This is to certify that the

thesis entitled

Internet Engineering Design Agents

presented by

Gary Joseph Gosciak

has been accepted towards fulfillment
of the requirements for

MS _____ degree in Mechanical Engineering

Clark Radcliffe
Major professor

Date 5/4/01

DATE DUE	DATE DUE	DATE DUE

INTERNET ENGINEERING DESIGN AGENTS

By

Gary Joseph Gosciak

A THESIS

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

MASTER OF SCIENCE

Department of Mechanical Engineering

2001

ABSTRACT

INTERNET ENGINEERING DESIGN AGENTS

By

Gary Joseph Gosciak

This paper presents Internet Engineering Design Agents (IEDA). The approach is a component-based agent methodology implemented with a strict input/output communication protocol. An individual Design Agent (DA) is a virtual product capable of encapsulating a corporation's expertise and knowledge base. Agents for sub-systems and/or components are linked via a network to form larger integrated model systems. IEDA utilize a global ontology agent containing the viable standardized engineering domain queries. User, system, sub-system, and component Design Agents interact through the global ontology, allowing non-inferable communication of information without divulging the proprietary models contained within the DA. The IEDA facilitates proprietary security and seamless integration of differing corporate software platforms. The structure and function of the IEDA is discussed and realized in illustration along with a working Client/Server software model. A two dimensional span model is used as an example to validate the distributed nature of assemblies and components registered as DA's on a network. IEDA form a distributed modeling environment that allows for communication and coordination required for effective and efficient global collaborative distributed design.

to the light gmg, to the path family and friends, to the way department and knowledgeable patient advisors

ACKNOWLEDGEMENTS

First I would like to give thanks and appreciation to my family. Although not always understanding my way they have provided the unconditional support and understanding as I proceeded through this endeavor. This is an understatement when considering my good advisor Dr. Clark J. Radcliffe. Through him and for him this work does exist.

Also, thanks and appreciation is due for Roel, Oleg, April, and Dr. Jon Sticklen for participation and contribution of their energy.

Gratitude is given to all the members making up the department, the lab eb2553, and the environment I was allowed to participate in. Thank you.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Introduction

Modern engineering is progressing toward a global, collaborative, distributed design process. A notable example is the automotive industry. In the late 1980's, the traditional automotive engineering design process was too slow and impeded collaborative design. The traditional "over the wall" design philosophy was sequential, exclusive and drastically inefficient. Automotive engineering design is evolving to a new paradigm initiated by computer technologies empowering communication and Computer Aided Design (CAD). These technologies bring globally distributed, cross-functional information to an engineer's desktop. Ford's Design and Manufacturing teams collaborate on projects around the world and around the clock (AEI, 2000a). GM boasts the largest CAD installation, linking 20,000 multi-disciplined engineers across the Globe (AEI, 2000b). The automotive industry has recognized the inefficiency of localized in-house engineering of products from conception through component design to market fabrication. For BMW in South Carolina only the powertrain is their design and the rest is from suppliers (Bucholz, AEI 1998). The Internet is connecting the world to generate a global market. With this global market comes global competition. Automotive companies are remaining competitive by evolving into integrators of components and subsystems from globally distributed external suppliers. Engineers and designers are more efficient and effective in their jobs by leveraging global design teams using the best software tools and workflow systems, and by outsourcing more responsibility to component suppliers (AEI, 2000a). With the global competition intensifying, many Original Equipment Manufacturers (OEMs) and suppliers are looking for ways to reduce development cost, parts count, and time-to-market (Bokulich, AEI 1999). The rapid

evolution of these technologies fuels advances in engineering design methods as well as hurdles for effective realization.

Current collaborative engineering design is based primarily on in-house practice. The walls between disciplines have been broken down, yet each discipline has retained its expertise. Computer Aided Engineering (CAE) has transformed the sequential relationships between engineering disciplines into efficient, concurrent, multidisciplinary product development (Dorf and Kusiak, 1994). The product design cycle and its in-house participants are less exclusive and more integrated. The expertise of the various in-house disciplines is represented in a comprehensive database. Computer Aided Design (CAD) provides a standardized representation of engineering design information for interaction and communication between the various teams involved in the development of a new product. All participating disciplines utilize the same software and interact with this database. The competency of these CAD tools allows for computer simulation, rather than costly and time-consuming physical testing, thereby reducing time to market for new products. The exchange of information is freely communicated within the boundaries of corporate organizational walls. External suppliers enter the organization and participate through adoption and/or coordinated acceptance of the integrating CAD system. Participation in current collaborative engineering design is based upon legal contracts and required software compatibility.

