Project

Five of the class instructional days (in fall, 1999, days 5, 9, 14, 19, and 24 as shown in Table 1) are reserved for in-class bridge tasks. These are the first opportunities the students have to take each of the bridge tasks. The days on which these bridge tasks are offered varies each semester depending on the other instructional revisions but are generally offered biweekly. Opportunities for students to repeat bridge tasks that they have not passed are offered during the weeks in which there are no in-class bridge tasks. This means that there is a total of 12 possible bridge task opportunities during the semester.

Since students must repeat any bridge task until they do pass it, students do not necessarily take the "scheduled" in-class bridge tasks. Rather, they receive the next bridge task that they have not yet passed. Since there is a finite number of bridge task opportunities during the semester, it is important that students remain "on schedule" so that they have sufficient opportunities to pass all of the bridge tasks. Student progress is tracked in the database and email messages are sent to students warning them if they fall behind schedule.

Assessment

Due to the large enrollments and institutional resource constraints, the cost-effectiveness of the instructional design is a major consideration. The design team decided to make exclusive use of performance-based assessments rather than attempting a hybrid model that would include more traditional assessments such as multiple choice exams, graded homework assignments, and so forth. Performance-based assessments often require a larger amount of labor to create, administer, and evaluate than do traditional multiple choice or computer-

based training exams (Schoenfeld, 1994). In order to optimize the use of resources and to ensure consistency of evaluation of bridge tasks while working with TAs from a wide variety of backgrounds, the design team considered several factors. First, students complete the bridge tasks in a standard instructional computer lab, where the computers are located close together. This meant that the bridge tasks had to be individualized so different students receive different bridge tasks. Furthermore, when a student repeats a bridge task, it has to be different from the version that student received the first time. Second, the evaluation criteria for the bridge tasks had to be clear enough to ensure a high degree of consistency among raters. Third, the instructors needed to track and analyze student performance to ensure that the bridge tasks meet the design goals. The creation, maintenance, delivery and record keeping needed to address these factors required an elaborate software infrastructure (Urban-Lurain & Weinshank, 1999a).

Creating bridge tasks

Each bridge task appears to the students as a long “story problem” in which they are given realistic scenarios of problems to solve. They must be able to determine what are the appropriate computing skills and concepts and apply them correctly to solve the problems. Figure 5 shows the logical structure of a bridge task. Each bridge task has a series of (M) dimensions. Each dimension consists of (n) instances. An “instance” is the smallest unit of text that can define a problem, sub-problem or a task that is representative of that dimension. For example, one dimension may address using reference tools. Within that

dimension, there are several instances. One instance may require the students to use a thesaurus to look up a synonym for a particular word. Another may ask students to use the dictionary to insert a definition of a word into a document. The next dimension may address text fonts, with one instance of 10 point Times Roman, another of 12 point Helvetica, and so on.

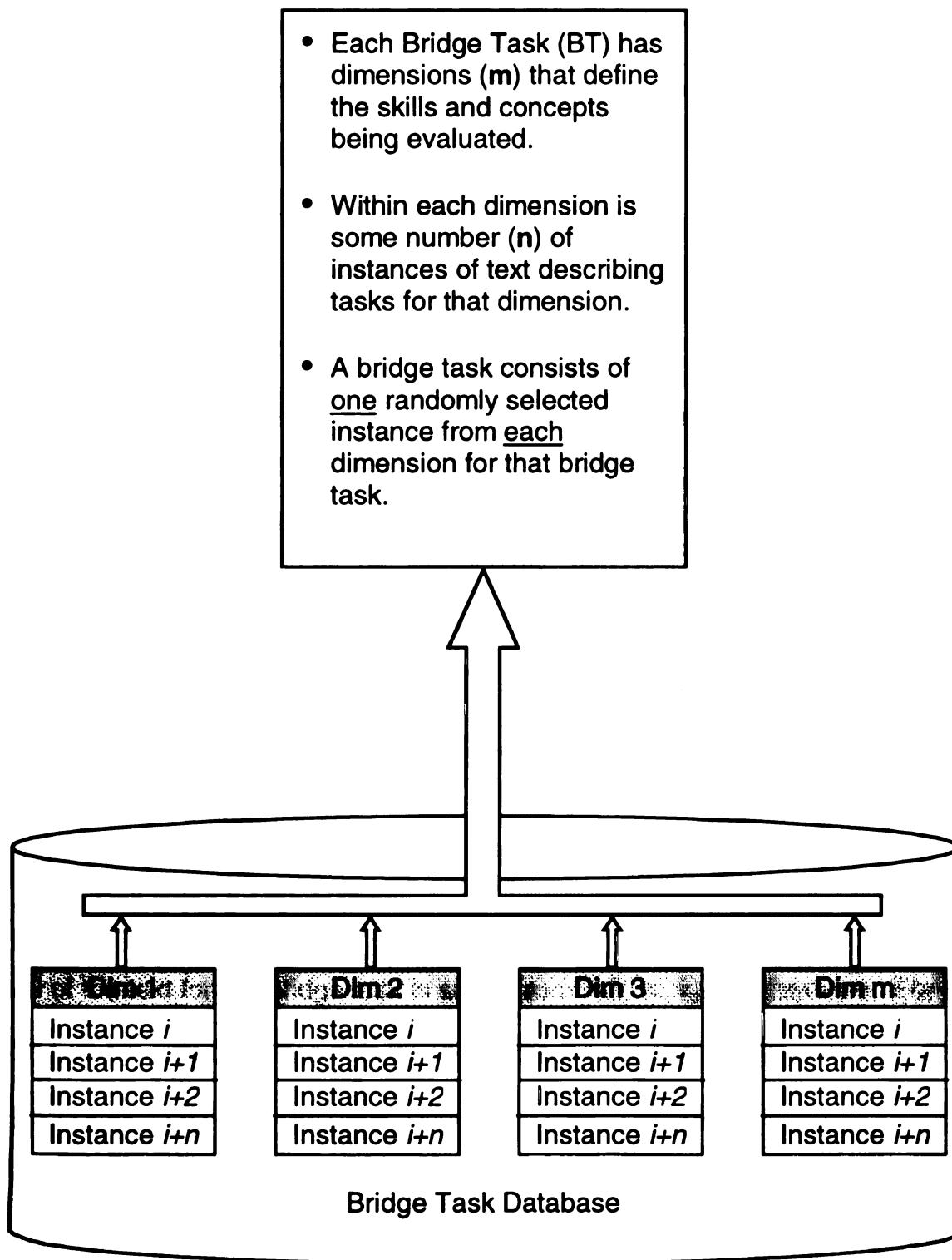


Figure 5 Logical structure of Bridge Tasks

A single bridge task is thus constructed by randomly selecting a single instance from each dimension so that each student receives a different bridge task. In addition, if a student repeats a bridge task, the instances are sampled without replacement for that student so repeated bridge tasks do not repeat any instances that a student had on previous attempts for that bridge task. The number of unique bridge tasks is given by:

$$BT_u = I_1 \times I_2 \times I_3 \dots I_n$$

where: BT_u is the number of unique bridge tasks

I_n is the number of instances in the n^{th} dimension

n is the number of dimensions

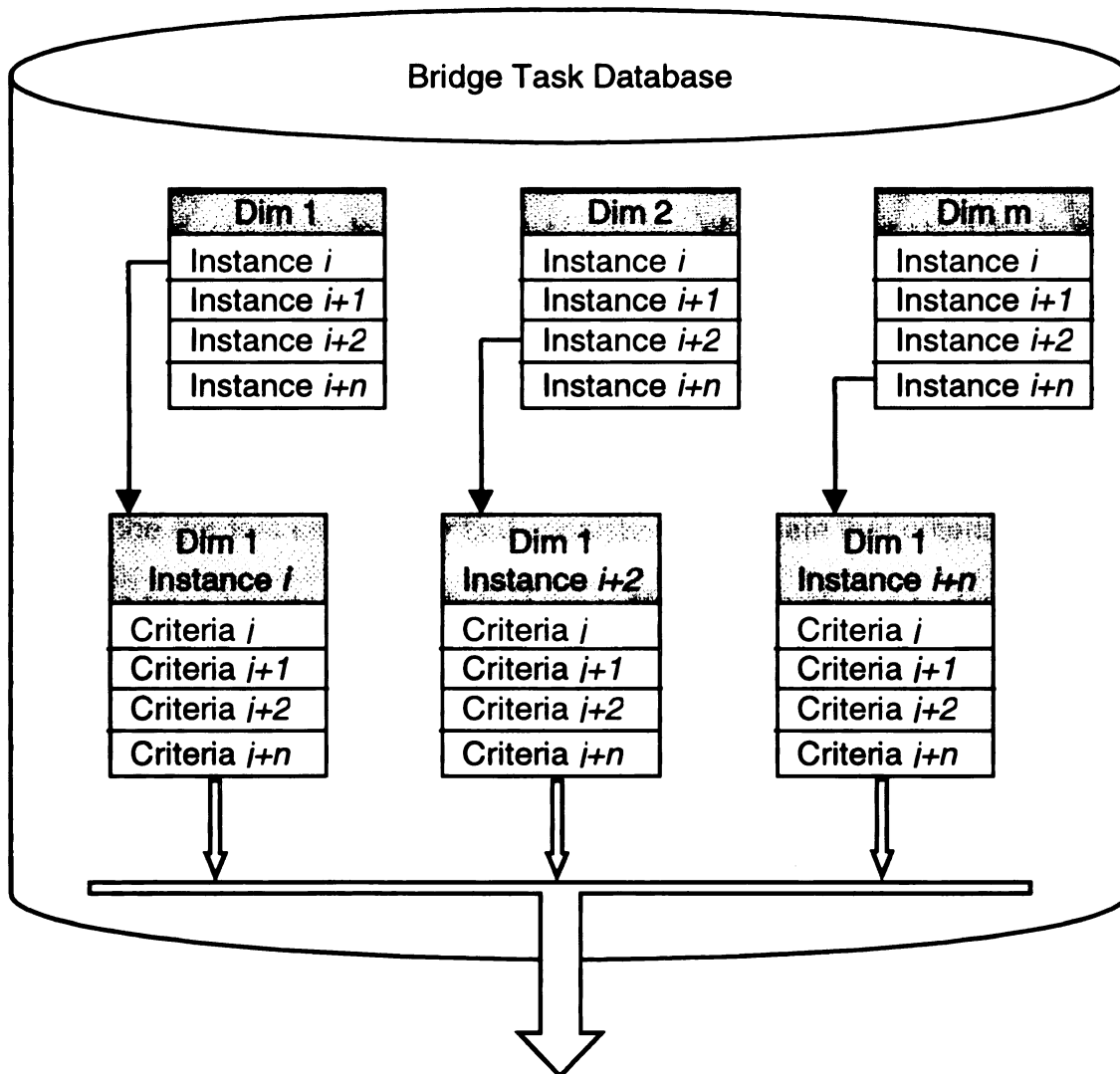
For example, the 1.0 bridge task has seven dimensions, with 17, 5, 8, 14, 4, 4 and 3 instances in each dimension respectively. This yields 456,960 unique 1.0 bridge tasks. Appendix B contains samples of the 1.0, 1.5, 2.0, 2.5, 3.0A, 3.0C, and 3.0D bridge tasks.

Delivering bridge tasks

All of the text for the bridge task is stored as HTML in a SQL database. For each bridge task, the database selects one instance from each dimension and generates a custom Web page. Bridge tasks are delivered in real time to the student's computer screen in a supervised laboratory. Security is maintained by software that requires the proctor authenticate each student after verifying the student's identity against a picture ID.

Evaluation criteria

Evaluating the performance of 1,800 students' bridge tasks requires maintaining a stringent set of evaluation criteria for each of the instances in the bridge task database to ensure inter-rater reliability. This is done with a set of database tables containing evaluation criteria in a one-to-many relationship for each of the bridge task instances. This relationship is depicted in Figure 6.



- Each dimension instance has some number (*j*) of criteria.
- Each criterion is mandatory or optional.
- Each criterion is evaluated as pass or fail.
- To pass a dimension, all mandatory and some number of optional criteria must be passed.
- Each dimension is mandatory or optional.
- To pass a bridge task, all mandatory and some number of optional dimensions must be passed.

Figure 6 Evaluation Criteria

Each instance of a bridge task dimension usually consists of more than one sentence – often a paragraph or more – of text because the dimensions must be combined randomly and still make sense in the context of the BT. While the granularity of multiple sentences of text is sufficient for creating the bridge task text, it is too coarse for the specification of the grading criteria. Therefore, each instance of every dimension in the bridge task has associated with it one or more criteria. Each criterion is designed to have the smallest possible granularity so that the graders can quickly and consistently evaluate that criterion as pass or fail. The pass/fail results for each of the criteria are combined to determine whether or not a student has passed a particular dimension. The pass/fail results for each dimension are then combined to determine whether a student has passed or failed the bridge task as a whole.

Although the goal of mastery model evaluation is for the student to demonstrate mastery, this does not require passing all criteria and all dimensions. To allow for minor student errors that do not indicate a lack of understanding, dimensions and criteria can be either mandatory or optional. To pass a dimension, students must pass all mandatory criteria and some portion (e.g., 3 out of 5) of the optional criteria for the dimension. Likewise, dimensions can themselves be mandatory or optional. Students must pass all mandatory dimensions and some portion of the optional dimensions to pass the bridge task.

For example, suppose a bridge task is composed of the three dimensions shown in Figure 6. Instance i of dimension 1 has the criteria shown in the figure. Assume that criteria j and $j+1$ are mandatory and criteria $j+2$ and $j+n$ are optional,

with a requirement for passing one of the two optional criteria. To pass dimension 1, the student must pass criteria j and $j+1$ and pass at least one of criteria $j+2$ and $j+n$. This hierarchical framework provides a flexible structure for creating, maintaining and updating assessments as the course content evolves. Appendix C lists the evaluation criteria for the sample bridge tasks in Appendix B.

Evaluating bridge tasks

Bridge tasks are graded by running an application that retrieves from the database the appropriate set of criteria for a student's particular bridge task. The application then presents each of the criteria, and the grader evaluates each criterion in a binary (pass/fail) fashion. The database computes the overall pass or fail for the bridge task based on the individual pass/fail scores on each of the mandatory and optional criteria. The grader may also enter comments for each of the criteria. These comments become part of the student's formative feedback for the bridge task.

The bridge task database has both operational and evaluative value. Operationally, the pass/fail rates on each criterion and each instance of each dimension can be analyzed to determine which items have results that deviate from the target performance goals. If such items are found, they can be revised or the instruction associated with the concepts these items test can be revised. Since each of the criteria on the bridge task is scored in a dichotomous manner, the distribution of responses follows a logistic function (Baker, 1985). By analyzing the repeat rates for the bridge tasks the instructors can evaluate the

over

me

con

dir

kn

en

ap

pa

m

D

s

overall difficulty and discrimination of the bridge tasks to determine if they are meeting the instructional and performance goals.

Final project

The bridge tasks provide a well-defined framework for evaluating students' competencies with the various software packages. The "extension task" dimensions of the bridge tasks test the students' ability to transfer their knowledge to new categories of problems. However, the same framework that ensures uniformity in evaluating the bridge tasks restricts the range of creative approaches to completing the bridge tasks. For that reason, once a student has passed the 3.0 bridge task, she or he may complete a final project that has a more open set of specifications and evaluation criteria than do the bridge tasks. Depending on the quality of the final project, it may have no impact on the student's final grade (if the project is of poor quality) or it may raise the student's grade to a 3.5 or 4.0. It cannot reduce the final grade.

The emphasis on the final project shifts from having students demonstrate their mastery of skills and concepts to having them demonstrate how well they can integrate what they have learned to solve a larger problem. The purpose is to evaluate how well students apply the computing skills and concepts to doing research, data gathering and analysis and preparing research reports and presentations in their own disciplines. Students must first define a topic or problem. There are no constraints on the topic, but it should be one that allows students to use a range of computer applications. The students are encouraged to use a term paper or project from another class to motivate their interest. Once

they

resc

with

and

par

typ

cou

sol

tra

be

br

ev

th

p

fr

s

c

s

t

they have defined the topic, students perform research using electronic resources available in the library and on the Web to gather data that they analyze with a spreadsheet. They then prepare a report that includes images and charts and cites the references they located in the library and on the Web. In the final part of the project, students demonstrate transfer by learning how to use a new type of application software. It must be a class of software that is not used in the course but which allows students to do something that cannot be done using the software they learned to use in the course. For example, in the data analysis track, they may decide to learn to use SPSS to analyze data in ways that cannot be done using a spreadsheet.

The final projects are evaluated in a more open-ended manner than the bridge tasks. There are two primary dimensions on which the projects are evaluated. First, how well students demonstrate mastery and understanding of the appropriate use of computing technology. For example, does the word processing document demonstrate the correct use of abstractions such as styles for controlling formatting and embedding objects from other applications? The second dimension is how well students can explain what, why and how they used different applications to do various parts of the project. Doing so requires that students reflect on their understanding of the material and be able to articulate this understanding in an authentic context.

the

cou

the

po

are

ra

hc

pr

e

p

d

Data And Hypotheses

Evaluation Framework

Figure 7 shows the *program implementation* from Figure 4. The boxes in the center labeled *instructional system* represent the course. However, the course is part of a larger system. *Incoming students* bring many attributes to their experience of the course. The course design is intended to be as neutral as possible with respect to individual incoming student differences. However, there are different *outcomes* in terms of performance on bridge tasks and student ratings of their experiences in the course. The intent of this study is to determine how the instructional system interacts with incoming student differences to produce different student outcomes. This model provides a framework for evaluating the results of the program implementation– and by inference the program design – of the course. The data and research hypotheses are discussed in the context this framework.

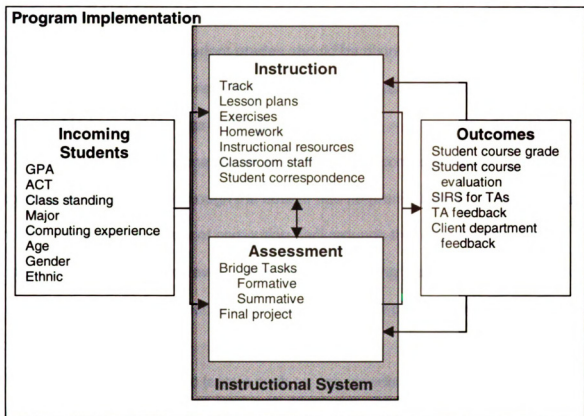


Figure 7 Evaluation of the implementation

Incoming Students

Each student comes to the course with a wide range of abilities and experiences that may have an impact on what and how the student learns in the course. While there are likely a number of latent variables that are not accessible to us, there are some behavioral variables that may be predictors of outcomes. These include general measures of academic ability, as shown by grade point average (GPA), ACT scores, class standing and major. To the degree that students differ in academic ability, we may see some difference in their outcomes from the course. Better students may master the material more

quickly than less able students but, in a course using mastery, criterion-referenced assessment, student grades should be distributed in a linear fashion, rather than the “bell curve” resulting from using norm-referenced assessments (Bloom, Madaus, & Hastings, 1981).

Another factor that may have an impact on outcomes is incoming computing experience. Although the course has no prerequisites, students come to the course with a variety of computing backgrounds that may have an impact on student outcomes. However, if the course is meeting the design goals, incoming computing experience should not be a major predictor of student outcomes.

Finally, demographic factors such as age, gender and ethnic background that have no direct relationship to the course design and implementation may confound students’ performance in the course. It is important to understand if – and how – these factors impact student outcomes.

Instructional Variables

Ideally, students’ outcomes should be a function of their experiences in the course. One design goal is for the instructional system to be a “black box” which is uniform with respect to student outcomes. However, there are two major categories of variables in the instructional model that may have impact on student outcomes. The first is student participation in the course. Students make choices about attending class, remaining engaged with the class exercises, completing the homework preparing for and completing the bridge tasks. While many of these are the result of latent variables such as motivation,

some *behaviors* may be measured. Class attendance and repeat rates on the bridge tasks are recorded, although they do not directly contribute to student grades (that is, no “points” are awarded for these variables.)

A second set of variables that may account for variance in the instruction is related to the classroom staff. Although each day’s instruction is planned in detail with the goal of consistent instruction across sections and TAs, each TA brings different abilities and experiences to the classroom. These may interact with individual student variables to affect outcomes. The number of semesters of teaching experience and student evaluations of the TAs’ classroom abilities may contribute to different student outcomes.

Electronic mail is the primary mode of communication between the course instructors and individual students. Students are encouraged to write to their teaching assistants or the course instructors with questions, problems or concerns. All of the email is stored in the course database so that when the instructors meet with a student during office hours they can access a student’s correspondence records along with bridge task performance data. The instructors also record the results of office visits and phone calls in the correspondence database. This helps the instructors quickly understand the context of the problem when working with students. This data also provides a rich set of qualitative data about student perceptions and concerns. The instructors can query this database for keywords or phrases to understand areas that cause problems or confusion for student and use this information to revise the instruction.

Assessments

Student bridge task performance has several metrics. These include a) the highest bridge task passed by each student; b) the number of times each student repeats each bridge task; c) the overall pass rates across the course for each bridge task; d) the distribution of pass rates by dimension within each bridge task; and e) the distribution of pass rates by instance within each dimension of each bridge task. The instructors evaluate the aggregate student bridge task data at the end of each semester to determine what instructional or assessment revisions are needed for the following semester. For example, if particular instances within dimensions have pass rates that are significantly higher or lower than the other instances in that dimension, the instructors analyze those instances to determine what factors contribute to the variance. Sometimes the particular wording of an instance may be confusing and need revision. Sometimes an instance may not assess the students' concepts in the way that the instructors intended. The instructors then revise those bridge task instances and/or the associated instruction for the next semester.

The individual student performance on each dimension of each bridge task provides measures of the difficulty of the skills and concepts that are being evaluated by that dimension. These data can be analyzed to determine how the skills and concepts are clustering and provide insight into how students organized their schemas and concepts.

Outcomes

There are several sets of data available from the outcomes in Figure 7. The instructional design incorporates these data for the continuous quality improvement (CQI) of the instructional system. Each semester, the instructors evaluate and revise the instruction and assessment in light of these data.

The students' final grades in the course are the primary outcome measures. The instructors analyze these data, along with the fine-grained BT performance data and TA feedback to determine what revisions may be needed in the instruction or the BTs.

Students complete a survey about the course and Student Instructional Rating System (SIRS) questions about their teaching assistants at the end of each semester. These data consist of both Likert scale responses to questions and open-ended comments. Appendix D lists the survey and SIRS questions. The instructors use the SIRS responses to improve both the course content and lesson plans. The instructors use the SIRS items about the teaching assistants as part of the evaluation of the teaching assistants' performance to help them improve their classroom skills.

Each lesson plan incorporates on-line evaluation instruments that the teaching assistants complete for each class. For each classroom exercise, the teaching assistants are asked to complete the survey shown in Figure 8. The course Web site also has feedback forms for the teaching assistants to complete on each day's overall class experience.

1) Time allotted for the exercise:

- Not Enough
- Just Right
- Too Much

2) About what percentage of the students seemed to understand the topic

after the exercise?

- 0%
- 25%
- 50%
- 75%
- 100%

3) Did students stay "on task" during the exercise?

- Not At All
- Somewhat
- Very Much

4) What features or parts of the exercise worked best?

5) What features or parts of the exercise worked worst?

6) Suggestions for improving the exercise?

Figure 8 Assistant TA exercise feedback questions

The instructors use this feedback as part of the CQI process to improve the instruction by revising individual exercises or the entire instruction for a day.

The final component in the CQI process is a framework for evaluating the external validity of student FITness after completing the course. There is an oversight committee made up of representatives from undergraduate deans of the client colleges for the course. This committee meets semi-annually to review the current content of the course and provide feedback on how well their students who have completed the course are prepared to use computers in their subsequent courses.

Analysis Framework

The instructional system can be evaluated using multivariate techniques. The general model for the observed data may be written:

$$\mathbf{Y} = \mathbf{XB} + \mathbf{E}$$

where: \mathbf{Y} is the $n \times p$ matrix of observations on the p dependent variables for the n students

\mathbf{X} is the $n \times q$ matrix of the q independent variables for each of the n students

\mathbf{B} is the $q \times p$ matrix of unknown parameters for which the equation will be solved

\mathbf{E} is the $n \times p$ matrix of random variables that accounts for measurement errors and unknown latent variables

(Krzanowski, 1988, p. 455).

The variables defined in the evaluation framework may be partitioned across these matrices. Matrix **Y** partitions into the vectors of variables that measure bridge task performance (e.g., highest bridge task passed, number of times a student repeated each bridge task) and vectors of variables that measure student ratings of the course. The Matrix **X** partitions into vectors of incoming student variables and each student's classroom variables. The vector of classroom variables consists of daily attendance records, the self-reported variables (e.g., participation in group exercises, how frequently they did the homework and the reading) and variables that measure the ability of the teaching assistant (e.g., amount of classroom experience, average SIRS ratings for the teaching assistant across sections and semesters.) The matrix **E** is partitioned into vectors of latent incoming student variables (unmeasured factors that impact the student's ability), vectors of latent instructional variables (unmeasured classroom factors), vectors of latent outcome variables (the extent to which the outcome measures do not capture what students learned in class) and a vector of measurement errors.

Although bridge tasks are sequential, the vectors of student bridge task variables are categorical; students receive a grade based on the highest bridge task passed (1.0, 1.5, 2.0, 2.5, 3.0). Therefore we may evaluate this model using classification techniques that predict categorical dependent variables based on the values of the independent variables. The goal of such a procedure is to determine the parameters (**B**) that best predict values on the categorical

depend

classif

into g

variab

mem

indivi

(Coo

sets

para

set

dete

non

non

dic

(B

rec

a r

lin

dependent variables (**Y**) given the values of the dependent variables (**X**). The classification rules are based on the general form:

$$Pr(H_j | \mathbf{B}, \mathbf{X}_i)$$

where the probability that the hypothesis of an individual being classified into group j (H_j) is conditional on the scores of that individual i on the dependent variables (**X**) and the estimated parameters (**B**). The probabilities of group membership in each group (j) are computed for each individual (i) and the individual is classified into the group with the highest computed probability (Cooley & Lohnes, 1971, p. 264).

To estimate and test the model, the data are randomly divided into two sets for analysis. The first data set is used to estimate the model. The estimated parameters (**B**) are then used to evaluate the model by classifying the remaining set of data. The accuracy of the resulting classifications are then evaluated to determine how well the model predicts group membership.

Since student outcomes are based on criterion-referenced, rather than norm-referenced assessments, it is not appropriate to assume a multivariate normal distribution of the underlying variables. Bridge tasks are evaluated on a dichotomous basis, so the underlying distribution follows a logistic function (Baker, 1985). Therefore, we need to use multivariate techniques that do not require the dependent variable to be normally distributed. Discriminant analysis is a multivariate technique similar to multiple regression that identifies an optimal linear combination of independent variables that best discriminates among two or

more gr

into the

t

discrim

interpre

deriva

studen

will be

and a

score

tha

more groups. The resulting discriminant functions can be used to classify cases into the groups defined by the dependent variable.

Discriminant Analysis

Discriminant analysis is usually applied in three stages: 1) derivation of the discriminant functions, 2) validation of the discriminant functions and 3) interpretation of the resulting functions (Hair, Anderson, & Tatham, 1987). In the derivation stage, the first step is identifying the categorical variable. In this study, student final grades will be the categorical variable. The independent variables will be obtained from data collected for each part of the evaluation framework and are described in the next section. The objective is to derive a discriminant score for each individual (Z_i) from the independent variables such that:

$$Z_i = b_0 + b_1x_{1i} + b_2x_{2i} + \dots + b_nx_{ni}$$

where: x_{ji} is the value on the j^{th} independent variable for individual i

b_j is the discriminant coefficient of the j^{th} variable

Z_i is the i^{th} individual's discriminant score

Classification boundaries (Z_{crit}) are defined in n-dimensional space such that if:

$Z_i > Z_{crit}$ individual i is classified in group 1

$Z_i < Z_{crit}$ individual i is classified in group 2

This process is shown graphically in Figure 9 (after Hair et al., 1987, p 80).

X₂

Fig

an

inc

so

th

"lo

di

bo

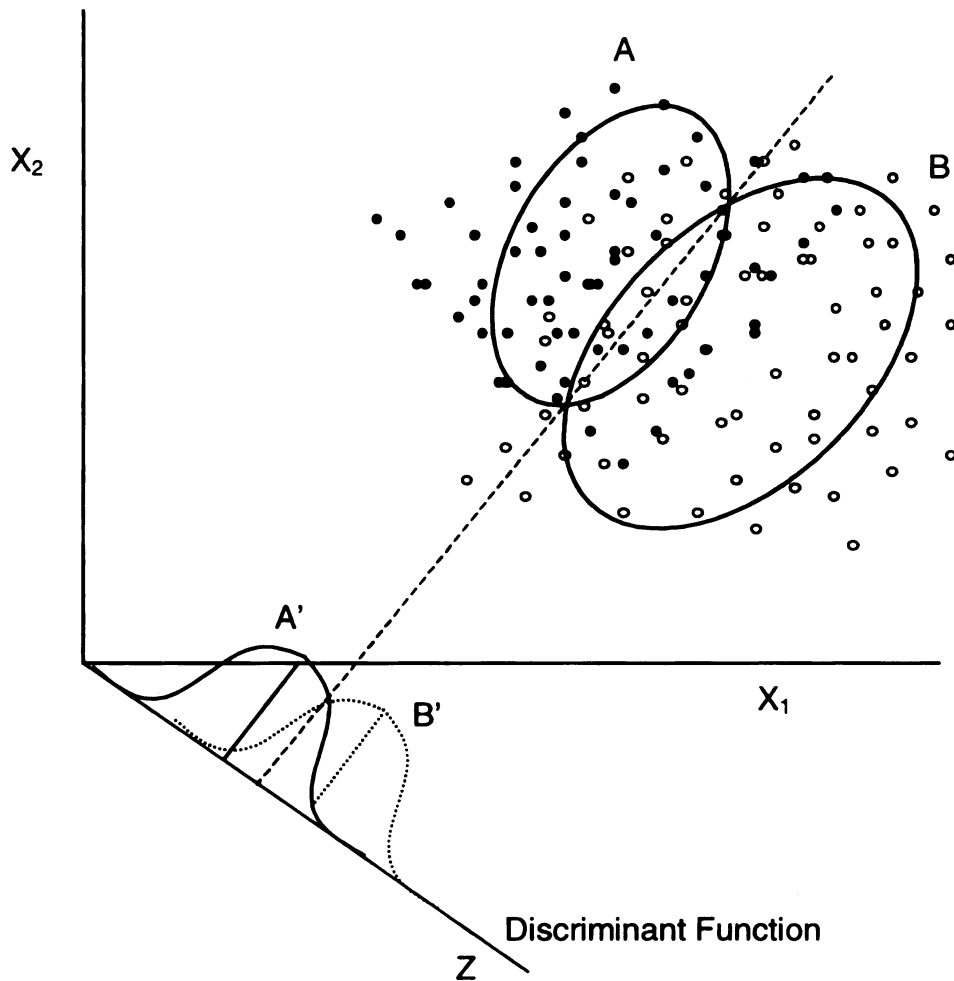


Figure 9 Two group discriminant analysis

The figure represents the scatter diagram of two groups, A (the solid dots) and B (the open dots) on two sets of variables, X_1 and X_2 . The dots are the individual measures for each case on the two variables. The ellipses represent some defined confidence interval for the distribution from each group. The line that is defined by the two points where the ellipses intersect is projected to a “lower” dimension on the Z-axis which represents the discriminant function that discriminate between the two groups. This line defines the classification boundary between the two groups. The means of the distributions of A' and B'

are t

the s

base

that

belon

class

grou

A ca

indiv

discr

being

accu

the g

dispe

dista

(Van

are the group centroids for each group on the discriminant function. Projecting the scores for each case onto the Z-axis allows us to assign group membership based upon the relative probabilities represented by A' and B'. Individual cases that are on the left side of the classification boundary line are classified as belonging to group A. All of the open dots from group B on the left side of this classification boundary are group B cases that are mis-classified as belonging to group A; the solid dots on the right side of the classification boundary are group A cases that are mis-classified as belonging to group B.

For more than two groups, the number of functions is increased so that individuals are classified into the group based on the combination of their discriminant scores on multiple functions, with the maximum number of functions being one less than the number of groups.

There are a number of ways to attempt to maximize the classification accuracy among groups. Most are based on maximizing the distances among the group centroids in n-dimensional space and/or reducing the multivariate dispersion of the resulting classification scores. One way of maximizing the distance is to maximize the Mahalanobis Distance (D^2) between group means (Van De Geer, 1971, p. 256):

$$D_{ij}^2 = (x_i - x_j)' V^{-1} (x_i - x_j)$$

where: x_i and x_j are vectors of coordinates in n-dimensional space

V is the variance-covariance matrix

an

de

do

va

inc

by

va

of

sta

fun

the

to a

the

The axes of the group ellipsoids are rotated to minimize correlations among variables and maximize the distance among group centroids. D^2 then defines the distance – and discriminatory power – among the groups.

Like multiple regression, variable selection in discriminant analysis may be done in a stepwise manner. There are several criteria that may be used to select variables, but most attempt to find combination of variables that maximize D^2 by including or deleting variables. These stepwise procedures are usually evaluated by comparing D^2 based on the current subset of variables to D^2 for the full set of variables. If D_k^2 is based on the k subset of variables and D_p^2 is based on the set of p variables, then, for sample sizes n_1 for group 1 and n_2 for group 2, the statistic:

$$\frac{((m - p + 1)/(p - k))c^2(D_p^2 - D_k^2)}{(m + c^2 D_k^2)}$$

where: $c^2 = \frac{n_1 n_2}{n}$

$$m = n_1 + n_2 - 2$$

is distributed $F_{p-k, m-p+1}$

When F is no longer significant, no additional variables are included in the function (Mardia, Kent, & Bibby, 1979, p. 322).

The second stage of using discriminant analysis, validation, involves using the computed classification scores for each individual to “assign” each individual to a predicted group, based on the group for which that individual's scores assign the highest probability. The number of individuals for whom the predicted group

matches the actual group determines the accuracy of the prediction. Accuracy may be evaluated using standard chi-square expected frequency measures to compare the accuracy of the model with chance expectations. Validation usually is done by splitting the original sample into two sub-samples. The first sample is used to derive the function. This sample is then tested for classification accuracy. Since this sample's data was used to derive the function, cross-validation with the second sub-sample is a check against upward bias of the prediction accuracy that may result from testing the function on the data from which it was derived. A certain "shrinkage" is expected from the first sub-sample to the cross-validation sample. Lower shrinkage indicates more stable classification boundaries.

If the first two stages of discriminant analysis are successful, the third stage is interpretation of the resulting functions (Hair et al., 1987). This stage is similar to interpreting factors in factor analysis. First, we examine the discriminant functions to determine the contribution of the independent variables to each function. After that, we attempt to characterize the differences among groups based on their multivariate means or group centroids. While factor analysis attempts to identify commonality among variables (factors) that underlie the observed variables, interpreting discriminant functions involves attempts to identify common factors that are associated with individuals who are classified in the same group (dependent variable.) This procedure should allow us to determine factors that distinguish among successful and unsuccessful students

i
t
c
t
s
c
t
i
i
t
f
n
s
f
c
L
s

in the course. It should also allow us to gain a better structural understanding of the issues that contribute to FITness.

A series of discriminant analyses was used to understand the contributions of each part of the evaluation framework. The first examined only the incoming student variables to identify the key incoming student factors. The second analysis added the instructional system variables, to see what contributions and interactions they have with the incoming student variables. For the third analysis, the data from the bridge tasks was added to the model to identify the underlying computing concepts and competencies – as represented in the bridge tasks – that are associated with success or failure in the course.

The outcomes portion of the model was analyzed by using factor analysis to reduce the dimensionality of the student course evaluations and SIRS. These factors were used as variables in regression analyses to understand the relationship among the factors and the variables from the other three parts of the system.

Data Sources

The analysis used data from the course for fall, 1998, spring, 1999 and fall, 1999. All individual identifiers were removed prior to analyses. The data consists of a number of quantitative and qualitative measures.

Incoming Student Data

Demographic data for each student from the semester in which the student took the course includes class standing, grade point average (GPA),

ACT scores, major, age, gender and ethnic classification. Data on incoming computing experience are collected as part of a Web-based survey that students complete on the first class day in the course. Students are given time to complete the survey during class and the survey remains available for the first week of the course. Participation is voluntary; students do not receive any grade or other credit for completing the survey. The course instructors use this data to track students' self-reported computing experience and adjust the pace of the first part of the course.

The survey is intended to identify the range of computing experience that students report having prior to entering the course. It does not attempt to measure the students' competence with any computer applications. One set of questions asks about general experience and exposure to computers. A second set of questions asks about the amount of experience with particular computer applications. The last set of questions is about the students' college algebra and computer programming background.

Instructional System Data

The data from the instructional system includes student and TA data. Student data includes attendance records, BT attempt rates, responses on questions about how often they completed their homework and readings before class, their participation in the group exercises and their use of instructional resources such as the course Web site, the textbook and the help room. The correspondence data in the course database for each student are also part of the instructional system data. The classroom staff data for each student includes the

teaching assistant's teaching experience with the course, along with the teaching assistant's average SIRS ratings.

Assessment Data

The data on student bridge task performance includes the highest bridge task passed, the number of times the student attempted each bridge task, and the particular dimension and instances attempted and passed by each student.

Outcomes Data

The final grade in the course is the dependent variable used for the analyses. The student evaluation of the course include several items asking students how well they feel the bridge tasks measure what they know, their perceptions of the grading of the bridge tasks, how well they feel their final grade will reflect what they have learned, if they would recommend the course to their friends and so forth. (See Appendix D for the specific wording of the items.)

The SIRS data for the lead TA and the assistant TA are also part of the outcomes data. For each of the SIRS questions, students are encouraged to include comments that elaborate on their quantitative responses. These responses are stored in the course database. These comments were examined to provide qualitative insights about student perceptions of the course.

Hypotheses

This analysis framework can be used test a number of hypotheses. There are four sets of hypotheses. The first address the effect of individual incoming student differences. The second address the effect of the instructional system.

The third address the validity of the assessments. The last set addresses the outcomes.

Individual Incoming Student Differences Hypotheses

One of the instructional goals was to create a course that would prepare students across the entire university to be FIT, regardless of their incoming experience or academic ability. These hypotheses are intended to measure the contribution of various incoming student attributes to the outcomes.

H 1: Outcomes are not linearly related to incoming ability. Because of the mastery-model assessment, students with lower GPAs and ACT scores may take more attempts to pass the bridge tasks, but should do better in this course than in their other courses.

H 2: Outcomes are not dependent on class standing. Because there is no prerequisite for the course, upper division students should not perform better than lower division students.

H 3: Outcomes are not dependent on major. Because the course has multiple tracks with focal problems appropriate for different majors, students from different majors should not have different outcomes.

H 4: Outcomes are not dependent on prior computing experience. Because the course presumes no prior computing experience, any effect of prior experience may only be evident at the onset of the course but should disappear by the end of the course.

H 5: Outcomes are not related to age. Younger students may have had more exposure to information technology and be more comfortable with it.

However, because the course presumes no prior computing experience, any effect of age should disappear by the end of the course.

H 6: Outcomes are not dependent on gender. When other incoming student variables are controlled for, gender should not interact with the instructional system.

H 7: Outcomes are not dependent on ethnic background. When other incoming student variables are controlled for, ethnic background should not interact with the instructional system.

Instructional System Hypotheses

These hypotheses are intended to evaluate the contribution of various instructional system features to the outcomes. If the course is meeting the design objectives, student activity in the instructional system should account for more of the variance in the outcomes than incoming student variables.

H 8: Student participation in class will be a predictor of bridge task performance. Because the course design is predicated on students collaboratively solving problems to build schemas and concepts, student preparation for and participation in class should be the primary predictor of outcomes.

H 9: Outcomes are a function of individual teaching assistants. Because each section is met by its own teaching assistants, there is may be differences across sections based on teaching assistant abilities and experience.

Assessment Hypotheses

These hypotheses are intended to evaluate the internal and external validity of the bridge tasks to determine if they actually measure student understanding of the computing concepts that are indicative of FITness.

H 10: The fitted model correctly predicts students' bridge task performance. The classification accuracy shows how well the model fits the data.

H 11: Students who repeat bridge tasks more frequently pass the extension tasks less frequently. Because the extension tasks test transfer, students who understand the concepts are more likely to pass the extension tasks than are students who are learning skills alone rather than concepts. Students who do not pass the extension tasks are more likely to have to repeat the bridge tasks.

Outcome Hypotheses

These hypotheses are intended to evaluate student attitudes and perceptions of the course.

H 12: Student perceptions of how well the bridge tasks assess their learning are a function of their performance on the bridge tasks. Students who do not do as well may be less inclined to believe that the bridge tasks are a fair reflection of their knowledge.

H 13: Student perceptions of the course are a function of teaching assistant abilities and experience. Because the teaching assistants are responsible for the daily instruction, student perceptions of the course and their learning

experiences are likely to be a function of the teaching assistant abilities and experience.

The next chapter reports on the results of the analyses and the evaluation of the hypotheses.

comp

The fi

comp

variab

comp

on the

system

from

Next,

prese

comp

to de

The

inter

regr

and

pres

in th

CHAPTER 4 RESULTS

This chapter presents the analyses conducted to evaluate the four components of the instructional system presented in Chapter 3 (see Figure 7.) The first component is based upon the incoming student variables. The second component is based upon the instructional system and adds student classroom variables and instructional staff variables to the first component. The third component of the system is the assessment subsystem. This adds performance on the various dimensions of each bridge task to the model. The final part of the system is outcomes and consists of the final grades in the course and feedback from the students on the end-of-the-semester surveys.

The analysis starts with the distribution of final grades of the students. Next, the analyses of the first three components of the instructional system are presented in parallel fashion. First the independent variables used for that component of the instructional system are described. Those variables are used to derive discriminant functions that predict final student grades in the course. The classification accuracy of the functions is then used to test their validity. The interpretations of the resulting functions are then presented.

The analysis of the fourth component, student evaluations, uses regression to understand the relationships among the three system components and student perceptions of the course and their teaching assistants. After presenting the results for each part of the model, the hypotheses are discussed in the context of the analytic framework.

Student outcomes

Data from all students who took the course in fall, 1998, spring, 1999 and fall, 1999 semesters were used for the analyses ($n = 5068$). The final course grade was the dependent, categorical variable used for evaluating the various components of the instructional system with discriminant analysis (see Chapter 3.) The university grades on a scale of 0.0, 1.0, 1.5, 2.0, 2.5, 3.0, 3.5 or 4.0 for each course. The final course grades for all students who took the course are shown in Figure 10.

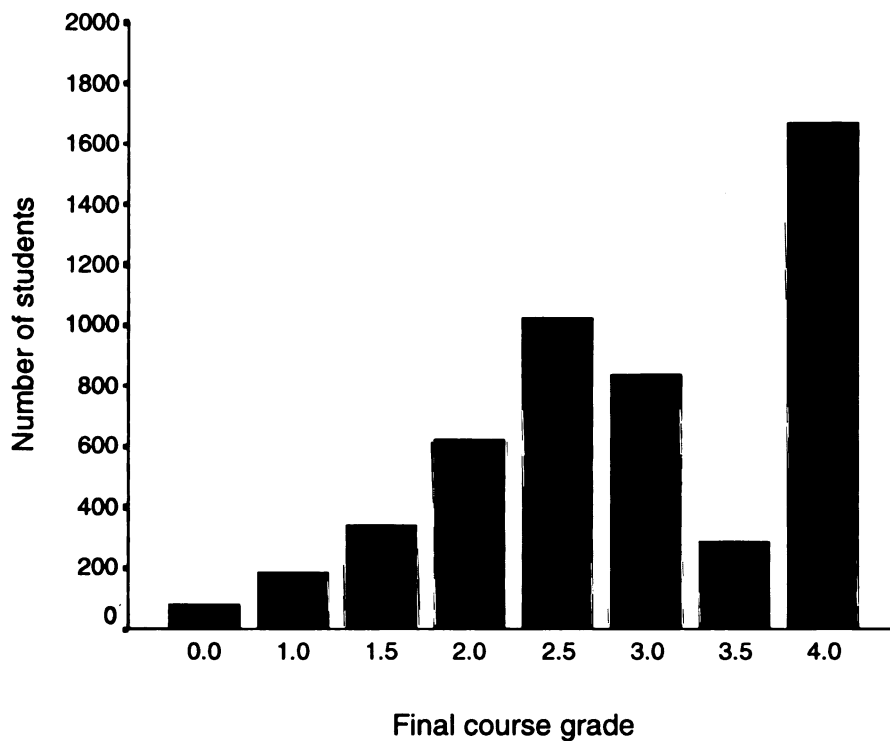


Figure 10 Distribution of final course grades

Recall that the highest bridge task passed determines course grades up through 3.0. After that, students may raise their grade to 3.5 or 4.0 by

completing a final semester project. Thus, all students receiving grades of 3.0 or higher passed the 3.0 bridge task by the end of the course. Because the bridge tasks and semester project are criterion referenced, we should not expect a normal distribution but rather a more linear one. (Bloom et al., 1981) Figure 10 shows that the grade distribution is more linear than normal ($M = 2.91$, $SD = 0.98$, $Skewness = -0.55$). However, there is a “dip” at the 3.5 grade since most students who complete the final project receive full credit for it and a 4.0 in the course. Since there are only eight discrete possible final grades, the course grade was used as the categorical variable for the discriminant analyses in the evaluation of each part of the instructional system.

Incoming Student Variables

Demographic Variables

Demographic data was obtained for each student for the semester in which the student took the course. It included the cumulative GPA for that semester, the term GPA for that semester, ACT scores, major, class standing, birth date, gender and ethnic classification. The number of students by semester is shown in Table 2.

Table 2

Number Of Students by Semester

Semester	Frequency	Percent
Fall 98	1713	33.8
Spring 99	1669	32.9
Fall 99	1686	33.3
Total	5068	100.0

Grade Point Averages (GPA)

Both the semester GPA ($M = 2.80$, $SD = 0.77$) and the cumulative GPA ($M = 2.80$, $SD = 0.61$) include the grades students received in the course. The influence of the course grade on each of these variables is a function of the number of credits. Therefore, the course grade has much greater impact on the cumulative GPA for first year students than for fourth year students as shown in Table 3.

Table 3

Correlations Between Course Grade, Semester GPA and Cumulative GPA by Class Level

Class level		Semester GPA	Cumulative GPA
1	Course grade	.745	.705
	Semester GPA	-	.941
2	Course grade	.744	.654
	Semester GPA	-	.853
3	Course grade	.701	.561
	Semester GPA	-	.799
4	Course grade	.716	.553
	Semester GPA	-	.693

Note: All correlations are significant at $p < 0.001$ (2-tailed)

The correlation between the course grade and the semester GPA is consistent across class levels, since the variation in the number of credits students can take during a particular semester is small. The smallest number of credits would be if a student enrolled only for the course, in which case the course grade would be the semester GPA. Few students enroll for more than 15 credits during any semester. The correlation between course grade and cumulative GPA decreases as the class level and number of credits earned increases. Thus, the course accounts for 20% of the cumulative GPA for first semester students taking a 15-credit course load. For graduating seniors, the

course accounts for only 2.5% of the cumulative GPA. Although the course grade is included in the cumulative GPA, without the total number of credits taken by the students, the course grade cannot be factored out of the cumulative GPA.

ACT Scores

Table 4 shows the minimums, maximums, mean and standard deviations for each of the ACT sub-scores and the ACT composite score for all students for whom the scores were available ($n = 4427$).

Table 4

ACT Scores

ACT sub-score	Minimum	Maximum	<i>M</i>	<i>SD</i>
ACT English	9	36	22.07	4.22
ACT Social Science	8	36	22.93	5.03
ACT Natural Science	10	36	22.73	3.84
ACT Mathematics	1	36	22.32	3.96
ACT Composite	12	34	22.65	3.55

These ACT scores are just above the national averages for college-bound students. From 1994 through 1999, the average ACT composite scores for students who took the core college preparatory curriculum in high school fluctuated between 22.0 and 22.1. The average ACT composite scores for all

students who took the ACT tests from 1994 through 1999 ranged from a low of 20.8 in 1994 through 21.0 in 1999 (ACT, 2000).

Students' Majors

There were 141 different majors among the students who took the course ($n = 5056$). Different majors require, recommend, prohibit, or accept the course as an elective. Some majors may require or recommend a specific track of the course. The number of students from each major varies widely but 14 majors account for 65.5% of the total enrollment. Of the remaining 127 majors no one constitutes as much as 2% of the total enrollment. Table 5 lists the number of students from each of the 14 majors with the greatest enrollment.

Table 5

Number of Students from the Majors with Highest Enrollments

Major	Track A	Track C	Track D	Total
General Business Admin	90	181	327	598
No Preference	295	89	181	565
Advertising	160	46	85	291
Marketing	35	57	142	234
Communication	136	28	49	213
Accounting	14	43	146	203
Finance	17	38	145	200
General Business Admin-Prelaw	18	56	92	166
Criminal Justice	129	22	13	164
Education	117	12	27	156
Hospitality Business	25	50	80	155
Telecommunication	99	20	25	144
Psychology	73	33	21	127
Supply Chain Management	11	32	65	108

Seven of the majors (General Business Administration, Marketing, Accounting, Finance, General Business Admin-Prelaw, Hospitality Business and Supply Chain Management) are from the College of Business and account for 50% of the students shown in the table. The course is required for these business majors, and the college recommends Track D – where business students comprise 71% of the students in this table – or Track C – where business majors are 65% of the students in the table. Both of these tracks emphasize spreadsheets. By contrast, business students are only 17% of the Track A students listed in this table. Track A students come mainly from no preference, Advertising, Communication, Criminal Justice, Education, Telecommunications and Psychology majors. Engineering students generally take an introductory programming course, so there are few engineering students in the course ($n = 52$ for seven engineering majors.) The track in which each student was enrolled was coded with a pair of binary variables: Track C and Track D. Values of zero on both indicate a student is in track A.

Class Standing

Student class standing is coded by year for undergraduate students (1 – 4), graduate students (M) and Life long education students (L). The class standings of students for each semester are shown Table 6.

Table 6

Percentage of Students Enrolled Each Semester by Class Standing

Class standing	FS98	SS99	FS99	Overall
1	29.9	36.7	38.5	35.0
2	39.6	39.4	37.2	38.7
3	20.7	13.8	16.5	17.1
4	8.9	9.6	7.4	8.7
M	0.0	0.0	0.1	0.0
L	0.3	0.1	0.3	0.3

Note: Totals are less than 100% due to missing data for 0.3% of the students.

The course is intended to prepare first year students for their subsequent courses in their majors. Because the course is fully enrolled, and upper division students receive enrollment priority, fall semesters have a larger proportion of upper division students than do spring semesters.

Student Age

Student date of birth was used to compute each student's age on the first day of the semester in which the student took the course. The students are primarily "traditional" undergraduate students ($M = 20.2$, $SD = 2.5$). There were 94 students older than two standard deviations above the mean (> 25.2 years.) Only seven of these students were lifelong education or graduate students. The others were enrolled in undergraduate degree programs.

Gender and Ethnic Classification

The university collects self-reported gender and ethnic classification data from students. There were 2692 female students (52.9%), 2388 male students (46.9%) and eight students for whom no gender was reported (0.2%). The ethnic classifications are summarized in Table 7.

Table 7

Ethnic Classifications

<u>Classification</u>	<u>Percent</u>
Caucasian	81.1
Black	9.5
Chicano	0.8
Hispanic	1.8
American Indian	0.5
Asian / Pacific Islander	5.6
Other	0.6
Missing	0.2

For analysis purposes the ethnic classifications were re-coded into three binary indicator variables. Asian / Pacific Islander was coded 1 if the student reported Asian / Pacific Islander, otherwise coded 0. Black was coded 1 if the student reported Black, otherwise coded 0. Chicano / Hispanic was coded 1 if

the

"An

da

fai

fo

co

ex

the

qu

ba

Th

is

ha

co

the student reported either Chicano or Hispanic, otherwise coded 0. The “American Indian” and “Other” categories accounted for only 0.5 percent of the data; they are not explicitly captured in this coding scheme.

The students in the course come from the across the university and are a fairly representative sample of non-engineering / technical students that may be found at many large, public colleges and universities.

Incoming Student Computing Experience

Chapter 3 discussed the survey of computing experience that students complete on the first day of class. One set of questions asks about general experience and exposure to computers. A second set of questions asks about the amount of experience with particular computer applications. The last set of questions is about the students’ college algebra and computer programming background. The trends across semesters are shown in the following tables. The average response rate on each of the survey questions across all semesters is 75%.

General Computing Experience

Table 8 shows the responses to the question about how long students have worked with computers. Note that few students report never using a computer.

Table 8

Length of Time Incoming Students Report Working with Computers

Semester	Never	1 - 3 months	3 - 6 months	6 - 12 months	> 1 year
FS98	4.4	8.1	6.5	7.2	73.9
SS99	4.6	4.6	5.3	5.8	79.7
FS99	4.3	5.7	5.0	5.5	79.5

Note: All values are percent of students who responded to the question.

Table 9 shows how students learned to type. The choices were intended to measure formal typing instruction – learning to “touch” type – versus self-taught typing – “hunt-and-peck” (H & P) typing. They also measured whether students learned to type on a typewriter or a computer.

Table 9

How Students Report Learning to Type

Semester	Never typed	Typewriter H & P	Typewriter touch typist	Computer H & P	Computer, touch typist
FS98	0.9	6.7	21.7	21.9	48.8
SS99	0.6	6.3	20.6	22.8	49.7
FS99	0.8	4.5	15.6	21.9	57.2
FS99	0.8	4.5	15.6	21.9	57.2

Note: All values are percent of students who responded to the question.

Almost no students report not being able to type. Furthermore, the majority of students report being able to “touch” type. What is particularly interesting is the trend for formal typing instruction. The proportion of students who reported learning to “touch” type on a typewriter is declining, while those who reporting learning to “touch” type on a computer is increasing. Keyboarding skills are now primarily associated directly with computing.

The next three questions were designed to measure students’ familiarity with some computer terminology. The first question asks the students if they know what the term “Windows” means. The assumption is that, regardless of actual computing experience, the term “Windows” has permeated the cultural lexicon in recent years. The results are shown in Table 10. Note that virtually no student claimed never to have heard of the term.

Table 10

Student Self-Reported Understanding of the Meaning of the Term “Windows”

Semester	Never heard of it	Heard but don't know	Vague idea of meaning	Clear idea of meaning but not used it	Clear idea and have used it
FS98	0.1	6.2	17.8	9.4	66.5
SS99	0.2	3.6	15.1	9.1	72.0
FS99	0.2	5.1	18.0	8.3	68.3

Note: All values are percent of students who responded to the question.

Contrast the results of the previous question with a more generic wording of the same question. Table 11 shows the responses to the question “Do you know what the term ‘User Interface’ means?” The assumption is that, unless students have had more formal instruction or experience with computers they are not as likely to know this as they are to know the term “Windows.” The vast majority of students claim to have never heard of the term or not know what it means. This trend has not changed much across the semesters.

Table 11

Student Self-Reported Understanding of the Meaning of the Term “User Interface”

Semester	Never heard of it	Heard but don't know meaning	Vague idea of meaning but not used it	Clear idea of meaning but not used it	Clear idea of meaning and used it
FS98	35.2	34.0	16.7	6.3	7.7
SS99	34.2	31.3	21.7	5.0	7.9
FS99	37.1	29.4	18.4	6.2	8.9

Note: All values are percent of students who responded to the question.

The final question in this cluster asks if students know what the term “Hypertext” means, shown in Table 12. The assumption here is that the concept of hypertext underlies the World Wide Web. While the instructors expected most students to know about the Web, those who have a deeper knowledge might know about hypertext. Again, the majority of students claim either not to have

heard of the term or not know what it means, with less than one eighth of the students ever reporting that they have used the term.

Table 12

Student Self-Reported Understanding of the Meaning of the Term “Hypertext”

Semester	Never heard of it	Heard but don't know meaning	Vague idea of meaning but not used it	Clear idea of meaning but not used it	Clear idea of meaning and used it
FS98	33.5	33.4	15.2	7.0	10.8
SS99	30.5	32.6	17.6	8.0	11.3
FS99	32.4	31.6	16.5	7.1	12.4

Note: All values are percent of students who responded to the question.

As these data show, student exposure to computing increased from fall, 1998 to fall, 1999, with the vast majority of students reporting over a year of experience using computers. Most students learned keyboarding skills on a computer rather than a typewriter. However, understanding of the abstract concepts of “user interface” and “hypertext” has remained constant over that time.

Computer Application Experience

The next set of questions measure the amount of time or experience that students have using particular types of computer applications. The applications include E-mail, the Web, word processing, spreadsheets, electronic databases, and on-line services. The scales were designed to measure experience from no

experience to the level of experience gained from using the applications for a year or more in informal settings.

Table 13 shows the student experience with E-mail. The scale ranges from “not sure what E-mail is” to having participated in one or more E-mail lists. The vast majority of students report using E-mail, with over half participating in E-mail lists. E-mail is also the computer application that most correlates with student class level, $r = .197, p < .001$.

Table 13

Student Self-Reported Experience with E-mail

Semester	Not sure what it is	Never used e-mail	Used 5 or fewer times	Used more than 5 times	Participated in e-mail lists
FS98	0.8	4.5	6.5	38.1	50.1
SS99	0.7	1.6	1.6	41.8	54.2
FS99	1.1	2.8	4.3	36.4	55.4

Note: All values are percent of students who responded to the question.

After E-mail, the Web is the area in which students have the most computing experience, as shown in Table 14. Over three-quarters reported using it more than five times and over one-eighth reported having their own home page.

Table 14

Student Self-Reported Experience with the World Wide Web

Semester	Not sure what it is	Never used	Used 5 or fewer times	Used more than 5 times	Have own home page
FS98	0.5	2.1	10.3	74.3	12.8
SS99	0.1	1.6	5.1	79.2	13.9
FS99	0.7	0.8	5.5	77.7	15.3

Note: All values are percent of students who responded to the question.

The next most frequently used application is word processing, as shown in Table 15. About two thirds of the students report using word processing for creating more than one paper of at least five pages in length and almost one fifth have used desktop publishing packages.

Table 15

Student Self-Reported Experience with Word Processing

Semester	Not sure what it is	Never used for 5 page paper	Used for one 5 page paper	Used for more than one 5 page paper	Used desktop publishing package
FS98	2.6	6.5	10.1	63.4	17.4
SS99	1.4	6.4	7.7	67.9	16.6
FS99	3.5	5.6	8.4	63.6	18.9

Note: All values are percent of students who responded to the question.

After word processing, spreadsheets are the next most frequently used applications. Spreadsheets are also the focus of two of the three tracks of the course, so we might expect students to have some familiarity with spreadsheets. Table 16 shows the experience with spreadsheets. Note that a much higher proportion of students report not knowing what a spreadsheet is or never having used one than did for word processing. It is interesting to note that over one third of the students report using a spreadsheet only in a computer class. This implies that students may have taken a class that taught spreadsheets in high school but have not used them since then. Only about 22% report using spreadsheets in non-computer classes and only about 15% report using them on the job.

Table 16

Student Self-Reported Experience with Spreadsheets

Semester	Not sure what it is	Never used a spreadsheet	Used only in computer class	Used in non-computer class	Used as part of a job
FS98	8.5	19.7	39.0	18.8	13.9
SS99	6.8	18.8	36.8	24.1	13.6
FS99	6.8	17.6	36.9	22.6	16.2

Note: All values are percent of students who responded to the question.

The last application surveyed is the use of electronic databases. The university library has an electronic “card catalog,” so the instructors used a different scale of responses for this item. The first three responses parallel the other application questions: a) not sure, b) never used and c) searched five or

fewer times. The last two responses compare the use of the library card catalog with searching other electronic databases. The distribution of responses to this question is shown in Table 17. Note that the pattern of experience increases across time, with use of databases other than the library catalog increasing by about 25% from fall, 1998 to fall, 1999. This indicates that students are making greater use of these resources, possibly in other courses.

Table 17

Student Self-Reported Experience with Electronic Databases

Semester	Not sure what it is	Never used	5 or fewer times	Only library catalog	Other than library
FS98	28.3	14.1	15.4	10.0	32.3
SS99	23.0	12.0	13.4	11.6	40.1
FS99	22.8	12.2	14.3	8.4	42.3

Note: All values are percent of students who responded to the question.

Student experience with the most common computer applications increased slightly from 1998 to 1999. Virtually all of them at least knew about E-mail and the Web, with most having used them regularly for over a year. Most students had word processing experience and had written papers using a word processor. Far fewer had experience with spreadsheets and electronic databases.

Algebra and Computer Programming Experience

The last set of questions ask about the students' college algebra and computer programming background, since these items have traditionally been associated with success in introductory CS courses. The first question asks when students took a college-level algebra course; it does not ask about how well students did in the course. See Table 18.

Table 18

When Students Report Taking College-Level Algebra

Semester	Not taken	Currently enrolled	Took last term	Took term before last	Took over 6 months ago
FS98	6.4	20.4	16.0	13.3	43.9
SS99	5.6	11.1	25.9	11.9	45.5
FS99	6.9	25.1	13.0	14.4	40.7

Note: All values are percent of students who responded to the question.

Examining the distribution of responses, it appears that the students tend to take algebra in the fall semester. The “currently enrolled” response is highest each fall, with the “took the course last semester” peaking during spring semesters. A linear regression of class level and semester with the responses to the algebra question shows that this question is actually a function of class standing ($\beta = .486$, 23.6% of the variance) rather than the semester in which the students enroll in the course ($\beta = .079$, 0.6% of the variance.) Thus, it appears that students tend to take algebra courses early in their academic careers.

Almost all of the students had taken algebra or were taking it concurrently with the course.

The last question was designed to determine exposure to computer programming, since traditional computer classes focus on teaching computer programming. Table 19 shows that almost three fourths of the students report having no programming experience. About 13% report having one to three months of experience, the amount of time spent taking a single programming course. Also note that the proportion of students reporting any programming experience remains constant across semesters. This contrasts with the trends in general computing experience and the use of computer applications, which have increased over time.

Table 19

Amount of Student Computer Programming Experience

Semester	Never	1 - 3 months	3 – 6 months	6 - 12 months	More than 1 year
FS98	71.8	14.1	5.2	2.8	6.1
SS99	73.7	11.8	7.5	2.5	4.5
FS99	72.6	13.5	5.4	4.3	4.2

Note: All values are percent of students who responded to the question.

College algebra is often a prerequisite for traditional introductory computing courses, particularly programming classes. While most of the students in the course report having taken algebra, the responses appear to be

associated with class standing. The next section will explore this relationship in greater depth. Finally, in spite of the almost ubiquitous exposure to computing, very few students have any programming experience and the proportion of students who have programmed remained constant over time.

Survey Factor Analysis

Factor analysis was used to reduce the dimensionality of the survey and identify underlying factors that represent the computing experience measured by the survey. This section summarizes the results of this analysis.

All of the survey variables were included in the initial factor analysis producing three orthogonal factors with eigenvalues larger than 1.0. The question about algebra loaded mainly on the third factor. The only other variable that loaded on factor 3 more than on factors 1 or 2 was the email question. Recall that there is a .486 correlation between class level and response to the algebra question and a .197 correlation between E-mail and class level. It appears that factor 3 primarily reflects class level.

The next factor analysis dropped the algebra question and produced only two factors with eigenvalues larger than 1.0. To produce the factors that had the most separation, questions that loaded on both factors were eliminated until the remaining variables clustered closely around the factor axes. An oblique rotation was used to maximize the loadings of the variables on one of the two factors. Table 20 shows the resulting loadings.

Table 20

Final Set of Survey Variables and Factor Loadings

Question	Factor 1	Factor 2
How long have you worked with a computer	.604	-.079
How did you learn to type	.591	.123
Computer programming	-.074	-.665
Know User Interface meaning	.053	-.819
Know Hypertext means	.112	-.759
E-mail	.629	-.044
Online service	.728	.018
WWW	.658	-.158

The two factors account for 49.4% of the variance of the survey variables used in the final analysis. Factor 1 loads on the questions about how long students have worked with a computer, how they learned to type and their use of E-mail, online services and the Web. This factor appears to reflect an acquaintance with using computers for communication. Factor 2 loads on the questions about computer programming and knowledge of abstract terms such as “user interface” and “hypertext.” This factor appears to measure more in-depth experience, moving beyond simply using computers to a deeper knowledge of computer operations and terminology. The questions that were eliminated – knowing the meaning of “windows,” and use of word processing, spreadsheets and databases – are experiences that loaded on both factors and

are common across students that use computers for communication and those with more abstract computing knowledge.

The scores for these factors were saved as new variables (*Computer communication* and *Computer terms*) for each student to be used in subsequent analyses. Mean values were substituted for any students for whom there were missing data on the survey questions.

Incoming Data Model

To what degree does the incoming student data alone predict student performance in the course? Recall from Chapter 3 that there are three stages in the application of discriminant analysis: 1) derivation, 2) validation and 3) interpretation (Hair et al., 1987). This section reviews the results of these three stages using the incoming student data as the independent variables and the course grade as the categorical dependent variable. Each of these stages is presented in detail for this analysis so that the reader understands the procedures; subsequent analyses of the remaining parts of the instructional system follow a parallel process.

Discriminant Function Derivation

In the derivation stage, the cases were randomly divided into two groups. The first group had 70% of the cases ($n = 5289$) and was used to derive the discriminant functions. The second group had the remaining 30% of the cases and was used for the validation stage. Cases with missing data on any of the independent variables were not included in the derivation phase of the analysis.

Table 21 shows the distribution of by the dependent grouping variable -- final course grade – for cases that were used to derive the functions. Note that it follows the distribution of course grades for the entire set of data shown in Figure 10.

Table 21

Distribution of Cases by Final Course Grade

Grade	0.0	1.0	1.5	2.0	2.5	3.0	3.5	4.0	Total
<i>n</i>	42	102	185	364	633	503	176	1033	3038

The discriminant functions were derived from the pooled within-groups correlation matrix. Table 22 summarizes *F* values for the univariate ANOVAs on the group means for each of the variables entered into this analysis.

Table 22

Tests of Equality of Group Means

Variable	Wilks' Lambda	<i>F</i> *	Sig.
Cumulative GPA	.602	286.243	.000
ACT English	.959	18.513	.000
ACT Social Science	.978	9.523	.000
ACT Natural Science	.947	24.037	.000
ACT Mathematics	.906	45.164	.000
ACT Composite	.935	30.232	.000
Class level	.992	3.300	.002
Age	.979	9.475	.000
Gender	.997	1.312	.240
Asian / Pacific Islander	.998	.904	.502
Black	.970	13.184	.000
Chicano / Hispanic	.993	2.867	.006
Computer communication	.978	9.732	.000
Computer terms	.992	3.404	.001
Track C	.994	2.520	.014
Track D	.984	6.900	.000

Note: * all *F* tests, *df* 1, 3030

The means of each of the independent variables except *Asian / Pacific Islander* and *Gender* are significantly different across groups. However, we do not know among which groups the differences are significant. More importantly, these ANOVAS do not give us information about the interrelationships among the variables. To better understand the relative importance of each of the variables, we need to examine the results of the stepwise inclusion and removal of these variables shown in Table 23. The variables were selected using step-wise inclusion, maximizing the Mahalanobis D^2 distance between groups.

Table 23

Stepwise Variable Inclusion and Removal Summary

Step *	Variable	Minimum D^2					
		Statistic	Between Groups	F	$df 1$	$df 2$	Sig.
1 E	Cumulative GPA	.114	1.0 – 1.5	7.525	1	3030	6.1E-03
2 E	Class level	.143	1.5 – 2.0	8.754	2	3029	1.6E-04
3 E	Track D	.159	1.5 – 2.0	6.481	3	3028	2.3E-04
4 E	Computer communication	.166	1.5 – 2.0	5.087	4	3027	4.4E-04
5 E	ACT Mathematics	.171	1.5 – 2.0	4.199	5	3026	8.5E-04
6 E	Track C	.173	1.5 – 2.0	3.541	6	3025	1.7E-03
7 E	Computer terms	.174	1.5 – 2.0	3.043	7	3024	3.4E-03
8 E	ACT Composite	.174	1.5 – 2.0	2.667	8	3023	6.4E-03

Note: At each step, the variable that maximizes the Mahalanobis distance between the two closest groups was entered. After 8 steps, the F level was insufficient for further inclusion or removal.

* E: variable entered at this step; R variable removed at this step.

At the first step, the variable *Cumulative GPA* was first variable entered. It maximized the Mahalanobis distance the two closest groups, the students who received a 1.0 and those that received a 1.5 in the course. In the second step, *Class level* was added to the function because it maximized the distance between the two groups that were closest, after including *Cumulative GPA*, the students who received a 1.5 and those that received a 2.0. After eight steps, no additional variables significantly improved the separation among groups. Also note that, although a stepwise procedure was used, no variables were removed once they were added. It is encouraging to note that, although the dependent variable is treated as nominal, not ordinal, in discriminant analysis, each of the steps maximized the separation between two adjacent grades.

After deriving functions, the next step is to determine if they are statistically significant. Table 24 shows the significance tests for the set of discriminant functions. The functions are numbered in order of the percentage of the total variance for which they account. The first row shows the overall Chi-square significance for all seven functions considered together. The second row shows the test of functions two through seven, without function one. It is possible to eliminate the first two functions, which account for 97.2% of the total variance, and the remaining functions (3 through 7) are still significant. However, functions 4 through 7 are not significant without the other functions.

Table 24

Significance of the Discriminant Functions

Test of Function(s)	Wilks' Lambda	Chi-square	df	Sig.
1 through 7	.534	1899.053	56	.000
2 through 7	.950	156.212	42	.000
3 through 7	.977	69.962	30	.000
4 through 7	.990	29.683	20	.075
5 through 7	.997	9.679	12	.644
6 through 7	.999	2.778	6	.836
7	1.000	.741	2	.690

This completes the derivation stage of the analysis. Even though the univariate ANOVAs for 14 of the 16 original variables were significant, it is possible to construct a set of statistically significant discriminant functions using only eight of the variables. Since the functions were significant, we can move to the next step, the validation stage.

Discriminant Function Validation

Even though the functions were statistically significant, with large n's it is possible to have statistical significance without any practical utility. Validation allows us to understand the predictive capability of the functions by using them to classify, or predict, the final grade for each student. To provide the most stringent test of the functions, equal prior probabilities were assigned to each grade for the

classification. Thus, with eight categories, we expect to classify 12.5% of the cases correctly if the functions performed no better than chance. The separate group covariance matrices of the discriminant functions were used for classification because Box's M test of equality for the matrices was significant. Although Box's M is often significant with large samples and multiple groups, and thus the covariance matrices may not be different (SPSS, 1999b, p. 280), using the separate covariance matrices provides a more conservative classification test.

Validation was done in two phases. First, cases from the original sample from which the functions were derived were classified. In this phase, cases were included that had no missing data on the variables that were included in the final functions. Thus, there are some additional cases that were not in the original derivation analysis ($n = 4420$) since they had missing data on variables that were not included in the final functions. Table 25 shows the classification results for the original sample.

Table 25

Incoming Data Classification Results for Cases Used to Derive the Discriminant Functions

Actual grade	n	Percentage classified by predicted grade							
		0.0	1.0	1.5	2.0	2.5	3.0	3.5	4.0
0.0	44	<u>63.6</u>	20.5	11.4	2.3	2.3	.0	.0	.0
1.0	104	14.4	<u>50.0</u>	12.5	7.7	11.5	1.0	1.9	1.0
1.5	186	11.8	28.5	<u>20.4</u>	15.6	18.3	1.6	2.7	1.1
2.0	367	7.1	19.3	15.5	<u>18.0</u>	22.9	4.6	7.6	4.9
2.5	637	3.9	9.4	6.8	11.5	<u>39.6</u>	7.8	14.8	6.3
3.0	507	2.4	7.1	4.9	12.0	24.1	<u>10.3</u>	19.3	19.9
3.5	177	1.1	3.4	5.1	6.2	18.1	7.3	<u>28.2</u>	30.5
4.0	1044	.2	3.4	1.6	4.4	11.0	6.5	20.5	<u>52.3</u>

Note: 35.4% of the original grouped cases correctly classified.

The rows contain the results for the cases based on the actual grade each student received. The number of students who received each grade is shown in the column labeled n. The remaining columns show the percentage of predicted grades for the cases in each row. For example, there were 44 students who received a grade of 0.0. Of those 44 students, 63.6% were correctly predicted to receive a 0.0. Another 20.5% were incorrectly predicted to receive a 1.0 and 11.4% to receive a 1.5. The underlined numbers on the diagonal show the percentages of each actual grade that were correctly predicted. Thus, 52.3% of

the students who actually received grades of 4.0 were predicted to receive a 4.0. Numbers above the diagonal are cases where the predicted grade was higher than the grade the student actually received. Numbers below the diagonal are cases where the predicted grade was lower than they actually received. The overall classification accuracy was 35.4%.

To control for the upward bias we expect when classifying cases that were used to derive the functions, we can use the functions to classify the cases that comprised the second sample ($n = 1354$). The results are shown in Table 26. There is little upward bias, since 32.8% of the second sample were correctly classified.

Table 26

Incoming Data Classification Results for Unselected Cases

Actual grade	n	Percentage classified by predicted grade							
		0.0	1.0	1.5	2.0	2.5	3.0	3.5	4.0
0.0	18	<u>66.7</u>	16.7	11.1	.0	5.6	.0	.0	.0
1.0	49	22.4	<u>44.9</u>	10.2	12.2	8.2	.0	.0	2.0
1.5	89	13.5	29.2	<u>15.7</u>	10.1	23.6	4.5	2.2	1.1
2.0	163	8.6	19.0	17.2	<u>19.0</u>	21.5	4.9	6.1	3.7
2.5	277	3.6	15.9	5.4	10.5	<u>33.6</u>	10.1	8.7	12.3
3.0	225	.9	7.1	5.8	12.4	25.3	<u>5.8</u>	20.9	21.8
3.5	82	.0	7.3	7.3	6.1	22.0	12.2	<u>15.9</u>	29.3
4.0	451	.9	2.2	2.0	5.1	11.1	6.4	17.7	<u>54.5</u>

Note: 32.8% of the unselected sample cases correctly classified.

Because the actual grades were not equally distributed – 34% of the students received a 4.0 – another way to evaluate the functions is to compare the classification accuracy to the results if we were to simply assign everyone to the 4.0 group. If we did that, we would be correct 34% of the time, which is as good as using the discriminant functions. However, examining the classification tables shows that the accuracy is far from random. For example, for 18.8% of the second sample the functions underestimated the final grades by only one grade (e.g., predicting a 1.0 when the student received a 1.5 grade.) It overestimated by one grade level for another 11.2% of the second sample. Thus, it correctly

predicted 32.8% of the grades and was within plus or minus one grade for another 30% of the sample.

Interpretation of the Discriminant Functions

Having validated the utility of the discriminant functions, we can next turn to the third step: interpretation. There are a number of ways to interpret the results of discriminant analysis. A popular one is to plot the group centroids against axes formed by the discriminant functions, similar to plotting variables against factor axes in factor analysis. This shows the structural relationship of the variables in analytic space. Another graphic method of displaying the results is a “territory map” that shows the boundaries for each of the groups. Cases are classified based on where they lie in this space and a territory map can provide a sense of the relationships among the group centroids and the boundaries of each group. However, graphic methods are limited in the number of dimensions that can be displayed. For a two or three group problem, graphic methods work well. For an eight-group problem in seven dimensional analytic space, projecting the seven dimensions onto two dimensions for plotting makes interpretation difficult. Fortunately, since the dependent variable is ordinal rather than nominal, interpretation is somewhat less complex. We already understand the relationship among the dependent groups – it is ordered. We primarily need to understand the relationships among the independent variables.

For high dimensional problems, we can examine the discriminant functions to determine the influence of each of the independent variables on each function. The most common approaches are a) examining the standardized

discriminant weights, which is similar to interpreting beta weights in regression; b) examining the discriminant structure correlations or loadings, similar to examining loadings in factor analysis and c) examining the stepwise partial F-values, which is the same as examining the discriminant weights, but also assigns significance values to each variable. While all of these methods may be subject to instability, examining the correlations between the independent variables and the functions is preferred (Hair et al., 1987).

Table 27 shows the pooled within-groups correlations between the independent variables and the standardized canonical discriminant functions. All variables from the original analysis are shown in the table, with the variables that were selected for the functions – *Cumulative GPA* through *Track D* – shown in the first eight rows.

Table 27

**Correlations Between Independent Variables and Standardized Canonical
Discriminant Functions**

Variable	Function						
	1	2	3	4	5	6	7
Cumulative GPA	.922	.055	-.177	.055	-.002	.145	.070
ACT Mathematics	.349	-.236	.683	.478	-.067	-.007	-.347
ACT Composite	.279	-.245	.734	.143	.124	.084	.023
Class level	-.032	-.325	-.303	.601	-.129	.082	.618
Computer communication	.156	-.197	.383	-.093	.441	.028	.501
Computer terms	-.066	.316	-.263	.170	.128	.818	-.323
Track C	-.024	.316	-.110	.382	.735	-.417	-.092
Track D	.054	.608	.400	-.070	-.594	.094	.270
ACT English *	.219	-.189	.560	.019	.170	.117	.117
ACT Social Science *	.181	-.176	.554	-.030	.160	.108	.180
ACT Natural Science *	.194	-.213	.633	.077	.113	.041	.056
Age *	-.124	-.228	-.280	.439	-.135	.011	.469
Gender *	.043	.029	-.168	-.042	.060	.167	.062
Asian / Pacific Islander *	-.023	.006	.026	.051	-.024	-.038	-.060
Black *	-.135	.108	-.244	-.101	-.087	-.093	.005
Chicano / Hispanic *	.000	.005	-.044	-.047	.002	.010	-.017

Note: * variable not used in analysis

Interpretation of these correlations is similar to interpreting the correlations between variables and factor loadings in factor analysis. We are interested in the magnitude of the correlation; the sign merely indicates the direction, which is arbitrary. For example, *Cumulative GPA* is most correlated with function 1 (.922). However, other variables do not correlate as strongly with single functions. *ACT Mathematics* has a moderate correlation with all of the functions except 5, and 6, making interpretation more difficult. The solution to this is to rotate the function axes to maximize the correlation between variables and the individual functions. By using an orthogonal rotation, the structure of the analytic space – and the classification accuracy – are preserved. Rotation merely simplifies the interpretation of the correlations between the variables and the functions, much like factor analysis. The rotated correlations are shown in Table 28.

Table 28

Correlations Between Independent Variables and Rotated Functions

Variable	Function						
	1	2	3	4	5	6	7
Cumulative GPA	.921	.189	.145	-.003	.034	-.039	.072
ACT Mathematics	-.023	.984	.157	-.015	.049	-.028	-.018
ACT Composite	-.071	.621	.551	-.069	.101	-.102	-.047
Class level	.109	-.055	.096	.961	-.114	-.037	-.044
Computer communication	-.004	-.010	.801	.064	-.014	.065	-.119
Computer terms	.082	-.049	-.193	-.021	-.020	-.008	.972
Track C	-.005	.037	.073	-.020	-.222	.962	-.003
Track D	.010	.019	-.065	-.002	.969	-.167	-.009
ACT English *	-.039	.372	.542	-.069	.076	-.096	-.013
ACT Social Science *	-.064	.300	.564	-.061	.106	-.110	-.041
ACT Natural Science *	-.101	.479	.490	-.067	.092	-.090	-.080
Age *	.014	-.112	-.005	.743	-.084	-.018	-.071
Gender *	.121	-.152	.024	.037	-.048	-.027	.159
Asian / Pacific Islander *	-.037	.064	-.055	-.002	.004	.026	-.014
Black *	-.011	-.253	-.219	-.004	.007	.028	-.054
Chicano / Hispanic *	.019	-.045	-.022	-.033	-.020	-.015	.010

Note: * variable not used in analysis

Compare the rotated correlations in Table 28 with the correlations before rotation in Table 27. Note that *Cumulative GPA* still is primarily correlated with function 1. There has been a slight reduction in the correlation between *Cumulative GPA* and functions 3, 4 and 6. However, there is also a slight increase in the correlation between functions 2, 5 and 7, though the overall effect is to “sharpen” the correlation mostly on function 1. Looking at *ACT Mathematics* we see that it now primarily correlates with function 2. We can also see that most functions correlate primarily with single variables. The exception is *ACT Composite*, which has moderate correlations with both functions 2 and 3.

After rotation, we can summarize the “meaning” of the functions based on the variables with which each has the strongest correlations. Function 1, which accounts for 76.7% of the variance after rotation, is associated with *Cumulative GPA*. Function 2 reflects *ACT Mathematics* scores and a moderate amount of *ACT Composite* and accounts for 11.5% of the variance. Function 3 loads primarily on *Computer communication* and moderately on *ACT Composite* with 4.6% of the variance. Function 4 is *Class Level* with 2.4% of the variance. Function 5 is *Track D* with 2.3% of the variance and Function 6 is *Track C* with 1.3% of the variance. These two binary variables together reflect the three tracks. Finally, *Computer terms* is associated with function 7 and 1.1% of the variance.

Now that we have an interpretation for the “meanings” of the factors, we can examine the group centroids to see if they are significant. Table 29 summarizes the F-tests of the differences between each pair of means.

Table 29

F-Statistics Among Each Pair of Group Centroid Means

Group	0.0	1.0	1.5	2.0	2.5	3.0	3.5
1.0	7.734						
1.5	12.610	2.489					
2.0	19.759	8.024	2.667				
2.5	31.911	21.934	19.554	13.976			
3.0	41.277	33.324	31.442	26.494	11.695		
3.5	47.320	40.689	38.085	31.349	15.701	4.612	
4.0	72.277	84.014	102.157	122.815	99.988	36.746	5.341

Note: All *F* tests $p < .01$, *df* 8, 3023

The *F*-statistics are a measure of the degree of separation among groups. What is interesting is that there is actually a greater separation between the 2.0 and 4.0 groups than between the 4.0 and other groups. However, recall that these distances are absolute distances and the vectors lie in different directions in 7-dimensional space. Table 30 shows the location of the centroids for each unstandardized function for each of the final course grades.

T

L

C

m

g

th

F

Table 30

Unstandardized Functions at Group Centroids

Course grade	Function						
	1	2	3	4	5	6	7
0.0	-2.831	-.378	-.005	.200	-.044	-.160	.036
1.0	-1.524	-.391	-.457	.526	-.146	.028	.181
1.5	-1.160	-.446	-.285	.204	-.284	-.007	.060
2.0	-.796	-.435	-.194	.099	-.150	-.074	.059
2.5	-.339	-.185	-.130	-.023	.203	.180	.128
3.0	.052	.024	.015	.055	-.057	.044	-.027
3.5	.445	.148	.036	-.137	.155	-.101	-.131
4.0	.861	.364	.231	-.121	-.033	-.083	-.094

Not surprisingly, since function 1, Cumulative GPA, accounts for the majority of the variance, the distances of the centroids on function 1 are much greater than the distances on the remaining functions. The relationship between the centroids on function 1 and the course grade is nearly linear, as shown in Figure 11.

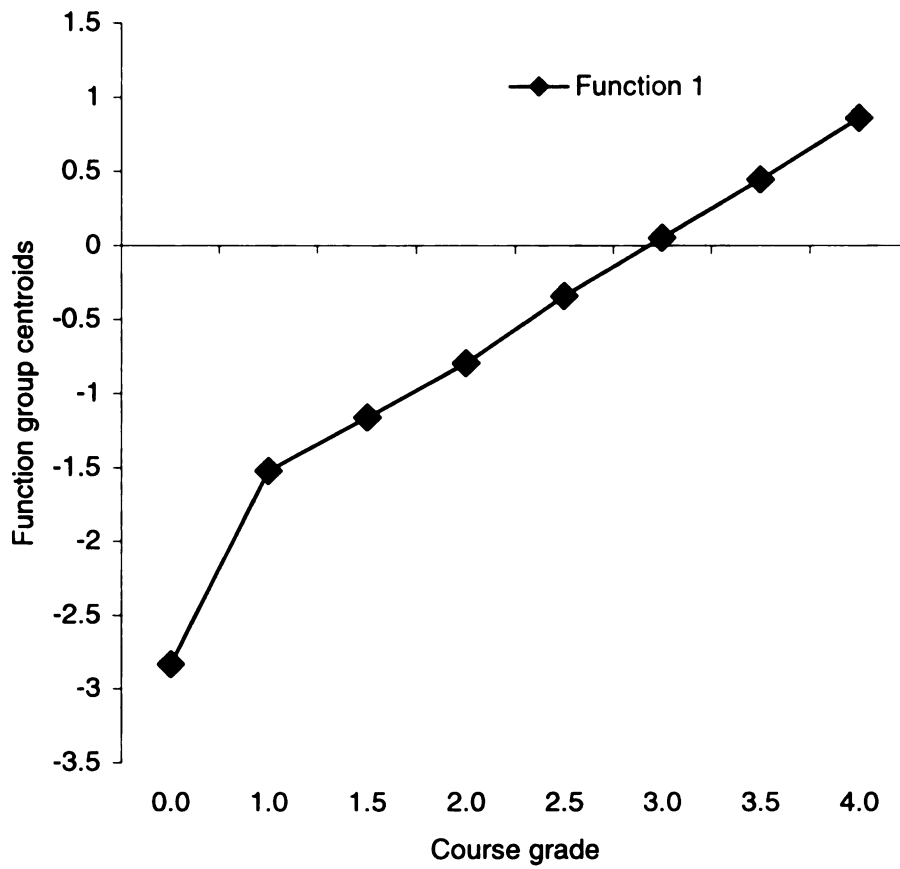


Figure 11 Group centroids for function 1

Given the strength of the relationship between function 1 and the final grade, what contributions do the other functions make? The centroids for Functions 2 through 7 are shown in Figure 12.

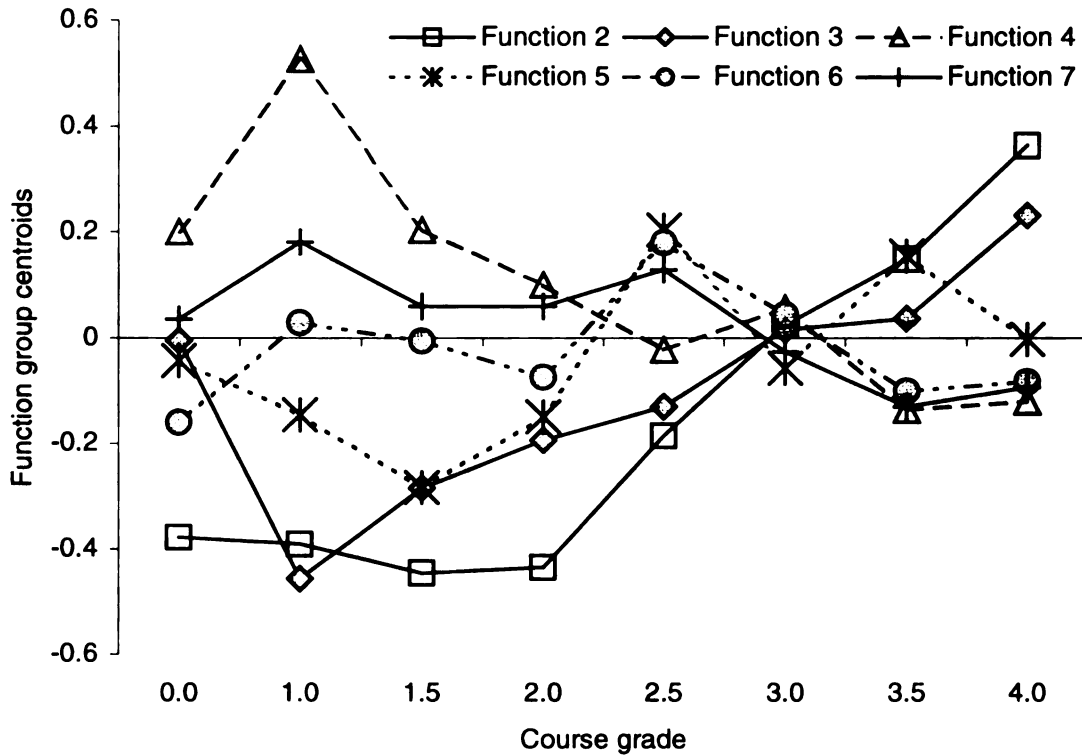


Figure 12 Group centroids for functions 2 through 7

Note that the Y axis scale in Figure 12 is smaller than the Y axis scale in Figure 11, since the range of the centroids for Functions 2 through 7 is smaller than for Function 1. Recall that the axes for all seven of these functions are orthogonal to each other in analytic space. They are plotted on the same graph only to show the non-linear nature of the relationships among each of the functions and the course grade. For example, Function 2 remains at about -0.4 for grades 0.0 through 2.0, then rises from 2.5 through 4.0. On the other hand,

functions 5, 6 and 7 have “sawtooth” shapes. It is the interaction of these non-linear relationships in the 7 dimensional analytic space that provides additional classification accuracy over a linear prediction on the *Cumulative GPA* alone.

Summary

In this section we examined the final course grades and saw that, as expected in a criterion-referenced assessment, they are distributed in a linear rather than a normal fashion. We next examined the various incoming student variables, what they measure, and their univariate the distributions. We then used these incoming variables to derive a series of discriminant functions that used course grades as the categorical grouping variable. A subset of the original data was classified with the discriminant functions to evaluate the functions' classification accuracy. Finally, the correlations between the independent variables and the rotated discriminant functions were used to interpret the meaning of the resulting functions. Although interpretation of the incoming student variables is relatively trivial – cumulative GPA is the strongest predictor of outcome in this model – this section demonstrated the analysis framework used for the remaining portions of the model. The next section follows a parallel structure to examine the classroom variables in the instructional system and determine the impact of adding these variables to the model.

Instructional Variables

The second part of the model is the instructional system. This includes instructional staff variables, student participation variables and records of interactions between the student and the course instructors.

Teaching Assistants

Recall from Chapter 3 that Teaching Assistants (TAs) meet each section of the course and conduct the actual classroom instruction. The instructors provide detailed lesson plans and resources for each day's instruction to ensure the maximum consistency among sections. However, each TA has his or her own teaching style, strengths and weaknesses. This section examines data about the TAs. This includes the number of semesters experience the TAs had teaching the course and the average instructor ratings each TA received from students across all sections taught by that TA. The final section of this chapter will examine the individual student ratings of the TAs.

Teaching Assistant Experience

Most of the teaching assistants are graduate students in computer science or other engineering disciplines. Few of them have prior teaching experience when they join the instructional staff so, for most, this is their first time teaching. Although the TAs receive training before they enter the classroom and at weekly staff meetings, we would expect their teaching to improve as they gain classroom experience. Because the TAs are graduate students, there is a substantial turnover. About half of the TA staff are new each year. Table 31 summarizes

the number of semesters of experience that the TAs had during each semester. These percentages are based on the number of students taught by each TA. Generally, novice TAs are assigned one or two sections of 30 students, where TAs with previous experience will have three, and occasionally four, sections. Almost half of the students were taught by a teaching assistant who was teaching for the first time; less than one third of the students were being taught by a TA in his or her second semester and less than one fourth of the students' TAs had a year or more of experience.

Table 31

Teaching Assistant Experience By Semester

Semester	First semester	Second semester	Over two semesters
FS98	73.1%	13.6%	13.3%
SS99	19.6%	61.1%	19.3%
FS99	56.2%	7.6%	36.2%
Total	49.9%	27.2%	22.9%

Student Ratings of Teaching Assistants

At the end of each semester, students complete a survey about the course and the university SIRS for their teaching assistants (see discussion in Chapter 3, the questions are shown in Appendix D). For each TA, the student ratings from

all students taught by that TA were averaged to create an average measure of that TA's ratings for each semester. These averages were used as variables for each student. Thus, each of the students has "TA ratings" variable that consist of the average of all student ratings for that TA from the semester they took the course. Because these ratings are based on n's > 30, the influence of any individual student's ratings of his or her TA is small.

Responses to the SIRS questions are strongly inter-correlated so factor analysis was used to reduce the dimensionality of the ratings to a single factor that accounts for 90.3% of the variance. Table 32 shows the factor loadings.

Table 32

Factor Loadings for Average SIRS Ratings of TAs

Variable	Loading
Lead TA available and willing to help	.923
Lead TA explained material clearly	.966
Lead TA well prepared	.926
Lead TA organized and explained well	.971
Lead TA communicated well	.929
Grade Lead GA	.984

There was a factor score for each TA for each semester in which that TA taught. These factor scores were saved as variables for each student to be used in further analysis.

Student Participation

Student participation data consists of attendance rates, number times students took bridge tasks, student self-reported data on how frequently they attended the help room sessions and how frequently they did the homework and assigned readings before coming to class.

Attendance

The assistant TA in each section records student attendance in the database. This data is then used to randomly assign students to their collaborative groups each day. Students receive no credit for attendance. The attendance rate is simply the percentage of instructional class days – not including in-class bridge tasks – on which the student was in class. Occasionally students come to class late and the assistant TA fails to record their attendance. Likewise, if students leave early, this is not recorded. However, the error rate on this data is quite low. Because attendance is not “mandatory,” there is substantial variance. Figure 13 shows the attendance distribution ($M = 72\%$, $SD = 24\%$).