The next step in collaborative engineering design as global competition increases is to progress to a broader base than a corporate in-house practice. External supplier expertise is being leveraged to form multidisciplinary teams no longer entirely located within corporate organizational walls. Collaborative engineering is increasingly being conducted

through multi-corporate discipline teams. Current CAD approaches augmenting concurrent design have been developed and executed under the limiting corporate in-house assumption. This assumption limits the engineering design process when multiple corporate organizations are involved. The participating companies are reluctant to adopt into the integrating software system. Conformity to the integrating system may require unwanted overhead in personnel training, economic strain in software compliance, and exposure of proprietary models. All participants need to have free access to up-to-date engineering information for a given project. All participants, however, do not need to have free access to the model details providing this information. Competitive proprietary engineering design information needs to remain secure. Yet, as each team, and each company, progresses through an evolving project, the results from the expertise that they provide must be continuously available to other components of the multi-corporate team. Research on a project called DOME (Distributed Object-based Modeling Environment), has begun to address this problem (Abrahamson, et al, 1999). The DOME project presents an example of integrated product development through building a computational service exchange network, thereby interconnecting the input and output services of different design participants. Global collaborative engineering design requires all corporate software platforms to operate in a seamless fashion across corporate organizational boundaries.

A global collaboration approach towards engineering poses a considerable undertaking in effective communication management, information resource development, and coordination of both. As the participants involved in realizing a product span the world, time zones are bridged. The normal day-to-day, 9-5 work regime is replaced with

a round the clock engineering practice. Human interaction using a network is not always possible. As workdays overlap or miss entirely all parties may not be active at any given time. Yet, synchronization of the work has to occur. Multiple teams comprised of various companies and geographically separated concurrently work on a project. Existing CAD systems are within corporate boundary walls. This acknowledges that the protection of participant's proprietary nature is a bottleneck for decreased time-to-market through the distributed integrator supplier approach. A company's communication techniques and information resource base must address challenges such as synchronization with 24 hour operation, proprietary security behind engineering information representation, and seamless information exchange with multiple software platform possibilities to achieve efficacious globally distributed engineering design.

Computer-aided tools are needed to facilitate the ongoing evolution of global collaborative engineering. The current state of collaborative engineering has been realized in part through CAE. Computer technologies are increasing at a rapid rate. This results in advances of these tools as well as new tools. The computer-based tool presented in this paper addresses the challenges in effective global collaborative engineering design. As the integrators have many components, sub-systems and systems, they remain nimble and procure suppliers best suited for their project without the need to form long term relations with specific suppliers. The requirement for legal contracts to protect proprietary engineering design information is eliminated further reducing the design cycle time. The suppliers are free to collaborate with the competition without fear of compromising their proprietary information. Integrators and suppliers freely sleep with their enemies (Esterman, Ishii, 1999). Suppliers can coordinate with

multiple integrators, suppliers on various projects, across geographical distances, and simultaneously as the necessity of long-term relations with a specific integrator is diminished. The competitive advantage is increased and secured for both integrators and suppliers. The advent of these tools can augment existing methodologies into a greater functional regime.

Agent Technology as a Solution

Agent software technology allows tools to assist global collaborative engineering. An agent (Tecuci, et al, 1998) is a Knowledge Based System (KBS) that perceives its environment; reasons to interpret perceptions, draw inferences, solve problems, and determine actions; and acts upon that environment to realize a set of goals or tasks for which it was assigned. Software agents have autonomous, responsive, proactive, reasoning/learning, collaborative behavior (Murch, Johnson, 1999, Brenner, et al 1999, Wooldridge, Jennings, 1998, Bradshaw, 1997). An agent's environment may be the physical world, a person using a graphical user interface, and a collection of other agents, the Internet, or other complex environment. Agents are able to act on their own in a changing environment prescribed by a set of goals and with other agents, further developing as they proceed with a task. Agents provide the opportunity to autonomously represent the engineering behavior of individual physical components in a distributed engineering design environment. Software with this capability can be coupled with existing corporate KBS comprised in part by CAE technology. This combination provides tools capable of bringing global collaborative engineering design to the next level.

An agent's autonomous, responsive, proactive learning nature addresses global collaborative engineering design's 24-hour synchronization challenge. The agent can function autonomously. Having a goal, rule orientated composition allows for operation without constant supervision from the corporation providing the agent. In any time zone

around the globe, the agent is operational, reacting promptly to changing operational conditions. Timely agent response facilitates synchronization.

An agent's collaborative behavior addresses global collaborative engineering design's proprietary and software platform handicaps. Agent software can effectively collaborate on a solution to a problem in a distributed environment. A global multi-corporate engineering design team is a distributed environment where expertise and knowledge is not contained in a central location. A MAS (Multi-Agent System) allows the integration of existing Design Agents of an individual corporation into a larger system without the need of collocating the individual DA's into a single DA at a single location (Brenner, et al, 1999). A collection of agents representing the individual corporate expertise and knowledge makes up the MAS. A Mult-Agent System's, as well as the global collaborative engineering design team's competency is the result of the aggregate competency of the individual Design Agents not contained at central DA. Appropriately developed DA's would allow communication and interaction of engineering information without compromising the protection of participant's proprietary nature. The partners of a multi-corporate design team through the DA's communication protocol may cooperate seamlessly through the MAS regardless of specific partner software platforms. Exclusive proprietary information and independent software platform challenges are circumvented through agent technology.