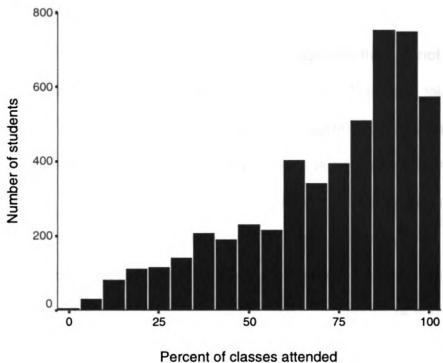


Figure 13 Class attendance distribution

Compare the attendance distribution with the final course grades distribution (Figure 10). The correlation between attendance and final grade is strong ($r = .53$) and has been consistent across semesters. The instructors noted this relationship after analyzing the data from the first year the course was offered (Urban-Lurain & Weinshank, 2000). As a result, the course instruction repeatedly emphasizes the importance of attendance throughout the semester. Beginning in spring, 1999, the instructors began sending E-mail to students when they missed class, reminding them of the material they missed and of the importance of attendance.

Number of Bridge Tasks Taken

Recall that students must repeat any bridge task they do not pass until they pass that BT or the end of the semester arrives. There is a total of five different BTs (1.0, 1.5, 2.0, 2.5 and 3.0) and a maximum of 12 opportunities for students to take BTs during the semester, so students may repeat each BT once and still pass the 3.0 BT if they take all available opportunities. (Only 5% of students pass all five BTs on the first try.) However, many students must repeat some BTs more than one time. Furthermore, many students do not take advantage of all available opportunities to take BTs. Figure 14 shows the distribution of the number of times students took BTs ($M = 7.41$, $SD = 2.42$).

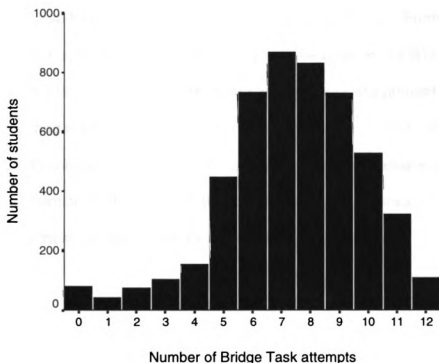


Figure 14 Number of bridge task attempts

Note the asymmetry of the distribution at the lower tail, with a number of students who take four or fewer BTs. There are five in-class BT opportunities; other BTs must be scheduled and taken outside of class. There are many students who never take advantage of the make-up opportunities.

Student Preparation

On the end of the semester survey, students are asked to report how frequently they did the homework assignments and readings before coming to class and how often they attended the optional help rooms that are available outside of class. Homework is not graded directly. However, the class exercises often require that students have their homework completed. Students who have not done so have difficulty completing the class exercises. Furthermore, any homework and classroom assignments may be used on the BTs. If students come to a BT without completed homework or class assignments, they must take time during the BT to complete the necessary work in order to complete the BT. Table 33 summarizes the responses to the questions “I usually did the homework before coming to class” and “I attended help room frequently.” The response rates to these questions are 74.6% and 69.1% respectively.

Table 33

Student Self-Reports of Doing Homework and Attending Help Room

Question	<i>n</i>	Strongly Agree	Agree	Neither Agree or Disagree	Disagree	Strongly Disagree
Usually did homework	3797	18.7	45.5	16.6	14.6	3.6
Attended help room frequently	3516	3.2	10.5	16.2	30.8	39.3

Note: All values are percent of students who responded to the question.

Interactions Between Students and Course Instructors

The student database is a key component of the course design, allowing the instructors to collect and manage a large amount of data about each student. Among the data tracked are a number of different interactions with the students. For each student these include: each E-mail message sent to the student; each E-mail message received from the student; records of phone calls to or from the student; records of instructor office visits; records of communication with the student's TA about the student; and student requests for reviews of their BT grading. This portion of the student database was implemented in spring, 1998.

During spring, 1998, only manual data was collected. Starting in summer, 1998, the instructors began using data about each student's progress (e.g, attendance, current BT status, etc.) to send customized E-mail messages to students during the semester (Urban-Lurain & Weinshank, 2000). The numbers

of E-mail messages sent to the students in spring, 1998, fall, 1998, spring, 1999 and fall, 1999 are shown in Figure 15.

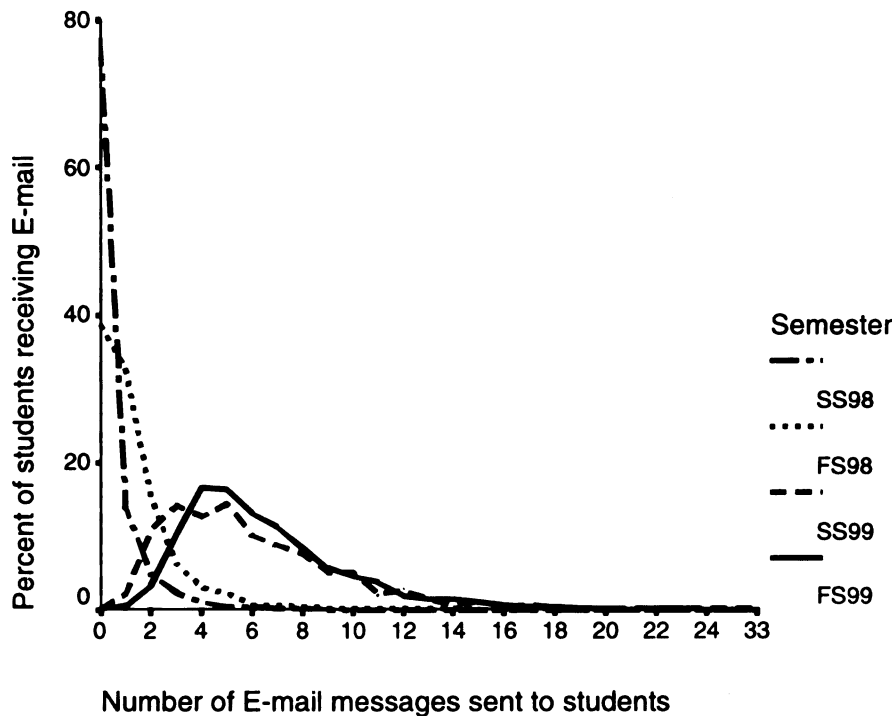


Figure 15 Number of E-mail messages sent to students by semester

The X-axis displays number of E-mail messages sent to each student. The Y-axis shows the percentage of students receiving that number of messages. Only individual E-mail sent to each student is shown on the graph. E-mail sent by the instructors to the entire class and messages sent by the TAs to their students are not included. In spring, 1998, 77.6% of the students did not receive any individualized E-mail and about 13.7% received only one message ($M = 0.39$, $SD = 0.93$). By fall, 1999, every student received at least one message, with 16.6% receiving four messages and another 16.3% receiving five messages ($M = 6.42$, $SD = 3.11$). E-mail received from students follows a

similar trend across the semesters. This makes sense since students frequently reply to the E-mail they receive from the instructors with questions or comments. This often results in an E-mail dialog between the student and the instructors, with the instructors answering questions or suggesting the students attend help room or come to instructor office hours.

Other interactions with students that are recorded in the database have been more constant across semesters and are much less frequent. Most students never come to instructor office hours (95.5%) and never submit requests to have their bridge task grading reviewed by a head grader (88.6%). Over 98% of the students never telephone the course office, since the instructors encourage E-mail as the preferred means of communication in the course. Similarly, the instructors exchange E-mail or have personal consultations with the teaching assistants about fewer than 2% of the students in the course.

The data about interactions between students and the course instructors were analyzed with factor analysis to reduce the dimensionality. Three factors accounted for 62.8% of the variance. The correlations between each of the variables and rotated oblique factors are shown in Table 34. The resulting factors were saved as variables for use in subsequent analyses.

Table 34

Correlations with Student Interaction and Factors

Variable	Factor 1	Factor 2	Factor 3
E-mail to TA about student	.104	.877	.050
Student phone call	-.003	.062	.821
Office visit	.241	.053	.710
E-mail to student	.869	.156	.221
E-mail from student	.859	.132	.170
E-mail from TA about student	.152	.873	.075
BT review request	.373	.045	.034

Factor 1 primarily represents E-mail between the student and the instructors. The BT review requests also correlate with this factor as some students are not satisfied with the results of their BT review request and they send E-mail to the instructors. Likewise, there is a modest correlation with office visits, since the E-mail from the instructors always encourages students to come to office hours. Factor 2 primarily represents communication between the instructors and the TAs about particular students. The TAs often contact the instructors if they are having problems or questions about helping students. The instructors will often send E-mail to a student's TA alerting the TA to the issues that came up during office visits or in E-mail with the student. Factor 3 represents phone calls and office visits. As noted above, the proportion of

students phoning or coming to instructor office hours is small. These are often students who are having academic or personal problems and who need personal interaction with the instructors. E-mail with students slightly correlates with this factor since the office visits or phone calls often result from preliminary E-mail contact, or are followed up with E-mail.

Instructional Data Model

The instructional data were added to the incoming student data and used as independent variables for discriminant analysis so that all of the incoming student variables – including those that were eliminated during the original analysis – were again included in this analysis. This allowed the analysis to identify interactions between any of the incoming variables and the classroom variables. The analysis process paralleled the analysis for the incoming student data model; see that description for details on the meanings and rationale of each step in the analysis.

Discriminant Function Derivation

As in the analysis of incoming student variables, the cases were randomly divided into two groups. The first group had 70% of the cases and was used to derive the discriminant functions ($n = 3038$). The second group had the remaining 30% of the cases and was used for the validation stage. Cases with missing data on the survey questions about doing homework and attending help room were included in the analysis on the assumption that non-respondents on these questions may be different than students who did respond. The non-

responses were treated as unique, valid values for the analysis (SPSS, 1999a, p. 307.) Cases with missing data on any of the remaining independent variables were excluded in the derivation phase of the analysis. Table 35 summarizes the *F* values for the univariate ANOVAs on the group means for each of the variables entered into this analysis.

Table 35

Tests of Equality of Group Means

Variable	Wilks' Lambda	F	Sig.
Cumulative GPA	.602	286.243	.000
ACT English	.959	18.513	.000
ACT Social Science	.978	9.523	.000
ACT Natural Science	.947	24.037	.000
ACT Mathematics	.906	45.164	.000
ACT Composite	.935	30.232	.000
Class level	.992	3.300	.002
Age	.979	9.475	.000
Gender	.997	1.312	.240
Asian / Pacific Islander	.998	.904	.502
Black	.970	13.184	.000
Chicano / Hispanic	.993	2.867	.006
Computer communication	.978	9.732	.000
Computer terms	.992	3.404	.001
Percent of classes attended	.699	186.140	.000
Number of BT attempts	.605	282.995	.000
Did homework	.821	94.111	.000
Attended help room frequently	.958	19.070	.000
Student Email Factor	.985	6.716	.000

(table continues)

Table 35 (cont'd).

Variable	Wilks' Lambda	F	Sig.
Comm with TA about student	.993	2.888	.005
Office phone	.994	2.759	.007
Track C	.994	2.520	.014
Track D	.984	6.900	.000
TA Experience	.992	3.487	.001
TA Average SIRS	.996	1.789	.085

Note: All *F* tests *df* 7, 3030.

The significance of some of the incoming student variables remains the same as in the first analysis (see Table 22). The means of all of the instructional variables except for the *TA Average SIRS* ratings are significant. Again, these ANOVAS do not provide information about which means are different nor do they tell us anything about the interrelationships among the variables. Table 36 shows the order that the variables were included or removed during the analysis and which pair of group Mahalanobis distances were maximized by each of the variables.

Table 36

Stepwise Variable Inclusion and Removal Summary

Step *	Variable	Minimum D^2					Sig.
		Statistic	Between Groups	F	$df 1$	$df 2$	
1 E	Cumulative GPA	.114	1.0 – 1.5	7.525	1	3030	6.2E-03
2 E	Attendance	.184	3.0 – 3.5	11.970	2	3029	6.8E-06
3 E	Student Email	.241	3.5 – 4.0	12.082	3	3028	7.7E-08
4 E	Did homework	.279	3.5 – 4.0	10.490	4	3027	2.1E-08
5 E	Number BT attempts	.315	3.5 – 4.0	9.456	5	3026	6.3E-09
6 E	Attended helproom	.360	3.0 – 3.5	7.810	6	3025	2.6E-08
7 E	Track D	.414	3.5 – 4.0	8.870	7	3024	1.1E-10
8 E	ACT Social Science	.419	3.0 – 3.5	6.820	8	3023	7.5E-09
9 E	TA Experience	.433	3.0 – 3.5	6.253	9	3022	9.7E-09
10 E	ACT Mathematics	.442	3.5 – 4.0	6.631	10	3021	4.0E-10
11 E	Computer communication	.443	3.0 – 3.5	5.237	11	3020	3.4E-08
12 R	Attended helproom	.395	3.5 – 4.0	5.923	10	3021	7.3E-09

(table continues)

Table 36 (cont'd).

Step *	Variable	Minimum D^2					
		Statistic	Between Groups	F	$df 1$	$df 2$	Sig.
13 E	Computer terms	.399	3.5 – 4.0	5.440	11	3020	1.4E-08
14 E	Track C	.399	3.5 – 4.0	4.987	12	3019	3.3E-08

Note: At each step, the variable that maximizes the Mahalanobis distance between the two closest groups was entered. After 14 steps, the F level was insufficient for further inclusion or removal.

* E: variable entered at this step; R variable removed at this step.

The first variable included was *Cumulative GPA*, as it maximized the distance between the two closest groups: 1.0 and 1.5. Next, *Percent of classes attended* was included, maximizing the distance between the 3.0 and 3.5 groups. It is interesting to note that the remaining steps maximized the distance between the 3.5 group and the 3.0 or 4.0 groups. *Attended helproom frequently* was included at step 6, maximizing the distance between the 3.0 and 3.5 groups. However, after including the *Computer communication* factor from the incoming survey data in step 11, which maximized the distance between the 3.0 and 3.5 groups, *Attended helproom frequently* was then removed in step 12 because doing so improved the separation between the 3.5 and 4.0 groups. After 14 steps, no further variables were included or removed because their F levels were below the required thresholds.

The significance tests of the resulting functions are shown in Table 37. The combinations of functions remain significant until only functions 6 through 7 and 7 alone are tested. Contrast these functions with the functions derived only on the incoming data (see Table 24). In this analysis, the Wilk's Lambda for the functions 1 – 7 is much lower, indicating more discrimination (SPSS, 1999b, p. 276), than in the first analysis and more functions remain significant.

Table 37

Significance of the Discriminant Functions

Test of Function(s)	Wilks' Lambda	Chi-square	df	Sig.
1 through 7	.272	3941.955	84	.000
2 through 7	.758	839.934	66	.000
3 through 7	.942	180.418	50	.000
4 through 7	.968	97.570	36	.000
5 through 7	.985	45.370	24	.005
6 through 7	.995	14.930	14	.383
7	.999	2.613	6	.856

Discriminant Function Validation

Validation was done in the same manner as the first analysis. The data used to derive the functions were first classified ($n = 3068$). The “hold out” sample was then classified for cross-validation ($n = 1355$). Cases that had

missing values on variables that were not included in the functions were used in the classifications. However, no cases with missing data on any of the variables used to derive the functions were used in the classification.

Equal prior probabilities were assigned to each grade for the classification. The separate group covariance matrices of the discriminant functions were used for classification because Box's M test of equality for the matrices was significant. The results for the original sample are shown in Table 38.

Table 38

Incoming and Classroom Data Classification Results for Cases Used to Derive the Discriminant Functions

Actual grade	n	Percentage classified by predicted grade							
		0.0	1.0	1.5	2.0	2.5	3.0	3.5	4.0
0.0	44	<u>100.0</u>	0.0	0.0	0.0	0.0	0.0	0.0	0.0
1.0	105	3.8	<u>82.9</u>	10.5	2.9	0.0	0.0	0.0	0.0
1.5	187	0.0	18.7	<u>56.1</u>	14.4	5.9	2.1	0.5	2.1
2.0	367	0.0	2.2	31.3	<u>30.5</u>	20.7	8.4	4.4	2.5
2.5	637	0.0	0.2	6.9	15.1	<u>39.2</u>	20.9	10.0	7.7
3.0	507	0.0	0.2	2.0	8.7	20.3	<u>37.7</u>	12.6	18.5
3.5	177	0.0	0.0	1.7	5.1	11.9	20.9	<u>31.6</u>	28.8
4.0	1044	0.0	0.0	1.3	5.0	8.6	13.4	20.7	<u>51.0</u>

Note: 44.9 % of the original grouped cases correctly classified.

The classification results for “hold out” sample are shown in Table 39. There is little shrinkage from the classification of the original data to the cross-validation sample (44.9% to 42.4%), indicating that upward bias is not a problem with the functions.

Table 39

Incoming and Classroom Data Classification Results for Unselected Cases

Actual grade	n	Percentage classified by predicted grade							
		0.0	1.0	1.5	2.0	2.5	3.0	3.5	4.0
0.0	18	<u>88.9</u>	11.1	0.0	0.0	0.0	0.0	0.0	0.0
1.0	49	4.1	<u>81.6</u>	12.2	0.0	2.0	0.0	0.0	0.0
1.5	90	0.0	17.8	<u>52.2</u>	15.6	12.2	2.2	0.0	0.0
2.0	163	0.0	6.1	27.0	<u>30.7</u>	20.9	12.9	1.2	1.2
2.5	277	0.0	0.4	4.7	18.4	<u>36.1</u>	21.3	10.5	8.7
3.0	225	0.0	0.0	3.6	10.2	20.9	<u>25.3</u>	19.1	20.9
3.5	82	0.0	0.0	2.4	7.3	17.1	30.5	<u>24.4</u>	18.3
4.0	451	0.0	0.0	1.6	4.7	8.6	15.3	15.5	<u>54.3</u>

Note: 42.4% of the unselected sample cases correctly classified.

This model under-classified 17.8% of the cases by only one grade and over-classified 11.1% by one grade, for an accuracy rate of 71.1%, plus or minus one grade. The only grade for which the predicted grade accuracy was not higher than all other predictions was the 3.5 grade, which had 24.4% correct, but had 30.5% predicted to receive a 3.0. Recall that separating the 3.5 grade from the 3.0 and the 4.0 grades was the focus of most of the variable selection during the function derivation and was inherent in the grading process as explained previously.

Interpretation of the Discriminant Functions

We again begin interpretation by examining the correlations between the variables and the rotated functions. Recall that we may perform an orthogonal rotation and maintain the classification accuracy. Table 40 shows the correlations between the variables used in the analysis and each of the rotated functions.



Table 40

Correlations Between Independent Variables and Rotated Functions

Variable	Function						
	1	2	3	4	5	6	7
Cumulative GPA	-.039	.923	.107	.143	-.053	.071	.036
ACT Social Science	-.095	-.086	.196	.387	-.372	.179	.127
ACT Mathematics	-.244	.091	.211	.512	.067	.081	.066
Computer communication	-.069	-.051	.118	.429	-.407	.026	.151
Computer terms	.072	.051	-.373	.076	.340	.099	-.015
Percent of classes attended	.146	.103	.725	-.084	.294	.013	-.184
Number of BT attempts	.982	-.025	-.041	-.076	.066	-.025	-.016
Did homework	-.107	-.053	-.080	-.611	-.098	.001	.113
Student Email Factor	.039	-.004	-.090	-.006	.130	-.007	.896
Track C	.044	.033	-.266	.168	.291	.019	-.341
Track D	-.023	-.002	.138	-.117	-.088	.876	.114
TA Experience	-.099	-.072	.068	.036	.633	-.005	.316

Recall that we can assign “meanings” to each of the functions based on the magnitude of the correlations. With 12 variables and seven functions, the interpretation becomes more complex. We can start with the functions that load strongly on individual variables. Function 2 is strongly correlated with *Cumulative GPA* and no other variables, so it mainly reflects the influence of *Cumulative GPA*. Function 6 correlates mainly with Track D ($r = .876$) with little other correlation with other variables. From here, we need to “blend” the interpretation.

Function 1 has the highest correlation ($r = .982$) with the *Number of BT attempts* but it also has a modest negative correlation ($r = -.244$) with *ACT Mathematics*. This implies that students with higher *ACT Mathematics* scores take somewhat fewer BTs to obtain the same grade.

Function 7 is strongly correlated with E-mail interactions between the student and the course instructors ($r = .896$). However, there is some interaction with the track in which the student is enrolled. Track C student interaction is somewhat less, all other things being equal. Function 7 also has some interaction with *TA experience*. It appears that function 7 reflects students who have more experienced TAs and who have more interaction with the instructors by E-mail. Perhaps more experienced TAs are able to recognize students who are having problems sooner and encourage them to seek assistance.

Function 3 has the strongest correlation ($r = .725$) with attendance, but there is also some interaction with incoming knowledge of abstract Computer terms ($r = -.373$). Due to the scale used, negative correlations reflect more knowledge of the terms. Function 3 also interacts somewhat with the tracks in

which the student is enrolled, reflecting student majors. The tracks also have slightly different attendance rates. Track A students ($M = 71.0\%$, $SD = 24.8\%$) attend slightly less than Track C ($M = 72.9\%$, $SD = 23.2\%$) or Track D ($M = 72.6\%$, $SD = 24.4\%$) students. The ANOVA of group means is significant ($F = 3.339$, $df = 2$, 5085, $p < .05$).

Function 5 is most strongly correlated with the TA experience ($r = .633$) but also has moderate correlations with several other variables. There are moderate negative correlations with the amount of experience that students report with *Computer communication* before the course ($r = -.407$) and their *ACT Social Science* scores ($r = -.372$). There are moderate correlations with less incoming knowledge of *Computer terms* ($r = .340$), attendance, and enrollment in *Track C* (again, possibly reflecting student majors.) Thus, function 5 appears to reflect the interaction of TA experience, incoming computing experience, attendance, major and ACT social science ability.

Finally, function 4 is most strongly correlated with doing homework ($r = -.611$). Due to the scale of the survey item, negative numbers reflect doing homework more frequently. There are also moderate correlations with *Computer communication*, *Cumulative GPA*, *ACT Social Science* and *ACT Mathematics* scores, indicating that students with higher GPAs and ability (as measured by the ACT tests) tend to do their homework more frequently.

We next examine the F-statistics to see how far these functions separate the groups. Table 41 shows the F-statistics for the differences among each pair of group centroids.

Table 41

F- Statistics Among Each Pair of Group Centroid Means

Group	0.0	1.0	1.5	2.0	2.5	3.0	3.5
1.0	10.642						
1.5	34.869	15.605					
2.0	71.061	57.483	18.074				
2.5	105.428	109.882	62.987	24.499			
3.0	127.278	143.393	96.504	55.052	15.777		
3.5	117.827	116.880	76.087	42.275	17.016	4.678	
4.0	143.601	173.130	143.642	120.676	79.276	28.668	4.987

Note: All *F* tests $p < .001$, *df* 12, 3019

The magnitude of the F-statistic indicates the distance between each pair of centroids. Recall that the classification accuracy for the 3.5 group was the lowest of any of the groups. The F-statistic is the smallest between the 3.0 and 3.5 and the 3.5 and 4.0 groups, indicating that the classification region for the 3.5 group is the smallest. Also note that, in contrast to the model based only on incoming data, this model is generally more ordinal. For example, the distance between the 4.0 and 3.0 group is larger than the distance between the 3.5 and 4.0 group. The distance from 4.0 to 2.5 is larger still, etc. The one exception is the distance from the 4.0 to the 0.0 group and the 3.5 and 0.0 group are smaller than the distance from the 4.0 or 3.5 to the 1.0 group. Students who received 0.0 grades often have personal, non-academic problems that impact their ability to

complete the course and receive 0.0's more frequently than they receive 1.0 grades. Other students receive 0.0 grades for academic dishonesty, which may not reflect their abilities or effort in the course. Table 42 shows the locations of the group centroids for each of the functions.

Table 42

Unstandardized Functions at Group Means

Course grade	Function						
	1	2	3	4	5	6	7
0.0	-4.306	-2.707	-1.832	-1.239	-.938	-.103	-.379
1.0	-2.857	-1.408	-1.698	-1.204	-.301	-.157	-.069
1.5	-1.372	-1.096	-1.111	-.805	-.388	-.298	-.226
2.0	-.261	-.766	-.665	-.552	-.225	-.174	.130
2.5	.337	-.343	-.216	-.093	.058	.244	-.137
3.0	.616	.024	.318	.212	.099	-.048	-.142
3.5	.445	.402	.647	.112	.196	.120	.181
4.0	.213	.845	.542	.443	.099	-.012	.139

The functions are ordered by the amount of variance for which they account. Function 1, associated with number of BT attempts, accounts for 36.8% of the variance. Function 2, associated with GPA, accounts for 28.2% of the variance. Function 3, attendance, accounts for 20.2% of the variance. Function

4, mainly reflecting homework, accounts for 10.7% of the variance. Functions 5, 6 and 7 account for 1.8%, 1.1% and 1.1% respectively. This is also reflected in the range of the group centroids, with function 1 having the largest range and function 7 the smallest. Cumulative GPA, which is now reflected in function 2, is much less important than it was in the incoming data model, where it accounted for over three-quarters of the variance. The variance for this model is more distributed among the functions. Figure 16 shows the plots of all seven of the function centroids. Remember that the functions are orthogonal; the centroids are plotted on the same graph only to show their relative distributions and magnitudes.

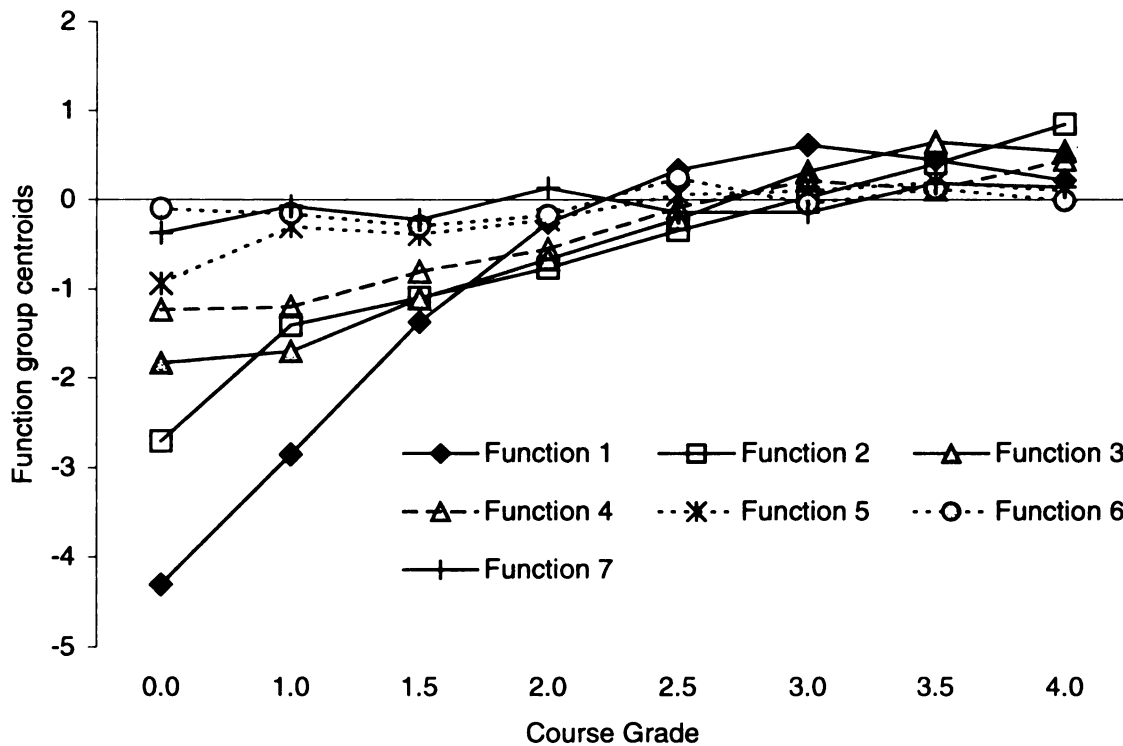


Figure 16 Group centroids for functions 1 through 7

Compare this figure to Figure 11 and Figure 12. The shape of function 2 (*Cumulative GPA*) is linear, much like function 1, which loaded on GPA in Figure 11. The overall centroids are approximately the same. Thus, the absolute contribution of *Cumulative GPA* remains the same. However, other factors now add additional discrimination. Function 1 (Number of BT attempts) is curvilinear, peaking at 3.0 and then declining somewhat. This reflects the fact that students who earn 3.5 and 4.0 grades actually complete their 3.0 BT in fewer attempts than do students who earn a 3.0. Functions 3 (Attendance) and 4 (Homework) are fairly linear, indicating that these functions are directly associated with student final grades in the course. The remaining functions – 5 through 7 – account for a small amount of the variance and have the “sawtooth” shape indicating that their main influence is via their interaction with the other functions.

Summary

This section presented the effects of adding classroom variables to the incoming student data model. We saw how the factors over which students have direct control – taking all bridge task opportunities, attending class, coming to class prepared and communicating with the instructors – have a significant impact on outcomes. The next section analyzes the assessments to see what computing concepts and skills – as measured by the bridge tasks – are the important predictors of outcomes.

Computing Concepts and Skill Variables

The third part of the instructional system is the assessment component. This section describes the variables that represent the computing concepts and skills that are evaluated by the bridge tasks. These variables are then added to the variables incoming student and classroom variables to generate discriminant functions. The results of this model are compared with the other models.

Bridge Tasks

Each bridge task consists of multiple dimensions, each of which is evaluated on a pass/fail basis. See Chapter 3 for details about the construction and evaluation of the bridge tasks. When a student fails a BT, s/he must repeat the entire BT. One metric that the instructors use to determine if the BTs are meeting their design goals is the bridge task repeat rate, defined as the number of times students must take any particular BT in order to pass it. Because the BTs are evaluated on a pass/fail basis, the cumulative pass rate should follow a logistic distribution (Hambleton, Swaminathan, & Rogers, 1991). To fall on the most linear portion of the logistic distribution, the initial pass rate for each BT should be between 30% and 70%, with optional rates between 40% and 60%. Rates much higher or lower than these would indicate that the BTs are not discriminating properly. Figure 17 shows the combined BT repeat rates for fall, 1998, spring, 1999 and fall, 1999.

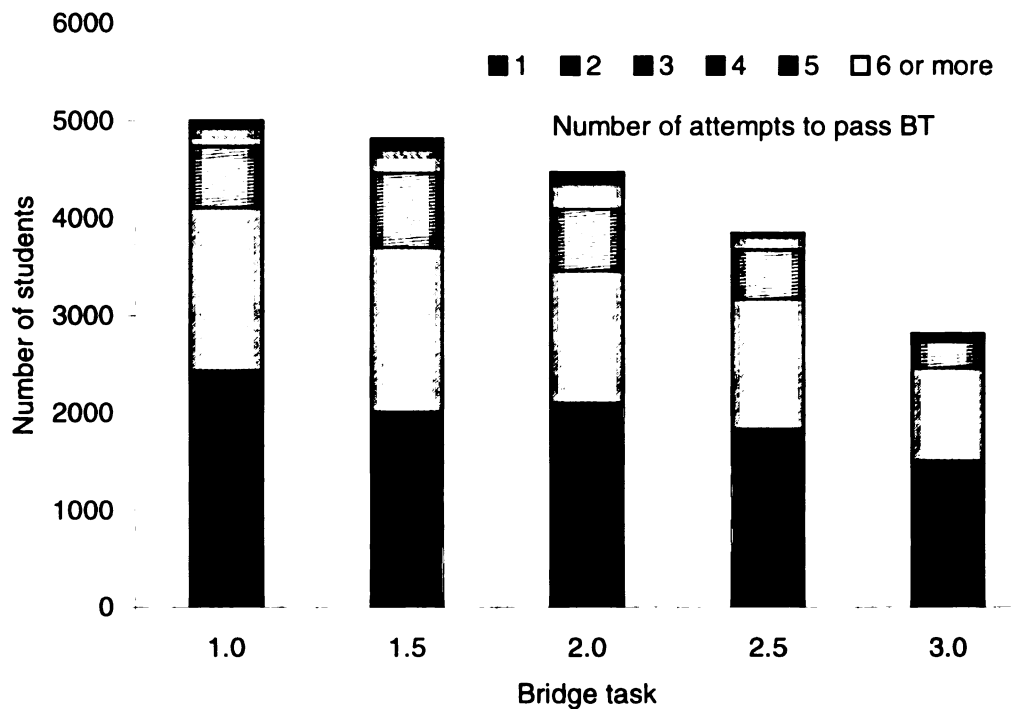


Figure 17 Bridge task repeat rates

About 49% of the students passed the 1.0 BT on the first attempt. Another 33% passed on the second attempt; 13% passed on the third try and a small number required 6 or more attempts. The initial pass rates on the other bridge tasks were: 1.5, 42%; 2.0, 47%; 2.5, 48%; and 3.0, 53%. These pass rates are very accurate measures of student performance, since inter-rater reliability on the BTs is high. Inter-rater reliability as measured by the grading error rate – defined as the number of BTs for which the grading was overturned on student appeal – was 2.08% for the 40,720 BTs administered during these three semesters. This error rate measures errors that caused students to fail a BT when they actually should have been passed. If we assume that grading

errors are randomly distributed, such errors would be equally likely to result in students passing dimensions when they actually should have failed those dimensions. Since students are more likely to appeal grading errors that change their BT results from fail to pass rather than from pass to fail, we can estimate that another 2% of the students were incorrectly passed when they should have been failed. Combining the known error rate for false fails and the estimated error rate for false passes produces an overall grading error rate of about 4%.

Usually, students pass some dimensions of the BT and fail others. For example, a student may pass all dimensions of the 1.5 BT except the specification of the URL for her or his Web page. The student must then repeat the entire 1.5 BT. This time, s/he may again fail the URL dimension and also fail the Boolean search dimension, again failing the 1.5 BT. Thus, s/he has failed the URL dimension twice and the Boolean search dimension once. These pass rates on each dimension provide data about the difficulty of each of the concepts or skills evaluated by each of the BT dimensions.

Table 43 shows an alphabetic list of the skills and concepts evaluated by dimensions on the 1.0, 1.5, 2.0 and 2.5 bridge tasks for fall, 1998, spring, 1999 and fall, 1999 semesters. The first four BTs (1.0, 1.5, 2.0 and 2.5) evaluate the basic computing concepts. The 3.0 BTs focus on more advanced application of these concepts and are unique to each track. Not all students pass the 2.5 BT and attempt the 3.0 BT. Therefore, including the 3.0 BT and partitioning the population by track would prevent us from contrasting the results with the

previous analyses in a meaningful manner, so only BT data through the 2.5 BT are included.

Only dimensions that were included in all three semesters are listed. The percentages of students who did not pass each dimension a particular number of times are listed in each column. If students never fail that dimension – however many times they take it – they are counted in the *0* column. If students do not pass a dimension one time, but then pass it on subsequent BTs – or never take the BT again – they are listed in the *1* column. Students who never took the BT associated with that dimension are listed in the *Did not take BT* column. For example, 63.1% of the students never failed the Boolean search dimension. Some may have passed the BT and moved on; others may have failed that BT for other reasons, but never failed the *Boolean search* dimension when repeating the BT. Another 25.8% of the students failed this dimension one time, with a small number failing two, three or more times. Only 2.9% of the students never took the BT on which this dimension appeared. Because spreadsheets appear on the 2.5 BT, and some students never get to the 2.5 BT, 17.2% of the students are listed in the *Did not take BT* column for dimensions associated with that BT.

Table 43

Bridge Task Skill and Concept Difficulty

Concept / Skill	Number of times did not pass dimension						Did not take BT
	0	1	2	3	4	5+	
Boolean search	63.1	25.8	6.4	1.3	0.4	0.1	2.9
Computer specifications	45.9	27.9	11.7	3.9	0.9	0.2	9.4
Create chart	59.3	19.4	3.4	0.6	0.0	0.0	17.2
Create link	85.3	10.0	1.5	0.2	0.0	0.0	2.9
Create private folder	82.0	9.5	1.4	0.2	0.0	0.0	6.9
Document margins	73.2	16.2	3.1	0.4	0.1	0.0	6.9
Extension task: backgrounds	76.5	14.2	2.0	0.4	0.0	0.0	6.9
Extension task: character styles	70.4	18.6	3.5	0.6	0.0	0.0	6.9
Extension task: Excel function	62.7	16.9	2.9	0.3	0.0	0.0	17.2
Find and rename file	80.6	16.1	2.4	0.5	0.1	0.0	0.4
Find new application	88.9	9.4	1.1	0.2	0.0	0.0	0.4
Footnotes in Word	75.7	14.3	2.4	0.6	0.0	0.0	6.9
Image on Web page	83.2	11.6	1.9	0.4	0.0	0.0	2.9
Modify styles in Word	65.7	19.8	5.1	1.7	0.6	0.1	6.9
New spreadsheet	61.0	18.5	2.5	0.7	0.0	0.0	17.2
Path to document	71.2	17.8	3.4	0.5	0.1	0.1	6.9
Path to own AFS	59.3	31.2	7.0	1.6	0.4	0.2	0.4

(table continues)

Table 43 (cont'd).

Concept / Skill	Number of times did not pass dimension						Did not take BT
	0	1	2	3	4	5+	
Public folder	78.5	16.5	3.5	0.8	0.3	0.1	0.4
Search for Web pages off-site	80.2	14.2	2.3	0.4	0.0	0.0	2.9
Table of Contents	60.7	24.1	6.5	1.5	0.3	0.0	6.9
Update payroll data	47.3	26.0	7.5	1.7	0.2	0.1	17.2
Web page text formatting	88.4	7.5	1.0	0.2	0.0	0.0	2.9
Web pages in Web folder	88.1	7.8	1.1	0.1	0.0	0.0	2.9
Web page URL	52.3	31.5	10.0	2.5	0.6	0.1	2.9

Note: All numbers are the percentage of students who failed the dimension listed in each row the number of listed in the column headings.

The dimensions range in difficulty from the *Computer specifications*, which only 45.9% of the students never fail, to the *Find new application* dimension, which 88.9% of the students do successfully each time they take that BT. Another metric for difficulty is the percentage of students who fail the dimension one time. Using this measure, understanding how to construct a *Web page URL*, with 31.5% of the students failing one time, is the most difficult.

Computing Concepts and Skills Model

The dimension pass rate statistics are approximately logistically distributed, as expected for criterion-referenced items that are evaluated on a

pass/fail basis. Since they are not normally distributed, we cannot use them as independent variables in discriminant analysis without transforming them.

Therefore, each of these pass rates was re-coded as a binary variable. If the student never failed the dimension, it was coded 0; all other students – including those who never took the BT – were coded 1. The transformed variables were used for the discriminant analysis.

The BT variables were added to the variables used in the instructional data model. All of the variables from the instructional data model – including those that were eliminated during that analysis – were again included in this analysis. This accounted for interactions among all of the variables. The analysis process paralleled the previous analyses.

Discriminant Function Derivation

The cases were divided into the same two groups used in the instructional model. The first group had 70% of the cases and was used to derive the discriminant functions ($n = 3038$). The second group had the remaining 30% of the cases and was used for the validation stage. Table 44 summarizes F values for the univariate ANOVAs on the group means for each of the BT variables entered into this analysis. The ANOVAs for the other variables are the same as they were in the instructional data model (see Table 35), since the same cases were analyzed. All ANOVAs were significant ($p < .001$).

Table 44

Tests of Equality of Group Means

Variable	Wilks' Lambda	F
Boolean Search	.857	72.440
Create chart	.666	216.828
Computer specifications	.801	107.326
Create link	.784	118.962
Create private folder	.644	239.599
Document margins	.717	170.695
Find and rename file	.924	35.483
Extension task: Excel function	.638	245.458
Extension task: character styles	.753	141.809
Extension task: backgrounds	.700	185.637
Find new application	.914	40.802
Footnotes in Word	.693	191.427
Image on Web page	.794	112.514
Modify styles in Word	.748	145.841
New spreadsheet	.617	268.809
Path to document	.734	156.658
Path to own AFS	.928	33.620
Public folder	.908	43.731
Search for Web pages off-site	.822	93.733

(table continues)

Table 44 (cont'd).

Variable	Wilks' Lambda	F
TOC	.780	121.972
Update payroll data	.756	139.862
Web page text formatting	.741	151.349
Web pages in Web folder	.757	139.001
Web page URL	.878	60.413

Note: All *F* tests $p < .001$, $df = 7, 3030$

Table 45 shows the order that the variables were included or removed during the analysis and which pair of group Mahalanobis distances were maximized by each of the variables.

Table 45

Stepwise Variable Inclusion and Removal Summary

Step *	Variable	Minimum D^2					
		Statistic	Between Groups	F	$df 1$	$df 2$	Sig.
1 E	Cumulative GPA	.114	1.0 – 1.5	7.525	1	3030	6.1E-03
2 E	Path to document	.188	3.0 – 3.5	12.234	2	3029	5.2E-06
3 E	Track D	.229	3.5 – 4.0	11.460	3	3028	1.9E-07
4 E	Percent of classes attended	.265	3.5 – 4.0	9.957	4	3027	5.6E-08
5 E	Search for Web pages off-site	.296	3.5 – 4.0	8.897	5	3026	2.3E-08
6 E	Did homework	.327	3.0 – 3.5	7.099	6	3025	1.7E-07
7 E	Number of BT attempts	.365	3.0 – 3.5	6.784	7	3024	5.6E-08
8 E	Footnotes in Word	.376	3.0 – 3.5	6.107	8	3023	8.4E-08
9 E	TA Experience	.386	3.5 – 4.0	6.440	9	3022	4.8E-09
10 E	Computer communication	.405	3.0 – 3.5	5.264	10	3021	1.0E-07

(table continues)

Table 45 (cont'd).

Step *	Variable	Statistic	Minimum D^2				
			Between Groups	F	$df 1$	$df 2$	Sig.
11 E	Find – rename file	.412	3.0 – 3.5	4.866	11	3020	1.8E-07
12 E	New spreadsheet	.425	3.5 – 4.0	5.311	12	3019	6.9E-09
13 E	Boolean Search	.433	3.0 – 3.5	4.325	13	3018	3.0E-07
14 E	Computer terms	.438	3.5 – 4.0	4.687	14	3017	1.7E-08
15 E	ACT Mathematics	.447	3.5 – 4.0	4.459	15	3016	2.3E-08
16 E	ACT Social Science	.459	3.0 – 3.5	3.720	16	3015	7.7E-07
17 E	Student E-mail	.470	3.5 – 4.0	4.132	17	3014	2.8E-08
	Factor						
18 E	Image on	.475	3.5 – 4.0	3.948	18	3013	4.2E-08
	Web page						
19 E	Modify styles	.478	3.5 – 4.0	3.762	19	3012	7.2E-08
	in Word						
20 E	Create link	.481	3.5 – 4.0	3.594	20	3011	1.2E-07
21 E	Web page	.487	3.5 – 4.0	3.461	21	3010	1.7E-07
	text formatting						
22 E	Extension task:	.489	3.5 – 4.0	3.317	22	3009	2.9E-07
	Excel function						
23 E	Update payroll	.492	3.5 – 4.0	3.192	23	3008	4.7E-07
	data						

(table continues)

Table 45 (cont'd).

Step *	Variable	Minimum D^2					Sig.
		Statistic	Between Groups	F	$df 1$	$df 2$	
24 E	Create private folder	.494	3.5 – 4.0	3.072	24	3007	7.6E-07
25 E	TOC	.497	3.5 – 4.0	2.963	25	3006	1.2E-06
26 E	Create chart	.498	3.5 – 4.0	2.859	26	3005	1.9E-06
27 E	Web page URL	.500	3.5 – 4.0	2.762	27	3004	3.0E-06
28 R	Image on Web page	.496	3.5 – 4.0	2.847	26	3005	2.1E-06
29 R	Path to document	.480	3.5 – 4.0	2.863	25	3006	2.8E-06
30 E	Path to own AFS	.481	3.5 – 4.0	2.759	26	3005	4.5E-06
31 R	TA Experience	.474	3.5 – 4.0	2.830	25	3006	3.6E-06
32 R	Computer communication	.455	3.5 – 4.0	2.829	24	3007	5.5E-06
33 E	Computer specifications	.457	3.5 – 4.0	2.727	25	3006	8.6E-06
34 E	Find new application	.458	3.5 – 4.0	2.626	26	3005	1.4E-05

(table continues)

Table 45 (cont'd).

Step *	Variable	Minimum D^2					
		Statistic	Between Groups	F	$df 1$	$df 2$	Sig.
35 E	Public folder	.458	3.5 – 4.0	2.528	27	3004	2.3E-05
36 E	Track C	.458	3.5 – 4.0	2.437	28	3003	3.8E-05

Note: At each step, the variable that maximizes the Mahalanobis distance between the two closest groups is entered. After 36 steps, the F level was insufficient for further inclusion or removal.

* E: variable entered at this step; R variable removed at this step

As in the previous models, the first variable included was *Cumulative GPA*, which maximized the distance between the two closest groups: 1.0 and 1.5. Next, a BT dimension that assesses understanding of hierarchical network file systems – *Path to document* – was included, maximizing the distance between the 3.0 and 3.5 groups. As in the instructional data model, the remaining steps maximized the distance between the 3.5 group and the 3.0 or 4.0 groups, indicating the difficulty discriminating the 3.5 group from the 3.0 and 4.0 groups. At step three, *Track D* was included, indicating that there is a difference among the tracks. Since the BTs through the 2.5 are common across tracks, this is likely due to different majors in the different tracks. Attendance is still a strong predictor of outcome as it was included in step four. Step five entered *Search for Web pages off-site*, a BT dimension that assesses student ability to use search engines and refine their searches. Student effort and participation in class as

measured by how often they reported doing their homework and the number of BT attempts are still important in this model, being included in steps six and seven. Step eight entered *Footnotes in Word*. This dimension assesses abstractions of text that are associated with one part of a word processing document, but displayed in another.

Four variables that were initially entered were eventually removed from the model. In step 28, the ability to correctly add an image to a Web page, a BT dimension that requires understanding of the relationship between the network file structure, the university's Web server and using a Web editor to insert the desired image, was removed. Next, in step 29, *Path to document*, which measured understanding of hierarchical file systems, was eliminated. It was replaced in step 30 by another measure of the same concept, *Path to own AFS*. In steps 31 and 32 *TA Experience* and student incoming experience using *Computer communication* were eliminated. They were replaced in step 33 by *Computer specifications*, a BT dimension that requires an understanding of the relationship between computer software specifications and hardware specifications so that students can select appropriate hardware to run particular software. After 36 steps, no further variables were included or removed because their *F* levels were below the required thresholds, leaving 28 variables in the final model.

The significance tests of the resulting functions are shown in Table 46. The combinations of functions remain significant until only function 7 is evaluated alone. Contrast these functions with the functions derived only from the incoming

data (see Table 24) or the functions based on the incoming and classroom data (see Table 37). The Wilk's Lambda for the functions 1 – 7 is much lower in this analysis (.070) than in the first (.534) or second (.272) analyses, indicating much more discrimination in this model (SPSS, 1999b, p. 276).

Table 46

Significance of the Discriminant Functions

Test of Function(s)	Wilks' Lambda	Chi-square	df	Sig.
1 through 7	.070	8007.535	196	.000
2 through 7	.584	1625.826	162	.000
3 through 7	.799	677.839	130	.000
4 through 7	.913	274.911	100	.000
5 through 7	.942	178.898	72	.000
6 through 7	.969	96.352	46	.000
7	.991	27.957	22	.177

Discriminant Function Validation

Validation was done in the same manner as the first two analyses. The data used to derive the functions were first classified ($n = 3068$). The “hold out” sample was then classified for cross-validation ($n = 1355$). The same cases that were classified in the instructional data model were classified using this model. Cases that had missing values on variables that were not included in the

functions were used in the classifications. However, no cases with missing data on any of the variables used to derive the functions were used in the classification.

Equal prior probabilities were assigned to each grade for the classification. The separate group covariance matrices of the discriminant functions were used for classification because Box's M test of equality for the matrices was significant. The results for the original sample are shown in Table 47

Table 47

Incoming Data, Classroom Data and Computing Concepts Classification Results for Cases Used to Derive the Discriminant Functions

Actual grade	n	Percentage classified by predicted grade							
		0.0	1.0	1.5	2.0	2.5	3.0	3.5	4.0
0.0	44	<u>97.7</u>	2.3	0.0	0.0	0.0	0.0	0.0	0.0
1.0	105	0.0	<u>95.2</u>	4.8	0.0	0.0	0.0	0.0	0.0
1.5	187	0.0	6.4	<u>82.4</u>	11.2	0.0	0.0	0.0	0.0
2.0	367	0.0	0.0	13.6	<u>71.4</u>	13.1	1.1	0.8	0.0
2.5	637	0.0	0.0	0.2	13.0	<u>54.6</u>	14.4	10.2	7.5
3.0	507	0.0	0.0	0.0	2.0	16.6	<u>38.7</u>	16.6	26.2
3.5	177	0.0	0.0	0.0	1.7	10.7	18.6	<u>34.5</u>	34.5
4.0	1044	0.0	0.0	0.0	1.0	8.2	16.4	18.1	<u>56.3</u>

Note: 57.1 % of the original grouped cases correctly classified.

The classification results for the “hold out” sample are shown in Table 48. There is little shrinkage from the classification of the original data to the cross-validation sample (57.1% to 56.5%), indicating that upward bias is not a problem with the functions.

Table 48

Incoming Data, Classroom Data and Computing Concepts Classification Results
for Unselected Cases

Actual grade	n	Percentage classified by predicted grade							
		0.0	1.0	1.5	2.0	2.5	3.0	3.5	4.0
0.0	18	<u>94.4</u>	5.6	0.0	0.0	0.0	0.0	0.0	0.0
1.0	49	0.0	<u>100.0</u>	0.0	0.0	0.0	0.0	0.0	0.0
1.5	90	0.0	7.8	<u>82.2</u>	10.0	0.0	0.0	0.0	0.0
2.0	163	0.0	0.0	14.1	<u>73.6</u>	10.4	1.2	0.6	0.0
2.5	277	0.0	0.0	0.0	14.4	<u>50.2</u>	16.2	11.9	7.2
3.0	225	0.0	0.0	0.0	1.8	17.3	<u>33.8</u>	22.2	24.9
3.5	82	0.0	0.0	0.0	3.7	9.8	25.6	<u>22.0</u>	39.0
4.0	451	0.0	0.0	0.0	0.9	7.5	14.6	16.6	<u>60.3</u>

Note: 56.5% of the unselected sample cases correctly classified.

This model significantly improved on the classification accuracy of the instructional model. It still under-classified 15.1% of the cases by only one grade and over-classified 11.4% by one grade for an overall accuracy rate of 83%, plus or minus one grade. Once again, the 3.5 group proved to be the most difficult to classify correctly, with only 22% of the 3.5's in the hold-out sample being correctly classified. However, the "off-diagonals" are very clean, with only 4.6% of the cases being mis-classified by more than two grades.

Interpretation of the Discriminant Functions

Table 49 shows the correlations between the variables and the rotated functions.

Table 49

Correlations Between Independent Variables and Rotated Functions

Variable	Function						
	1	2	3	4	5	6	7
Cumulative GPA	.025	-.841	-.116	.034	-.013	.078	.105
ACT Social Science	.074	-.051	-.232	-.039	-.202	.052	-.070
ACT Mathematics	.099	-.286	-.261	-.007	-.084	.089	-.098
Computer terms	-.045	-.050	.108	.140	.111	-.303	-.024
Percent of classes attended	-.156	-.038	.052	.059	.020	.603	.100
Number of BT attempts	-.498	.271	.379	.057	.018	-.009	.097
Did homework	.065	.178	.138	.013	.070	.014	.244
Student E-mail Factor	-.061	-.093	-.031	-.040	.013	-.054	.542
Track C	-.022	-.030	.072	.100	.092	-.224	-.257
Track D	.017	.037	.052	.422	-.108	-.017	.253
Boolean Search	.003	.244	.028	-.012	.152	-.042	.321
Create chart	.076	.074	.309	-.432	-.037	-.123	.074
Computer specifications	.008	.199	.340	-.062	.088	-.100	-.100
Create link	.150	.210	-.345	.139	.234	-.071	.265
Create private folder	.144	-.031	.212	.039	.814	.056	-.164
Find and rename file	.079	.219	.109	.083	-.071	.190	.035
Extension task: Excel function	.107	.022	.302	-.579	-.021	-.010	.067
Find new application	.194	.154	.017	.162	-.292	.184	-.020

(table continues)

Table 49 (cont'd).

Variable	Function						
	1	2	3	4	5	6	7
Footnotes in Word	.161	-.024	.382	.093	.436	.101	-.097
Modify styles in Word	.153	.003	.492	.156	.196	.076	-.195
New spreadsheet	.061	.114	.296	-.588	-.020	-.097	.130
Web page URL	.020	.158	.127	.207	.067	-.232	.226
Path to own AFS	.151	.146	.198	.264	-.228	.175	-.083
Public folder	.120	.162	.112	.130	-.095	.115	-.053
Search for Web pages off-site	.079	.209	-.247	.116	.238	-.108	.424
Table of contents	.092	.002	.264	.072	.126	-.287	-.024
Update payroll data	-.001	.158	.169	-.460	-.033	-.098	.243
Web page text formatting	.174	.261	-.555	.010	.295	-.109	.111

We can assign “meanings” to each of the functions based on the magnitude of the correlations between the variables and each function. With 28 variables and seven functions, the interpretation becomes even more complex than for the instructional model. Again, we can start with the functions that load strongly on individual variables.

Function 2 has the strongest correlation with *Cumulative GPA* ($r = -.841$) and also has a moderate ($r = -.286$) correlation with *ACT Mathematics*. In the previous models, these correlations were positive. However, the direction is

arbitrary as we will see in the plots of the function centroids. The BT dimensions which correlate most with this function are *Find and rename file* and *Public folder*. Both of these require understanding networked file structures and file access permissions. So function 2 is associated with academic ability and network file system permissions.

The next largest correlation is between *Create private folder* and function 5 ($r = .814$). This dimension also requires understanding network file structures and permissions, though it requires understanding how to prevent, rather than permit, other users from accessing files. There are two other hierarchical file system items that have moderate negative correlations with this function: *Find new application* ($r = -.292$), which requires searching for files and *Path to own AFS* ($r = -.228$), which requires understanding how to specify the path in a network file structure. *Footnotes in Word* has a moderate positive correlation ($r = .436$) with this function. This requires understanding abstractions of text in a word processing document. Function 5 is primarily associated with the concepts of hierarchical file systems, file system permissions and data abstractions.

Function 6 has the next largest overall correlation ($r = .603$) with attendance. It also has a moderate negative correlation with student incoming knowledge of abstract computer terms ($r = -.303$). The BT dimensions that are most strongly associated with this function are *Web page URL* ($r = -.232$) which requires understanding how to construct a URL to locate Web pages and constructing a *Table of contents* in Word ($r = -.287$). Tables of contents require understanding the abstractions of “styles” – collections of formatting commands –

and “fields” – which act like variables to hold data that will be updated and are required to generate a table of contents. The computing concepts associated with function 6 reflect an understanding of client server networks and data abstractions.

The next largest correlation is between *New spreadsheet* and function 4 ($r = -.588$). This BT dimension requires students to construct a spreadsheet to solve a problem for which they are given a verbal description. They must understand how to translate this problem into the necessary algorithm, then construct the spreadsheet correctly. This function is associated with several other spreadsheet dimensions. The dimension *Extension task: Excel function* also correlates most strongly with this function ($r = -.579$). This extension task requires correct use of new functions in a spreadsheet. Function 4 is also most strongly correlated with *Create a chart* in a spreadsheet ($r = -.432$), and with *Update payroll data* ($r = -.460$), which requires that students modify a spreadsheet they created in class. Function 4 also is associated with *Track D*, the spreadsheet track that uses spreadsheets for financial modeling. Function 4 reflects students' overall ability to use spreadsheets and shows that these concepts are different than other computing concepts since they cluster on a single discriminant function.

Function 3 has the next largest overall correlation with *Web page text formatting* ($r = -.555$). This BT dimension requires applying HTML styles to Web text. Function 3 is also associated with *Modify styles in Word* ($r = .492$), which requires a similar understanding of abstractions of formatting commands as

represented by “styles.” There is a moderate correlation between this function and *Create link* ($r = -.345$) on a Web page and understanding *Computer specifications* ($r = .340$), being able to match computer hardware and software specifications. Function 3 also has a moderate negative correlation with *ACT Social Science* ($r = -.232$). The computing concepts associated with function 3 are objects as collections of attributes and data representation.

Student E-mail Factor, the measure of E-mail communication between the student and the course instructors, is most strongly correlated with function 7 ($r = .542$). This function is also associated with how frequently the students do homework ($r = .244$) and *Track C* ($r = -.257$). On the BT dimensions, it correlates with *Boolean search* ($r = .321$) and *Search for Web pages off-site* ($r = .424$). Both dimensions require knowledge of Boolean operators and search strategies.

Finally, function 1 has the strongest correlation with only one variable, *Number of BT attempts* ($r = -.498$). No other variables have their strongest correlation with this function. Recall that the functions are ordered based on the amount of variance they account for and that the number of times students take BTs also loaded on function 1 in the instructional model. Student effort, as measured by the number of BT attempts, still accounts for most of the variance in this model.

We can now see how far these functions separate the groups. Table 50 shows the F-statistics for the differences among each pair of group centroids.

Table 50

F- Statistics Among Each Pair of Group Centroid Means

Group	0.0	1.0	1.5	2.0	2.5	3.0	3.5
1.0	12.802						
1.5	52.922	29.027					
2.0	113.199	106.958	36.526				
2.5	176.642	213.830	139.045	55.132			
3.0	215.014	274.512	204.588	111.525	21.976		
3.5	186.761	207.350	138.335	65.885	13.374	2.861	
4.0	227.981	307.121	251.643	164.845	54.045	12.960	2.437

Note: All *F* tests $p < .001$, *df* 28, 3003

The magnitude of the F-statistic indicates the distance between each pair of centroids. As in the previous models, the classification accuracy for the 3.5 group was the lowest of any of the groups. The F-statistic is once again smallest between the 3.0 and 3.5 and the 3.5 and 4.0 groups, making the classification region for that group the smallest. As with the instructional data model, this model is generally ordinal. Again, the exceptions are the distances to the 0.0 and 1.0. For final grades 2.0 through 4.0, the distances to the 0.0 group are actually closer than to the 1.0 group. As mentioned in the discussion of the instructional model, students who receive 0.0 grades often have personal, non-academic problems that impact their ability to complete the course or are

involved in academic dishonesty. Table 51 shows the locations of the group centroids for each of the functions.

Table 51

Unstandardized Functions at Group Means

Course grade	Function						
	1	2	3	4	5	6	7
0.0	10.156	3.390	.061	-.790	1.989	-1.138	.699
1.0	7.196	1.835	.991	-.899	2.251	-1.167	.868
1.5	4.299	1.242	1.582	-1.167	.586	-.759	.256
2.0	1.577	.881	.957	-.923	-.006	-.682	.516
2.5	-.335	.367	.212	.361	-.178	-.242	-.108
3.0	-1.507	-.075	-.296	.163	-.122	.278	-.380
3.5	-1.304	-.413	-.445	.466	-.161	.596	.118
4.0	-1.288	-.970	-.631	.275	-.210	.449	-.111

Function 1 accounts for 71.9% of the variance; function 2 accounts for 10.4% of the variance; function 3 accounts for 5.9% of the variance; function 4 accounts for 3.7% of the variance; functions 5 and 6 each account for 3.4% of the variance; and function 7 accounts for the remaining 1.1%. Figure 18 plots the centroids for function 1.

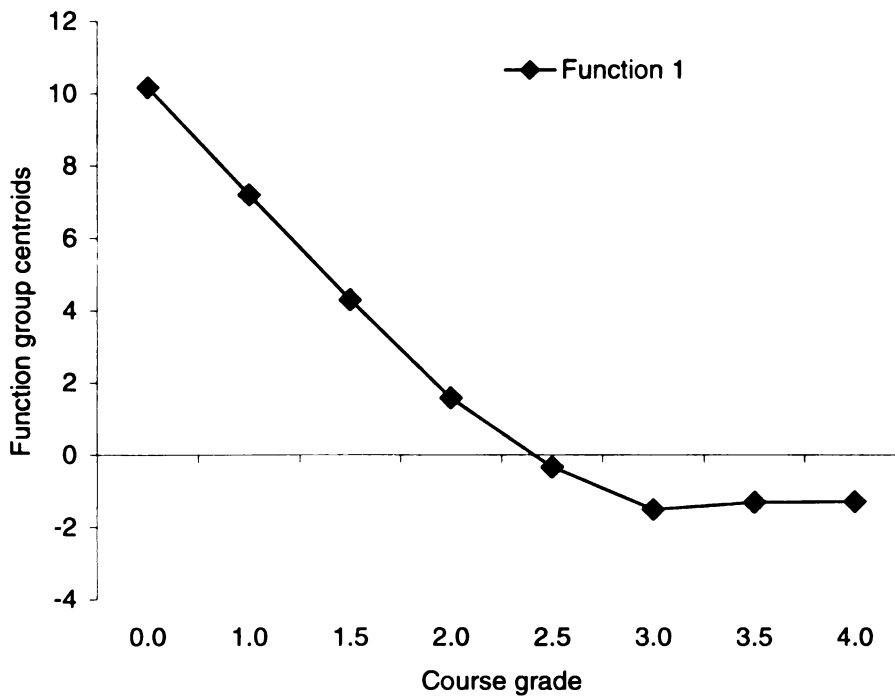


Figure 18 Group centroids for function 1

This figure shows the same relationship between number of BT attempts and final grades as the instructional model. The curve is reversed because the orientation of the axes in analytic space is arbitrary. Recall that the correlation between *Number of BT attempts* and function 1 was negative in this model and positive in the instructional model. Figure 19 shows the plots of the remaining six function centroids which have a smaller scale. Remember that the functions are orthogonal; the centroids are plotted on the same graph to show their relative distributions and magnitudes.

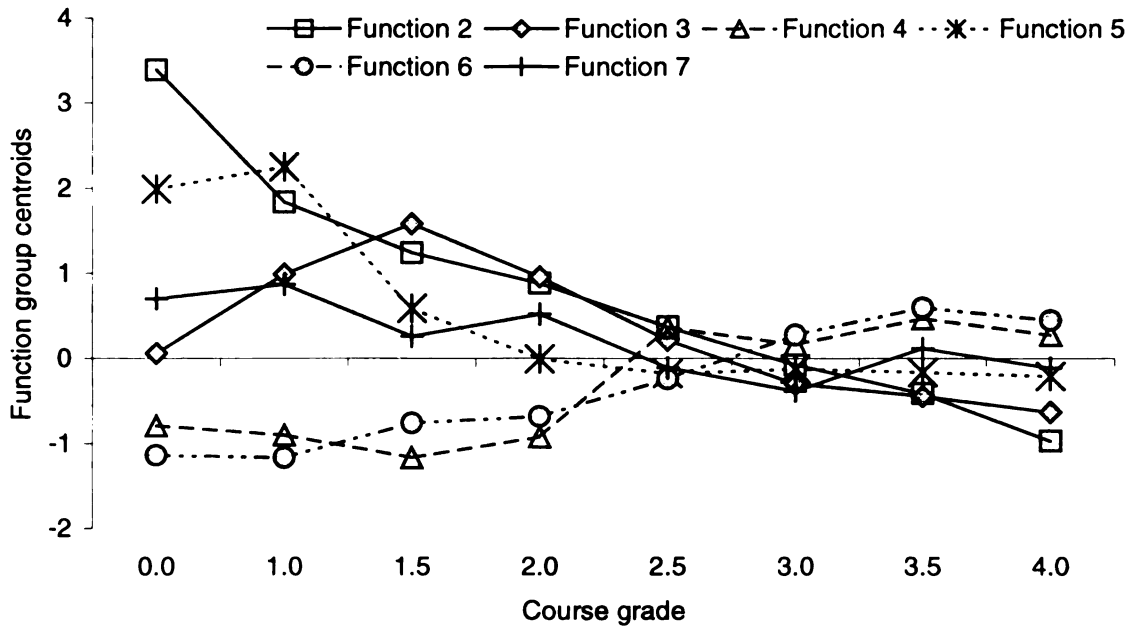


Figure 19 Group centroids for functions 2 through 7

Once again, incoming student ability, as reflected in function 2, has a near linear relationship with course grade. Function 3 has a curvilinear relationship, peaking at 1.5. This make sense, since the Web formatting dimensions are on the 1.5 BT. Likewise, function 4, which reflects spreadsheets, peaks at 2.5, the BT on which most of these dimensions are found. Function 5 has a peak at the 1.0 BT. However, although some of the hierarchical file system dimensions are found on the 1.0, others appear on the remaining BTs. It is important to remember that it is the constellation of these functions that provides the powerful classification accuracy. None of the functions alone can classify the outcomes as well as the combination of the functions.

Summary

This section presented the effects of adding the computing concepts and skills as measured by the various bridge task dimensions to the instructional data model. Several of these concepts cluster in the same manner as the instructors organized them. Although many of the dimensions that evaluate these concepts appear on the same bridge tasks, the pass rates that are measured by these dimensions are independent of each other. The instructors grouped these items together based upon their conceptual similarity, and it is encouraging that this analysis of the dimensions that measure the concepts clusters them in a similar fashion. We also saw how the factors over which students have direct control – taking all bridge task opportunities, attending class, coming to class prepared and communicating with the instructors – still have a significant impact on outcomes. This indicates that students with less academic ability who may be struggling with these concepts can still master them by taking advantage of the opportunities that the course structure provides, such as repeating failed BTs, preparing for and attending class and seeking assistance from the course instructors.

Student Evaluations

Towards the end of each semester, students complete surveys in which they rate various aspects of their experience in the course and complete the Student Instructional Ratings (SIRS) for their teaching assistants. Students are allowed to complete the survey before the first opportunity to take the 3.0 BT, but they may complete it at any time through the end of the course. The analysis

began by using factor analysis to extract factors for the course survey questions, the TA SIRS and the ATA SIRS. These factors were saved as variables for additional analysis. To understand the relationship among the student ratings of the course and their ratings of their TAs, several regressions were performed using the variables that were significant from the incoming student, classroom and computing concept models. Student comments from the course evaluations and SIRS were sampled to illuminate the quantitative data.

Student Course Evaluations

The surveys use five point Likert scales (Strongly Agree; Agree; Neither Agree or Disagree; Disagree; Strongly Disagree) for each question. (See Appendix D for the complete questions.) Students also may enter free response comments to each of the questions to elaborate on their responses. Table 52 shows the percentage of students who selected each of the responses to the questions.

Table 52

Student Responses to Course Survey Questions

Question	<i>n</i>	Strongly Agree	Agree	Neither Agree or Disagree	Disagree	Strongly Disagree
Textbooks helpful	3994	4.3	30.7	24.8	26.8	13.4
Web helpful	3992	45.7	46.2	6.1	1.4	0.7
Notes helpful	3788	23.0	45.1	24.6	5.9	1.4
Did homework	3797	19.7	45.5	16.6	14.6	3.6
Learned in group exercises	3831	8.1	32.1	26.4	22.7	10.7
Attended help room frequently	3516	3.2	10.5	16.2	30.8	39.3
Helproom useful	2598	8.5	21.1	50.4	9.5	10.6
BTs fair	3847	10.1	49.5	14.6	16.5	9.3
Extension tasks connected	3590	6.9	45.0	23.5	18.1	6.6
BT grading fair	3782	6.0	34.8	16.5	27.3	15.4
Graders comments explained	3725	11.5	47.9	16.9	17.5	6.1

(table continues)

Table 52 (cont'd).

Question	<i>n</i>	Strongly Agree	Agree	Neither Agree or Disagree	Disagree	Strongly Disagree
Grade reflects my understanding	3719	11.1	39.3	15.9	20.6	13.1
Recommend course to friends	3634	11.0	36.4	23.9	15.4	13.3

Note: responses are percentages of all students answering that question.

Factor analysis was used to reduce the dimensionality of these questions. When all of the questions were used in the factor analysis it produced four factors. The fourth factor loaded only on the questions about the help room. Since that data had already been incorporated into the previous models, those questions were eliminated. The remaining variables produced three factors accounting for 56.0% of the variance before rotation. Table 53 shows the correlations for each variable with the factors after oblique rotation.

Table 53

Correlations Between Survey Questions and Rotated Factors

Survey question	Component		
	1	2	3
Textbooks helpful	-.035	.582	-.308
Web helpful	.142	.064	-.710
Notes helpful	.013	.015	-.824
Did homework	.114	.644	.203
Learned in group exercises	-.021	.746	-.093
BTs fair	.830	-.010	-.006
Extension tasks connected	.682	-.035	-.075
BT grading fair	.838	-.002	.102
Graders comments explained	.520	-.115	-.184
Grade reflects my understanding	.760	.081	.084
Recommend course to friends	.638	.256	-.057

Factor 1 loads primarily on questions about student attitudes towards the assessments and if they would recommend the course to their friends; it is a measure of perceived “fairness.” Factor 2 loads primarily on how much the students thought they learned in the group exercises, if they did their homework and if they found the textbook helpful. It seems to reflect student preparation and participation in the course. Factor 3 loads on questions about the course Web

site and the daily summaries of class that appear on the Web site. It also has a moderate correlation with the textbook question, the homework question and a small correlation with the summative feedback students receive on the bridge tasks. This factor seems to reflect the student perceptions of course resources. All three factors were saved as variables for subsequent analysis.

Student Evaluations of Their Teaching Assistants

Students complete an expanded set of the standard university Student Instructional Rating System (SIRS) teaching evaluations for their Teaching Assistants. The averaged responses for the TAs were used in the analysis of the classroom model. This analysis examines the individual student ratings of their TAs. Table 54 summarizes the responses to the Likert scale items.

Table 54

Student Instructional Rating System Responses for Their Lead TAs

Question	<i>n</i>	Strongly Agree	Agree	Neither Agree or Disagree	Disagree	Strongly Disagree
Willing to help	3659	42.6	42.1	9.8	4.1	1.4
Explained clearly	3647	26.0	40.9	15.8	13.1	4.1
Prepared and enthusiastic	3601	32.5	43.0	15.9	6.8	1.7
Organized and competent	3582	34.4	44.4	12.8	6.4	2.0
Communicated well	3556	24.0	35.5	17.4	15.7	7.4
Accessible and prompt	2766	29.5	30.1	37.1	2.2	1.1

email response

Note: responses are percentages of all students answering that question.

Students are also asked to “grade” their TAs on a scale of 4.0, 3.0, 2.0, 1.0 and 0.0. Table 55 summarizes their responses.

Table 55

Lead Teaching Assistant Grades Assigned by Students

<i>n</i>	4.0	3.0	2.0	1.0	0.0
3641	38.5	40.3	15.5	4.3	1.4

Note: responses are percentages of all students answering that question.

The various student ratings of their TAs are highly correlated with each other. The question about the TA responding promptly to E-mail is less correlated than the rest. It also had a much lower response rate than the other questions. The student free responses to this item show that many students never sent E-mail to their TAs, so did not know if the TA responded quickly or not. Hence, 37.5% selected the *Neither Agree or Disagree* response to this question. Factor analysis of all SIRS questions except for the E-mail question resulted in a single factor that accounted for 75.3% of the variance of the remaining questions. This factor was saved for subsequent analyses. Table 56 shows the factor loadings.

Table 56

Lead Teaching Assistant SIRS Factor Loadings

SIRS question	Component1
Willing to help	.824
Explained clearly	.895
Prepared and enthusiastic	.851
Organized and competent	.874
Communicated well	.855
Grade the TA	.904

Most students rate their Teaching Assistants highly, with almost 80% giving them a grade of 3.0 or 4.0. Examining the student comments about their TAs reveals that the primary negative comments focus on the TA's communication skills. The majority of the TAs are international students for whom English is their second language. Over 90% of the students in the course are domestic, and many feel that they cannot understand their TAs.

Student Evaluations of Their Assistant Teaching Assistants

Students also complete an expanded set of the SIRS teaching evaluations for their assistant Teaching Assistants (ATA). Table 57 summarizes the responses to the Likert scale items.

Table 57

Student Instructional Rating System Responses for Assistant TAs

Question	<i>n</i>	Strongly Agree	Agree	Neither Agree or Disagree	Disagree	Strongly Disagree
Available to help student	3584	48.1	39.3	7.3	3.7	1.6
Explained clearly	3473	32.5	37.6	18.1	8.3	3.5
Prepared and enthusiastic	3447	31.7	38.8	19.8	6.5	3.1
Organized and competent	3446	34.1	40.4	16.1	6.6	2.8
Communicated well	3397	31.1	35.1	17.5	11.3	5.0

Note: responses are percentages of all students answering that question.

Students are also asked to “grade” their Assistant TAs on a scale of 4.0, 3.0, 2.0, 1.0 and 0.0. Table 58 summarizes their responses.

Table 58

Assistant Teaching Assistant Grades Assigned by Students

<i>n</i>	4.0	3.0	2.0	1.0	0.0
3533	43.1	35.4	14.1	5.3	2.1

Note: responses are percentages of all students answering that question.

All student ratings of their ATAs are highly correlated with each other. Factor analysis of all of the ATA SIRS questions resulted in a single factor that accounted for 79.2% of the variance of the questions. This factor was saved for subsequent analysis. Table 59 shows the factor loadings.

Table 59

Assistant Teaching Assistant SIRS Factor Loadings

SIRS question	Component 1
Available to help student	.841
Explained clearly	.908
Prepared and enthusiastic	.893
Organized and competent	.902
Communicated well	.878
Grade the ATA	.917

Students also rate their ATAs highly, with almost 80% grading them 3.0 or 4.0. The ATAs are generally domestic undergraduate students. It is interesting to note that the students rate the lead TA knowledge somewhat higher than they rate the knowledge of their ATAs. However, they rate the communication skills of their ATAs somewhat higher than they rate their lead TAs' communication. The comments reflect the ratings. Often the students will comment that their TA is

more knowledgeable than the ATA, but that they feel the ATA is better able to communicate with them.

Analysis of the Student Evaluations

To understand the important predictors of the student evaluations, the three factors from the course evaluation – the “fairness” factor, the “preparation and participation” factor and the “course resources” factor – and the two factors from the lead TA and ATA SIRS were analyzed with stepwise regression analysis. Each factor was used as the dependent variable for a separate analysis. The students’ final grade in the course and all of the variables that were significant from the incoming student model, the classroom model and the computing concepts and skills model were used as independent variables. The average lead TA SIRS variable was not included, since the individual student SIRS were used in these analyses. The TA and ATA SIRS factors were also included each of the analyses of the three course factors. Likewise, the ATA SIRS factor was included in the TA model and the TA SIRS factor was included in the ATA model.

The Course “Fairness” Factor

The “fairness” factor was the first factor in the course survey analysis. It loaded primarily on the questions about whether or not the students felt the BTs were a fair test of what they learned, if the grading was fair and the graders’ comments were helpful, if they felt that their final course grade would reflect their understanding of the computing concepts and if they would recommend the

course to their friends. The grade the student received in the course, all of the significant variables from the previous models and the TA and ATA SIRS factors were used as independent variables. The variables were added to the model in a stepwise manner. At each step, the variable that maximized the R-square for the model was added until no more variables resulted in a significant change of the R-square. Table 60 shows the model summary as each variable was added to the model. The final model included a constant and all variables listed in the table; no other variables were significant.

Table 60

Summary of Regression Model for the Course “Fairness” Factor

Model	Beta	R	Adjusted R Square	F	df 1	df 2	Sig.
Course grade	-.369	.394	.155	369.999	1	2015	.000
TA SIRS	.269	.500	.249	253.549	1	2014	.000
Number of BT attempts	.262	.573	.328	237.145	1	2013	.000
ATA SIRS	.131	.587	.343	47.782	1	2012	.000
Cumulative GPA	.100	.590	.346	11.409	1	2011	.001
ACT Mathematics	-.064	.593	.349	10.667	1	2010	.001
Computer specifications	.056	.595	.351	7.069	1	2009	.008
Computer communication	.045	.596	.353	5.954	1	2008	.015

Note: Variables are listed in the order they were included in the stepwise analysis. Each F-test is for the model with the variables included at that step. No additional variables were included because they were not significant.

Not surprisingly, the strongest predictor of student ratings of “fairness” is their final grade, accounting for 15.5% of the variance. (The beta for the grade is negative because the scales for the surveys are scored with zero as “strongly agree.”) Even though students complete these surveys before they have completed the course, most have a good idea of what their final grade will be. The TA SIRS factor was the second variable included in the model, accounting for an additional 9.4% of the variance. The number of BT attempts was the third variable included, accounting for another 7.9% of the variance. This beta is

positive, indicating that the more times students take BTs, the less fair they think they are. The ATA SIRS account for an additional 1.5% of the variance.

The remaining variables account for a small amount of the variance. Only one computing concept variable that enters this model: the computer specifications dimension. It accounts for only 0.1% of the variance. These eight variables account for 35.3% of the total variance in the students' "fairness" ratings.

The "Preparation and Participation" Factor

The second factor extracted from the course survey loaded primarily on the questions about learning in the group exercises and doing the homework before class. It also loaded somewhat on the question about the usefulness of the textbook and if the students would recommend the course to their friends. The partial loading on the textbook question was interpreted as a measure of how much the students actually studied the textbook, since this question also loaded on the "course resources" factor. The partial loading on the "recommend the course to my friends" question was interpreted as reflecting how much the students enjoyed the classroom experience. The grade the student received in the course, all of the significant variables from the previous models and the TA and ATA SIRS factors were used as independent variables. Table 61 shows the model summary as each variable was added to the model. The final model included a constant and all variables listed in the table; no other variables were significant.

Table 61

Summary of Regression Model for the Course "Preparation and Participation"Factor

Model	Beta	R	Adjusted R Square	F	df 1	df 2	Sig.
TA SIRS	.289	.353	.124	286.930	1	2015	.000
Attendance	-.132	.388	.150	61.460	1	2014	.000
ATA SIRS	.124	.408	.166	39.320	1	2013	.000
ACT Social Science	.120	.428	.181	39.383	1	2012	.000
Number BT attempts	.079	.434	.186	12.788	1	2011	.000
Create chart	.059	.437	.188	6.529	1	2010	.011
Computer terms	-.053	.440	.191	6.823	1	2009	.009
ACT Mathematics	.059	.442	.193	5.996	1	2008	.014
Find and rename file	.059	.445	.194	5.280	1	2007	.022
Path to own AFS	-.057	.448	.196	6.285	1	2006	.012
TOC	.045	.450	.198	4.366	1	2005	.037

Note: Variables are listed in the order they were included in the stepwise analysis. Each F-test is for the model with the variables included at that step. No additional variables were included because they were not significant.

This model accounts for much less of the variance in the “participation and preparation” factor (19.8%) than the previous model accounted for the “fairness” factor. However, different variables are significant in this model. The TA SIRS ratings are the most important, accounting for 12.4% of the variance. Attendance is the next variable and accounts for an additional 2.6% of the variance. (The beta for attendance is negative because the scales for the surveys are scored with zero as “strongly agree.”) The ATA SIRS account for another 1.6% of the variance. It is interesting that the final course grade does not enter into this model, although the number of BT attempts does. There are also some computing concepts: creating spreadsheet charts, finding and renaming a file in a hierarchical file system, network file system paths and creating a table of contents. These concepts are spread throughout the course and do require understanding abstractions rather than simply skills.

The “Course Resources” Factor

The final factor extracted from the course survey loaded primarily on the questions about course resources: the course Web site, summaries of the classes and the textbook. It also was moderately loaded on the question about doing homework and the clarity of the graders’ comments on the BT grading. The grade the student received in the course, all of the significant variables from the previous models and the TA and ATA SIRS factors were used as independent variables. Table 62 shows the model summary as each variable was added to the model. The final model included a constant and all variables listed in the table; no other variables were significant.

Table 62

Summary of Regression Model for the “Course Resources” Factor

Model	Beta	R	Adjusted R Square	F	df 1	df 2	Sig.
TA SIRS	-.246	.260	.067	146.510	1	2015	.000
Attendance	-.180	.296	.087	43.546	1	2014	.000
ATA SIRS	-.144	.329	.107	46.412	1	2013	.000
Extension task: backgrounds	-.061	.335	.110	8.734	1	2012	.003
Web pages in Web folder	-.047	.337	.112	4.433	1	2011	.035
Number of BT attempts	.047	.341	.113	4.821	1	2010	.028

Note: Variables are listed in the order they were included in the stepwise

analysis. Each F-test is for the model with the variables included at that step.

No additional variables were included because they were not significant.

The “course resources” model accounts for the least variance (11.3%) of the three course factor models. This makes sense, since the factors are orthogonal, and each subsequent factor accounts for less of the variance in the course survey questions. Again, the TA SIRS, attendance and the ATA SIRS have the strongest relationships. The grades the students received in the course do not enter this model. What is interesting is that the betas for all variables except the number of BT attempts are negative, indicating that students who rate their TAs lower and attend class less perceive the course resources as more

useful, perhaps using them to compensate for their lack of classroom participation.

The Teaching Assistant SIRS

To understand the factors that influence student ratings of their teaching assistants, the TA SIRS factor was used as the dependent variable in a regression analysis. The grade the student received in the course, all of the significant variables from the previous models, the three course survey factors and the ATA SIRS factors were used as independent variables. The final model included a constant and all variables listed in the table; no other variables were significant. Table 63 summarizes the regression results.

Table 63

Summary of Regression Model for the Teaching Assistant Ratings

Model	Beta	R	Adjusted R Square	F	df 1	df 2	Sig.
Fairness factor	.224	.383	.146	345.969	1	2015	.000
Preparation and participation factor	.199	.456	.207	155.196	1	2014	.000
TA Experience	-.201	.493	.242	94.339	1	2013	.000
Course resources factor	-.136	.509	.258	44.681	1	2012	.000
ATA SIRS	.095	.518	.267	24.229	1	2011	.000
Attendance	-.074	.523	.272	15.705	1	2010	.000
Private folder	-.062	.526	.274	7.410	1	2009	.007
Extension task: Excel function	.058	.529	.277	8.714	1	2008	.003
Number of BT attempts	-.040	.530	.278	4.052	1	2007	.044

Note: Variables are listed in the order they were included in the stepwise

analysis. Each F-test is for the model with the variables included at that step.

No additional variables were included because they were not significant.

This model accounts for 27.8% of the variance in the student ratings of their TA. The most prominent variable in the model is the course “fairness” factor, accounting for 14.6% of the variance. The student “participation and preparation” factor is the second most important variable followed by the TA experience. The “course resources” factor has a negative beta, consistent with the analysis of the “course resources” factor in which the TA SIRS had a negative beta. The students’ ratings of their ATA and student attendance also have some relationship to their rating of the lead TA. There are two computing concepts that are related to the rating of the TA: understanding folder permissions in a networked file system, which comes early in the course, and an extension task involving spreadsheet functions, which comes midway through the course. Finally, the number of BT attempts has a small relationship to the student ratings of their Teaching Assistant. However, the student grade in the course is not included in the model; the grade students anticipate, while significant in their rating of the course, is not a factor of their ratings of their TAs.

The Assistant Teaching Assistant SIRS

A similar analysis of the Assistant TA SIRS factor was used as the dependent variable in a regression analysis. The grade the student received in the course, all of the significant variables from the previous models, the three course survey factors and the TA SIRS factors were used as independent variables. Table 64 shows the model summary as each variable was added to the model. The final model included a constant and all the variables listed in the table; no other variables were significant.

Table 64

Summary of Regression Model for the Assistant Teaching Assistant Ratings

Model	Beta	R	Adjusted R Square	F	df 1	df 2	Sig.
Fairness factor	.132	.260	.067	145.764	1	2015	.000
Preparation and participation factor	.101	.299	.089	48.607	1	2014	.000
Student E-mail factor	-.123	.320	.101	29.224	1	2013	.000
TA SIRS	.115	.339	.113	28.418	1	2012	.000
Course resources factor	-.108	.349	.120	16.392	1	2011	.000
Attendance	-.088	.360	.127	17.128	1	2010	.000
Path to own AFS	.080	.366	.131	10.878	1	2009	.001
TA Experience	.028	.371	.134	8.301	1	2008	.004

Note: Variables are listed in the order they were included in the stepwise analysis. Each F-test is for the model with the variables included at that step. No additional variables were included because they were not significant.

Analysis of the assistant TA SIRS accounts for half as much of the variance (13.4% total) as the analysis of the lead TA SIRS. However, the “fairness” and “preparation and participation” factors were again the top two variables in this model, as they were in the lead TA model. The factor derived for E-mail communication with the instructor is the third variable, with a negative beta. Students who rate their ATA highly have less communication with the course instructors. Just as the ATA SIRS was a significant variable in the lead TA SIRS, the lead TA SIRS are significant predictors of the ATA SIRS. The lead and assistant TAs must work together as a team with the lead mentoring the ATA. More experienced leads probably do a better job of this than their less experienced colleagues. Again, student grade in the course is not related to their ratings of their assistant TAs.

Summary of the Student Evaluations

As with many large, required undergraduate courses, student responses to the course are widely distributed. This section first discusses the student course ratings. The student SIRS ratings of their TAs and ATAs are discussed after that.

Student Course Evaluations

Not surprisingly, the assessment method (how the students' grades are determined) is the part of the course that draws the most negative student response. Student perceptions of the “fairness” of the assessments are related to their grades: students who do poorly in the course rate the assessments lower

than students who do well. However, student comments about the assessments reveal how traumatic changing the “grading game” can be for all students, strong as well as weak ones. A student from fall, 1999 expresses this frustration that “This class should not be different than any others in the sense that if you miss one thing you completely fail. It becomes frustrating to the students which results in more failed BTs.” Another student from fall, 1999 wrote that “I hate the bridge tasks. One stupid little mistake held me up so many times. Just give me a 90% and let me take the next one.”

One reason that students resist this new assessment is that they have learned to succeed in the current educational assessment system. As a fall, 1998 student wrote:

I have learned alot [sic] about computers and have learned things that I would have never learned, but I would not recommend this class to my worst enemy after all the stress that I have been under over these bridge tasks. I have explained the way of grading to other professors and they agree that it is ridiculous.

Throughout their schooling, faculty have imbued students with an overwhelming expectation that grades are an accumulation of points that are redeemed at the end of the course for a final grade; this course violates that assumption.

The “extension tasks” are another part of the bridge tasks that some students perceive as unfair. A student from spring, 1999 says that “I think for the most part they covered material that had been discussed in class but sometimes included things that were not even mentioned and I don't feel that was fair at all.”

A fall, 1998 student complains that “There was always something more expected than what we learned in class. Meaning, some of the mandatory questions had us doing things we never talked about in class.” On the other hand, a student from fall, 1999 who rated the extension task question as “strongly disagree” still acknowledged having the conceptual framework to solve the problem. “There were things on my BT that we never learned about in class, luckily I know enough so I was able to pass.” Other students do not have problems with the extension tasks. A spring, 1999 student says “They were good in that they were made to cover all topics. They also made us learn new things but the new stuff was not difficult to learn, it just made sure that we knew how to use the Help System.”

Some students do understand that the bridge tasks are more authentic assessments than more traditional tests. A student from spring, 1999 wrote:

I am tired of hearing students complain about not doing what was asked on the BT. If they would have done it correctly, a passing grade would be given. Not only are you asked to perform computer concepts, but also follow specific directions in finishing the BT properly. Students will not be able to whine later in life when they demonstrate the skill, but did not follow all directions to the “T”.

Because of the unique nature of the assessments, the instructors have made communicating how the grading system works a prominent part of the course syllabus and culture. In the time since the course has was first offered, almost 10,000 students have taken the course; the course’s reputation has now

permeated the undergraduate culture. During the first day of class each semester, students use the course Web site to locate the usual information that instructors put into a course syllabus. Part of that day is spent discussing how the course works in small groups. Students are asked to provide comments or questions that they have about the course as a result of talking with their friends. Students commonly report that they hear that it is important to come to class and not to fall behind on the bridge tasks. However, in spite of the fact that awareness of the assessments has spread through the student culture, there is always apprehension about the grading system on the first day of class. It is the focus of most of their questions on day one and remains the focus of their concerns at the end of the course.

Aside from the grading system, students do appreciate most other aspects of the course. A fall, 1998 student found the bridge tasks to be formative, not just summative, and a way to enhance problem solving skills. "I enjoyed the challenge of bridge tasks. I think the class has taught me to be self-teaching, so when new problems arise, I have the tools to figure out the problem." Another fall, 1998 student also enjoyed the problem solving aspects of the course "it taught me the tools I need to become a 'Problem Solver.'"

The course is different from other courses students take, and also different from most other CS0 courses. The conceptual orientation of the course does provide more challenge than the skills only approach to teaching CS0. A fall, 1998 student wrote about the course:

I liked it very much. I have a friend at [a community college] who took the equivalent course to my own and hers compared to mine is a snap. They didn't learn HALF the things that we did in our class. This class was MUCH BETTER.

A fall, 1999 student also recognized the unique approach of the course. "I know that computer skills are very important today. What I really appreciated from the course was that it emphasized the importance of being able to learn new things."

Student Ratings of the Teaching Assistants

Students' ratings of teaching assistants' teaching ability are related to student ratings of the course. The TA is the only person that most of the students see, since fewer than 5% of the students ever come to instructor office hours. Because the analyses of the student surveys and SIRS are based on correlations, it is not possible to prove a causal relationship. However, the course instructional design is consistent – all TAs have the same detailed daily lesson plans – and all students take the same assessments. The TA and ATA SIRS are prominent in the "participation and preparation" factor and the "course resources" factor. They are also significant in the course "fairness" factor, along with the final grade and the number of BT attempts. Therefore, it is likely that the variance in the course ratings is a result of the different TAs, rather than the variance in TA ratings being a function of the course content and assessments.

However, although TA experience has some impact in the instructional model, the teaching assistants' teaching abilities – as rated by the students – are not related to student outcomes in the instructional model or the computing concepts model. This is important, since fiscal constraints make the use of teaching assistants the only option available to maintain small ($n = 30$) classes with live instructor/student interaction. The course design is robust enough that a weak teaching assistant does not disadvantage students, even though it does appear to impact student affect towards the course.

Evaluation of Hypotheses

We can now return to the hypotheses outlined at the end of Chapter 3. Recall that there were four sets of hypotheses that corresponded to the four components of the program implementation shown in Figure 7: Hypotheses about the impact of incoming student differences, hypotheses about the instructional system, hypotheses about the assessments and hypotheses about student evaluations of the course. These are addressed in light of the analyses presented in this chapter.

Individual Incoming Student Differences Hypotheses

One of the instructional goals was to create a course that would prepare students across the entire university to be FIT, regardless of their incoming experience or academic ability. These hypotheses are intended to measure the contribution of various incoming student attributes to the outcomes.

H 1: Outcomes are not linearly related to incoming ability. Because of the mastery-model assessment, students with lower GPAs and ACT scores may take more attempts to pass the bridge tasks, but should do better in this course than in their other courses.

As we saw in the analysis of the incoming student model, student GPA ACT Mathematics and ACT Composite scores are significant predictors of final grade in the course. However, the analysis of the instructional model shows that student effort (the number of BT attempts, class attendance, homework and taking advantage of help room) accounts for over two thirds of the variance in final grade, with GPA and ACT scores accounting for less than one third of the variance. Students who work harder can earn higher grades than their GPA and ACT scores alone predict. This hypothesis is accepted.

H 2: Outcomes are not dependent on class standing. Because there is no prerequisite for the course, upper division students should not perform better than lower division students.

Class standing is a factor in the incoming student model. Analysis of the grades by class standing alone shows that the upper division students' grades are actually lower than those of lower division students. However, class standing is not significant in the classroom model. Upper division students take fewer BTs and attend class less frequently than lower division students. Therefore, the difference in grades appears to be a function of student effort as predicted by the classroom model. This hypothesis is accepted.

H 3: Outcomes are not dependent on major. Because the course has multiple tracks with focal problems appropriate for different majors, students from different majors should not have different outcomes.

There were students from 141 different majors in the course. The distribution of the students differs by tracks with Tracks C and D having mostly students from the College of Business and track A having students who are no preference majors or from the colleges of Communication Arts, Education and Social Science. Track is a significant factor in the incoming student, but accounts for less than 5% of the variance in final grade in that model. In the instructional model, track accounts for about 2% of the variance, but is still significant. Tracks remain significant in the computing concepts and skills model. They do not load on any single functions in that model. However, track D correlates most with function 4, spreadsheets and track C correlates with function 7. These functions account for about 5% of the variance in final grade. Therefore, there is a difference in outcome by tracks. This hypothesis is rejected.

H 4: Outcomes are not dependent on prior computing experience. Because the course presumes no prior computing experience, any effect of prior experience may only be evident at the onset of the course but should disappear by the end of the course.

The incoming student experience was summarized as two factors: one factor that accounts for experience using computers for communications and a second factor that reflects programming experience and an understanding of

abstract computer terms. In the incoming student model, these account for about 6% of the variance in final student grade. When analyzed in the context of the instructional model, these factors do not load exclusively on any functions. Computer communication loads on function 5, which reflects the interaction of TA experience, ACT Social Science scores, and attendance. Knowledge of computing terms loads on function 4, along with doing homework, GPA and ACT scores. By the time we get to the computing concepts and skills model, only incoming knowledge of computing terms is included in the model. Again, it does not load exclusively on any function, but does correlate negatively ($r = -.303$) with function 6, attendance. Therefore, it appears that student self-reported incoming experience has some impact on student participation (attendance and homework) which in turn are predictors of outcomes. Students who report that they have computing experience are somewhat less inclined to participate in class. This hypothesis is accepted.

H 5: Outcomes are not related to age. Younger students may have had more exposure to information technology and be more comfortable with it.

However, because the course presumes no prior computing experience, any effect of age should disappear by the end of the course.

Age was never a significant variable in any of the models. Although there were 94 non-traditional students who were older than two standard deviations above the mean and upper division students are older than lower division students are, when controlling for class standing and other variables such as student effort, age was not significant. This hypothesis is accepted.

H 6: Outcomes are not dependent on gender. When other incoming student variables are controlled for, gender should not interact with the instructional system.

Gender was never a significant variable in any of the models. This hypothesis is accepted.

H 7: Outcomes are not dependent on ethnic background. When other incoming student variables are controlled for, ethnic background should not interact with the instructional system.

Table 65 shows the means and standard deviations for the course final grades by ethnic classification. Post hoc multiple comparisons show that the means for Black and Chicano/Hispanic students are significantly lower than the means for Caucasian and Asian/Pacific Islander students ($p < .001$).

Table 65

Final Grades by Ethnic Classification

Ethnic classification	<i>n</i>	<i>M</i>	<i>SD</i>
Caucasian	4112	2.98	.95
Black	479	2.42	.92
Chicano / Hispanic	129	2.51	1.16
Asian / Pacific Islander	284	3.04	.93

However, recall that ethnicity did not enter into any of the models, even the incoming student data model. Cumulative GPA and ACT scores were key variables in that model and Cumulative GPA remains significant in all of the models. ANOVA of the Cumulative GPA ($F = 79.656$, $df = 3$, 5015), ACT Mathematics ($F = 181.213$, $df = 3$, 4382) and ACT Social Science ($F = 138.611$, $df = 3$, 4382) scores show that the means are all significantly different by ethnic classification ($p < .001$). Again, the means for Black and Chicano / Hispanic students are all significantly lower than for Caucasian and Asian / Pacific Islander students ($p < .05$). Table 66 shows the distributions of ACT Mathematics and Social Science scores and cumulative GPA by ethnic classification.

Table 66

Cumulative GPA and ACT Scores by Ethnic Classification

Incoming variable	Ethnic classification	<i>n</i>	<i>M</i>	<i>SD</i>
ACT Social Science	Caucasian	3677	23.53	4.75
	Black	426	18.74	4.86
	Chicano / Hispanic	103	21.44	5.49
	Asian / Pacific Islander	180	21.17	5.39
ACT Mathematics	Caucasian	3677	22.77	3.76
	Black	426	18.40	3.19
	Chicano / Hispanic	103	21.18	4.10
	Asian / Pacific Islander	180	23.29	4.15
Cumulative GPA	Caucasian	4124	2.86	.59
	Black	481	2.43	.56
	Chicano / Hispanic	129	2.56	.66
	Asian / Pacific Islander	285	2.77	.64

There may be a difference in incoming computing experience by ethnic classification. Recall that the hypothesis H4, that incoming computing experience has no effect, was rejected because it appears that students who perceive they have computing experience have a tendency not to participate in class. Table 67 shows the experience using computers for communication by ethnic classification.

Table 67

Experience Using Computers for Communication by Ethnic Classification

Ethnic classification	<i>n</i>	<i>M</i>	<i>SD</i>
Caucasian	4127	.237	.84
Black	482	-.089	.96
Chicano / Hispanic	130	.167	.81
Asian / Pacific Islander	285	.069	.97

Multiple comparison tests of these means show that Caucasian students have significantly more experience using computers for communication than do Black students ($p < .001$) or Asian / Pacific Islander students ($p < .05$). However, there is no difference between Caucasian and Chicano / Hispanic students. Black students have significantly less computer communication experience than Chicano / Hispanic students ($p < .05$), but there is no significant difference between Black and Asian / Pacific Islander students. Asian / Pacific Islander

students have significantly less experience than Caucasian students ($p < .05$), but there is no difference between Asian / Pacific Islander students and Black or Hispanic / Chicano students. The other measure of incoming computing experience is the computer terms factor. The distributions on that factor are show in Table 68. There are no significant differences among any pairs of groups.

Table 68

Incoming Knowledge of Computing Terms by Ethnic Classification

Ethnic classification	<i>n</i>	<i>M</i>	<i>SD</i>
Caucasian	4127	-.041	1.-3
Black	482	-.036	.96
Chicano / Hispanic	130	.047	1.00
Asian / Pacific Islander	285	.006	1.01

Recall that the instructional data model – in addition to including ACT Social Science, ACT Mathematics, Cumulative GPA and incoming computing experience – had several other variables. Some reflect student effort: number of BT attempts, attendance, homework, use of the help room. ANOVAs on these variables help us understand these variables. Table 69 shows the number of BT attempts by ethnic classification.

Table 69

Number of BT Attempts by Ethnic Classification

Ethnic classification	<i>n</i>	<i>M</i>	<i>SD</i>
Caucasian	4127	7.42	2.34
Black	482	7.51	2.68
Chicano / Hispanic	130	6.82	2.90
Asian / Pacific Islander	285	7.74	2.41

A multiple comparison test on the means shows that the mean for Asian / Pacific Islander students is significantly higher than for Chicano / Hispanic students ($p < .05$). No other pairs of means significantly differ.

Attendance is another critical factor in outcomes. Table 70 compares the attendance rates by ethnic classification.

Table 70

Attendance Rates by Ethnic Classification

Ethnic classification	<i>n</i>	<i>M</i>	<i>SD</i>
Caucasian	4127	73.05%	23.73%
Black	482	67.07%	24.88%
Chicano / Hispanic	130	62.87%	26.52%
Asian / Pacific Islander	285	70.77%	24.67%

Multiple comparison tests on the pairs of means show that Black and Chicano / Hispanic students attend class significantly ($p < .001$) less than Caucasian students. Chicano / Hispanic students attend significantly less than Asian / Pacific Islander students ($p < .05$). No other pairs of means differ significantly.

There are no differences among groups in the responses to the statement “I usually did my homework before coming to class.” as shown in Table 71. The scale ranges from *Strongly Agree* (0) to *Strongly Disagree* (4).

Table 71

How Often Students Report Doing Homework by Ethnic Classification

Ethnic classification	<i>n</i>	<i>M</i>	<i>SD</i>
Caucasian	3118	1.36	1.07
Black	324	1.41	1.05
Chicano / Hispanic	86	1.41	1.08
Asian / Pacific Islander	227	1.48	1.07

The help room is another resource for students who are having difficulty. Table 72 shows the responses to the statement “I attended help room frequently.” The scale ranges from *Strongly Agree* (0) to *Strongly Disagree* (4).

Table 72

How Often Students Report Attending Help Room by Ethnic Classification

Ethnic classification	<i>n</i>	<i>M</i>	<i>SD</i>
Caucasian	2868	3.00	1.09
Black	317	2.48	1.20
Chicano / Hispanic	79	2.85	1.01
Asian / Pacific Islander	210	2.60	1.21

Multiple means comparison tests show that Black and Asian / Pacific Islander students attend help room more frequently ($p < .001$) than Caucasian students. Black students also attend help room more frequently than Chicano / Hispanic students ($p < .05$). No other pairs of comparisons are significant.

Because of the importance of attendance, the instructors send database-driven E-mail messages to students based on their attendance and BT status. Table 73 shows the distribution of the E-mail factor.

Table 73

Instructor – Student E-mail by Ethnic Classification

Ethnic classification	<i>n</i>	<i>M</i>	<i>SD</i>
Caucasian	4127	.277	.97
Black	482	.364	1.07
Chicano / Hispanic	130	.369	1.00
Asian / Pacific Islander	285	.429	1.05

Multiple means comparisons show no differences in the frequency of E-mail communication between the instructors and students among any of the ethnic groups. These communications are based solely on student attendance and BT performance and do not vary by ethnicity.

Although the final course grades for Black and Chicano/Hispanic students are significantly lower than the grades of Caucasian and Asian/Pacific Islander students, ethnicity does not enter into any of the models. The difference in grades appears to be a function of student effort and incoming ability, regardless of ethnicity. The hypothesis that outcomes are not dependent on ethnic background is accepted.

Instructional System Hypotheses

These hypotheses are intended to evaluate the contribution of various instructional system features to the outcomes. If the course is meeting the

design objectives, the instructional system should have more impact than incoming student variables.

H 8: Student participation in class will be a predictor of bridge task performance. Because the course design is predicated on students collaboratively solving problems to build schemas and concepts, student preparation for and participation in class should be the primary predictor of outcomes.

The instructional data model was based on all of the variables from the incoming student model. Incoming student ability, as measured by GPA and ACT scores, remained a significant predictor of student outcome, accounting for less than one third of the variance in final grades. However, student participation as measured by number of BT attempts (over a third of the variance), attendance (over 20% of the variance) and doing homework (over 10% of the variance) contributes much more than incoming variables in this model. Student effort accounts for twice the variance of incoming ability; this hypothesis is accepted.

H 9: Outcomes are a function of individual teaching assistants. Because each section is met by its own teaching assistants, there is may be differences across sections based on teaching assistant abilities and experience.

The average SIRS ratings of the TA were not significant enough to be included in the instructional model. Neither are SIRS predictors of student outcomes. However, the amount of experience that the lead TA has teaching the course loads significantly on function 5 in this model ($r = .633$), although function 5 accounts for less than 2% of the variance in the final student grade. It appears

that TA experience is a factor, albeit a small one, in student outcomes. This hypothesis is accepted.

Assessment Hypotheses

These hypotheses are intended to evaluate the internal and external validity of the bridge tasks to determine if they actually measure student understanding of the computing concepts that are indicative of FITness.

H 10: The fitted model correctly predicts students' bridge task performance. The classification accuracy shows how well the model fits the data.

The instructional data model correctly predicted the final course grade for 42.4% of the cross-validation cases; this is significantly ($p < .000$) better than expected by chance (12.5%). The computer concepts model increases the classification accuracy to 56.5% of the cross-validation cases ($p < .000$). The model selected 18 computing dimensions from the bridge tasks that provided statistically significant contributions to the classification accuracy. The model grouped the computing variables into clusters that reflect the underlying concepts that they were intended to evaluate. Six of the seven functions had significant loadings on computing concepts, whereas function 1 loaded primarily on the number of BT attempts, with small correlations with ($.001 < r < .194$) with several computing concepts. However, functions 2 through 7 loaded significantly on computing concepts. Function 2 had BT dimensions that evaluate the concepts of hierarchical network file systems and file access permissions. Function 3 grouped several items that require understanding data representation in Web and word processing documents along with computer hardware and software

specifications. Function 4 loaded on several spreadsheet concepts. Function 5 had BT dimensions that reflect file privacy and data representations in word processing documents. Function 6 loaded on understanding URLs, “styles” and “fields” in word processing documents. Function 7 clustered searching and Boolean operator dimensions.

These basic concepts were incorporated into the BTs up through the 2.5. Yet, in the cross-validation test, the model correctly classified 33.8% of the students who received 3.0 grades, 22.0% of the students who received 3.5 grades and 60.3% of the students who received 4.0 grades, significantly better than chance ($p < .000$). The computing skills and concepts, as measured through the 2.5 BT, predict higher level problem solving as indicated by grades of 3.0 through 4.0 in the course. This hypothesis is accepted.

H 11: Students who repeat bridge tasks more frequently pass the extension tasks less frequently. Because the extension tasks test transfer, students who understand the concepts are more likely to pass the extension tasks than are students who are learning skills alone rather than concepts. Students who do not pass the extension tasks are more likely to have to repeat the bridge tasks.

Of the 18 BT dimensions that were included in the computing concepts model, four are extension tasks. *Find and rename file* loads on mainly on function 2; *Excel function* loads on function 4 and function 3; *Find new application* loads on function 5; and *New spreadsheet* loads on functions 4 and 3. The number of BT attempts has a negative correlation ($r = -.498$) with function 1, the function on

which it primarily loads. It also has moderate positive correlations with function 2 ($r = .271$) and function 3 ($r = .379$), two of the functions on which the extension tasks load. This means that students who fail the concepts reflected by the extension tasks in functions 2 and 3 have to take more BTs before mastering those concepts. This hypothesis is accepted.

Outcome Hypotheses

These hypotheses are intended to evaluate student attitudes and perceptions of the course.

H 12: Student perceptions of how well the bridge tasks assess their learning are a function of their performance on the bridge tasks. Students who do not do as well may be less inclined to believe that the bridge tasks are a fair reflection of their knowledge.

The regression analysis of the course “fairness” factor as the dependent variable included the course grade (beta = $-.369$, 15.5% of the variance) in the first step and the number of BT attempts (beta = $.262$, 7.9% of the variance) in the third step of the analysis. Student free response comments on the course survey are consistent with the regression equation. Students who do well in the course are more inclined to evaluate the BTs as “fair” than are students who do not do as well in the course. This hypothesis is accepted.

H 13: Student perceptions of the course are a function of teaching assistant abilities and experience. Because the teaching assistants are responsible for the daily instruction, student perceptions of the course and their learning

experiences are likely to be a function of the teaching assistant abilities and experience.

Student SIRS ratings of the TAs are not related to student outcomes in the course, although TA teaching experience is a significant factor in student grades. However, the analyses of the course survey questions indicates that students' ratings of their TA's teaching abilities are related to their ratings of the course. Because of the homogeneity of the daily lesson plans and assessments, it is likely that variance in the student ratings of the course are a function the student perceptions of their TA's and ATA's teaching abilities. This hypothesis is accepted.

Summary

This chapter presented the analyses that were conducted to evaluate the four components of the instructional system presented in Chapter 3 (see Figure 7). The analyses show that each of the components has significant variables associated with it.

Student incoming abilities, as shown by GPA and ACT scores, are significant predictors of student performance in the course. They remain so even when adding instructional variables and the measures of computing concepts in the analyses of those parts of the instructional system. Student incoming experience with computers has a minor impact on student performance in the course. By the time the computing concepts model is evaluated, incoming knowledge of abstract computer terms has minor impact. It may actually be

associated with decreased attendance and have a detrimental impact on student performance in the course.

Student effort and participation in the course are the major predictors of student outcomes. Because of the criterion-referenced assessments, students who do their homework, attend class regularly and take advantage of the opportunities to take BTs do better than predicted by their incoming academic ability. This means that students who are academically disadvantaged can excel by making extra effort. Unfortunately, many Black, Chicano and Hispanic students who have lower GPAs and ACT scores are failing to take advantage of the opportunities that the course affords them. On average, they attend fewer classes than do other students. It appears that efforts to reach these students by E-mail and to encourage them to seek additional help are not yet successful.

The analysis of the assessment component shows that understanding of the computing concepts is crucial to success in the course, indicating that the course is meeting the goals of FITness. Students who do well in the course are learning the contemporary computing skills that they need to succeed in their future courses and understand the computing concepts that underlie those skills in a manner that allows them to transfer their understanding to solving new problems with computing technology.

THESIS

5

2001

**LIBRARY
Michigan State
University**

PLACE IN RETURN BOX to remove this checkout from your record.
TO AVOID FINES return on or before date due.
MAY BE RECALLED with earlier due date if requested.

DATE DUE	DATE DUE	DATE DUE
MAY 20 2007 01 25 07		

**TEACHING FOR FLUENCY WITH INFORMATION TECHNOLOGY:
AN EVALUATIVE STUDY**

VOLUME II

By

Mark George Urban-Lurain

A DISSERTATION

**Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of**

DOCTOR OF PHILOSOPHY

Department of Counseling, Educational Psychology and Special Education

2000

CHAPTER 5 DISCUSSION

This chapter summarizes the major findings of this study in the context of the research questions outlined in Chapter 1. It next presents a discussion of the implications of this study for FITness, instruction, and higher education. Finally, it proposes several areas for future research.

Major Findings

In Chapter 1, we saw that there are several questions that need to be answered if we are going to prepare students to be Fluent with Information Technology (FIT). What constitutes FITness? What is an appropriate curriculum and how do we determine it? What is the best instructional design to meet these goals? How do we assess student outcomes to determine if we are meeting the goals? To better understand these issues, this study evaluated a large (1800 students per semester) introductory computer science course that was designed to teach FITness.

Incoming Students and Generalizability of Findings

The students in this study represent a cross-section of undergraduates from non-technical majors at a large, public university. Their ACT scores are normally distributed with a mean slightly above the national average. Over half of the students are women. There are no gender differences in student outcomes.

The students are ethnically diverse; 81% are white, 10% are black; Asian students outnumber Hispanic, Chicano and Native American students. Black, Hispanic and Chicano students receive lower grades in the course than do white and Asian students. However, ethnicity is not a factor when controlling for incoming academic ability and student active participation in the course.

The students have a variety of incoming computing experience, but incoming experience using computers is not a factor in student success. Previous programming experience and some incoming knowledge of abstract computer concepts do have a small impact on outcomes.

As with most college courses, student GPA is the most significant incoming student variable for predicting outcomes. Because of the number ($N = 5068$) and diversity of the students in this study, the results should generalize to students in other U. S. higher educational institutions that have a wide distribution of academic skills.

What Constitutes FITness?

The course conforms to the broad definition of FITness suggested by the Committee on Information Technology Literacy (1999). FITness requires three types of knowledge: (a) contemporary skills, the ability to use various computer applications; (b) foundational concepts, the basic principles and concepts of computing that form the basis of computer science; and (c) intellectual capabilities, the ability to apply information technology in particular situations and use this technology to solve new problems.

The course design team used the instructional design process shown in Figure 4 resulting in two key design features of the course. First, it focuses on using computers to solve problems that epitomize the types of problems that on which students work in their subsequent courses. Second, the instruction takes an inductive approach to teaching the computing concepts needed for problem solving and transfer without teaching computer programming.

Conceptual understanding is crucial to problem solving. Chapter 2 describes the course's inductive approach to developing conceptual understanding with information technology. Students solve a series of problems that epitomize classes of problems for which a variety of computing skills are the solution (Reigeluth & Stein, 1983). Figure 3 represents the process. As students learn a variety of skills, they build mental schemas of the different skills. The intersection of these schemas triangulates on a more accurate representation of the underlying computing concepts (Anderson, 1984; Shuell, 1986).

Chapter 4 presents the analysis of the computing competencies and skills model. That analysis produced six discriminant functions on which 18 of the BT skills and concepts loaded. Consistent with the framework outlined in Chapter 2, each of the 18 skills and competencies loads on multiple functions. The concept map in Figure 20 represents the function loadings of these skills and concepts.

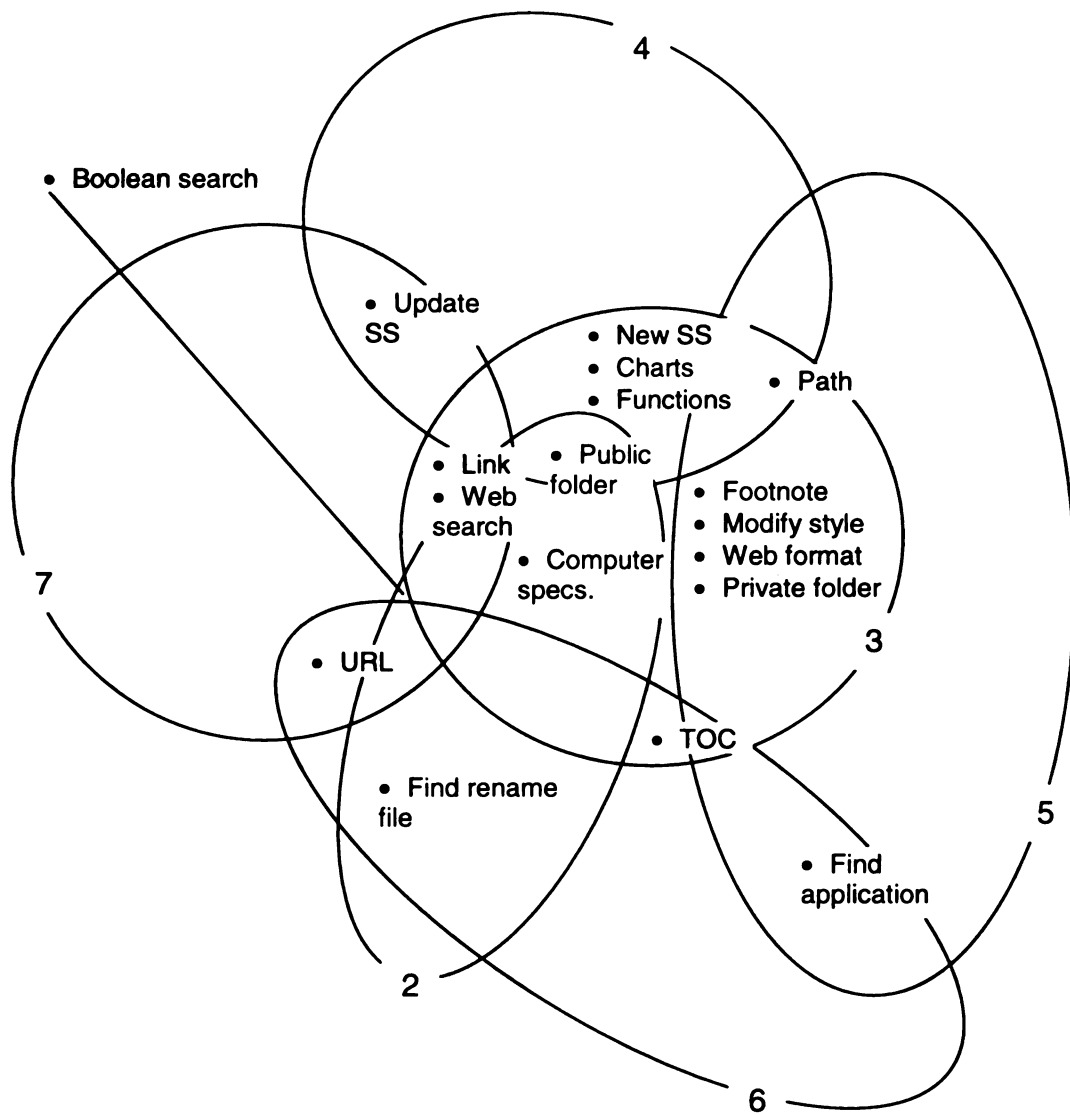


Figure 20 Concept map of computing skill schematic structure

Figure 20 portrays the conceptual space defined by the functions 2 through 7 in Table 49. The ellipses are actually seven-dimensional ellipsoids. The volumes, shapes, and locations of these ellipsoids in seven-dimensional space are defined by the six discriminant functions. The figure is a two-dimensional representation of the relationships among the ellipsoids.

Each of the 18 skills and competencies shown in the figure correlates – to various degrees – with each of the six functions. Each of the 18 skills and concepts is located in the intersections of the ellipses representing the two or three functions with which it most strongly correlates. For example, *Footnotes*, *Modify styles*, *Web page formatting* and *Creating a private folder* are each most strongly correlated with functions 3 and 5. *Public folder* correlates with functions 2 and 4 – and to a lesser degree – with function 3. It also has a moderate correlation with function 6, but these higher dimensional intersections cannot be represented in two dimensions. Function 3 appears to be central in this model because 11 of the 18 BT skills and concepts have their top two correlations with this function.

Each ellipse represents a different schema (cf., Figure 3). The intersection and clustering of the ellipses represents the way the students' schemas intersect and triangulate on the underlying computing concepts. The actual schematic structure is too complex to display in two-dimensions but is captured by the discriminant functions. However, the concept map does give a sense of the clustering of the schematic structure. The closer that a skill or competency lies to the middle of the graph, the more abstract and inter-related

with other concepts it is. For example, *Find and rename file* and *Find application* both require knowledge of hierarchical file systems and appear near the outside of the graph. *Private folder* also requires knowledge of hierarchical file systems plus knowledge of networked file system permissions. It is also grouped with data representation abstractions that are necessary to create footnotes, modify styles and control Web page formatting.

Several spreadsheet concepts also cluster together. Updating a spreadsheet is conceptually easier than creating a new spreadsheet. Updating requires an understanding of the syntax of spreadsheet formulas. Creating a new spreadsheet is more complex. It requires the ability to analyze a problem, determine the correct formulas to solve the problem and implement the solution by using the correct spreadsheet syntax.

This clustering of the computer skills and concepts is consistent with the theoretic framework adopted by the course's design team. The course's inductive approach produces a clustering of computing skills based on their underlying concepts. This analysis shows that it is possible for students to learn and apply important computing concepts to solve a variety of problems that exemplify FITness without learning computer programming.

Instructional Findings

Although it is possible to teach FITness without teaching programming, this study found that the course's instructional design was a crucial part of its success. Instruction must be based upon an analysis of the critical concepts, their interrelationships, and what types of problems epitomize these concepts at

each stage of the learning process. For the inductive approach to work, the concepts need to be the right “depth” in relationship to the problems, about one “level” lower than the scope of the problem students are trying to solve. For example, logical operators are a critical computing concept that can be approached from many conceptual levels. One approach would be that of many introductory deductive logic classes, teaching truth-functional composition, consistency trees and derivations. Another approach is to teach programming and the implementation of the underlying bitwise operators. Some courses go even lower, down to the implementation of logic gates in hardware (e.g., Biermann et al., 1994). Although these are all valid representations of logical operations, knowledge of VLSI circuitry is not what is important when trying to locate the appropriate information in a vast sea of electronic databases. Translating a vague question into the appropriate set of keywords, combining the keywords with Boolean operators, evaluating the quality of results returned by the search –refining it to locate the relevant information – help the learner build an understanding of logical concepts that will meet the needs of FITness.

This study provides evidence that a constructivist-based approach to teaching FITness is effective. The course does not use lectures but has students working actively in small groups each day to solve a series of carefully selected problems. While the course is not designed to be an independent study class, it is theoretically possible for students to do well without ever attending class. Course grades are based only on the bridge tasks; no points are awarded for attendance or class participation. However, active student participation accounts

for most of the variance in outcomes. Students who prepare for class, attend classes regularly, remain actively engaged in the group exercises, and take advantage of the opportunities to take the bridge tasks do substantially better than students who are not as active. These are key components of successful learning from either a learner-centered (e.g., Beilin, 1985; Berliner, 1992; Cahan, 1992; Greeno et al., 1996) or a social constructivist perspective (e.g., Gergen, 1994; Rogoff, 1994; St. Julien, 1994; Vygotsky, 1978).

This study also examined the impact of using Teaching Assistants (TAs) to teach the course. The amount of experience the TA had teaching the course had a small impact on student outcomes, but student effort can offset the impact of having an inexperienced TA. Student instructional ratings of their TAs' teaching abilities are not related to the grade the students receive in the course. However, student ratings of their TAs' teaching do predict their ratings of the course.

Authentic, Performance-Based Assessments

The final question this study explored is how to assess FITness. Generally, assessments are used to classify and stratify students (Bloom et al., 1971). The course designers created unique, criterion-referenced, performance-based assessments – the bridge tasks – that are a key component of the modified-mastery model course design.

There are two broad categories of performance assessments. Task-centered performance assessments evaluate particular skills and competencies and have clear-cut scoring rubrics. Construct-centered performance

assessments emphasize general skills, but do not have clear-cut scoring guidelines and are more difficult to use for summative evaluation (Khatti et al., 1998). Bridge tasks fall into the task-centered category, allowing for detailed scoring rubrics to ensure high inter-rater reliability. This study found that it is possible to create sophisticated, problem-based, performance assessments on a large scale (over 40,000 BTs in this study) with scoring rubrics that produce grading error rates of less than 5%.

Inter-task reliability is another concern with a large number of complex performance measurements (Raymond & Viswesvaran, 1993). Student performance can be affected by variability in the sampling of the particular tasks that they receive. The design of the bridge tasks addresses this problem. The granularity of the bridge task dimensions allows for analysis of the variability of each instance within the dimensions. The instructors use these analyses to reduce the variance within each dimension of the BTs. Students also have up to 12 opportunities to take BTs, so that their grades are not dependent upon a particular instance of a BT (Shavelson et al., 1993). These two factors allow the BTs to produce ratings that have high generalizability.

The bridge tasks are associated with most of the negative student feedback about the course. Some students do not adapt to the different way of grading and either flounder or complain about having to demonstrate mastery before moving on. On the other hand, many students report that they profit from the formative feedback and find the BTs to be a valuable learning tool.

Implications

The results of this study have implications in several areas. First, there are implications for teaching FITness. Beyond teaching about technology, there are also implications for the use of technology in instruction in any number of disciplines. Finally, there are implications for institutions of higher education. This section discusses each of these sets of implications.

How to Teach FITness

This study has several implications for faculty charged with teaching FITness. The first is that – at least for the foreseeable future – higher education will need to address FITness. Some argue that it is not going to be necessary to continue to teach FITness because, as computers become ubiquitous, students will arrive at college FIT. This argument goes back to the 1980's (e.g., Parker & Schneider, 1987). However, this study found that students are not arriving at college any more FIT than they did in the 1980's. Students' incoming computing experience has increased over time. They have a great deal of exposure to using computers, but they have little conceptual understanding when they come into the course. As a result, they have little ability to apply technology to the solution of new problems in ways that technology uniquely enables.

One reason that that we cannot yet assume students will arrive from high school FIT is because FITness is contextual. FITness for K-12 students is not necessarily sufficient for college students. Students need to use computing technology to solve problems in domains that they are studying in college and

will need instruction that focuses on problems that are typical of those they will encounter in their college courses.

A second implication for teaching FITness is that it requires conceptual understanding. It will not emerge from a superficial training on particular features of applications software. Teaching for FITness must focus on problem solving, with an emphasis on transfer, so students will be able to keep abreast of rapidly evolving technology and use it to solve new problems.

The impact and use of computing technology evolves differently across disciplines. Originally, computing was restricted to large numeric problems, either in the sciences or business. At that time, art students had little use for computing. Now, many art courses treat computer graphics as a unique artistic medium, and students need to be as facile with this medium as with any other. Not long ago, communications students had little need for computing. Now, computer networks are the backbone of most communication media, and computers are transforming mass communication.

Even in disciplines that have traditionally used computers for numeric purposes, the definition of FITness has changed. Business students used to learn COBOL programming, not with the expectation that they would actually program in their business careers, but so that they understood what was involved in creating business reports with computers. Now, business students must do modeling with spreadsheets that requires an understanding of both the computing tool – the spreadsheet – and the business problems that they are trying to solve. Spreadsheets are also transforming how students in technical

and scientific disciplines use computers. Disciplines that once required FORTRAN programming to do data analysis now do much more sophisticated data analysis with spreadsheets and databases. Again, this requires both knowledge of the tool (information technology) and knowledge of how to use the tool to solve the problem at hand in a particular discipline.

Finally, Chapter 2 discussed the fact that the Association for Computing Machinery (ACM) provides guidelines for CS curriculum and courses intended for CS majors. However, there are no such guidelines for faculty charged with teaching the CS0 course. This study provides a research foundation for developing CS0 curriculum guidelines that prepare students to be FIT by focusing on problem solving and computing concepts without teaching programming.

Single CS0 Course or Computing Across the Curriculum?

One of the goals of FITness is to prepare students to use information technology in all facets of their lives. This means that students will need to use computing technology as an integral part of their other courses, much as they use writing in many courses. As with teaching writing fluency, one debate about teaching FITness concerns teaching FITness in a single course vs. “computing across the curriculum.” As with the writing debate, there are arguments for and against both perspectives. This study has implications for this debate.

Computing across the curriculum has several advantages from the perspective of learning theory. First, it is contextualized, that is students learn how computing is used in a particular context while studying that discipline. This

improves student motivation, which should enhance retention. On the other hand, there are several pragmatic disadvantages to this approach. As we have seen, FITness requires conceptual understanding. If computing across the curriculum is going to develop that conceptual knowledge, the computing aspects of the instruction must be designed to facilitate that conceptual understanding. This means that to design the instruction, faculty across domains must have a deep conceptual understanding of computing, much deeper than is required for a practitioner in the discipline to be FIT. Given that discipline-specific courses have as their primary focus the discipline content, it seems unlikely that faculty in all disciplines are going to have the time or conceptual knowledge of computing to design instruction that effectively builds conceptual understanding.

The advantages and disadvantages of a single CS0 course are the inverse of “computing across the curriculum.” A single course can develop the material in depth and focus on teaching FITness as the primary task. It can provide uniform content so that all students have same foundation as they arrive in their subsequent courses. Because of the broad demand for FITness, a single course may have a high enrollment demand, so economies of scale can reduce the costs. On the other hand, there are many disadvantages to this approach. Foremost is that a single course isolates FITness from the other curriculum in the minds of the faculty who teach the course, the students who take the course and the faculty in other domains. As the review of CS0 courses in Chapter 2 shows, faculty who teach CS0 courses usually develop them in isolation from the other curriculum, deciding what they think is appropriate to teach without knowledge of

how students will use computing in their subsequent courses. Many students see a CS0 course simply as a requirement, a hurdle to clear. If students do not see the relevance to their other courses motivation suffers. Faculty in other domains may not feel the need to integrate information technology into their courses because the students “had” a computing course. Finally, if students do not have opportunities immediately and continuously to apply what they learn in the CS0 course in other courses, retention suffers. If students take a CS0 course during their first year of college and do not use computers again until their senior year, they are not likely to be FIT upon graduation.

The results of this study suggest two solutions to this dilemma. For institutions that chose to approach FITness with a single course, this study shows the importance of designing the course collaboratively among CS faculty and faculty from other disciplines. The course designers must determine the contexts in which computing is used in the various disciplines to provide exemplar problems that can be solved using computing technology. From these problems, faculty can determine how to approach the problems so that they highlight the relevant computing concepts. Doing so will produce a course in which the students solve problems that are contextualized in domains but epitomize the underlying computing concepts. Crucial to this approach is the cooperation between the CS and other discipline faculties to ensure that the course’s problems are relevant to the domains and that the subsequent discipline courses build upon the problems and concepts from the course.

1

For institutions that prefer to implement computing across the curriculum, the same principles can apply. In this context, the CS faculty can work also with faculty from other domains. The discipline faculty can provide problems within the discipline and their typical solutions. The CS faculty can work with the discipline faculty to design instruction that highlights the computing concepts and demonstrates using these concepts to solve the problems. This approach will require allocating sufficient time within the curriculum so that students are able to learn the underlying computing concepts by struggling with the problems. It will also require the discipline faculty to treat computing as an important part of the curriculum, not simply as a “tool” to be used to accomplish an end.

Technology in Instruction

This study has implications for the use of information technology in instruction across domains. Historically, the use of technology in education has paralleled technology adoption in industry. The first stage is using the technology to do things the way they were traditionally done, albeit faster or more efficiently (Cuban, 1986). For example, using television to broadcast teachers lecturing at a blackboard or, more recently, using the Web to distribute materials that were previously printed and handed out in class. Instructional technology is moving beyond the first stage. Teachers no longer use technology only to transmit information but have their students using technology to actively solve problems.

The assessment uses of technology have been at the first stage of adoption for some time, delivering assessments and managing student records.

This study demonstrates how computing technology can be extended beyond the first stage – delivering multiple-choice exams – into the area of performance assessments. However, there are several impediments to the use of performance assessments, portfolios and other “open” assessments. First, they are labor intensive. Second, it is difficult to ensure consistency in their evaluation. Third, there is much less of a research base for the validity of performance assessments than for more standardized testing. This study has implications for all three problems.

This study demonstrated how it is possible to use database technology to create, deliver and evaluate complex, performance-based assessments on a large scale, with high inter-rater reliability, in a cost-effective manner. While it would be feasible to do such assessments without technology, the technology enabled the use of these richer assessments on a large scale. This study also demonstrates how it is possible to use database technology not only to collect, but also to analyze large amounts of data about student performance. Analyzing the underlying structure of the computing concepts would not have been possible without the rich student performance database.

This study suggests new ways to understand how learners construct their knowledge by building models from the large data sets that technology allows us to collect. Such modeling can potentially be applied in any discipline where students need to develop a conceptual understanding. These conceptual models may be used to identify individual student conceptual difficulties and help instructors intervene when students are struggling to learn.

This study also has implications for simple intervention strategies to help students succeed. Based on data about attendance and bridge task attempts, the course instructors sent E-mail to students who missed classes, or were behind schedule on their BTs. These interventions were helpful and had a positive impact on student outcomes. Instructors in any course, regardless of the discipline or class size, could adopt this strategy. For example, instructors could use a spreadsheet to record class attendance and performance, then use mail merge to send E-mail to students based upon their attendance and standing in the course.

Institutional Implications

The course's instructional design process has application in other domains. For example, the Accreditation Board for Engineering and Technology (ABET) now mandates that all accredited engineering curricula have an assessment process in place that specifies desired outcomes and provides evidence that the curriculum meets those specifications. It also requires that there be a process for using the results of the evaluations to improve the effectiveness of the program. They suggest such items as "student portfolios and nationally-normed subject content examinations" as possible ways to meet these requirements (Engineering Accreditation Commission, 2000, p. 38-39). This study provides one model of how institutions might meet these goals.

The instructional design process shown in Figure 4 can be used in any discipline. The ongoing collection and analysis of data and the resulting feedback loops in the process are one way to meet the ABET requirements for evaluation

1

and improvement of engineering programs. By using instructional databases to track student performance not only within but across courses, engineering programs could provide more detailed data on student learning than is reflected simply by a final course grade.

No study about information technology in education would be complete without a discussion of the impact on distance education. A review of the Chronicle of Higher Education's Distance Education Web site (2000) shows that universities are scrambling to bring courses on-line; most are still trying to determine how to provide education in an on-line environment. This study has implications for distance education. On the face of it, a CS0 course is the perfect candidate for distance education. It has broad demand, so there is a large potential market. The topic itself requires the use of computers, so the information technology should not be a distraction from the instruction, as it can be in some distance education classes.

This study demonstrates that it is possible to deploy instruction at the scale that is necessary to make distance education fiscally viable. Producing distance education courses with the technology that is currently available requires a team of specialists (e.g., instructional designers, content experts, technology experts) and the concomitant investment of resources that cannot be recovered without large numbers of students. This study shows that such investments in instructional design can produce high quality instruction and is cost effective when amortized across a large number of students.

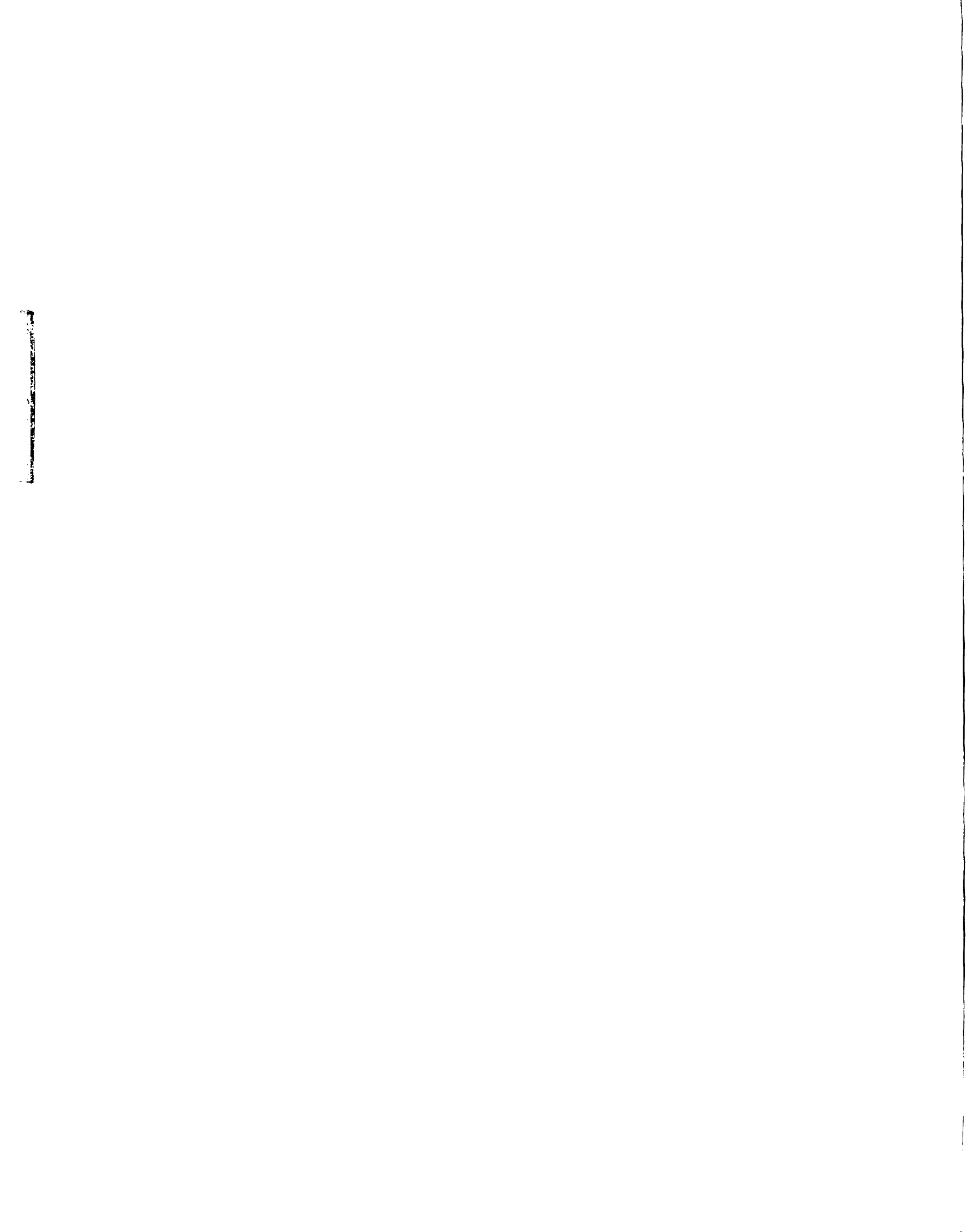
In addition to content delivery, distance education requires some form of assessment. The assessment model in this study has potential for distance education, since the bridge tasks are delivered on-line and the students receive their feedback on-line.

However, with today's technology, there are features of the course that are difficult to implement in an on-line environment. The course design is based upon constructivist learning theory: real-time student-student and student-teacher interaction is crucial. For this reason, the design team – although they had decades of instructional television and computer-based learning expertise among them – chose to design a learning environment in which "live" student-to-student and instructor-to-student interaction in the zone of proximal development was key, abandoning transmission models of instruction.

Although this study did not examine a distance education approach to teaching FITness, the results do show the importance of classroom interaction among students and instructors. Even though there are no distance education sections in the course, all course materials are available on the course Web site and in the textbook. It is theoretically possible for students to use the textbooks and the Web site, do the assignments on their own without attending class, take the bridge tasks and earn a 4.0 in the course. However, only 1% of the 1673 students who received a grade of 4.0 in the course attended class fewer than 25% of the time. Active student participation in the course accounts for the majority of the variance in the outcomes. Providing an active, constructivist learning experience in a distance education environment will require extensive

research and development to support or replace the type of interactions that students have with their peers and with the teacher in the classroom.

Distance education is just one aspect of the changes facing higher education today. Buderer (2000) described the "Harvardization" of U.S. universities after World War II as schools tried to excel in all areas: research, teaching and service across all disciplines. Today universities are "de-Harvardizing," identifying areas of expertise and focusing their resources there. Buderer quotes Kenneth Wilson, President of OSU, as saying, "each university has to have its own character now. Each can figure out a way to pick its areas of strength, and then leverage those to partner with somebody else." This study has implications for such partnerships. Although FITness is a need that all students will have, not every college and university has the expertise and resources to design and implement a CS0 course as elaborate as the course evaluated in this study. Rather than attempting to implement the course as a distance education course, it may be possible to create partnerships among colleges and universities. One institution could provide the course content and assessments to client institutions. Those clients could provide the local teaching assistants and instructional supervision to provide personal interaction with the students. This allows the providing institutions to amortize the development and infrastructure expenses across multiple client institutions. Such partnerships could allow the client institutions to provide high quality instruction without the expertise required for developing such a course.



Future Research Directions

This study suggests several avenues for future research. Some involve additional research on the course to better understand questions raised as a result of this study. Others involve the bridge tasks and suggest additional research questions that have implications beyond the course.

This study found that intervention with students who fall behind taking bridge tasks or who fail to attend class had a positive impact on outcomes. On the other hand, not all students responded to these interventions. It would be useful to better understand the impact of these interventions. What factors are involved with students who respond to the interventions? What factors are involved with students who resist the interventions? What other types of interventions might be more successful reaching students who do not respond to the E-mail? A better understanding of how to successfully intervene may help more struggling students succeed.

The qualitative data received little attention in this study. Student comments on the surveys and the content of the student correspondence are potentially lucrative areas for discourse analysis that could illuminate the statistical profiles, provide new insights and prompt new research questions. For example, the previous questions about intervention are one area that might benefit from such an analysis. The current study examined only the quantity of E-mail messages that were sent and received, there was no analysis of their content. The contents of these messages, particularly those sent by the students, is likely to provide more insight into student motivation than simply

counting the number of E-mail messages. The student free response comments on the course and TA evaluations provide another area for qualitative analysis. This may provide a deeper understanding of student motivation, which is critical to successful learning. Finally, analyzing the qualitative data may suggest additional refinements to the statistical models used in this study. For example, comparing the comments and correspondence data from students that “fit” the model with those who were incorrectly classified by the model may illuminate important factors that are not captured in the current model and suggest further refinements.

This study provides strong evidence that students are forming schemas of computing skills and concepts that they learn in the course. In-depth research on individual learners is likely to provide insight into how students are forming these schemas. For example, what are their cognitive processes as they are learning about the technology? What kinds of schema are they developing? Which mental models are fruitful? Which are not? These factors could be studied in real time, e.g., with talk-aloud protocols. A less intrusive protocol might be to capture keystroke and mouse movement data while students work, then use those to replay the student’s actions in a stimulated recall session. Either of these protocols may help us better understand how learners’ conceptual models of computing systems change and evolve.

The course designers emphasized collaborative learning in the classroom. Although collaborative learning already has a rich research literature, most of the studies have been in classroom environments that do not use computers. There

are several questions that may provide insight into the impact of computers in a collaborative classroom. How are the dynamics of collaborative learning different in a classroom with computers? What role does the computer play in the interaction among group members? How does it benefit group dynamics? In what ways is it detrimental? How do computers in the classroom change the student-teacher interactions? Are there particular arrangements of computers in the classroom that enhance or detract from the group interactions? As instruction in all domains moves more towards computing across the curriculum, understanding how computers impact the dynamics of the classroom will be important.

Retention and transfer are important parts of FITness if students are to be prepared for a lifetime of constant technological change. After they leave the course and take other courses or move into the workplace, they must retain – and build upon – the skills and concepts from the course. This study did not follow students after they left the course, so retention and transfer are other areas for further study. There are several questions to explore in this area. After they complete the course, how do students use computing technology outside of their other courses? How soon after taking the course are students taking other courses that use computing technology? How is computing technology used in these courses? How well are students prepared to solve problems in these courses by using computing technology? What are the shortcomings in student preparation? How well do students adapt to new technology when it becomes

10-20-20

available? The answers to these questions could be used to improve the instructional design of this and other CS0 courses.

Analyses of the bridge task performance in this study provide evidence that the bridge tasks are effective assessments. However, before extending and adopting this assessment model to other course or domains, further study and validation is necessary. How well do these assessments correlate with other assessments? What other variations on the bridge task model might work? One complaint from students is that they must repeat an entire bridge tasks after missing “only one” mandatory dimension. Is there an optimum mix of mandatory and optional dimensions on bridge tasks? On what theoretical basis should instructors determine which of the dimensions are mandatory and which are optional? Could this type of performance assessment be used without the mastery requirement (i.e., what is the impact of assigning “points” to each dimension and calculating a score?)

Creating bridge tasks is a complex, time-consuming task. It requires a knowledge of the content being assessed, an understanding of the bridge task framework, facility with databases and knowledge of Extensible Markup Language (XML.) Validating bridge tasks requires a detailed analysis of the performance on each instance of each dimension in the bridge task (see Figure 6.) Before faculty in other disciplines can adopt this assessment model, tools are needed that allow instructors who do not have this degree of FITness to focus on the content being assessed rather than on the structural aspects of the assessments. Ideally, a set of tools could guide faculty through the creation of

bridge tasks by asking a series of questions. The software could ask which items are most important and assign initial probabilities and estimates of performance before the items are used. After students take the bridge task, the software could analyze the performance characteristics of the various instances, allowing the instructor to adjust them as appropriate. A software development project of this scope requires a large investment in research and development. However, the potential for improving assessment in all areas of instruction makes such an investment worthwhile.

**APPENDIX A
SAMPLE LESSON PLAN FOR DAY 6**

This is a sample lesson plan for the instruction on day six of the course. The lesson plans are Web pages on the staff portion of the course Web site. These Web pages are printed for the staff so that they may take them to class for their convenience.

Each day's lesson plan is organized to show the summary of the day, general administrative instructions for the staff, and the daily "script" for the problems. The script portion of the lesson plan is displayed in a three-column table. The first column contains the name of the exercise, the start and ending times and the running time allocated for the exercise. The second column contains instructions for the Lead TA. This includes the overview of the exercise and which PowerPoint slides are to be used during the presentation. The third column contains instructions for the Assistant TA.

Day 6

Summary of Today

Introduction to WWW authoring; HTML; creating Web pages.

Sign In

Read Me First

- Before each class period starts, go to the staff web page for that day and check the **Read Me First** link at the top of the page. This will contain last minute announcements, changes, and warnings of problems. You should check it each time you teach as it may be updated even since the previous class period.

Lab Problems

- If you have severe problems that impact your entire lab (all computers down, no network, very slow server) call the Computer Lab Trouble Line: **3-4602**. Do not call the instructors or send E-mail. The instructors cannot help you with this!
- If only one computer is down or has hardware problems, please log on to aral and use the "report" function to report the down computer. You must also put a sign on the computer to let others know that it is down and that the problem has been reported. If the computer is not fixed within a few days, please report the problem again.
- If the light bulb in your ceiling mounted projector burns out or the unit malfunctions in some other way, call Microlab Operations at **5-3297**. Stress that we use the projector very heavily. These people will arrange for repair of the projector. Remind them that *Engineering Services* told us to report projector failures in this way.
- If you are using the transparency projector (often called the "overhead projector") and it burns out or runs out of transparency film, call IMC Distribution at **3-3960**. (Be sure to check first for additional light bulbs. Note that some transparency projectors have two light bulbs, and you can switch from one to the other. Note also that there generally are fresh rolls of transparency materials in a basket on the stand under the projector.)
- If you have any other problem about course policy or implementation, call the Course Coordinator at **3-8749** (press 3, and then press 4 to be transferred directly to the office).

To start the sign-in applet

- Double-click on the *sign in applet* icon in **U:\msu\course\cse\101\staff**

To start the group generation applet

- Double-click on the *generate group applet* icon in **U:\msu\course\cse\101\staff**

Script Conventions

- Throughout the **TA section** of the script, materials in normal font are instructions for the class. You may read this directly or elaborate on it as necessary.
- *Material in italics are instructions to you and are not meant to be read to the class.*
- **XX-YY**Tell the students to open their textbooks to chapter **XX** and read page **YY**. Give them exactly one minute to do so, **no more** (unless otherwise indicated in the script). The goal is to show the students that they **must** read the textbook before coming to class. (There are other pointers to the textbook, e.g., Review pp. MM-NN, but without the textbook logo. Here, you specifically tell them the page numbers but do not give them reading time.)

At the end of class

- Encourage students to work on homework in the ten minutes between the end of our class (1 hr. 50 min.) and the end of the MSU class hour (2 hours). Tell them that the homework is essential and almost always appears on Bridge Tasks.
- Turn off the ceiling-mounted projectors by clicking on the control as described in the staff FAQ, ***How do I operate the ceiling-mounted projector?*** Since the lamps in the projectors are very expensive, shutting down the units at the end of class is important. Please do so.

Get Today's Power Point Slides from the Course AFS Space

Script

Time Allocated	TA Info.	Assistant TA Info.
<p>Sign In 0-5 5 minutes</p>	<p><i>Start a copy of Notepad and type the HTML from PPT 4 into it. Do NOT save the file, but minimize Notepad. You will use this to demo saving a Web page.</i></p> <p><i>Make sure you can answer the questions on slide 1</i></p> <p><i>Generate pairs.</i></p> <p><i>It is important that you not get distracted reviewing students' BTs during class. Tell them that you will look at their results AFTER class if they have any questions. If they are looking at their BT feedback rather than doing the assignments, have them stop and come back to it after class so that they don't fall behind.</i></p>	<p>Sign in Students</p> <p>It is important that you not get distracted reviewing students' BTs during class. Tell them that you will look at their results AFTER class if they have any questions. If they are looking at their BT feedback rather than doing the assignments, have them stop and come back to it after class so that they don't fall behind.</p>
<p>Textbook questions 5-10 minutes 5 minutes</p>	<p>We will review your BT results at the END of today's class. If you are looking at your BT results now, please close the BT Feedback software so that you can work on today's material.</p> <p>Today we will creating a Web page. How many of you have already got a HOME page here at MSU?</p> <p>Those of you who already have a HOME page don't worry: you will create another Web page so that you will not lose your current Web page.</p> <p>While your machines are booting, open your texts. Work with your</p>	<p>The answers are in the textbook. Make sure that the students are finding the answers, not playing with the computers.</p>

	<p>partner to find the answers to these questions AND the page on which the answer appears.</p> <p>(PPT 1)</p> <ul style="list-style-type: none"> • What are Clients? p. 6-2 • What are Servers? p. 6-2 • How do they relate to each other? p. 6-2 • What does HTML stand for? p. 6-2 through 6-3 • What are the characteristics of HTML tags? p. 6-2 through 6-3, 6-7 through 6-9 <p><i>Randomly call on students to answer the questions.</i></p>	
<p>Web Page Creation Cycle Class Exercise 10-30 20 minutes</p>	<p>Creating Web pages is not very difficult. However, you need to understand how Web pages are stored and how Web servers make the Web pages available so that you can understand how to create Web pages.</p> <p>(PPT 2)</p> <p>At MSU, anyone with a pilot ID and AFS can create and store Web pages. However, AFS files are <u>only</u> available to computers at MSU.</p> <p>Since anyone in the world with access to a Web browser should be able to see the pages, MSU's Web server is configured to look for Pilot users' Web pages in the WEB folder in their AFS space. This diagram shows the relationship.</p> <p>If you are browsing on the web, to see one of SPARTY's Web pages you browse to the URL:</p>	<p>How did the exercise go? Fill out the exercise rating ON THE WEB.</p> <p>You can make notes here and transfer them to the web after class.</p> <p>1) Time allotted for the exercise: Not Enough -- Just Right -- Too Much</p> <p>2) About what percentage of the students seemed to understand the topic after the exercise? 0% -- 25% -- 50% -- 75% -- 100%</p> <p>3) Did students stay "on task" during the exercise? Not At All -- Somewhat -- Very Much</p> <p>4) What features or parts of the exercise worked best?</p> <p>5) What features or parts of the exercise worked worst?</p>

	<p>http://www.msu.edu/~sparty/day06.html</p> <p>There is no real pilot id SPARTY, we are just using this as an example.</p> <p>When you browse to this URL, the MSU Web server finds the AFS path to SPARTY's WEB folder in AFS and looks for the Web page there:</p> <p><code>u:\msu\user\s\p\sparty\web\day06.html</code></p> <p>You can only browse -- that is view or READ -- Web pages with a Web browser such as Netscape Navigator. However, in order to create Web pages, you must be able create -- that is WRITE -- files.</p> <p>Since <u>you</u> are the only person who can create files in your AFS space, <u>you</u> must create Web pages by editing files in <u>your</u> WEB folder in AFS.</p> <p>(PPT 3)</p> <p>The Web page development cycle looks like this:</p> <ul style="list-style-type: none"> • You use an <u>editor</u> to create or edit your Web page. • You save that file in your WEB folder in AFS. • You then test your Web page by using a browser to browse to the URL for your Web page. We will use Netscape Navigator as we have been using all semester. Here you see if the page looks OK and test to make sure that your links are working. 	<p>6) Suggestions for improving the exercise?</p> <p>Circulate and make sure that students are finding Notepad, that they are editing the Web pages as they should and they are saving the files correctly.</p> <p>If you get students who are using Composer or some other Web Editor, have them stop and do this using Notepad as we instructed. Tell them that we'll use these tools later on today, but that they need to do these HTML exercises first. If they complain, tell them that they need to learn this or they'll fail the BT.</p> <p>Problems to watch for:</p> <ul style="list-style-type: none"> • No Web folder because they deleted it. If this happens, creating a WEB folder will not work as it won't have the correct permissions. Have them create the Web folder, then follow the directions in the NEXT item • They have a WEB folder but get a FILE PERMISSION error when they browse. There's an FAQ on how to fix this. Work with the
--	---	---

	<ul style="list-style-type: none"> You then go back to the editor and make any corrections, resave the file, and view it in the browser to make sure it is working. <p>(PPT 4)</p> <p>P. 6-2 then 6-7</p> <p>The first thing you need to know is that <u>all</u> HTML files have this basic structure.</p> <ul style="list-style-type: none"> They begin with an <HTML> tag. <i>Point this out on the slide.</i> They end with a closing </HTML> tag. <i>Point this out on the slide.</i> NOTE: All closing tags have a <u>forward slash</u>. Do not confuse the forward slash with the backslash. Generally, every "opening" tag must have a matching "closing" tag somewhere in the document. As you can see, there are other tags between the HEAD tags <p>Between the opening and closing HTML tags, every HTML document must have a set of opening and closing HEAD tags.</p> <p><i>Indicate these</i></p> <p>Generally, the only thing that will go between these tags is the TITLE of your Web page. We'll come back that that in a moment. The other set of tags that every HTML document must have is a set of BODY tags.</p> <p><i>Indicate these</i></p>	<p>student following those directions on fixing their Web permissions.</p> <ul style="list-style-type: none"> Out of disk space. Check quota. Capitalization of filenames. Make sure they are all lowercase. The letter "O" rather than the number zero in the file name or letter "l" rather than the number one.
--	--	---

The actual material that will be displayed on your Web page goes between these tags.

There are many different editors that you can use to create Web pages. However, they all provide ways of creating and manipulating the HTML that underlies all Web pages.

We are going to start with the simplest of these editors: Notepad.

We'll be using Notepad to edit HTML so that you understand how HTML tags control the appearance of Web pages. After you learn the basics of creating Web pages with Notepad, we'll learn how to use a more sophisticated editor.

Since the first stage of the "Web Page development Cycle" is to edit and save a file, we'll need to start the Notepad editor.

Randomly call on students and ask:

How do we start Notepad?

Have them tell you while you demonstrate.

Toggle back to PowerPoint with

(PPT 4)

Go ahead and type the HTML from this slide into your editor. The exact spacing is not critical, that is, you can have the tags indented, or on the same line. However, formatting the page so

that the matching opening and closing tags are lined up the way you see them on the slide makes it easier to edit your Web pages,

Certain things must be done absolutely correctly, or your Web page will not work:

- The tags must appear in the correct order
- The tags must be spelled correctly
- Each tag must begin with the "less than" symbol.
Point this out.
- Each tag must end with the "greater than" symbol.
Point this out.
- Each ending tag must have the forward slash.
Point this out.

Give them a minute to enter the HTML. Have the Assistant TA circulate and let you know when they have typed this. NOTE: They have to have SOME text in Notepad before they can SAVE the file as Notepad won't save an empty file. Bring up the copy of Notepad you have with the HTML entered.

Now you are ready to SAVE your Web page. First, a bit of housekeeping. All Web pages for Pilot users must be located in your **WEB** folder in order for the Web server to be able to locate them. See the reading in today's text about **Clients and Servers**. p. 6-2.

From the File menu in Notepad, select SAVE AS. You will get the Dialog box which is prompting

you with a proposed folder in which to save the file and a default file type of TEXT.

Randomly call on students and ask:

Where do I have to save this file?
Keep prompting to have them tell you how to navigate to your **P:\WEB** folder.

Randomly call on students and ask:

What file extension must a Web page have?

Keep prompting. If students say HTM, indicate that this will work, but ask if there are any other extensions until you get HTML. Tell them that HTML is the preferred standard.

Enter the filename day06.html and SAVE your file. NOTE: Web servers are CASE-SENSITIVE. That is CAPITAL and lowercase letters are considered different. Type your filenames with all lowercase letters to maintain consistency and ensure that users will be able to locate your Web pages.

After you save the file, you should NOT close Notepad, just minimize it with your file still in it.

(PPT 5)

Next, start Netscape and browse to this Web page. The URL to this

Web page in your AFS WEB folder is:

<http://www.msu.edu/~PILOTID/day06.html>

The key for the TILDE is located in the upper left corner of your keyboard. **PILOTID** is YOUR Pilot ID

Give them a minute to browse to their pages. Answer questions and make sure the Assistant is helping students who may have problems.

At this point, you have created and browsed to your Web page, but it is blank. The tags you have entered tell the Web browser that this a "legal" Web page, but it doesn't contain any content.

Now you are going to do two things:

- You'll give the page a TITLE with the TITLE tags
- You'll put some text on the page.

Go back to Notepad. You should still have your Web page in it.

(PPT 6)

Edit your file so that it looks like this. Enter YOUR NAME and YOUR MAJOR in the file. Save your file. Do not use SAVE AS: just use SAVE so that the filename doesn't change.

Give them a minute to do this. Have the Assistant TA tell you when they have it done.

Randomly call on students and ask:

How do you test your changes?

Keep prompting until they say to look at it in Navigator. Ask them to try this and see what happens. They will say they don't see their changes.

The edits will not show because your Web browser doesn't know that the page has been changed on the Web server. In order to see your changes, you must press the **RELOAD** button.

*Toggle to Navigator and show them the **RELOAD** button.*

(PPT 7)

At this point, you have now been through one "round" of the Web page development cycle.

No matter how complex your Web page becomes, and no matter what Editor and Browser you are using, the development cycle is the same.

(PPT 8)

Now we want you to tell us more about yourself on your Web page.

Edit the page so that it has text something like this in the BODY of the page.

You page will have your name, your major, a sentence telling us that you are creating the page for

	<p>CSE 101 and a sentence inviting the user to come back to your web page.</p> <p>Be sure to save and test your page.</p>	
<p>Adding Information to Web Page Exercise 30-35 5 minutes</p>	<p><i>Circulate and make sure that they are getting the text entered correctly and that they are testing their Web pages correctly.</i></p> <p><i>They will probably notice that if they type the text into Notepad with layout like that on the PowerPoint slide that it is not being displayed with formatting on their Web browser. Tell them that we'll be addressing this in the next exercise.</i></p>	<p>Circulate and make sure that they are getting the text entered correctly and that they are testing their Web pages correctly.</p> <p>They will probably notice that if they type the text into Notepad with layout like that on the PowerPoint slide that it is not being displayed with formatting on their Web browser. Tell them that we'll be addressing this in the next exercise.</p>
<p>Setup Formatting with HTML Tags Exercise 35-45 10 minutes</p>	<p>Many of you noticed that even though you typed your text into Notepad so that it was formatted something like the text on PowerPoint slide, when you tested it in your Browser, it was one big block of text.</p> <p><i>Randomly call on students and ask:</i></p> <p>Why didn't the text look the same in the Browser as it did in Notepad?</p> <p>P6-7 to 6-9</p> <p><i>Keep prompting until they talk about the formatting being controlled by HTML. If necessary, give them a hint to look in today's reading about Data Representation.</i></p> <p>Recall that Web page formatting is controlled by HTML tags.</p>	<p>How did the exercise go? Fill out the exercise rating ON THE WEB.</p> <p>You can make notes here and transfer them to the web after class.</p> <p>1) Time allotted for the exercise: Not Enough -- Just Right -- Too Much</p> <p>2) About what percentage of the students seemed to understand the topic after the exercise? 0% -- 25% -- 50% -- 75% -- 100%</p> <p>3) Did students stay "on task" during the exercise? Not At All -- Somewhat -- Very Much</p> <p>4) What features or parts of the exercise worked best?</p> <p>5) What features or parts of the</p>

<p>These tags give instructions to the Web Browser telling it how to format the text it is displaying.</p> <p>There are a number of Web sites that give you information about HTML tags. We've included a link to one on Today's CSE 101 Web page.</p> <p>However, one way to find out about HTML tags is to examine the HTML for a Web page that is formatted or does something you want to do.</p> <p>Browse to Today's CSE 101 Web page. At the bottom of the page is a link to the Sample HTML Web Page. Click on that link. This will open a NEW Navigator window with that page in it.</p> <p><i>Show them this and wait for them to find the page.</i></p> <p>When you are browsing on any Web page, you can look at the HTML for that page. Go to the View menu and select Page Source.</p> <p><i>Show them.</i></p> <p>This window displays the HTML codes for the sample web page. The window color-codes the HTML tags so that you can find them easily.</p> <p>Note that the HTML for this page looks similar to the page you just created. There is a HTML tag, HEAD, TITLE and BODY tags. However, there are lots of other</p>	<p>exercise worked worst?</p> <p>6) Suggestions for improving the exercise?</p>
--	---

TAGS here too. You can look at appearance of the Web page in the Browser and see how the tags are controlling the formatting.

Show them how they can arrange windows so that they can see the page and the source to examine both.

You should examine this Sample page and its tags and figure out what tags you need to make your page look like this.

(PPT 9)

Note that the first two lines are to be "headings." HTML supports several levels of "headings" to make text stand out. You are to make these headings ONE and TWO.

Your text in the next paragraph should be one paragraph. It will NOT necessarily appear on this many lines. It should reformat itself as you resize the Browser window. Be sure to test that this works.

The last line should also be in its own paragraph.

Figure out how to make text italic, and underlined. Note that your name should be BOTH italic and underlined.

So, you are looking at the Sample HTML page and its source to figure out the HTML tags. Then you will edit YOUR page and test it to make sure that it is formatted

	correctly.	
<p>Formatting with HTML Tags Exercise 45-55 10 minutes</p>	<p><i>Circulate and make sure that they are finding the tags. Be sure that they are testing their page. Ask if anyone identified any other tags and point these out as time permits.</i></p>	<p>Circulate and make sure that they are finding the tags. Be sure that they are testing their page. Ask if anyone identified any other tags and point these out as time permits.</p>
<p>Setup Exercise on Creating a Link 55-60 5 minutes</p>	<p>One thing that distinguishes Web pages from other kinds of computer files is that Web pages can contain links that allow you to jump to other pages. Like everything else on Web pages, these links are controlled by HTML tags.</p> <p><i>Show them the Sample HTML Web page and the source. You may want to tile these horizontally. Find the link to the Dryden Web page.</i></p> <p>You may have noticed that this page contains a link to the home page for the publisher of your textbook. Let's examine the HTML for that link so that you understand how it works.</p> <p>Note that like every other HTML tag, this one begins with the "less than" symbol. Following the "less than" symbol is the letter "A." This means that this is an anchor tag.</p> <p>The type of anchor tag is indicated by the attribute HREF, which means that it is a Hypertext Reference.</p> <p>After the HREF attribute is the actual URL to which you will be linking. This is the fully specified URL, including HTTP://. Even though you do not need to</p>	<p>How did the exercise go? Fill out the exercise rating ON THE WEB.</p> <p>You can make notes here and transfer them to the web after class.</p> <p>1) Time allotted for the exercise: Not Enough -- Just Right -- Too Much</p> <p>2) About what percentage of the students seemed to understand the topic after the exercise? 0% -- 25% -- 50% -- 75% -- 100%</p> <p>3) Did students stay "on task" during the exercise? Not At All -- Somewhat -- Very Much</p> <p>4) What features or parts of the exercise worked best?</p> <p>5) What features or parts of the exercise worked worst?</p> <p>6) Suggestions for improving the exercise?</p>

	<p>include the HTTP:// when you browse to a Web page, it <u>must</u> be included when you create a link. Note that the URL is enclosed in quotation marks. This is very important! If you don't do this, your link will not work.</p> <p>After the URL is the "greater than" symbol that indicates that this is the end of this tag.</p> <p><i>Randomly call on students and ask:</i></p> <p>Which text on this page is "linked" to the publisher's Web site? How is that link controlled by the HTML?</p> <p><i>Keep prompting until they tell you that the text that follows the opening anchor tag up to the closing anchor tag is the "linked" text.</i></p> <p>Note that the anchor tag is like other tags. It has an opening tag.</p> <p><i>Show this.</i></p> <p>It has a closing tag. <i>Show this.</i></p> <p>And the text between these tags is "formatted" to be a link to the URL specified in the opening tag.</p> <p>(PPT 10)</p> <p>For this exercise, you need to make the text CSE 101 a link to the CSE 101 home page.</p>	
<p>Exercise on Creating a Link</p>	<p><i>Circulate among the groups and facilitate the exercise.</i></p> <p><i>Watch for syntax errors in their</i></p>	<p>Circulate among the groups and facilitate the exercise.</p> <p>Watch for syntax errors in their</p>

<p>60-70 10 minutes</p>	<p><i>links. Ask them leading questions to get them to figure out why the link doesn't work. Don't just tell them what to type.</i></p>	<p>links. Ask them leading questions to get them to figure out why the link doesn't work. Don't just tell them what to type.</p>
<p>Setup Composer Exercise 70-80 10 minutes</p>	<p>At this point you should now understand that Web pages have two major components:</p> <ul style="list-style-type: none"> • The content: text, images, etc. • The formatting: controlled by HTML tags <p>By the end of the next class, you will be doing some much more sophisticated things with Web pages such as</p> <ul style="list-style-type: none"> • Including images • Using different colors for text and background • Making bulleted lists <p>We can do all of these things and many others the just as we have been, by using Notepad. However, the details of creating and managing all of the necessary tags can get pretty picky.</p> <p>Like many other facets of computing, people have created a number of different types of editors to make this job easier. These editors take care of managing the details of creating many of the HTML tags. The one we are going to use is Netscape Composer.</p> <p>First, CLOSE your copy of Notepad. Make sure that you have saved any changes you have made to your Web page.</p> <p>Next, we are going to use</p>	<p>How did the exercise go? Fill out the exercise rating ON THE WEB.</p> <p>You can make notes here and transfer them to the web after class.</p> <p>1) Time allotted for the exercise: Not Enough -- Just Right -- Too Much</p> <p>2) About what percentage of the students seemed to understand the topic after the exercise? 0% -- 25% -- 50% -- 75% -- 100%</p> <p>3) Did students stay "on task" during the exercise? Not At All -- Somewhat -- Very Much</p> <p>4) What features or parts of the exercise worked best?</p> <p>5) What features or parts of the exercise worked worst?</p> <p>6) Suggestions for improving the exercise?</p>

Netscape Composer to edit the same file.

You start Netscape Composer from the **Communicator** menu in Netscape Navigator.

Show them how to do this and have them do it with you.

Notice that the menus and toolbars in Composer are different than those in Navigator. This is because Composer is designed to EDIT pages and Navigator is designed to BROWSE the Web.

You'll notice that, like other software, there is a File menu. Use the file menu to OPEN the page you have been working on.

Notice that you don't get the same file open dialog box we have seen in other software. This is because Netscape is a different company than Microsoft and they did things in different ways. You can think of this as being like different cars having the wiper controls in different places. They do the same thing, just somewhat differently.

You want to CHOOSE FILE to get a dialog box that will let you locate the day06.html file you have been working on. Find that file and open it.

Note that ANOTHER copy of Composer will start when you do a FILE OPEN. Students can wind up with many copies open and not be able to find the one

with their work. They can close the original one that has no file in it.

Give them a moment to find it, do not go on until the Assistant TA tells you they have their files open.

You'll note that your page looks very similar to the way it looks in the Browser. The formatting is preserved and you do not see the HTML tags. This is because Composer is a WYSIWYG (what you see is what you get) editor. That is, it "hides" the underlying HTML from you so that you can concentrate on using higher-level tools to control the way your page looks.

Let's review some of the tools that are available in Netscape Composer.

Demonstrate each of these features as you describe them.

As in most software, all of the commands that are available in Composer are available on the menus. Many of the menus contain sub menus. When you use a piece of software for the first time, you should take a few minutes to explore the menus to see what commands are available in the software.

You have already used the FILE Menu to Open and Save your file

Let's look at the FORMAT menu. Note that there is an entry for STYLES. The arrow to the right of the STYLES entry indicates that

there is another submenu available. This is true for any menu item that contains submenus. *Point this out on your display.* This submenu contains commands to change the appearance of the text for such things as bold, underline and italic.

Some commands are so common that the software has "shortcuts" as ways to get at the commands. Note that there are "toolbars" below the menu that contains "buttons." If you put your mouse pointer over these buttons and pause for a moment. You will see a description of the command that button performs. *Show this for the bold, italic, and underline buttons.*

To apply any command to specific text, you must first "select" the text to which you want to make changes. You usually do this with your mouse by clicking and dragging to highlight it. After the text is selected, you may then make changes to it. *Show this.*

(PPT 11) (repeat of slide 7)

The important thing to remember is that Netscape Composer is an editor, much like Notepad. You have to do exactly the same cycle. Edit in Composer, Save the file, View it in the Browser.

This is important because, even though Composer is a WYSIWYG editor, you need to TEST you Web pages to verify that the

	<p>formatting really is what you intended and to test that your links work.</p> <p>(PPT 12)</p> <p>Using Composer, make the following changes to your day06.html Web page.</p> <ul style="list-style-type: none"> • Add this line (as a HEADING 3): <ul style="list-style-type: none"> • Here are some of my favorite things about MSU • Under that, create a bulleted list of 4 or more items listing your favorite things • Make some of the text <ul style="list-style-type: none"> • Underlined • bold • italic • two or more of these <p>Be sure that you SAVE your file and RELOAD it in Navigator to test it.</p>	
<p>Composer Exercise 80-95 (15 minutes)</p>	<p><i>Circulate and make sure that they are on-task. If they can't find commands, have them explore the menus. Do not just tell them where the menus are.</i></p> <p><i>Make sure students are not using the PREVIEW button to see their edits. They must SAVE in Composer and then use Navigator to Browse. If they use the Preview button, it will browse to the local file, not the Web URL.</i></p> <p><i>Watch out for students typing the HTML into composer.</i></p>	<p>Circulate and make sure that they are on-task. If they can't find commands, have them explore the menus. Do not just tell them where the menus are.</p> <p>Make sure students are not using the PREVIEW button to see their edits. They must SAVE in Composer and then use Navigator to Browse. If they use the Preview button, it will browse to the local file, not the Web URL.</p> <p>Watch out for students typing the HTML into composer.</p>

<p>Setup Homework and BT Feedback 95-110 (15 minutes)</p>	<p>(PPT => 13)</p> <p>Remember that there is a homework assignment to be done before day 7.</p> <ul style="list-style-type: none"> • Finish the Web page you started in class today if it is not already done. • Using Composer, create a brand new Web page with information about the courses you are taking. • Name and describe the courses you are taking (or took last semester), using a bulleted list • Include in the page a link to the MSU Home Page at http://www.msu.edu/home/ <p>See the homework link for more information.</p> <p>Part of your homework asks you to E-mail me the URL for the Web page you are creating. Please do this at least 24 hours before the next class so that I can review your Web page before class.</p> <p>(PPT => 14)</p> <p>Hopefully, you took the 1.0 BT. How do you get your results?</p> <p>There is a computer program that works much like the OnLine BT software.</p> <p>From the Start button, you need to go to: Start >> Program Files >> Class Software >> CSE >> 101 >> Review BT Results</p> <p>This sequence is also shown as a</p>	<p>How did the exercise go? Fill out the exercise rating ON THE WEB.</p> <p>You can make notes here and transfer them to the web after class.</p> <p>1) Time allotted for the exercise: Not Enough -- Just Right -- Too Much</p> <p>2) About what percentage of the students seemed to understand the the topic after the exercise? 0% -- 25% -- 50% -- 75% -- 100%</p> <p>3) Did students stay "on task" during the exercise? Not At All -- Somewhat -- Very Much</p> <p>4) What features or parts of the exercise worked best?</p> <p>5) What features or parts of the exercise worked worst?</p> <p>6) Suggestions for improving the exercise?</p> <p>Circulate and help them get the BTFeedback software running. Review any questions they have about their BT results.</p> <p>Remind students to logout before leaving.</p>
---	---	--

link on the Today page.

Enter your Pilot ID and PW.

You should be presented with a list of all the BTs you have taken so far. In this case, it should be one. Select the BT.

Note that the display shows you the overall PASS/NO PASS for your BT. It also contains the text of your BT, along with each of the criteria and whether or not you passed each one.

You must review the results of each BT carefully so that you understand the parts on which you had problems.

You may ask the Assistant TA or me to help you if you have questions, or you can go to help room.

You may use this software at anytime to review any of your BTs. This can be a good study aid.

(PPT => 15)

Point them at the Web pages for Bridge Task.

At this time, you can be in one of several states:

- You passed the 1.0 BT. In this case, you now have a 1.0 in the course, even if you elect not to do any further work.
- You took the 1.0 BT and did not pass the 1.0 BT. In this case, you need to

review your BT results to understand what you did not pass. See me or the assistant TA after class if you have questions about what you did incorrectly.

You need to sign up for a makeup BT immediately. If you get to Day 9 -- the day scheduled for the 1.5 BT -- and you have not yet passed the 1.0, you will receive another 1.0 BT on that day. Beware of falling behind in class.

- You did not take the 1.0 BT. In this case, you will need to sign up for a makeup BT immediately. If you get to Day 9 -- the day scheduled for the 1.5 BT -- and you have not yet passed the 1.0, you will receive another 1.0 BT on that day. Beware of falling behind in class.

(PPT 16)

If you feel that your BT was misgraded, you need to have a TA -- me, our Assistant TA, or a TA in the Helproom -- review your feedback. If the TA agrees that your BT may have been misgraded, you must immediately fill out the Web form to submit grading concerns.

Demonstrate how to find this on the student Web pages.

Back to PPT 16

Do not send E-mail to the

instructors or visit them during office hours with concerns about your BT grading. Do not call the CSE 101 office. Doing so will only slow down the resolution of your questions as they will tell you to fill out the Web form. See me or the assistant TA after class if you have any questions about how this works.

If you had difficulty understanding the materials that were on the BT and need more help than the assistant TA or I can provide after class, be sure to go to the help rooms.

Show them the help room hours on the Web pages.

(PPT 17)

Recall that you cannot take the 1.5 BT until you pass the 1.0 BT. While you may repeat the 1.0 as many times as you need to to pass it, there are only 11 more BT opportunities left in the semester including all in class and makeup opportunities. If you do not promptly schedule a makeup, you will lose that chance to pass the BT on schedule.

If you repeat any BT more than one time, you are likely to fall behind and have difficulty passing the 3.0 BT by the end of the course.

Also remember that you may not receive any credit for the Ultimate Bridge Task unless you pass the 3.0 BT.

	<p>Therefore, you should not fall behind thinking that you can continue to repeat BTs indefinitely. You cannot.</p> <p><i>Point them to the further information about BTs on the BT Web page.</i></p> <p><i>Ask the students if they have questions and answer the questions. If they don't have questions and there is time, they can continue working on their Web pages.</i></p> <p><i>Circulate and answer any questions about the BT or their Web pages.</i></p> <p><i>Remind students to logout before leaving.</i></p>	
<p>Help Room ten minutes after class.</p>	<p>Help Room on all topics. General Day 6 TA feedback</p>	<p>Help Room on all topics. General Day 6 Asst. TA feedback</p>

**APPENDIX B
SAMPLE BRIDGE TASKS FROM FALL, 1999**

This appendix contains sample bridge tasks from fall, 1999. The bridge tasks are administered at the students' computers using special software that displays the BT, but does not allow the BT file to be saved or printed. Because the software displays the BTs in color, text and graphics that appear in color on the BT are reproduced in gray here.

Sample 1.0 BT From Fall 1999

Note: Many times students fail bridge tasks not because they do not understand how to do the tasks, but because they do not carefully read the bridge task. Be sure that you take the time to *read the instructions carefully and double-check your work when you are through.*

For this bridge task, you are Willy the mail boy's assistant. Due to Willy's general incompetence, he is in danger of being fired. If you successfully assemble and edit the files listed below, Willy is out, and you are the new head mail carrier. You will need to copy some files into your AFS Space and create some files of your own.

1. Using Windows 95 Help, figure out how to create a new folder in your AFS space.

The files you are collecting will be used by your pointy haired boss as part of the annual shareholders' report. Create a new folder named **Report** in your AFS space so that anyone can read from the files in it.

2. There is a file stored on AFS that you need.

Copy the file named **Fax-Jokes** from the Day 5 CSE 101 Course AFS space for your track into the new folder you created in your AFS space.

3. Your pointy haired boss lost one of the files that you need. Use the Help System to learn how to find files on your computer.

All that he can remember is that it is named **hightech.dot** and is stored on the C: drive. Find it and copy it to the folder that you made for this bridge task. After you have copied it, use the Windows 95 Help System to figure out how to rename a file. Rename this file **Headhunters**. Do not give this file any extension. You should ignore the warning you will receive that the file may become unusable.

4. Using Windows 95 Help, you need to find a particular application. You'll use this to create a file for your boss which will help you get Willy's job.

Find the application named **Notepad**. The **Notepad** program will let you create and save a text file. Start **Notepad** and enter in the following sentence (replace the blank with the name of a famous person):

My assigned partner is _____.

Save this file in the folder that you made for this bridge task with the filename **Partner**. Notepad will automatically add the file extension **.txt** to your file name. Do not remove or change the **.txt** extension.

Do not close this application as you will be adding more information to this file in the next question.

5. For the next question, you will need a message from your pointy haired boss giving you instructions that you need to follow to complete the question. To get this message, you will need to send an E-mail message to the correct address. If you do this, you will receive an E-mail message with the remaining information you need.

Send an email message to **cse101bt1a@cse.msu.edu** (Note that the character 1 in the address is the **number one**, not the **letter l**.) The subject should be *AFS file*. You do not need to include any information in the body of the message.

If you do this correctly, you should receive a reply within one or two minutes.

Follow the directions in the reply you receive to complete this question on the Bridge Task.

6. Your pointy-haired boss has been enrolled in **CSE 101 – Section 99 Track A** to improve his fledgling computer skills.

- Your boss has passed the 1.0 BT and wants to know whether he can delete the file **CAMPUSMAP.BMP** from his AFS space. From the CSE 101 web site find out whether he can delete the file. Based on what you discover at the end of the file you used in the previous question write

You can delete CAMPUSMAP.BMP.

if he may delete the file or

You must not delete CAMPUSMAP.BMP.

if he should keep the file.

7. Your boss has trouble finding things on the CSE 101 web site--- he's heard there is a search engine on the site but he can't find that either. After the information you just added to the file write

Finding the CSE 101 web site search engine:

and then give the sequence of links you should follow from the CSE 101 home page for information on how to get to the **CSE 101 web site search engine**.

Be sure to save your file with the changes you have just made.

Double check your work and make sure all of your files have been saved in the proper folder.

Use the Handin program to hand in all of the bridge task files from the folder which you copied, edited, or created for this bridge task. The Handin program can be started by selecting "File" and "Handin Files" from the menu in the bridge task software window.

You have completed bridge task 1.0. We will show you how to read your bridge task feedback in class on Day 6.

Sample 1.5 BT From Fall 1999

Note: Many times students fail bridge tasks not because they do not understand how to do the tasks, but because they do not carefully read the bridge task. Be sure that you take the time to read the instructions carefully and double check your work when you are through

SAVE EARLY, SAVE OFTEN!!!

If the next question does not specify a new file, continue making changes to the file you used in the previous question. When you are done, **SAVE YOUR WORK!!!**

Check your bridge Task. There should be three questions. If you have fewer than three questions, notify the proctor immediately.

You have discovered that being a mail carrier is not as glamorous as Willy made it appear. Wally, a research technician, has posted an opening for an intern on his research project. If you impress Wally, he'll hire you.

1. The first thing Wally wants to see in your internship application is a web page. Wally likes people who like him, but he doesn't like navigating AFS so you need to save the web page in the correct folder in your AFS space so that he'll be able to access the page by typing the correct URL in his web browser.

Netscape Page Composer can be started by selecting **Communicator - Page Composer** from the menu in Netscape Navigator.

- Start a new web page. Save it with the file name **gofer.htm** and the page title **The Office Retriever** .
- At the top of the web page, type the following text: **I like being an intern**
- Format the text you just typed so that it is **Heading 1** , **blue** , and **center aligned** .
- Wally would also like one of his profound observations on the web page. Add the following paragraph, and format it with the paragraph style **Normal** :

Down in the south where bananas grow. A small flea stepped on an elephant's toe. The elephant said with tears in his eyes. Why don't you step on someone of your size!

Make the word "**bananas** " in the paragraph that you just typed a link to the Bobby Banana Home Page . Bobby Banana's URL is **<http://www.dole.com/bobby/index2.ghtml>**.

- Format the text of the paragraph that you just typed so that the first sentence (only the first sentence) of the paragraph is the font **Book Antiqua** and **italic** .
- Add a horizontal line to the web page between the header and the paragraph. Set the width of the line to 250 pixels.
- At the bottom of your web page, add a line that says:

The URL for this page is: _____ .

Replace the blank with the URL for the web page you are creating for this bridge task.

Wally doesn't have access to your account, so make sure that if Wally uses this URL in Netscape Navigator, he can see this web page and that all the links are working correctly. **Test this by opening your page in Netscape Navigator using the URL you just typed.**

2. Wally insists on organization on all of his projects. You will need to create a new folder to store the files you will create in the remaining parts of this project.

Create a new folder named **Apply** in your AFS space so that foreign competitors won't be able to steal company secrets from the files that you store there.

3. Wally wants to be sure that you can do library research. You'll need to find articles using WinSPIRS which can be used by Wally to answer a question that he is researching. You can open WinSPIRS using **Start - Programs - Class Software - cse - 101 - CSE 101 Library Training . The AGRIS (database of agricultural articles), ERIC (database of educational articles), and MEDLINE (database of medical articles) databases are available for you to search.**

Select the most appropriate database, and refine your search so that it returns 10 or fewer records. Save these records to the folder which you created for this bridge task.

Wally thinks that it's easier to learn about computers while listening to music. Find articles about music and learning about computers. Save your search in a file named **learning.txt** .

Check that your web page displays correctly and your links work using Netscape Navigator *before* you hand it in.

After completing the files for this bridge task, use the Handin program to hand in the files you edited. This includes the web page you created, any graphics you used on the web page, and the WinSPIRS file.

You have completed the 1.5 Bridge Task. Your feedback will be available within a few days.

Sample 2.0 BT From Fall 1999

Note: Many times students fail bridge tasks not because they do not understand how to do the tasks, but because they do not carefully read the bridge task. Be sure that you take the time to read the instructions carefully and double check your work when you are through

SAVE EARLY, SAVE OFTEN!!!

If the next question does not specify a new file, continue making changes to the file you used in the previous question. When you are done, SAVE YOUR WORK!!!

Check your bridge Task. There should be four questions. If you have fewer than four questions, notify the proctor immediately.

Dogbert has noticed the superior quality of the work you've been doing for Wally, and he wants to borrow you to work on a project for him. Dogbert has a master plan for world domination, but he needs a governing document for his new ruling class, and he needs a computer to disseminate information to his followers.

1. Dogbert wants this information kept secret (for obvious reasons). Create a new folder named **Dogbert's Files in your AFS space so that nobody else will be able to read the files you store there.**

2. The first thing Dogbert wants from you is a governing document. Copy the constitution_styles.doc file that you worked on in class on Day 11 and Day 12 and which you modified for your homework into the folder that you created for this bridge task. This file **must contain all the formatting that was required in class and also any formatting required in the Day 11 and Day 12 homework assignments.**

Rename the copy in your bridge task folder **constitution_btMMDD.doc** where **MMDD** are today's month and day. For example, if today were February 7, your filename would be *constitution_bt0207.doc*.

(Note: It is important to make a copy of the file for this bridge task rather than editing the original file in your AFS space. If you need to retake this bridge task, you will need to have the original file again!)

3. Dogbert likes that this constitution is long and has lots of confusing legal terms, but he thinks it could look a little bit nicer. He wants you to use **styles to change the formatting of some of the text of the document.**

Format the text of the **preamble** to the font Matura MT Script Capitals, engraved, 18 point, blue, 1 inch hanging indent, 16 point line spacing, and 12 points after the paragraphs. The preamble is the paragraph at the beginning of the constitution that begins with "We the people...". You need to make a new style to apply to the preamble to do this. Name the new style **The Beginning**.

Dogbert wants to subliminally program the In-duh-viduals to show respect to the Ruling Class in his new world order, so he wants to put subtle messages

throughout this governing document that he is going to make all of his followers read.

In the preamble (the paragraph at the beginning of the constitution that begins with "We the people..."), add a footnote just after the word **Justice** in the first sentence. The footnote must be numbered **1**. It must contain the text **We love the Minister of the Snack Machine**. In the preamble, also add an endnote just after the text **Tranquility**. The endnote must be numbered **i**. It must contain the text **Dogbert is very clever**.

Dogbert still isn't satisfied with the appearance of the document.

- At the end of the document, after all of the rest of the text, add a bulleted list of people who will be in Dogbert's Ruling Class. These people are:

Dogbert
Gillian Anderson
Colin Powell
Larry Gonick
Pat Schroeder

Format the list to use check marks for the bullets.

- At the end of the document, after all of the text, insert a table that contains exactly 3 rows and exactly 2 columns. Fill the table with the following text, as shown:

Ruling Class	In-duh-viduals
Dogbert	Willy
	Pointy haired boss

Type your name in the empty cell in the table. The words may wrap differently in the cells in the table in your document than they appear on this bridge task. Format the text in the table with the style Main Text 2.

- Set the left and right margins of the entire document to 0.75 inches.

Dogbert doesn't like the word "ratified" because it sounds like it involves rodents. Find a synonym for the word "ratified" and replace all occurrences of the word "ratified" in the document with the synonym you found.

4. Dogbert wants to make sure that you're not an In-duh-vidual, so he wants one last test of your computer skills. At the end of the constitution document enter the text:

The latest version of the document is found at:

After this statement type the **full** path on the U: drive to the Word file containing the document that you are currently editing for this bridge task. This is the path to the copy of the file you made for this bridge task. (You must type the path for the file no matter whether you think the file is public or private.) Be sure to include the filename in the path.

Check your files before you hand them in.

After completing the files for this bridge task, use the Handin program to hand in the Word file you edited.

You have completed the 2.0 Bridge Task. Your feedback will be available within a few days.

Sample 2.5 BT From Fall 1999

Note: Many times students fail bridge tasks not because they do not understand how to do the tasks, but because they do not carefully read the bridge task. Be sure that you take the time to

READ THE INSTRUCTIONS CAREFULLY

and double check your work when you are through.

SAVE EARLY, SAVE OFTEN!!!

If the next question does not specify a new file, continue making changes to the file you used in the previous question. When you are done, SAVE YOUR WORK!!!

Check your bridge Task. There should be eight questions. If you have fewer than eight questions, notify the proctor immediately.

Catbert, the evil human resources director, has noticed that you have been changing jobs frequently. He suspects that you don't have the skills to keep a job for more than a week, and he's going to give you a chance to prove that you really are amazingly clever and highly talented.

1. Catbert wants you to create a folder to store the file that you will use to prove your worth. This is a confidential personnel matter so make the folder in your AFS space so that other employees won't be able to read the file that you store there. Name this new folder **Catbert's File.**

2. Catbert wants you to show that you understand the company payroll. Copy the payroll file that you worked on in class on Day 17 and which you modified for your Day 17 homework into the folder that you created for this bridge task. This file **must contain all the formatting and calculations that were required in class and also any formatting and calculations required in the Day 17 homework assignment.**

Rename the copy in your bridge task folder **payroll_btMMDD.xls** where **MMDD** are today's month and day. For example, if today were February 7, your filename would be *payroll_bt0207.xls*.

(Note: It is important to make a copy of the file for this bridge task rather than editing the original file in your AFS space. If you need to retake this bridge task, you will need to have the original file again!)

3. Catbert has noticed that the employee data in your payroll spreadsheet is not accurate.

A new employee named Alvin Kurtzweil was hired by the company last month. He has the following data.

- His starting pay is \$31.15 per hour and he works a 80 hour work week.

- He is in the 22.14% federal tax bracket.
- He has 5 dependents.
- He can only afford to contribute \$55.00 to the community fund.

See the **Notes** under the table of Data for new employees from the Day 17 homework to determine the rest of the necessary tax rates.

Add Alvin to your payroll spreadsheet in a new row below Joe Smith. Be sure to calculate all of the outputs specified in the original Day 17 **Payroll Spreadsheet Design Exercise**. Verify that the results for all of your calculations reflect the data values currently on the worksheet and that all data is formatted correctly.

4. Catbert wants a link from this payroll spreadsheet to the Internal Revenue Service's web page. Insert a new row at the top of your spreadsheet. Find and use the appropriate **function** to make a hyperlink in cell **A1**. The IRS's URL is **<http://www.irs.gov>**, but Catbert wants the text "**Tax Collectors**" to be displayed in the cell as the link.

Be sure to test your link and verify that it works.

5. Catbert wants to see how much money all of the employees are making. Create a pie chart of **net pay** (and only net pay) of all of the employees in the company. Your pie chart should be the default chart sub-type. Make the chart title "**Net Pay of Employees**".

Create a legend on the left side of the chart. The legend must list the employees' names so that Catbert can tell which pie slice corresponds to each employee. The **data labels** must show the percentage of the total wages for each employee.

Place this chart as a new sheet and name the sheet "**Net Pay Chart**".

6. Catbert is still not convinced of your skills. He has some data that he wants you to analyze in a new Excel spreadsheet. Be sure to include **all** of the inputs he gives you, and calculate and display **all** of the outputs he wants.

Catbert is not a mind reader, so all data must have descriptive headings so he can understand your spreadsheet, otherwise he may spontaneously terminate you. All text shown below in red should be used as headings to label input or output values.

Catbert doesn't like messy spreadsheets so he also has a list of formatting requirements. Make sure that all of the cells in your spreadsheet are wide enough to display all of the data in them. Use appropriate formatting for all cells (i.e. Accounting or Currency format for cells with dollar values, Percentage for cells with percentage values, Date or Time for cells with date and time values).

The company produces high quality exploding widgets for markets worldwide. Catbert wants you to calculate this year's revenues from widget sales and calculate how many widgets should be produced and shipped next year. Save your new Excel workbook file as **Production.xls**.

The data that Catbert currently has is as follows:

Continent	Widgets sold this year
North America	150,000
Europe	79,000
Africa	53,000
Antarctica	225,000

Catbert wants you to calculate:

- The total Revenue for each continent (the revenue for each widget is \$0.75)
- Projected Sales of the number of widgets sold in each continent next year (sales increase by 5% each year)
- The total amount of revenue this year for all continents
- The average number of widgets sold across all continents this year

Use appropriate **functions** to calculate the **total amount of revenue** and the **average number of widgets sold**.

Make all of your headings the font **Impact, italic, and green**.

The data for each continent must be in a separate row.

7. Catbert needs the research on purchasing a computer that your group did on Day 17.

Copy the computer specifications file that you worked on in class on Day 17 into the folder that you created for this bridge task. This file **must** contain all the information you were required to gather in class, any additional data as required in the Day 17 homework assignment, and the charts added to this file in class on Day 18.

Rename the copy of the spreadsheet that is in your bridge task folder **computer_spec_btMMDD.xls** where **MMDD** are today's month and day. For example, if today were February 7, your filename would be *computer_spec_bt0207.xls*.

(Note: It is important to make a copy of the file for this bridge task rather than editing the original file in your AFS space. If you need to retake this bridge task, you will need to have the original file again!)

Catbert wants to know more about machines in your spreadsheet. Catbert isn't familiar with the format of your spreadsheet and needs hints to lead him to the information he requires.

The class assignment suggested that you set the fill color of cells in your spreadsheet to make it easier to enter your data. If your spreadsheet has fill color in any cells you need to remove the fill color. To remove the fill color:

- Select all of the cells in your spreadsheet that contain any data.

- Select the **FORMAT** menu.
- Select the **CELLS** menu.
- Select **PATTERNS**.
- Set the **CELL SHADING** to **NO COLOR**.

In the problem below you will be asked to highlight cells fitting a given criteria. You do this highlighting by setting the fill color of the cells to red.

For example, if you were finding the largest amount of video RAM you would highlight the indicated cell as in the picture at the right:

If more than a single machine satisfies a criteria highlight all of the cells meeting a criteria. For example, if two machines had the fastest DVD speeds you would highlight cells as in the picture below:

CD/DVD	Speed	DVD 8	DVD 8	DVD 6	DVD 6
Modem	Included?	Yes	Yes	Yes	Yes
	Speed(kps)	56	56	56	56

You must only highlight the cells that meet the specified criteria.

Catbert wants to know which machine or machines have:

- The largest amount of RAM.
- The largest capacity hard disk drive.
- The largest size monitor.

For each of the above categories, highlight the cells in your spreadsheet which contain the appropriate data.

After highlighting the cells for the machines that fit all three categories, Catbert now wants you to help him decide which machine or machines to buy. There are three possible recommendations you can make:

1. No machine is the best in all three categories.
2. Only one machine is the best in all three categories.
3. More than one machine is the best in all three categories. In this case Catbert wants to know which machine is the cheapest.

Insert a **NEW ROW** at the top of your spreadsheet.

Examine your spreadsheet to determine which of the three above statements is true.

1. If no machine is the best in all three categories, enter the text:

No machine is the best in all three categories.

in cell **A1** of your spreadsheet.

2. If only one machine is the best in all three categories, enter the text:

Only machine *model* from *manufacturer* is the best

in cell **A1** of your spreadsheet. Replace *model* with the model name or number of the computer and *manufacturer* with the manufacturer of the computer.

3. If more than one machine is the best in all three categories, select the cheapest one and enter the text:

Machine *model* from *manufacturer* is the cheapest

in cell **A1** of your spreadsheet. Replace *model* with the model name or number of the computer and *manufacturer* with the manufacturer of the computer.

Be sure to save your worksheet with these changes.

8. Catbert's assistant wants you to tell him which machine or machines are able to run certain software.

Insert a **NEW ROW** below row 1 (the row you inserted for the previous question) at the top of your spreadsheet. You will use cell **A2** to tell Catbert's assistant which machine(s) will run the software.

Determine which, if any, of the computer(s) on your spreadsheet can run this set of software. All of the programs need to be stored on the hard drive at the same time. Catbert's assistant plans on running only one program at a time, so the computer(s) only need as much RAM as it would take to run the one program that requires the largest amount of RAM.

You only need to consider the computer's RAM and Hard Disk when making this decision.

Examine the list of software in the table below.

Software Package	RAM Required	Hard Disk Space Required
Windows 98 Second Edition	24 MB	260 MB
Microsoft Office 2000 Premium	32 MB	626 MB
Adobe Premiere	64 MB	15.4 GB
Corel Draw 9	64 MB	100 MB

Determine which computer or computers can install and run the software.

There are three possible recommendations you can make:

1. No machine can run this set of software.
2. Only one machine can run this set of software.
3. More than one machine can run this set of software.

Examine your spreadsheet to determine which of the three above statements is true.

1. If no machine can run the software, enter the text:

No machine can run the software.

in cell **A2** of your spreadsheet.

2. If only one machine can run the software, enter the text:

Only machine *model* from *manufacturer* can run the software

in cell **A2** of your spreadsheet. Replace *model* with the model name or number of the computer and *manufacturer* with the manufacturer of the computer.

3. If more than one machine can run the software, enter the text:

The following machines can run the software: and list the *models* and *manufacturers* of all of the computers that can run software.

in cell **A2** of your spreadsheet.

Check your files before you hand them in.

After completing the file for this bridge task, use the Handin program to hand all three of the Excel files you edited. You should handin THREE files. You have completed the 2.5 Bridge Task. Your feedback will be available within a few days.

Sample 3.0 Track A BT From Fall 1999

Note: Many times students fail bridge tasks not because they do not understand how to do the tasks, but because they do not carefully read the bridge task. Be sure that you take the time to

READ THE INSTRUCTIONS CAREFULLY

and double check your work when you are through.

SAVE EARLY, SAVE OFTEN!!!

If the next question does not specify a new file, continue making changes to the file you used in the previous question. When you are done, SAVE YOUR WORK!!!

Check your bridge Task. There should be seven questions. If you have fewer than seven questions, notify the proctor immediately.

Note on the length of this Bridge Task: This Bridge Task is fairly long. Be sure to watch your time. If you get stuck or are having problems with parts of the Bridge Task, move on to other parts and then return to the parts where you are having problems at the end. In particular, be sure that you complete the navigation toolbar question and test your navigation tools as you did in class. This is a required part of the BT, so you need to have it working in order to pass the BT.

You have finally had enough of your pointy haired boss and his legion of evil minions. You're ready to pack it in, move to Sandusky, Ohio and ride roller coasters until you make yourself sick. All you need to do before you leave is line up a new cushy web page design job to support your theme park habit, and you'll finally be free of the travails and tedium of hard core computing. Adams, Inc. is holding a competition to select a new webmaster for their corporate site. If you submit a site that meets all of their requirements, you'll be selected for the job.

1. You must create a new folder in your **web folder to organize all of the web pages and graphics you may need for the new web site that you'll make. This web site will be submitted to Adams, Inc.'s marketing department to demonstrate your web site building skills.**

You must save all of the files that you create, modify, or use for this bridge task in this folder.

Name the new folder **MyEscape_MMDD where *MM* is the current month and *DD* is the current date.**

For example, if today was February 14 the name of the folder would be **MyEscape_0214.**

You need to be sure that the marketing department will be able to view your

pages. Be sure to test the pages as you build them by using the **full** URLs for the pages in the site.

Refer to pages 1-28 and 1-29 of the text for information on constructing URLs. The two folder names in the URL for this bridge task site are:

- **~pilotid** where **pilotid** is your pilot ID.
- **MyEscape_MMDD** (the folder you created for this bridge task)

You should be able to construct the URL to this site from this information. Thus, the URL to the web page **dogbert.htm** in the above folder would be:

http://www.msu.edu/~pilotid/MyEscape_MMDD/dogbert.htm

2. Adams, Inc. has a template that they want you to use to design the web site.

Copy the file

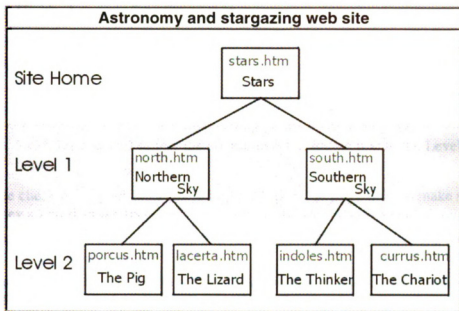
U:\msu\course\cse\101\general_track_a\Day24\Gothic_template.htm to the folder you created for this bridge task. Use this template to create all of the pages for this web site.

3. The Adams, Inc. marketing department has some guidelines about how to implement the template to design the web site.

- All of the text that is already in the template is formatted with a **single** font. Determine what font that is, and use it for **all** text that you add to the web site so that the site will be consistently formatted.
- You must not change the font formatting of the text **Page Top** and **top of the page**.
- You must not remove any horizontal lines or tables that are in the template.
- Use Help to find out about **targets**. Put a target at the beginning of the words **Page Top** on the template.
- Name this target **TheTop**.
- Create a link from the text **top of the page** to the target you just created so that clicking on the text **top of the page** takes you back to the text **Page Top**.

Be sure to save the changes to your template before you continue. (You can test that the link works correctly by adding several blank lines to the template - enough to make the page so long you have to scroll up and down to see the whole page - and clicking on the link. The link should bring you back to the target you placed at the text **Page Top**. The link may not appear to do anything if your page is not at least one "screen" long. Be sure to remove the blank lines before handing in your files.)

4. You must implement the web site flow chart below. Included in the flow chart are each of the web pages that must exist in your web site. The **file name** for each page is in RED, and the **page title** for each page is in BLUE.



You must do the following in your web site implementation:

- Use Netscape Composer to create each of the web pages in the web site flow chart represented by a rectangle. Each of the pages you create must be based on the template that you copied for this bridge task.
- For each of your web pages in the web site flow chart:
 - Set the **page title property** of the page to the **Page title** given in the flow chart.
 - Replace the text **Page Top** on each page with that page's **Page title** from the flow chart. Be careful that you do not accidentally delete the target that you created when you make this edit.
 - Below the text *PUT BODY TEXT HERE* create links to **all** the web pages that are connected directly to that web page in the flow chart, both above and below. You may enter text or use images for your links. **HINT:** This means create a link for a web page, even if it also appears in the Navigation Tool that you make in question 5.

You need to add text to the **Site Home** page and each of the **Level 1** pages. On each of the pages, replace the text *PUT BODY TEXT HERE* with the following text:

- On the stars.htm page: **This is the web site of Adams, Inc. We produce the world's finest telescopes.**

- On the north.htm page: **These are constellations of the Northern Hemisphere.**
- On the south.htm page: **These are constellations of the Southern Hemisphere.**

The marketing department has left text for the **Level 2** pages in a file in the course AFS space.

- Find the file **U:\msu\course\cse\101\general_track_a\Day24\stars.doc** and open it in Microsoft Word.
- Read this file, and follow the instructions for adding text to the Level 2 pages.

Double check all of your links (including the **top of the page** link) to make sure that they all work correctly).

5. The marketing department would like to see something a little more exciting than just text on the web site.

A. You need to add a java applet to the **site home** page. Go to the Day 24 web page for Track A. Find the link to the **spotLinks** java applet.

- Read the information about adding the applet to your web page.
- Add the **spotLinks** applet to the **site home** page above the text **top of the page** .
- Make the following customizations to the applet:
 - Set the button color code to **251,4,66**
 - Set the text color code to **250,254,71**
 - Set the highlighted text and button spot color code to **240,240,240**
 - Set the selected text and button spot color code to **0,255,64**
 - Set the first button to be highlighted when the applet is started.
 - Display **exactly 4** buttons
 - The first button must link to the Adams, Inc. home page, and the text displayed on the button must read **Adams, Inc.** . The URL of the Adams, Inc. home page is **<http://www.unitedmedia.com/comics/dilbert/career/>**
 - The second button must link to the CSE 101 web site. The text displayed on the button must read **CSE 101**

- The third button must link to the Extension-o-matic home page, and the text displayed on the button must read **Extension-o-matic** . The URL of the Extension-o-matic home page is **<http://www.student.com/feature/extensionomatic>**
- The fourth button must link to the Seinfeld-o-matic home page, and the text displayed on the button must read **Seinfeld-o-matic** . The URL of the Seinfeld-o-matic home page is **<http://www.student.com/feature/seinfeldomatic>**
- Make sure that the height and width of the applet are set correctly so that the entire applet is visible.
- Test the applet to be sure that it is working before you hand in your files.

B. You need to add an image to the **site home** page. Go to the Day 24 web page for Track A. Find the link to the image files for the site you are creating.

- Select **one** image from the page of images provided for your site, and save it to the folder you created for this bridge task.
- Create a table with 1 row and 1 column on the **site home** page for the site you are creating.
 - Set the border line width of the table to 10 pixels.
 - Uncheck the **Table width:** option and the **Equal column widths** option.
- Put the image you just saved in the cell of the table you just created.
- Make this image a link to the Adams, Inc. home page. The URL for Adams, Inc. is **<http://www.unitedmedia.com/comics/dilbert/career/>**
- The marketing director wants the Web browser status bar to show visitors to your web site where the link will take them before they click on the link. To do this, you need to add **extra HTML** to the **link** tag for the image. You add extra HTML in the same dialog box where you create the link on the image.

The HTML code you need to add is:

```
onMouseOver='window.status="Adams, Inc. Home  
Page"; return true;'  
onMouseOut='window.status=" "; return true;'
```

There are two lines of code; you may need to make your BT window wider to see them displayed correctly.

Be careful to type the code *exactly* as it appears. If you have typed the code correctly, the status bar at the bottom of the web browser window will display **Adams, Inc. Home Page** when your mouse is over the image and will remove it when your mouse is not over the image. Test this before you hand in your files.

- Set the image properties so that the image has a solid border of 5 pixels.

6. You need to include a navigation tool so that the head of the marketing department can find her way around your site.

- The navigation tool must be formatted as a table.
- The navigation tool must have a border width of 1.
- The navigation tool must have a cell padding of 1.
- The links in the navigation tool must be text, not images. You can use the **Page Title** of each of the pages to which you are linking as the link text.
- Each link in the navigation tool must be in its own cell.
- The text in the cells of the navigation tool must be right aligned.
- The navigation tool must appear in the correct location. Delete the text *PUT NAVIGATION TOOL HERE*, and replace it with your navigation tool.
- The navigation tool must appear on every page in your site.
- The navigation tool must contain links to the **Site Home** page, all **Level 1** pages from the site flow chart, and the **Centre for the Easily Amused** home page (the URL for the Centre for the Easily Amused home page is <http://www.amused.com>).
- Verify your navigation tool works appropriately. Check that each link in the Navigation Tool opens the correct web page.

7. You need to include contact information on your web site so that you can be informed when you are selected for the new job.

Beneath the text **top of the page** on your site's **Home Page**, add the following text:

Site design by _____ . copyright 1999, CCC Web Design

Replace the blank with your name. Make your name a link to the home page of the **MyMSU site** you created in class on Days 20-23. Specify the **full** URL of the page in the link. You do **not** have to hand in the files in your **MyMSU site**.

Check your files before you hand them in. Thoroughly test your web site to make sure that all links work correctly and all required components are present.

After completing the files for this bridge task, use the Handin program to hand in all of the files for your web site. You should handin ALL web pages and image files or Java applet files you used to create this web site. This includes the edited template file, SEVEN web pages for the web site, and any image or applet files you used. You do not have to hand in the files in your MyMSU web site.

You have completed the 3.0 bridge task. Your feedback will be available within a few days.

Note: Many times students fail bridge tasks not because they do not understand how to do the tasks, but because they do not carefully read the bridge task. Be sure that you take the time to

READ THE INSTRUCTIONS CAREFULLY

and double check your work when you are through.

SAVE EARLY, SAVE OFTEN!!!

If the next question does not specify a new file, continue making changes to the file you used in the previous question. When you are done, SAVE YOUR WORK!!!

Check your bridge Task. There should be five questions. If you have fewer than five questions, notify the proctor immediately.

You have finally had enough of your pointy haired boss and his legion of evil minions. You're ready to pack it in, move to somewhere more pleasant, and goof off until your money runs out. All you need to do before you leave is decide on where you want to go, and you'll finally be free of the travails and tedium of computing.

1. You don't want your pointy haired boss finding out about your plans to abandon the company, so create a new folder in your AFS so that he won't be able to read the files stored there. Name this folder **MyEscape_MMDD where *MM* is the current month and *DD* is the current day.**

For example, if today was February 14 the name of the folder would be **MyEscape_0214.**

You need to examine the weather data for a city you are considering moving to. A travel agency that you have contacted has left the weather data file in the **CSE 101 Day 24 AFS space.**

Copy the file

U:\msu\course\cps\101\spreadsheet_c\Day24\Georgia_weather.prn to the folder you created for this bridge task.

This file contains the weather data for the city that you are considering moving to. Import this file into an Excel spreadsheet. The data in this file is formatted with the same fields as the weather files that you imported on Day 20. Each column of data in the text file has a heading that should also appear in the first row of the spreadsheet after the data is imported. Be sure that your data and headings are imported correctly, with each data item in its own cell.

Save the file as an Excel workbook with the name **Georgia_escape_MMDD.xls using the same value for MMDD as you used for the folder name.**

2. You notice that the same incompetent dope that saved the weather data files that you imported on Day 20 created the weather data file that you just imported,

and some temperatures are in Celsius while others are in Fahrenheit. All of the temperature data need to be in the same scale so they can be compared.

Add two new columns to the right of the existing data in the spreadsheet and label them **MAX TEMP C** and **MIN TEMP C**.

Create a formula that will automatically detect whether the temperatures for each date are in Celsius or Fahrenheit, convert all Fahrenheit temperatures to the Celsius scale, and display Celsius temperatures **without** converting them. The same formula must be used for both maximum and minimum temperature columns and will require mixed cell references.

Use this formula to calculate **MAX TEMP C** values based on the raw data values in **TMAX24** and to calculate **MIN TEMP C** values based on the raw data values in **TMIN24**. There must be **MAX TEMP C** and **MIN TEMP C** values for every record in the data set. The values must be rounded to the nearest integer (i.e. there are no digits to the right of the decimal point) using a built-in Excel function.

The formula for converting values from Fahrenheit to Celsius is:

$$\text{Degrees_Celsius} = 5/9 * (\text{Degrees_Fahrenheit} - 32)$$

3. Now that you have your data organized, it needs to be summarized and displayed so it's easier to see the advantages of moving.

a) Use a pivot table to summarize the data.

- Use the **Month** column from your worksheet as the column labels.
- Use the **PSUN24** values for the data values of the pivot table.
- Summarize by the **AVERAGE** of the **PSUN24** values.
- The pivot table must appear on a new sheet.
- Rename the sheet that the pivot table is on to **Sun Table**.

b) Display the pivot table summary in a chart.

- Make a column chart using the **Clustered column with a 3-D visual effect** format.
- Add a meaningful chart title, x-axis title, and y-axis title.
- The legend should include meaningful text from the pivot table datasheet (not "Series1").
- Display the legend on the left side of the chart.
- Save the chart as a new chart sheet named **Average Sun**.

- Format the data series columns with the texture **purple mesh**. Clicking on a texture in the texture dialog window will display its name.
- Format the background chart area of the chart with the texture **white marble**.

4. You want to be able to easily compare your new weather data with the weather data you collected earlier.

- Copy the spreadsheet file which contains your Key West data to the folder that you created for this bridge task. This file must contain all of the data and formatting from the classwork and homework exercises for days 20 through 24. (Note: Be sure to make a **copy** of this file. If you have to retake this bridge task, you will need to use this file again!)
- Use Help to find out how to copy a worksheet from one workbook to another workbook.
- Copy the **worksheet** which contains the Key West weather data (not the histogram, pivot table, or any charts) to the workbook that contains the weather data that you imported for this bridge task. If you do this correctly, the file which contains the data you imported for this bridge task will also contain a sheet with the Key West data.
- Rearrange the columns in the worksheet containing the data which you imported for this bridge task.
 - The columns containing the data you imported in question **1** of the bridge task must be in the same order as the columns of data in your Key West weather worksheet (i.e. if TMAX24 data is in column **E** in your Key West worksheet, it should also be in column **E** of your new worksheet).
 - Put the converted temperature data from question **2** of the bridge task in the columns which correspond to the columns that contain MAX TEMP F and MIN TEMP F in your Key West worksheet. Note: do not be concerned if your two worksheets use different temperature scales. Just verify that the MAX and MIN columns correspond.
 - There is no sunrise / sunset data for the city you imported in question **1** of the bridge task so do not worry about those columns.

5. You want to look closer at some of the weather data for the two locations in your workbook.

Use Help and your textbook to find out how to reference cell addresses from other worksheets in a workbook.

On a new blank sheet in the weather workbook you are editing for this bridge

task, add the following entries:

- In cell B1, enter the location name of the city whose weather data you imported for this bridge task
- In cell C1, enter the location name of the city whose weather data you imported on Day 20 and are using for this bridge task
- In cell A2, enter the text "Average minimum:"
- In cell B2, use a FUNCTION to calculate the average daily minimum temperature for the city whose data you imported for this bridge task. Use the column of minimum daily temperatures that you calculated for this bridge task, not the raw data which you imported. This function must correctly reference data on the sheet containing that city's data.
- In cell C2, use a FUNCTION to calculate the average daily minimum temperature for the city whose data you imported on Day 20. Use the column of minimum daily temperatures that you converted to Fahrenheit, not the raw data which you imported. This function must correctly reference data on the sheet containing that city's data.
- Rename this worksheet **Minimum averages**.

This worksheet must have text, data or formulas only in the five cells specified above.

Check your files before you hand them in.

After completing the file for this bridge task, use the Handin program to hand in the Excel file you edited. You should handin ONE spreadsheet file consisting of a chart sheet and at least 3 worksheets which contain data. You have completed the 3.0 bridge task. Your feedback will be available within a few days.

Sample 3.0 Track D BT From Fall 1998

Note: Many times students fail bridge tasks not because they do not understand how to do the tasks, but because they do not carefully read the bridge task. Be sure that you take the time to

READ THE INSTRUCTIONS CAREFULLY

and double check your work when you are through.

SAVE EARLY, SAVE OFTEN!!!

If the next question does not specify a new file, continue making changes to the file you used in the previous question. When you are done, SAVE YOUR WORK!!!

Check your bridge Task. There should be six questions. If you have fewer than six questions, notify the proctor immediately.

You have finally had enough of your pointy haired boss and his legion of evil minions. You're ready to pack it in and head for the Bahamas where you'll do nothing but sunbathe and build sandcastles until your money runs out. All you need to do before you leave is clean up a few financial matters, and you'll finally be free of the travails and tedium of computing.

1. You don't want your pointy haired boss finding out about your plans to abandon the company, so create a new folder in your AFS so that he won't be able to read the files stored there. Name this folder **MyEscape_MMDD where *MM* is the current month and *DD* is the current day.**

For example, if today was February 14 the name of the folder would be **MyEscape_0214**.

Copy the credit card spreadsheet you worked on from Day 20 through Day 23 to the folder that you just created. Rename your credit card spreadsheet to **CreditCard_MMDD.xls** using the same value for *MMDD* as you used for the folder name.

Copy the mortgage spreadsheet you worked on from Day 21 through Day 23 to the folder that you just created. Rename your mortgage spreadsheet to **Mortgage_MMDD.xls** using the same value for *MMDD* as you used for the folder name.

You are responsible for insuring that all of the calculations and formatting required for all exercises with these files were completed correctly. (**Note: It is important to make a copy of the file for this bridge task rather than editing the original file in your AFS space. If you need to retake this bridge task, you will need to have the original file again!**)



2. For this question, you will be working with the copy of the credit card spreadsheet.

You have enormous debts, and you need to determine exactly what amount you currently owe. Your bank gave you a teaser rate to lure you into using their credit card, which they changed after you had become dependent on it. After a year of loyal charging, they have decided to reduce the interest rate they will charge you for the remainder of the time that you hold their credit card. Your credit card spreadsheet already includes calculations for the **teaser rate** and the **full rate**. You need to verify that you have the correct interest rate values in your calculations and that you have the correct values for the opening value, monthly payment, and new purchases. You will need to add data and the appropriate calculations for the **new rate**.

- Extend the spreadsheet to include payment numbers 13 through 19.
- Insert a new row at the top of the spreadsheet to hold the **new rate**.
- Teaser Rate (Payments 1 through 6) = 9.0% APR
- Full Rate (Payments 7 through 12) = 18% APR
- New Rate (Payments 13 through 19) = 11% APR
- The three rates must appear in only one cell each in the spreadsheet. (You may have separate cells for the APR and monthly rates if you wish.)
- You must use **absolute cell references** when referencing interest rates in formulas.
- Your opening balance must be \$500.00.

The monthly payment and new purchase values are as follows.

Payment Number	New Purchases	Payment
1	\$300.00	\$250.00
2	\$300.00	\$250.00
3	\$300.00	\$250.00
4	\$300.00	\$250.00
5	\$300.00	\$250.00
6	\$300.00	\$250.00
7	\$250.00	\$275.00
8	\$250.00	\$275.00
9	\$250.00	\$275.00
10	\$250.00	\$275.00
11	\$250.00	\$275.00
12	\$250.00	\$275.00
13	\$200.00	\$175.00
14	\$200.00	\$175.00
15	\$200.00	\$175.00
16	\$200.00	\$175.00
17	\$200.00	\$175.00
18	\$200.00	\$175.00
19	\$200.00	\$175.00

3. You need to create a chart of the data in your **credit card** spreadsheet to track the balance on your credit card over time to see if you'll pay it off before you run away. The chart must adhere to the following criteria:.

- It must appear on a new sheet.
- It must be a **line graph** (use the default format - Line with markers displayed at each data value).
- It must use the **Payment Number** for the labels for the **x-axis**.
- It must plot only the columns **Opening Balance** and **Payment Amount**.
- You must give the chart an appropriate title.
- You must give the X and Y axes titles for Amount and Payment Number.
- It must have a legend with labels for Opening Balance and Payment Amount.
- You must **rename the sheet** that the chart appears on to "**My Balance**".

4. You want to buy a house in the Bahamas so you can have wild parties every weekend. Since you have your Monthly Payment and Loan Amortization spreadsheet from CSE 101 you can determine how much your payments will be without doing much work. You decide to purchase a house advertised on the web. From the **CSE 101 Day 25** web page there is a link to the **Adams Realty**

site; find the listing with the address **1313 Mockingbird Lane** and follow the instructions given there.

5. You want to summarize some data from your mortgage and loan amortization so you can see how much it will cost you to escape your thankless job. On a **new sheet** in your **mortgage** workbook you must make the following entries:

- In cell A1 enter the text "Total Monthly Payment:"
- In cell A2 enter the text "Total Principal Paid:"
- In cell B1: You must enter a formula using absolute cell references that references the total monthly payment made on the loan, including insurance and taxes. Basically, you want the value from the monthly mortgage payment sheet to appear in this cell without typing the value into this cell.
- In cell B2: You must enter a formula using absolute cell references that references the total cumulative principal paid. Basically, you want the value from the loan amortization sheet for cumulative principal at the month the loan is paid off (this is the month in which the loan balance goes negative). This value must appear on your new sheet without typing the value into the cell on the new sheet.
- **Rename the sheet** containing this information "**Total Paid**".

6. You want to know **exactly** how many more days you have to work before the beach house mortgage is completely paid off. Your last day of work is the day that the mortgage Closing Balance goes to \$0 or a negative number. The only holiday you have is Independence Day (the Fourth of July).

- Use Help to find a function in Excel which will allow you to calculate the number of workdays from today until the mortgage is paid off.
- Add a new row at the top of your loan amortization worksheet.
- In cell B1, use the appropriate function to calculate the number of work days until the loan is paid off.

Check your files before you hand them in.

After completing the file for this bridge task, use the Handin program to hand in both of the Excel files you edited. You should handin TWO files. You have completed the 3.0 bridge task. Your feedback will be available within a few days.

APPENDIX C EVALUATION CRITERIA FOR SAMPLE BRIDGE TASKS

This appendix lists the evaluation criteria for each of the sample bridge tasks in Appendix B. Each bridge task consists of multiple dimensions. Each dimension has multiple criteria associated with it. (See Figure 6.) Each of these criteria are either mandatory or optional. To pass a dimension, a student must pass all of the mandatory criteria and some specified number of the optional criteria. Similarly, each dimension is mandatory or optional. To pass the bridge task, a student must pass all mandatory dimensions and some specified number of optional dimensions.

For each of the sample BTs in Appendix B, the tables list each dimension and if it is mandatory or optional. For each dimension, it lists the criteria and whether the criteria are mandatory or optional. The mandatory criteria are listed as "M." For the optional criteria, it indicates the number of optional criteria that must be passed, and the total number of optional criteria for that dimension. For example, if a dimension has five optional criteria and students must pass at least three of the optional criteria to pass the dimension, they are listed as "3/5."

Sample 1.0 BT Criteria

Table 74 shows the criteria for the sample 1.0 BT. It has seven dimensions: five are mandatory and two are optional. To pass, a student must pass all five mandatory and at least one of the two optional dimensions.

Table 74

Sample 1.0 Bridge Task Evaluation Criteria

Dimension			Evaluation
Num.	M / O	M / O	Criteria
0	M	M	The files handed in by the student were stored in a folder named Report. Capitalization does not matter.
		1 / 2	The Report folder is located in a public folder.
		1 / 2	The Report folder is in the student's PUBLIC folder and not in the WEB folder.
1	M	M	There must be a file named "Fax-Jokes" in the files handed in by the student. Capitalization does not matter.
		M	A file comparison shows that this file is identical to the original file, thus it was copied correctly. Grader note: See the grader web pages for instructions on doing the file comparison.
2	M	M	There must be a file named "Headhunters" in the files handed in by the student. Capitalization, spacing and file extensions do not matter.
		M	A file comparison shows that this file is identical to the original file (hightech.dot) thus it was copied and renamed correctly. Grader note: See the grader web pages for instructions on doing the file comparison.
3	M	M	There must be a file named "Partner" in the files handed in by the student.
		M	The file must have the extension ".txt".
		M	The file must contain text that says something like: "My assigned partner is _____". (The blank can be filled in with a person's name.)
4	M	M	The text of the file that the student handed in for the previous question must contain a sentence like "The path to my AFS space is _____" and include a path at the end.

(table continues)

Table 74 (cont'd).

Dimension			Evaluation
Num.	M / O	M / O	Criteria
		M	The path at must include: \MSU\USER\first letter in Pilot ID\second letter in Pilot ID\Pilot ID. It may include MSU.EDU instead of MSU. It may or may not include the folder for this BT or the PUBLIC folder. The separator (\ or /) is not important on this criteria
		1 / 2	The drive letter U: is included at the beginning of the path.
		1 / 2	The backwards slash (\) is used between each folder.
5	O	M	The text of the file that the student handed in for the previous question must contain a sentence like "You can delete CAMPUSMAP.BMP."
6	O	M	The text of the file that the student handed in for the previous question must contain a sentence like "Finding the CSE 101 web site search engine:" followed by a list of hyperlinks.
		M	The list contains the Student Pages for a track or the Site Map for a track.
		1 / 2	One of the following and only one of the following criteria is true: The list ends with "Search CSE101 WEB".
		1 / 2	OR the list ends with "Search the CSE 101 Website".

Note: The column heading "Num" refers to the dimension number of the BT, which are zero-ordinal. The column headings "M / O" under dimensions indicate if the dimension is mandatory or optional. The column headed "M / O" under evaluation criteria indicate "M" for mandatory criteria and the number of optional criteria needed to pass out of the total number of optional criteria (i.e., 1 / 2 is one required of two total optional.)

Sample 1.5 BT Criteria

Table 75 shows the criteria for the sample 1.5 BT. It has nine dimensions: six are mandatory and three are optional. To pass, a student must pass all six mandatory and at least two of the three optional dimensions.

Table 75

Sample 1.5 Bridge Task Evaluation Criteria

Dimension			Evaluation
Num.	M / O	M / O	Criteria
0	M	2 / 3	There is a file named something like gofer.htm handed in by the student.
		2 / 3	The web page was handed in from the student's WEB folder.
		2 / 3	The title of the student's web page is something like "The Office Retriever".
1	O	M	Something like "I like being an intern" is at the top of the student's web page.
2	O	2 / 3	The text is formatted as Heading 1.
		2 / 3	The text is a different color than the default color. There should be tags around this text, where XXXXXX is a hexadecimal color code, indicating that the font color was changed.
		2 / 3	The text is formatted as center aligned.
3	M	M	There is a paragraph on the student's web page that begins with something like "Down in the south where bananas grow."
		M	The word "banana" in the paragraph is a link.

(table continues)

Table 75 (cont'd).

Dimension			Evaluation
Num.	M / O	M / O	Criteria
		M	Clicking on the link takes you to the web site for Bobby Banana. The URL for Bobby Banana is http://www.dole.com/bobby/index2.ghtml .
4	M	2 / 3	The first sentence of the paragraph on the student's web page is the font Book Antiqua.
		2 / 3	The first sentence of the paragraph on the student's web page is italic.
		2 / 3	The rest of the paragraph on the student's web page is neither Book Antiqua or italic.
5	O	2 / 3	There is a horizontal line on the student's web page.
		2 / 3	The line is between the header and the paragraph on the page.
		2 / 3	The width of the line is 250 pixels.
6	M	M	There is a sentence something like "The URL for this page is:" at the bottom of the student's web page which ends with a URL.
		M	Copy the URL at the bottom of the student's web page into the Location field of Netscape Navigator. The student's web page must appear.
7	M	M	A text file was handed in from a folder named something like "Apply".
		M	The folder named "Apply" is in a private folder, but is not in the .ce, .elm, or mail folder.
8	M	M	The student handed in a file named something like "learning.txt".
		M	Open the file with Notepad. The search must have returned 10 or fewer records.

(table continues)

Table 75 (cont'd).

Dimension			Evaluation
Num.	M / O	M / O	Criteria
8	M	M	The student handed in a file named something like "learning.txt".
		M	Open the file with Notepad. The search must have returned 10 or fewer records.
		M	The records in the file must relate to music and learning about computers in some way.
		M	The records in the file are from the ERIC database.

Note: The column heading "Num" refers to the dimension number of the BT, which are zero-ordinal. The column headings "M / 0" under dimensions indicate if the dimension is mandatory or optional. The column headed "M / O" under evaluation criteria indicate "M" for mandatory criteria and the number of optional criteria needed to pass out of the total number of optional criteria (i.e., 1 / 2 is one required of two total optional.)

Sample 2.0 BT Criteria

Table 76 shows the criteria for the sample 2.0 BT. It has nine dimensions: six are mandatory and three are optional. To pass, a student must pass all six mandatory and at least two of the three optional dimensions.

Table 76

Sample 2.0 Bridge Task Evaluation Criteria

Dimension			Evaluation
Num.	M / O	M / O	Criteria
0	M	M	The student handed in two Word documents from a folder named something like "Dogbert's Files".
		M	The folder "Dogbert's Files" is in a private folder.
1	M	M	The student handed in a Word file that contains the constitution.
2	M	M	The document contains a style named something like "Main Text 1" and a style named something like "The Beginning".
		M	The preamble is formatted with the style "The Beginning".
		M	The text of Article 1 is formatted with a style named something like "Main Text 1". This style was defined in a class exercise.
		5 / 7	The style "The Beginning" must be font Matura MT Script Capitals.
		5 / 7	The style "The Beginning" must be font formatted as engraved.
5 / 7	The style "The Beginning" must be font formatted as blue.		

(table continues)

Table 76 (cont'd).

Dimension			Evaluation
Num.	M / O	M / O	Criteria
		5 / 7	The style "The Beginning" must be font formatted as size 18 point.
		5 / 7	The style "The Beginning" must be paragraph formatted with a 1 inch hanging indent.
		5 / 7	The style "The Beginning" must be paragraph formatted with a 16 point line spacing.
		5 / 7	The style "The Beginning" must be paragraph formatted with 12 points after the paragraph.
3	M	5 / 6	In the preamble of the document, after the text "Justice", there must be a footnote.
		5 / 6	The footnote must be numbered "1".
		5 / 6	The footnote must contain something like "We love the Minister of the Snack Machine"
		5 / 6	After the text "Tranquility", there must be an endnote.
		5 / 6	The endnote must be numbered "i".
		5 / 6	The endnote must contain something like "Dogbert is very clever".
4	O	2 / 3	There is a bulleted list at the end of the student's document.
		2 / 3	There are five items in the bulleted list, starting with something like "Dogbert".
		2 / 3	The "bullets" in the list are not round bullets.
5	O	3 / 5	There is a table at the end of the student's document.
		3 / 5	The table consists of exactly 3 rows and exactly 2 columns.

(table continues)

Table 76 (cont'd).

Dimension			Evaluation
Num.	M / O	M / O	Criteria
		3 / 5	The first cell in the first column contains something like "Ruling Class" and all of the cells in the column contain text.
		3 / 5	The first cell in the second column contains something like "In-duh-viduals" and all of the cells in the column contain text.
		3 / 5	The text in the table is formatted using the style "Main Text 2".
6	M	M	The left and right margins of the document must be 0.75 inches.
7	O	M	The word "ratified" must not occur in the document.
8	M	M	There is a path at the end of the document.
		M	The path includes \msu\user\p\l\pilotid\ including the first two letters from the student's Pilot ID and where PILOTID is the student's Pilot ID.
		2 / 4	The path must begin with U:.
		2 / 4	The path must end with with a filename of the constitution file the student handed in for the BT. The filename must be spelled correctly; capitalization does not matter.
		2 / 4	The folder name after the student's pilot id in the path is the folder from which the file was handed in.
		2 / 4	The backwards slash (\) is used between each folder.

Note: The column heading "Num" refers to the dimension number of the BT, which are zero-ordinal. The column headings "M / O" under dimensions indicate if the dimension is mandatory or optional. The column headed "M / O" under evaluation criteria indicate "M" for mandatory criteria and the number of optional

criteria needed to pass out of the total number of optional criteria (i.e., 1 / 2 is one required of two total optional.)

Sample 2.5 BT Criteria

Table 77 shows the criteria for the sample 2.5 BT. It has five dimensions: three are mandatory and two are optional. To pass, a student must pass all three mandatory and at least one of the two optional dimensions.

Table 77

Sample 2.5 Bridge Task Evaluation Criteria

Dimension			Evaluation
Num.	M / O	M / O	Criteria
0	M	M	The student handed in an Excel file that contains the payroll exercise.
1	M	8 / 11	There are eight employees listed in the student's payroll spreadsheet.
		8 / 11	There is an employee named something like "Alvin Kurtzweil" (Grader: accept "Kurtzweil, Alvin", "Kurtzweil Alvin" and other variations) listed in the student's payroll spreadsheet.
		8 / 11	The cell which contains Alvin Kurtzweil's community fund contribution is formatted as either Accounting or Currency.
		8 / 11	There must be a formula in a cell which calculates the average community fund contribution.
		8 / 11	The average community fund contribution formula uses the AVERAGE function
8 / 11		The parameter of the AVERAGE function must be the range of values for the employees (something like I2:I7 that includes the appropriate cells for community fund). Blank cells may be included in this range.	

(table continues)

Table 77 (cont'd).

Dimension			Evaluation
Num.	M / O	M / O	Criteria
		8 / 11	The average community fund contribution must be \$79.44 and formatted as an accounting or currency value.
		8 / 11	There must be a formula in a cell which calculates the average net pay.
		8 / 11	The average net pay formula uses the AVERAGE function.
		8 / 11	The parameter of the AVERAGE function must be the range of values for the employees (something like I2:I7 that includes the appropriate cells for net pay). Blank cells may be included in this range.
		8 / 11	The average net pay must be \$988.48 and formatted as an accounting or currency value
2	O	M	The HYPERLINK function appears in cell A1 of the student's payroll spreadsheet.
		2 / 4	There is text displayed in cell A1 of the student's payroll spreadsheet.
		2 / 4	The text displayed in cell A1 is something like "Tax Collectors".
		2 / 4	Clicking on the link opens a web page.
		2 / 4	The web page that the link opens is http://www.irs.gov .
3	O	M	The chart contains only net pay data and employee names.
		6 / 8	The student's spreadsheet file contains a chart worksheet.
		6 / 8	The chart worksheet is named something like "Net Pay Chart".

(table continues)

Table 77 (cont'd).

Dimension			Evaluation
Num.	M / O	M / O	Criteria
		6 / 8	The chart sheet contains a pie chart.
		6 / 8	The title of the chart is something like "Net Pay of Employees".
		6 / 8	There is a legend on the chart which contains employee names. If the legend contains "Series 1", "Series 2", etc. the student fails this criteria.
		6 / 8	The chart legend is on the left side of the chart.
		6 / 8	There are data labels next to each pie wedge.
		6 / 8	The data labels displayed contain the percentage of the chart that each wedge represents.
4	M	M	The student handed in a file named something like "Production.xls"
		M	The worksheet contains the names of four continents.
		M	The worksheet contains a cell which contains a value for the total revenue.
		M	The worksheet contains a cell which contains a value for the average number of widgets sold.
		13 / 17	There is a cell that contains text that is something like "Continent".
		13 / 17	The cell that contains "Continent" is formatted with the font Impact.
		13 / 17	The cell that contains "Continent" is formatted as italic.
		13 / 17	The cell that contains "Continent" is formatted with the font color green. (Grader: as long as the font color has been changed from the default of Black, this passes.)
		13 / 17	"Antarctica" has a value of 225,000 widgets sold.

(table continues)

Table 77 (cont'd).

Dimension			Evaluation
Num.	M / O	M / O	Criteria
		13 / 17	Antarctica has a value for revenue.
		13 / 17	The cell with the revenue value for Antarctica contains a formula.
		13 / 17	The value in the revenue cell for Antarctica is \$168,750.00.
		13 / 17	Antarctica has a value for the projected widget shipping.
		13 / 17	The cell with the projected widget shipping value for Antarctica contains a formula.
		13 / 17	The value in the projected widget shipping cell for Antarctica is 236,250.
		13 / 17	The cell with the total revenue value contains the SUM FUNCTION.
		13 / 17	The parameter of the SUM function must be the range of values for all four of the continents (something like I2:I7 that includes the appropriate cells for revenue). Blank cells may be included in this range.
		13 / 17	The value in the total revenue cell is \$380,250.00.
		13 / 17	The cell with the average number of widgets sold value contains the AVERAGE FUNCTION.
		13 / 17	The parameter of the AVERAGE function must be the range of values for the continents (something like I2:I7 that includes the appropriate cells for widgets sold). Blank cells may be included in this range.
		13 / 17	The value in the average number of widgets sold cell is 126,750.

(table continues)

Table 77 (cont'd).

Dimension			Evaluation
Num.	M / O	M / O	Criteria
5	M	M	The student handed in an Excel file containing a computer specifications worksheet.
		M	The computer specifications worksheet contains data on three desktop models. This data was gathered in a class exercise on Day 17.
		M	The computer specifications worksheet contains data on one laptop model. This data was gathered for the Day 17 homework.
		5 / 7	There is a label something like "CD/DVD speed".
		5 / 7	The cells containing the CD/DVD speed have entries for all three desktop models and the laptop model.
		5 / 7	There is a label something like "Printer Brand" or "Printer Manufacturer."
		5 / 7	The cells containing the "Printer Brand" or "Printer Manufacturer" has entries for all three desktop models. It is OK if the data for the laptop computer is missing.
		5 / 7	The spreadsheet contains a Line-Column on 2 Axes chart graphing Hard Disk Capacity and Price. This chart appears on a separate chart sheet. This chart was created as an in-class exercise.
		5 / 7	The chart has Hard Disk Capacity as the columns and Price as the line. No additional data are charted.
6	M	M	The text in cell A1 correctly indicates the computer model or models that have: 1) The largest amount of RAM; 2) The largest capacity hard disk drive; and 3) The largest size monitor. Graders: See the grading web pages for details.

(table continues)

Table 77 (cont'd).

Dimension			Evaluation
Num.	M / O	M / O	Criteria
		4 / 6	In the computer specifications spreadsheet there are cells labeled something like "Amount of RAM" (or Memory).
		4 / 6	All of the cells containing the "Amount of Ram" for the machine(s) with the most memory are highlighted. No cells with values other than the maximum memory are highlighted.
		4 / 6	In the computer specifications spreadsheet there are cells labeled something like "Hard Disk Capacity".
		4 / 6	All of the cells containing the hard disk capacity for the machine(s) with the most storage are highlighted. No cells with values other than the maximum storage are highlighted.
		4 / 6	In the computer specifications spreadsheet there is a set of cells labeled something like "Monitor Size."
		4 / 6	All of the cells containing the "Monitor Size" for the machine(s) with the largest monitor are highlighted. No cells with values other than the largest monitor(s) are highlighted.
7	M	M	In cell A2, all computers with at least 64 MB of RAM and 16.4 GB of Hard Disk space are listed. No computers with less than these values are listed.

Note: The column heading "Num" refers to the dimension number of the BT, which are zero-ordinal. The column headings "M / O" under dimensions indicate if the dimension is mandatory or optional. The column headed "M / O" under evaluation criteria indicate "M" for mandatory criteria and the number of optional criteria needed to pass out of the total number of optional criteria (i.e., 1 / 2 is one required of two total optional.)

Sample 3.0 Track A BT Criteria

Table 78 shows the criteria for the sample 3.0 A BT. It has eight dimensions: four are mandatory and four are optional. To pass, a student must pass all four mandatory and at least two of the four optional dimensions.

Table 78

Sample 3.0 A Bridge Task Evaluation Criteria

Dimension			Evaluation
Num.	M / O	M / O	Criteria
0	M	M	The student handed in all of their Bridge Task files from a folder named something like MyEscape_MMDD (where MM is the month that their files were handed in and DD is the day their files were handed in) in their web folder in AFS.
1	M	M	All of the web pages in the student's web site use the template Gothic_template.htm
2	O	M	All of the text in the web site is formatted with the correct font for the template used.
		M	Select one of the level 2 pages from the student's web site. Verify that it contains a target near the Page Top location. Grader note: Refer to the sample templates for the correct location on each template.
		M	On this page, text something like "top of the page" appears and is a link.
3	M	M	On this page, the text something like "top of the page" links to the target at the top of the page.
		M	A file with a filename similar to stars.htm must be handed in by the student.
		M	A file with a filename similar to north.htm must be handed in by the student.

(table continues)

Table 78 (cont'd).

Dimension			Evaluation
Num.	M / O	M / O	Criteria
		M	A file with a filename similar to south.htm must be handed in by the student.
		M	A file with a filename similar to porcus.htm must be handed in by the student.
		M	A file with a filename similar to lacerta.htm must be handed in by the student.
		M	A file with a filename similar to indoles.htm must be handed in by the student.
		M	A file with a filename similar to currus.htm must be handed in by the student.
8 / 13			The stars.htm web page must have text that is something like "Stars" in the "Page Top" position.
8 / 13			The stars.htm web page must have a link to the north.htm page that is not in the Navigation Tool table. Verify the link works. There may be an additional link to this page in the student's navigation tool.
8 / 13			The stars.htm web page must have a link to the south.htm page that is not in the Navigation Tool table. Verify the link works. There may be an additional link to this page in the student's navigation tool.
8 / 13			The north.htm web page must have text that is something like "Northern Sky" in the "Page Top" position.
8 / 13			The page title of the north.htm page is something like "Northern Sky".
8 / 13			The north.htm web page must have links to both the porcus.htm web page and the lacerta.htm web page that are not in the Navigation Tool. Verify the links work. There may be an additional link to this page in the student's navigation tool.

(table continues)

Table 78 (cont'd).

Dimension			Evaluation
Num.	M / O	M / O	Criteria
		8 / 13	The north.htm web page must have a sentence in the "Body Text" position that is something like "These are constellations of the Northern Hemisphere."
		8 / 13	The porcus.htm web page must have a block of text in the "Body Text" position that begins with something like "The constellation of the pig commemorates the memory of a very important pig."
		8 / 13	The porcus.htm web page must have a link to the north.htm page that is not in the Navigation Tool table. Verify the link works. There may be an additional link to this page in the student's navigation tool.
		8 / 13	The currus.htm web page must have a block of text in the "Body Text" position that begins with something like "The constellation of the chariot was first described by the priests of Chilton."
		8 / 13	The currus.htm web page must have a link to the south.htm page that is not in the Navigation Tool table. Verify the link works. There may be an additional link to this page in the student's navigation tool.
		8 / 13	The currus.htm web page must have something like "The Chariot" in the "Page Top" position.
		8 / 13	The page title of the currus.htm page is something like "The Chariot".
4	O	M	The spotLinks java applet is displayed on the site home page.
		4 / 8	The applet displays exactly 4 buttons.
		4 / 8	The text on the first button (the topmost button in the applet) is a different color than the text on the other buttons in the applet.

(table continues)

Table 78 (cont'd).

Dimension			Evaluation
Num.	M / O	M / O	Criteria
		4 / 8	The fourth button (the bottommost button in the applet) displays text something like "Seinfeld-o-matic".
		4 / 8	Clicking on the fourth button opens a new web page. This may or may not open in a new browser window.
		4 / 8	The page that is opened is the Seinfeld-o-matic web page. The URL for the Seinfeld-o-matic web page is http://www.student.com/feature/seinfeldomatic
		4 / 8	The buttColor PARAM tag has the value "251,4,66".
		4 / 8	The color of the text on the buttons changes when the mouse is moved over the buttons.
		4 / 8	The entire applet is visible on the web page (the WIDTH and HEIGHT values are large enough to display the entire applet).
5	O	M	The image is from the Day 24 Track A BT image file web pages.
		M	The SRC value of the IMG tag is of the form SRC="filename.*" or SRC="http://www.msu.edu/~pilotid/... ../filename.*" where * is either a jpg or gif extension.
		6 / 8	An image appears on the site home page of the student's web site.
		6 / 8	The image has a solid border of 5 pixels.
		6 / 8	The image is a link.
		6 / 8	Words something like "Adams, Inc. Home Page" appear on the status bar of Navigator when the mouse is moved over the image.

(table continues)

Table 78 (cont'd).

Dimension			Evaluation
Num.	M / O	M / O	Criteria
		6 / 8	The words something like "Adams, Inc. Home Page" on the status bar are removed when the mouse is moved off the image.
		6 / 8	The link option of the image contains extra HTML.
		6 / 8	Clicking on the link opens the "Catbert's Anti-career Zone" home page. The URL of the "Catbert's Anti-career Zone" home page is http://www.unitedmedia.com/comics/dilbert/career/ .
		6 / 8	The image appears in a table which has exactly one row and exactly one column.
6	M	M	There is a table that is a navigation tool on each web page in the student's web site.
		M	Open a Level 2 page from the student's web site in Netscape Composer. Each of the links in the navigation tool must be in its own cell.
		M	Verify that there is a correctly functioning link in the Navigation Tool to every Level 1 page in the student's web site.
		M	The navigation tool must contain a link which opens the Centre for the Easily Amused Home page (the URL for the CEA home page is http://www.amused.com).
		2 / 3	The navigation tool's border width is 1.
		2 / 3	The navigation tool's cell padding is 1.
		2 / 3	The text in the cells of the navigation tool is right aligned.

(table continues)

Table 78 (cont'd).

Dimension			Evaluation
Num.	M / O	M / O	Criteria
7	O	M	Clicking on the link takes you to another page in the web folder of the student's AFS space. The next five criteria deal with this MyMSU site, which was used during class exercises and homework.
		6 / 8	This page contains a navigation tool which uses a table.
		6 / 8	The links in the navigation tool function correctly on all pages in the site.
		6 / 8	There are at least 7 pages in this site.
		6 / 8	All of the pages in the site use the same background graphics/colors and have the same link colors.
		6 / 8	There is a functional java applet on the MyMSU home page. This must not be displayed as an empty gray box and must not display any error messages.
		6 / 8	Something like "Site design by (student's name). copyright 1999, CCC Web Design" appears at the bottom of the site's Home Page.
		6 / 8	The student's name is a link.
		6 / 8	The URL in the link begins with "http://".

Note: The column heading "Num" refers to the dimension number of the BT, which are zero-ordinal. The column headings "M / O" under dimensions indicate if the dimension is mandatory or optional. The column headed "M / O" under evaluation criteria indicate "M" for mandatory criteria and the number of optional criteria needed to pass out of the total number of optional criteria (i.e., 1 / 2 is one required of two total optional.)

Sample 3.0 Track C BT Criteria

Table 79 shows the criteria for the sample 3.0 C BT. It has five dimensions: three are mandatory and two are optional. To pass, a student must pass all three mandatory and at least one of the two optional dimensions.

Table 79

Sample 3.0 C Bridge Task Evaluation Criteria

Dimension			Evaluation
Num.	M / O	M / O	Criteria
0	M	M	A spreadsheet file containing a worksheet with weather data from Athens, Georgia was handed in from the student's private AFS space.
		M	The worksheet which contains the Georgia data has separate columns for DAY, MONTH, and YEAR.
		M	The worksheet which contains the Georgia data has a single column for location. Grader note: verify that there are not separate columns for each word in the location name.
1	M	M	The MAX TEMP C and MIN TEMP C columns must use the built in Function IF.
		M	The MAX TEMP C and MIN TEMP C columns must use the built in Function ROUND. This function can be used either outside or inside the IF Function like this: 1) =ROUND(IF()) or 2) =IF(ROUND()).
		M	The condition in the IF function refers to cells in the SCALE column.
		M	A mixed cell reference is used to refer to the SCALE column in the IF function.

(table continues)

Table 79 (cont'd).

Dimension			Evaluation
Num.	M / O	M / O	Criteria
		2 / 3	The spreadsheet must contain columns for MAX TEMP C and MIN TEMP C. The names do not need to be exact.
		2 / 3	Rows with a SCALE value of "C" display the same values in the TMAX24 and MAX TEMP C columns, and also display the same values in the TMIN24 and MIN TEMP C columns.
		2 / 3	Rows with a SCALE value of "F" display correctly converted values in the MIN TEMP C and MAX TEMP C columns on the Celsius scale.
2	M	M	The student's workbook contains a worksheet named something like "Sun Table".
		M	The "Sun Table" worksheet contains a pivot table.
		M	The values in the pivot table are grouped by month with the month numbers on the column labels.
		M	The pivot table contains data from 365 rows of the student's data worksheet.
		M	The data section of the pivot table contains the AVERAGE of the PSUN24 values from the student's worksheet. The data section contains no other data values.
		M	The student's workbook contains a chart sheet named something like "Average Rain" or "Average Sun".
		M	The chart sheet contains a column chart.
		M	The data series in the chart is based on the AVERAGE values of the pivot table. Grader note: The graph may or may not include the Grand Total column from the pivot table

(table continues)

Table 79 (cont'd).

Dimension			Evaluation
Num.	M / O	M / O	Criteria
		M	There is only one data series in the chart.
		3 / 5	The column chart has a chart title, x-axis title, and y-axis title.
		3 / 5	The column chart contains a legend on the left side of the chart.
		3 / 5	The legend text comes directly from a cell in the pivot table datasheet; it does not contain the text "Series x" (where x is some number.)
		3 / 5	The columns of the chart are formatted with the texture "purple mesh".
		3 / 5	The background chart area of the chart is the texture "white marble".
3	O	7 / 9	There is a separate worksheet in the student's workbook which contains weather data for Key West. Grader note: this data must be on a different worksheet than the data from the previous criteria.
		7 / 9	The Key West data sheet contains at least 14 columns of data.
		7 / 9	The Key West data sheet contains columns labelled with something like MAX TEMP F and MIN TEMP F.
		7 / 9	The MAX TEMP F and MIN TEMP F columns use the IF function.
		7 / 9	The MAX TEMP F and MIN TEMP F columns use the ROUND function. The ROUND function can be used either inside the IF function [=IF(ROUND())] or outside the IF function [=ROUND(IF())].

(table continues)

Table 79 (cont'd).

Dimension			Evaluation
Num.	M / O	M / O	Criteria
		7 / 9	The worksheet contains a column labelled something like "Daylight".
		7 / 9	The Daylight column contains a formula of the form "=Sunset-Sunrise".
		7 / 9	The values in the Daylight column are formatted as TIME values.
		7 / 9	The columns of data in the two weather worksheets are in the same order.
4	O	M	There is a worksheet called something like "Minimum averages" in the student's workbook.
		M	Cell B2 contains a formula which uses the AVERAGE function.
		M	The AVERAGE function references a range of at least 365 cells on another worksheet in the student's workbook.

Note: The column heading "Num" refers to the dimension number of the BT, which are zero-ordinal. The column headings "M / O" under dimensions indicate if the dimension is mandatory or optional. The column headed "M / O" under evaluation criteria indicate "M" for mandatory criteria and the number of optional criteria needed to pass out of the total number of optional criteria (i.e., 1 / 2 is one required of two total optional.)

Sample 3.0 Track D BT Criteria

Table 80 shows the criteria for the sample 3.0 D BT. It has six dimensions: three are mandatory and three are optional. To pass, a student must pass all three mandatory and at least two of the three optional dimensions.

Table 80

Sample 3.0 D Bridge Task Evaluation Criteria

Dimension			Evaluation
Num.	M / O	M / O	Criteria
0	M	M	A file containing the student's credit card spreadsheet was handed in from the student's PRIVATE AFS space.
1	M	M	The final ending balance in payment number 19 must be \$979.67.
		5 / 7	The spreadsheet must contain columns for Payment Number, Finance Charge, New Purchases, Payments, and Ending Balance. The names do not have to be exact, but equivalent information has to exist. It may contain a column for Opening Balance, but this is not required.
		5 / 7	There must be entries for the teaser rate, full rate, and new rate. These values must have labels and appear in only ONE cell each (or two if they use an annual and monthly rate cell) in the spreadsheet. That is, there must not be columns in the spreadsheet specifically for the three rates.
		5 / 7	For Payments 1 through 6, the Finance Charge column must use absolute cell references to reference the monthly Teaser Rate. GRADER NOTE: An absolute cell reference is of the form \$B\$2.

(table continues)

Table 80 (cont'd).

Dimension			Evaluation
Num.	M / O	M / O	Criteria
		5 / 7	For Payments 7 through 12, the Finance Charge column must use absolute cell references to reference the monthly Full Rate. GRADER NOTE: An absolute cell reference is of the form \$B\$2.
		5 / 7	For Payments 13 through 19, the Finance Charge column must use absolute cell references to reference the monthly New Rate. GRADER NOTE: An absolute cell reference is of the form \$B\$2.
		5 / 7	In the Payment Number column, there must be a formula used to make the payment numbers automatically increment. For example, =A1+1, =A2+1, etc.
		5 / 7	There must be at least 19 payment rows in the spreadsheet.
2	M	M	The graph must contain exactly two lines, one for Opening Balance, one for Payment Amount.
		M	The legend must contain the labels for the lines: Opening Balance and Payment Amount.
		4 / 6	In the credit card spreadsheet, there must be a chart inserted on a new chart sheet named something like "My Balance".
		4 / 6	The chart must be a line graph.
		4 / 6	The X axis must contain the payment number 1 through the final payment number in their spreadsheet.
		4 / 6	The Y axis must contain dollar amounts.
		4 / 6	There must be a chart title.

(table continues)

Table 80 (cont'd).

Dimension			Evaluation
Num.	M / O	M / O	Criteria
		4 / 6	There must be axis titles. For example: Payment Number for the X Axis, Dollar Amounts for the Y Axis).
3	O	M	On the Monthly Payment sheet, the Monthly Payment on the Loan must be \$1434.30.
		M	On the Monthly Payment sheet, the Total Monthly Cost must be \$1609.30.
		M	The column labeled something like Monthly Interest must contain formulas using the built in function IF.
		M	The IF function must return a zero if Monthly Opening Balance is less than zero. Otherwise it returns a positive amount of interest.
		M	The column Monthly Interest must not contain any negative values.
		M	On the Loan Amortization sheet, the Monthly Closing Balance must become negative in payment 239.
		M	On the Loan Amortization sheet, there must be entries in the Additional Principal Payment column ONLY for payment numbers 5, 6, 7, 8, 9, and 10 all with the amount \$100.
		5 / 10	A file containing the student's mortgage spreadsheet was handed in from the student's PRIVATE AFS space.
		5 / 10	In the mortgage spreadsheet, there is a Monthly Payment sheet which contains a cell which calculates the Monthly Loan Payment on the Loan using the built-in function PMT.
		5 / 10	On the Monthly Payment sheet, the value for the Monthly Payment on the Loan must have a double underline.

(table continues)

Table 80 (cont'd).

Dimension			Evaluation
Num.	M / O	M / O	Criteria
		5 / 10	In the mortgage spreadsheet, there is a Loan Amortization sheet which contains a column labeled something like Payment Date. This column must contain dates that increment monthly down the column. Make sure they are not incrementing daily.
		5 / 10	Columns named something like Cumulative Principal and Cumulative Interest must be included on the Loan Amortization sheet.
		5 / 10	The columns Cumulative Interest and Cumulative Principal must contain formulas that keep a running balance of the Monthly Interest. The Interest formula must be of this form: =(Previous Month of Cumulative Interest) + (Current Monthly Interest). The Principal formula must be of this form: =(Previous Month of Cumulative Principal) + (Current Monthly Principal) + (Current Additional Principal Payment).
		5 / 10	There must be a column labeled something like Escrow Account Balance.
		5 / 10	The formula in the Escrow Account Balance column must use the built in function IF.
		5 / 10	On the Monthly Payment sheet and the Loan Amortization sheet, all entries that are dollar amounts must be formatted as Accounting.
		5 / 10	On the Monthly Payment sheet and the Loan Amortization sheet, all entries that are percentages must be formatted as Percentage with 2 decimal places.
		5 / 10	There must be a chart inserted as a new chart sheet that is a line graph plotting Interest and Principal.
4	O	M	There must be a sheet named something like "Total Paid" in the mortgage workbook.

(table continues)

Table 80 (cont'd).

Dimension			Evaluation
Num.	M / O	M / O	Criteria
		M	Cell B1 must contain a formula that uses an absolute cell reference to reference the cell containing the Total Monthly Payment from the mortgage worksheet. GRADER NOTE: An absolute cell reference is of the form ='Sheet Name'!\$B\$2. Grader Note: This value could be from the row in which the loan balance was paid off (a negative balance) or it could be the last row on the amortization sheet.
		M	Cell B2 must contain a formula that uses an absolute cell reference to reference the cell containing the ending Cumulative Principal Paid from the loan amortization sheet.
5	O	M	Cell B1 of the student's loan amortization worksheet contains a formula which uses the NETWORKDAYS function.
		2 / 4	The first argument of the NETWORKDAYS function is either the date on which the student took the bridge task or the NOW function.
		2 / 4	The second argument of the NETWORKDAYS function is the cell containing the date of the final loan payment. Grading note: Use the date value where the closing balance goes to zero or a negative number.
		2 / 4	The third argument of the NETWORKDAYS function is some date representation of the Fourth of July. This may be of the format "July 4", "7/4", "4 July", or something similar.
		2 / 4	The formula returns a value and not an error message.

Note: The column heading "Num" refers to the dimension number of the BT, which are zero-ordinal. The column headings "M / O" under dimensions indicate if the dimension is mandatory or optional. The column headed "M / O" under evaluation criteria indicate "M" for mandatory criteria and the number of optional

criteria needed to pass out of the total number of optional criteria (i.e., 1 / 2 is one required of two total optional.)

APPENDIX D STUDENT COURSE SURVEY AND SIRS QUESTIONS

This appendix contains the questions from the end-of-the-semester survey that students complete about the course and the Student Instructor Rating System (SIRS) questions that they complete about their TAs and ATAs. The surveys are completed on the course Web site. Students must enter their student numbers to verify their enrollment in the course and to ensure that each student completes the survey and SIRS only one time. The responses are anonymous and the course instructors and TAs do not see the summary results until the semester is complete.

To ensure a high completion rate, students must submit the surveys before Day 26 of the course. If they do not do so, the BT Feedback software will not work after Day 26 until they submit the surveys. They may elect to submit "no response" to and or all of the survey questions if they wish.

Course Survey Questions

The following questions are Likert scale questions. The response options for each question are Strongly Agree, Agree, Neither Agree or Disagree, Disagree, Strongly Disagree and No Response. For each question, student may also enter free-response comments if they wish.

1. I found the two textbooks to be helpful.
2. I found the CPS 101 Web pages to be helpful.
3. The computer facilities usually worked well.
4. I usually did my homework before coming to class.
5. I learned a lot in the group exercises in class.
6. I attended Help Room frequently.
7. I found Help Room to be useful.
8. The Bridge Tasks were a fair test of the material I learned in class.
9. I felt that the **extension tasks** on the Bridge Task, which asked me to do something I had not previously done, were reasonably connected to what I had already learned.
10. The grading was fair for the Bridge Tasks.
11. The grader's comments explained what I did wrong on the Bridge Tasks.
12. I feel that my course grade will reflect my understanding of the computer concepts we covered during the course.
13. I would recommend this course to my friends.

The following are free-response questions.

14. If you had it to do over again, what would you do differently to prepare for the Bridge Tasks?
15. What is the one thing you think most needs to be added to this course?
16. What are the five hardest concepts in this course?
17. What are the five easiest concepts in this course?
18. What is the worst part of this course?
19. What is the best part of this course?
20. What advice would you give to other students as they start CPS 101 to help them succeed?
21. What is the most important thing you want to tell the Instructors and the design team for this course?
22. How could you, as a student, have done a better job in this course?
23. What is your major?
24. Please enter any additional comments about CSE 101 that you have.

Lead TA SIRS

The following questions are Likert scale questions. The response options for each question are Strongly Agree, Agree, Neither Agree or Disagree, Disagree, Strongly Disagree and No Response. For each question, student may also enter free-response comments if they wish.

1. The Lead TA was available and willing to help the student.
2. The Lead TA explained course material clearly.
3. The Lead TA was well prepared for classes and enthusiastic about teaching the course section.
4. The Lead TA was organized and explained the materials for this section well and, generally, displayed a high-level of competence in the subject matter of this course.
5. The Lead TA communicated, in both written and oral modes, well and with ease.
6. The Lead TA was accessible and responded promptly to Email questions.

The response options for this question are 4.0 (excellent), 3.0 (good), 2.0 (average), 1.0 (below average) and 0.0 (poor). Students may also enter free-response comments if they wish.

7. Grade the Lead TA on the following scale:

Assistant TA SIRS

The following questions are Likert scale questions. The response options for each question are Strongly Agree, Agree, Neither Agree or Disagree, Disagree, Strongly Disagree and No Response. For each question, student may also enter free-response comments if they wish.

1. The assistant TA was available and willing to help the student.
2. The assistant TA explained course material clearly.
3. The assistant TA was prepared for class sessions and enthusiastic about teaching the course section.
4. The assistant TA was organized and explained the materials for this section well and, generally, displayed a high-level of competence in the subject matter of this course.
5. The assistant TA communicated, in both written and oral modes, well and with ease.

The response options for this question are 4.0 (excellent), 3.0 (good), 2.0 (average), 1.0 (below average) and 0.0 (poor). Students may also enter free-response comments if they wish.

6. Grade the Assistant TA on the following scale:

REFERENCES

- ACM Curriculum Committee on Computer Science. (1968). Curriculum 68: Recommendations for the undergraduate program in computer science. *Communications of the ACM*, 11(3), 151-197.
- ACM Curriculum Committee on Computer Science. (1979). Curriculum 78: Recommendations for the undergraduate program in computer science. *Communications of the ACM*, 22(3), 147-166.
- ACM Special Interest Group on Computer Science Education. (1998). ACM SIG fact sheet . http://www.acm.org/sigcse/sigcse_fact_sheet.html.
- ACT. (2000). 1999 ACT National and State Scores: Summary (Vol. 2000,). <http://www.act.org/news/data/99/tsum.html>: ACT.
- Allen, J. T., Porter, H., Nanney, T. R., & Abernethy, K. (1990, February, 1990). *Reexamining the introductory computer science course in liberal arts institutions*. Paper presented at the Twenty-First SIGCSE technical symposium on computer science education, Washington, DC.
- American Association of School Librarians, A. L. A. (1999). Information literacy: A position paper on information problem solving (Vol. 2000,): Wisconsin Educational Media Association.
- Anderson, R. C. (1984). *The architecture of cognition*. Cambridge, MA: Harvard University Press.
- Arnou, D. (1991, March, 1991). *The Iliad and the WHILE loop: computer literacy in a liberal arts program*. Paper presented at the Twenty-Second SIGCSE technical symposium on computer science education, San Antonio, TX.
- Arnou, D. (1994, March, 1994). *Teaching programming to liberal arts students: Using loop invariants*. Paper presented at the Twenty-Fifth SIGCSE technical symposium on computer science education, Phoenix, AZ.
- Asbjörnsen, P. C., & Moe, J. I. (1859/1969). The three Billy-goats gruff, *Popular tales from the Norse* (pp. 184-185). London: The Bodley Head Ltd.

- Bailey, M. G. (1987, February, 1987). *Spreadsheets and databases - alternatives to programming for non-computer science majors*. Paper presented at the Eighteenth SIGCSE technical symposium on computer science education, St. Louis, MO.
- Baker, F. B. (1985). *The basics of item response theory*. Portsmouth, NH: Heinemann Educational Books.
- Baron, N. S. (1984, February, 1984). *Should everyone learn anything?: the question of computer literacy*. Paper presented at the Fifteenth SIGCSE technical symposium on computer science education, Philadelphia, PA.
- Barrett, M. (1996). Emphasizing design in CS1. *SIGCSE Bulletin*, 28(Q), 315-318.
- Baruch, M. (1986, February, 1986). *An experience is worth 1K words*. Paper presented at the Seventeenth SIGCSE technical symposium on computer science education, Cincinnati, OH.
- Baxter, G. P., & Glaser, R. (1997). *An approach to analyzing the cognitive complexity of science performance assessments* (Technical Report 452). Los Angeles, CA: National Center for Research on Evaluation, Standards, and Student Testing (CRESST), Center for the Study of Evaluation (CSE) Graduate School of Education & Information Studies, University of California.
- Beidler, J., Cassel, L., Lidtke, D., & Owens, B. (1985, March, 1985). *Trends in service courses*. Paper presented at the Sixteenth SIGCSE technical symposium on computer science education, New Orleans, Louisiana.
- Beilin, H. (1985). Dispensable and nondispensable elements in Piaget's theory. In J. Montangero (Ed.), *Genetic epistemology: Yesterday and today*. (pp. 107-125). NY: City University of New York.
- Berliner, D. C. (1992, August, 1992). *The science of psychology and the practice of schooling: The one hundred year journey of educational psychology from interest, to disdain, to respect for practice*. Paper presented at the American Psychological Association, Washington, D.C.

- Besser, H. (1993). Education as marketplace. In R. Muffoletto & N. Knupfer (Eds.), *Computers in education: Social, political, and historical perspectives* (pp. 37-69). Cresskill, NJ: Hampton Press, Inc.
- Biermann, A. W. (1990, February, 1990). *An overview course in academic computer science: A new approach for teaching nonmajors*. Paper presented at the Twenty-First SIGCSE technical symposium on computer science education, Washington, DC.
- Biermann, A. W., Fahmy, A. F., Guinn, C., Pennock, D., Ramm, D., & Wu, P. (1994, March, 1994). *Teaching a hierarchical model of computation with animation software in the first course*. Paper presented at the Twenty-Fifth SIGCSE technical symposium on computer science education, Phoenix, AZ.
- Block, J. H., Efthim, H. E., & Burns, R. B. (1989). *Building effective mastery learning schools*. New York: Longman.
- Bloom, B. S., Hastings, J. T., & Madaus, G. F. (1971). *Handbook on formative and summative evaluation of student learning*. New York: McGraw-Hill Book Company.
- Bloom, B. S., Madaus, G. F., & Hastings, J. T. (1981). *Evaluation to improve learning*. New York, NY: McGraw-Hill.
- Bowers, C. A. (1988). *The cultural dimensions of educational computing: Understanding the non- neutrality of technology*. New York: Teachers College Press.
- Bransford, J. D., Brown, A. L., & Cocking, R. R. (Eds.). (1999). *How people learn: Brain, mind, experience and school*. Washington, DC: National Academy Press.
- Bruner, J. S. (1960). *The process of education*. New York: Vintage Books.
- Buderi, R. (2000). From the ivory tower to the bottom line, *Technology Review* .
- Cahan, E. D. (1992). John Dewey and human development. *Developmental Psychology*, 28(2), 205-214.

- Canup, M. J., & Shackelford, R. L. (1998, February, 1998). *Using software to solve problems in large computing courses*. Paper presented at the Twenty-ninth SIGCSE technical symposium on computer science education, Atlanta, Georgia.
- Chandler, P., & Sweller, J. (1991). Cognitive load theory and the format of instruction. *Cognition and Instruction*, 8(4), 292-332.
- Cherry, J. (1986). Introduction to computer use: a course for non-computer science majors at a large university. *SIGCSE Bulletin*, 18, 40-43, 48.
- Clark, A. R. (1979, June, 1979). *Computer literacy for liberal art students: the case of computer mapping*. Paper presented at the National Educational Computing Conference, University of Iowa.
- Committee on Information Technology Literacy. (1999). *Being fluent with information technology* (Book and web site NSF Contract Number CDA-9616681). Washington, DC: National Academy of Sciences.
- Cooley, W. W., & Lohnes, P. R. (1971). *Multivariate data analysis*. New York: John Wiley & Sons, Inc.
- Cuban, L. (1986). The use of instructional television, *Teachers and machines : the classroom use of technology since 1920* (pp. 27-38). New York: Teachers College Press.
- Curl, L. A., & Hussin, B. J. (1993, March, 1993). *Introductory computing: a new approach*. Paper presented at the Twenty-Fourth SIGCSE technical symposium on computer science education, Grand Valley State University, MI.
- Dalbey, J. (1991). Applying zen principles in an introductory programming course. *SIGCSE Bulletin*, 23, 21-23.
- Dyck, V. A., Black, J. P., & Fenton, S. L. (1987, February, 1987). *Beyond traditional computer literacy*. Paper presented at the Eighteenth SIGCSE technical symposium on computer science education, St. Louis, MO.

- Ellison, R. J. (1980, June, 1980). *Microcomputers and computer literacy: a case study*. Paper presented at the National Educational Computing Conference, Christopher Newport College, Newport News, Virginia.
- Engineering Accreditation Commission. (2000). Criteria for accrediting engineering programs (pp. 59). Baltimore, MD: Engineering Accreditation Commission.
- Feinstein, D., & Langan, D. (1985). Computers and society -- another look at that general purpose course. *SIGCSE Bulletin*, 17, 32-33, 41.
- Foshay, R. (1991). Sharpen up your schemata. *Data Training*, May, 18-25.
- Gardner, D. P., & and others. (1983). *A nation at risk: The imperative for educational reform. An open letter to the American people. A report to the Nation and the Secretary of Education*. (Stock No. 065-000-00177-2). Washington, DC: National Commission on Excellence in Education.
- Gergen, K. J. (1994). Social construction and the educational process. In L. P. Steffe & J. Gale (Eds.), *Constructivism in Education* (pp. 17-39). Hillsdale, NJ: Lawrence Erlbaum Associates.
- Gick, M. L., & Holyoak, K. J. (1983). Schema induction and analogical transfer. *Cognitive Psychology*, 15, 1-38.
- Goldweber, M., Barr, J., & Leska, C. (1994, March, 1994). *A new perspective on teaching computer literacy*. Paper presented at the Twenty-Fifth SIGCSE technical symposium on computer science education, Phoenix, AZ.
- Greeno, J. G., Collins, A. M., & Resnick, L. B. (1996). Cognition and learning. In D. Berliner & R. Calfee (Eds.), *Handbook of educational psychology* (pp. 15-46). New York: Macmillan.
- Gurwitz, C. (1998, February, 1998). *The internet as a motivating theme in a math/computer core course for nonmajors*. Paper presented at the Twenty-ninth SIGCSE technical symposium on computer science education, Atlanta, Georgia.

- Hair, J. F., Jr., Anderson, R. e., & Tatham, R. L. (1987). *Multivariate data analysis with readings*. (2nd ed.). New York: Macmillan Publishing Company.
- Halaris, A., & Sloan, L. (1985, March, 1985). *Towards a definition of computing literacy for the liberal arts environment*. Paper presented at the Sixteenth SIGCSE technical symposium on computer science education, New Orleans, Louisiana.
- Hambleton, R. K., Swaminathan, H., & Rogers, H. J. (1991). *Fundamentals of item response theory*. (First ed.). Newbury Park: Sage Publications.
- Herrmann, N., & Popyack, J. L. (1994, March, 1994). *An integrated, software-based approach to teaching introductory computer programming*. Paper presented at the Twenty-Fifth SIGCSE technical symposium on computer science education, Phoenix, AZ.
- Herrmann, N., & Popyack, J. L. (1995, March, 1995). *Creating an authentic learning experience in introductory programming courses*. Paper presented at the Twenty-Sixth SIGCSE technical symposium on computer science education, Nashville, TN.
- Holt, R. W., Boehm-Davis, D. A., & Schultz, A. C. (1987). Mental representations of programs for student and professional programmers. In G. M. Olson, S. Sheppard, & E. Soloway (Eds.), *Empirical studies of programmers: Second workshop* (pp. 33-46). Norwood, NJ: Ablex Publishing Corporation.
- Howe, J. A. M., Ross, P. M., Johnson, K. R., Plane, F., & Inglis, R. (1989). Teaching mathematics through programming in the classroom. In E. Soloway & J. Spohrer (Eds.), *Studying the novice programmer* (pp. 43-56). Hillsdale, New Jersey: Lawrence Erlbaum Associates, Inc.
- Johnson, D. W., Johnson, R. T., & Smith, K. A. (1991). *Active learning: Cooperation in the college classroom*. Edina, MN: Interaction Book Company.
- Joyce, D. (1998, February, 1998). *The computer as a problem solving tool: A unifying view for a non-majors course*. Paper presented at the Twenty-ninth SIGCSE technical symposium on computer science education, Atlanta, Georgia.

- Kaufman, R. A., & Zahn, D. (1993). *Quality management plus : the continuous improvement of education*. Newbury Park, Calif.: Corwin Press.
- Kay, D. G. (1993, March, 1993). *An honors computer science seminar for undergraduate non-majors*. Paper presented at the Twenty-Fourth SIGCSE technical symposium on computer science education, Grand Valley State University, MI.
- Kay, D. G. (1998, February, 1998). *Large introductory computer science classes: strategies for effective course management*. Paper presented at the Twenty-ninth SIGCSE technical symposium on computer science education, Atlanta, Georgia.
- Keller, J. M. (1983). Motivational design of instruction. In C. M. Reigeluth (Ed.), *Instructional-design theories and models: An overview of their current status* (pp. 383-434). Hillsdale, New Jersey: Lawrence Erlbaum Associates.
- Khatti, N., Reeve, A. L., & Kane, M., B. (1998). *Principles and practices of performance assessment*. Mahwah, NJ: Lawrence Erlbaum Associates, Inc.
- King, L. A. S., & Barr, J. (1997, March, 1997). *Computer science for the artist*. Paper presented at the Twenty-Eighth SIGCSE technical symposium on computer science education, San Jose, CA.
- Koffman, E. P., Miller, P. L., & Wardle, C. E. (1984). Recommended curriculum for CS1, 1984: A report of the ACM curriculum task force for CS1. *Communications of the ACM*, 27(10), 998-1001.
- Koffman, E. P., Miller, P. L., & Wardle, C. E. (1985). Recommended curriculum for CS2, 1984: A report of the ACM curriculum task force for CS2. *Communications of the ACM*, 28(8), 815-818.
- Kolesar, M. V., & Allan, V. H. (1995, March, 1995). *Teaching computer science concepts and problem solving with a spreadsheet*. Paper presented at the Twenty-Sixth SIGCSE technical symposium on computer science education, Nashville, TN.

- Kolodny, N. H., & Ott, G. (1981, June, 1981). *Developing computer literacy at a liberal arts college*. Paper presented at the National Educational Computing Conference, North Texas State University, Denton, Texas.
- Konstam, A., & Howland, J. E. (1994). Teaching computer science principles to liberal arts students. *SIGCSE Bulletin*, 26, 29-34, 40.
- Krzanowski, W. J. (1988). *Principles of multivariate analysis: A user's perspective*. Oxford: Clarendon Press.
- Kurland, D. M., Pea, R. D., Clement, C., & Mawby, R. (1989). A study of the development of programming ability and thinking skills in high school students. In E. Soloway & J. Spohrer (Eds.), *Studying the novice programmer* (pp. 83-112). Hillsdale, New Jersey: Lawrence Erlbaum Associates, Inc.
- Lee, G. C., & Wu, C. C. (1997). On teaching computer literacy to future secondary school teachers. *SIGCSE Bulletin*, 29, 2-6.
- Lee, J. F., & Pruitt, K. W. (1984). *Providing for individual differences in student learning: a mastery learning approach*. Springfield, Ill.: C. C. Thomas.
- Levine, D. U. (1985). *Improving student achievement through mastery learning programs*. San Francisco: Jossey-Bass.
- Levine, L., Woolf, B., & Filoramo, R. (1984, February, 1984). "Do I press return?". Paper presented at the Fifteenth SIGCSE technical symposium on computer science education, Philadelphia, PA.
- Lewis, C., & Olson, G. (1987). Can principles of cognition lower the barriers to programming? In G. M. Olson, S. Sheppard, & E. Soloway (Eds.), *Empirical studies of programmers: Second workshop* (pp. 248-263). Norwood, NJ: Ablex Publishing Corporation.
- Linn, M. C., & Dalbey, J. (1989). Cognitive consequences of programming instruction. In E. Soloway & J. Spohrer (Eds.), *Studying the novice programmer* (pp. 57-81). Hillsdale, New Jersey: Lawrence Erlbaum Associates, Inc.

- Littlefield, J., Delclos, V. R., Lever, S., Clayton, K. N., Bransford, J. D., & Franks, J. J. (1988). Learning LOGO: Method of teaching, transfer of general skills, and attitudes toward school and computers. In R. E. Mayer (Ed.), *Teaching and learning computer programming: Multiple research perspectives* (pp. 111-135). Hillsdale, NJ: Lawrence Erlbaum Associates, Inc.
- Mardia, K. V., Kent, J. T., & Bibby, J. M. (1979). *Multivariate analysis*. London: Academic Press.
- Martin, J. B. (1986, February, 1986). *A profile of today's computer literacy student*. Paper presented at the Seventeenth SIGCSE technical symposium on computer science education, Cincinnati, OH.
- Mayer, R. E., Dyck, J. L., & Vilberg, W. (1986). Learning to program and learning to think: what's the connection? *Communications of the ACM*, 29(7), 605-610.
- McFall, R., & Stegink, G. (1997, March, 1997). *Introductory computer science for general education: Laboratories, textbooks, and the internet*. Paper presented at the Twenty-Eighth SIGCSE technical symposium on computer science education, San Jose, CA.
- McInerney, V., McInerney, D. M., & Sinclair, K. E. (1994). Student teachers, computer anxiety and computer experience. *Journal of Educational Computer Research*, 11(1), 27-50.
- Merrill, M. D. (1983). Component display theory. In C. M. Reigeluth (Ed.), *Instructional-design theories and models: An overview of their current status* (pp. 279-333). Hillsdale, New Jersey: Lawrence Erlbaum Associates.
- Michaels, S., & O'Connor, M. C. (1990,). *Literacy as reasoning within multiple discourses: Implications for policy and educational reform*. Paper presented at the Council of Chief State School Officers Summer Institute.
- Nulden, U. (1998, February, 1998). *The ExCon project: advocating continuous examination*. Paper presented at the Twenty-ninth SIGCSE technical symposium on computer science education, Atlanta, Georgia.

- Olson, G. M., Catrambone, R., & Soloway, E. (1987). Programming and algebra word problems: A failure to transfer. In G. M. Olson, S. Sheppard, & E. Soloway (Eds.), *Empirical studies of programmers: Second workshop* (pp. 1-13). Norwood, NJ: Ablex Publishing Corporation.
- Osin, L., & Lesgold, A. (1996). A proposal for the reengineering of the educational system. *Review of Educational Research*, 66(4), 621-656.
- Ourusoff, N. (1986). The computational view of nature: a liberal arts course in computer science. *SIGCSE Bulletin*, 18, 54-56, 64.
- Papert, S. (1980). *Mindstorms: Children, computers and powerful ideas*. New York: Basic Books.
- Parker, J. D., & Schneider, G. M. (1987, February, 1987). *Problems with and proposals for service courses in computer science*. Paper presented at the Eighteenth SIGCSE technical symposium on computer science education, St. Louis, MO.
- Peterson, J. T. (1987, February, 1987). *Goals for and lessons from a computer literacy course*. Paper presented at the Eighteenth SIGCSE technical symposium on computer science education, St. Louis, MO.
- Piaget, J. (1977). The role of action in the development of thinking. In W. F. Overton & J. M. Gallagher (Eds.), *Knowledge and development* (Vol. 1, pp. 17-42). New York: Plenum Press.
- Popyack, J. L., & Herrmann, N. (1993, March, 1993). *Mail merge as a first programming language*. Paper presented at the Twenty-Fourth SIGCSE technical symposium on computer science education, Grand Valley State University, MI.
- Price, B. A., Archer, C. B., & Moressi, W. J. (1988). A successful approach to the computer literacy course. *SIGCSE Bulletin*, 20, 13-17, 19.
- Rabung, J. (1994, March, 1994). *Introducing computer concepts to novices by "practical" immersion*. Paper presented at the Twenty-Fifth SIGCSE technical symposium on computer science education, Phoenix, AZ.

- Raymond, M. R., & Viswesvaran, C. (1993). Least squares models to correct for rater effects in performance assessment. *Journal of Educational Measurement, 30*(3), 253-368.
- Reigeluth, C. M., & Stein, F. S. (1983). The elaboration theory of instruction. In C. M. Reigeluth (Ed.), *Instructional-design theories and models: An overview of their current status* (pp. 338-381). Hillsdale, New Jersey: Lawrence Erlbaum Associates.
- Ricardo, C., Adair, J. H., Dubinsky, E., Ruston, H., Sprague, E., & Ricardo, C. (1986, February, 1986). *Computer science as a service department: What do we offer non-majors?* Paper presented at the Seventeenth SIGCSE technical symposium on computer science education, Cincinnati, OH.
- Rodriguez, R. V., & Anger, F. D. (1981, June, 1981). *A novel approach to computer literacy for natural science students.* Paper presented at the National Educational Computing Conference, North Texas State University, Denton, Texas.
- Rogoff, B. (1994). Developing understanding of the idea of communities of learners. *Mind, Culture, and Activity, 1*(4), 209-229.
- Rogoff, B., & Lave, J. (Eds.). (1984). *Everyday cognition: Its development in social context.* Cambridge, Mass.: Harvard University Press.
- Ropp, M. M. (1997). *Exploring individual characteristics associated with learning to use computers and their use as pedagogical tools in preservice teacher preparation.* Unpublished Doctor of Philosophy, Michigan State University, East Lansing, MI.
- Rosenberg, R. (1987). A critical analysis of research on intelligent tutoring systems. *Educational Technology, 27*(11), 7-13.
- Ryder, B. G. (1984, February, 1984). *A "Hands-On" approach to computer literacy.* Paper presented at the Fifteenth SIGCSE technical symposium on computer science education, Philadelphia, PA.
- Schimming, B. B. (1980, June, 1980). *Computer literacy: a case for information literacy.* Paper presented at the National Educational Computing Conference, Christopher Newport College, Newport News, Virginia.

- Schneider, G. M. (1986). A proposed redesign of the introductory service course in computer science. *SIGCSE Bulletin*, 18, 15-21.
- Schoenfeld, A. H. (1994). Toward a unifying framework for assessment: A conceptual frame and fundamental issues it highlights. *Draft*, 28.
- Sellars, H. L. (1988). Why a college course in computer literacy? *SIGCSE Bulletin*, 20, 58-59, 64.
- Shavelson, R. J., Baxter, G. P., & Gao, X. (1993). Sampling variability of performance assessments. *Journal of Educational Measurement*, 30(3), 215-232.
- Shuell, T., J. (1986). Cognitive conceptions of learning. *Review of Educational Research*, 56(4), 411-436.
- Simon, H. A. (1980). Problem solving and education. In D. T. Tuma & R. Reif (Eds.), *Problem solving and education: Issues in teaching and research* (pp. 81-96). Hillsdale, NJ: Lawrence Erlbaum.
- Smith, J. P., III, diSessa, A. A., & Roschelle, J. (1993). Misconceptions reconceived: A constructivist analysis of knowledge in transition. *The Journal of the Learning Sciences*, 3(2), 115-163.
- Soloway, E. (1993). Should we teach students to program? *Communications of the ACM*, 36(10), 21-24.
- Spooner, D. L., & Skolnick, M. M. (1997, March, 1997). *Science and engineering case studies in introductory computing courses for non-majors*. Paper presented at the Twenty-Eighth SIGCSE technical symposium on computer science education, San Jose, CA.
- Spresser, D. M. (1985, March, 1985). *A moderate approach to computer literacy*. Paper presented at the Sixteenth SIGCSE technical symposium on computer science education, New Orleans, Louisiana.
- SPSS. (1999a). *SPSS 10.0 syntax reference guide*. Chicago, IL: SPSS, Inc.

SPSS. (1999b). *SPSS Base 10.0 applications guide*. Chicago, IL: SPSS, Inc.

St. Julien, J. A. (1994, April 4-8, 1994). *Social constructivism: Vygotskian activity and connectionist representation*. Paper presented at the Annual meeting of the American Educational Research Association, New Orleans.

Taffe, W. J. (1991, March, 1991). *Simulation and modeling with Stella: A general education course*. Paper presented at the Twenty-Second SIGCSE technical symposium on computer science education, San Antonio, TX.

Tennyson, R. D., & Cocchiarella, M. J. (1986). An empirically based instructional design theory for teaching concepts. *Review of Educational Research*, 56(1), 40-71.

The Chronicle of Higher Education. (2000). Distance education (Vol. 2000,).

Tomasello, M., Kruger, A. C., & Ratner, H. H. (1993). Cultural learning. *Behavioral and Brain Sciences*, 16, 495-552.

Townsend, G. C. (1998, February, 1998). *Turning liabilities into assets in a general education course*. Paper presented at the Twenty-ninth SIGCSE technical symposium on computer science education, Atlanta, Georgia.

Tu, J. J., & Johnson, J. R. (1990). Can computer programming improve problem-solving ability? *SIGCSE Bulletin*, 22, 30-33, 37.

Tucker, A. B., & ACM/IEEE-CS Joint Curriculum Task Force. (1991). Computing curricula. *Communications of the ACM*, 34(6), 68-84.

Turk, J., & Wiley, S. (1997, March, 1997). *Teaching social and ethical issues in the literacy course*. Paper presented at the Twenty-Eighth SIGCSE technical symposium on computer science education, San Jose, CA.

U. S. Department of Labor. (1996). Generating productivity growth: A review of the role of workplace practices and computers (Vol. 2000,). Washington, DC.



- Urban-Lurain, M. (1996). *Intelligent tutoring systems: An historic review in the context of the development of artificial intelligence and educational psychology*. East Lansing, MI: Michigan State University.
- Urban-Lurain, M., & Weinshank, D. J. (1999a, March 26, 1999). *"I Do and I Understand:" Mastery model learning for a large non-major course*. Paper presented at the Special Interest Group on Computer Science Education, New Orleans, LA.
- Urban-Lurain, M., & Weinshank, D. J. (1999b). *Mastering computing technology: A new approach for non-computer science majors*.
<http://aral.cse.msu.edu/Publications/AERA99/MasteringComputing.html>.
- Urban-Lurain, M., & Weinshank, D. J. (2000). *Attendance and outcomes in a large, collaborative learning, performance assessment course*.
<http://aral.cse.msu.edu/Publications/AERA2000/Attendance.html>.
- Van De Geer, J. P. (1971). *Introduction to multivariate analysis*. San Francisco, CA: W. H. Freeman and Co.
- Vygotsky, L. (1978). *Mind in society: The development of higher psychological processes*. Cambridge, MA: Harvard University Press.
- Webb, N. M., Nemer, K. M., Chizhik, A. W., & Sugrue, B. (1999). Equity issues in collaborative group assessment: group composition and performance. *American Educational Research Journal*, 35(4), 607-651.
- Weinshank, D. J., Urban-Lurain, M., & Danieli, T. (1990). *Introduction to computing: Telecourse with Waterloo BASIC*. (2nd ed.). Dubuque, Iowa: Kendall/Hunt Publishing Company.
- Weinshank, D. J., Urban-Lurain, M., Danieli, T., & McCuaig, G. (1992). *Integrated introduction to computing*. (1st ed.). Dubuque, Iowa: Kendall/Hunt Publishing Company.
- Weinshank, D. J., Urban-Lurain, M., Danieli, T., & McCuaig, G. (1995). *Integrated introduction to computing*. (updated first edition, revised and enlarged ed.). Dubuque, Iowa: Kendall/Hunt Publishing Company.

Weinshank, D. J., Urban-Lurain, M., & Olds, S. (1988). *Introduction to computing: Telecourse with Waterloo BASIC*. (1st ed.). Dubuque, Iowa: Kendall/Hunt Publishing Company.

Wetmore, D. E. (1980, June, 1980). *Required freshman computer education in a liberal arts college*. Paper presented at the National Educational Computing Conference, Christopher Newport College, Newport News, Virginia.

Winograd, T., & Flores, F. (1986). Computation and intelligence, *Understanding computers and cognition* (pp. 93-106). Reading, MA: Addison-Wesley Publishing Company, Inc.

Woodson, M. I. C. E. (1982, February, 1982). *Computer literacy by computer*. Paper presented at the Thirteenth SIGCSE technical symposium on computer science education, Indianapolis, Indiana.

Yelon, S. L. (1996). *Powerful principles of instruction*. White Plains, NY: Longman.

MICHIGAN STATE UNIVERSITY LIBRARIES



3 1293 02102 1773