Recent years have shown agent methodology as a tool for assisting the workforce is becoming realized more and more every day as a viable asset to keeping a competitive edge. Agent software technology has been finding its way into industrial and commercial applications such as process control, manufacturing, information and business process

management, and electronic commerce (Wooldridge, Jennings, 1998). This has lead to current research and exploration into the use and validity of agents in engineering design. The CADOM project (Component Agent Design Orientated Modeling) utilizes agent methodology to assist application integration at the design data level (Rosenman, Wang 1999). This methodology functions to facilitate in-house collaborative engineering. The DOME project has agents as modules, which perform the service exchange in an integrated product model via a network. There work suggests that after the overhead of evaluation time in creating the first integrated system model, subsequent system evaluations with design tradeoffs can be reduced from months to minutes (Abrahamson, et al, 1999). The DOME project has recognized that detailed proprietary information is a barrier to this methodology's success.

Software agent technology provides autonomous, collaborative capability towards evolving globally distributed engineering design environments. The system of Internet Engineering Design Agents (IEDA) described below uses a MAS approach to represent physical components developed for a distributed modeling mechanism environment. IEDA form virtual components leveraged by individual corporations for assisting in effective and efficient global collaborative engineering design. The encapsulation of a corporation's knowledge in IEDA protects detailed proprietary engineering information.

Internet Engineering Design Agents

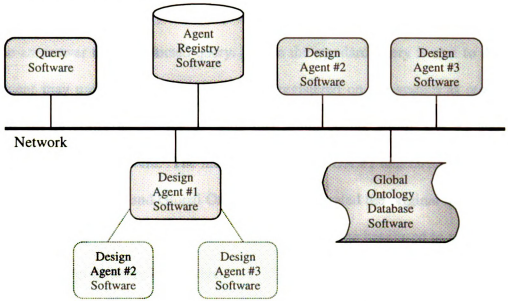


Figure 1. System of (IEDA) connected to a Network.

The system of Internet Engineering Design Agents is organized to facilitate the exchange of engineering design performance data between corporate organizations while protecting the proprietary design information yielding that performance. A minimal implementation of a system of IEDA (Figure 1) includes Query Software, an Agent Registry, Global Ontology, and a distributed set of Design Agents. The Query Software is used by a customer seeking design performance information and is responsible for generating queries. The Agent registry is the Agent that houses the locations of all the Design Agents connected to the network. A list of all valid queries for any Design Agent is standardized and resides in the Global Ontology. Each DA is a virtual model representation of a physical component, sub-system, or system. The collaboration of these entities through the system of IEDA allows distributed design performance modeling.

An IEDA system user uses Query Software (Figure 1) to select a valid agent query from the Global Ontology. The user's query software then selects a Design Agent (#1) to provide an answer to that selected query. When the standard query is sent to Agent #1, that Agent may use other Agents (#2 and #3) registered on the system as resources to generate the response to the query. This interaction occurs within DA #1 and is not visible to the Query Software. The interaction between DA #1 and DA's #2 and #3 uses the same Agent Registry and Global Ontology that generated the original query to DA#1. The IEDA topology allows decomposition of the knowledge base used to answer queries while protecting proprietary knowledge.

A global collaborative engineering system model formed using IEDA is comprised of sub-system and components from participating corporations. This functioning flexible system model can be realized through the collaboration of IEDA. The participating corporations would create their sub-system or component in DA form. When the product is connected to the network the IEDA is kept in-house and only the location of the agent is published in the Agent Registry. This helps insure proprietary protection behind the corporate firewalls. The Query Agent accesses the Agent Registry to find a particular product of interest. The Agent Registry in turn provides the location of the desired product. Once the product is found the Query Agent utilizes the Global Ontology as the proper standardized communication protocol. The strict communication protocol through Global Ontology of standardized queries provides a non-inferable method that ensures the IEDA retains proprietary nature internally, preventing external access. An independent organization would provide and maintain the standardized allowable queries in the Global Ontology. The query agent then generates queries to the cooperating DA in

the IEDA. The cooperating DA may be a complex product assembly consisting of many components and subassemblies. An individual DA in the system of IEDA represents each component or sub-assembly. Through the ontology, these component-based IEDA cooperate above application dependant platforms to formulate an answer to a particular collaborative design query. Although DA's within the IEDA may respond to the entire global ontology, it is anticipated that most will only respond to queries from the global ontology relevant to the agent's physical component and for which a knowledge base is available from the DA's parent organization. Relevant ontology responses are developed from the in-house knowledge that experts, physical testing, or CAE models that the corporation may provide. As the design of product evolve, the available ontology responses evolve. Submission and approval of additional queries into the global ontology will allow for flexible expansion of the engineering design queries that can be accommodated by the IEDA. This expansion of the global ontology allows for a wider range of competency for all the participating agents in the IEDA. When dealing with the global ontology it may prove efficient in terms of network overhead to periodically download the desired query set to product Design Agents. As more and more IEDA are connected to the network, a larger free seamless market takes form. The registered IEDA are available for use as is or by another corporation in a greater schema of sub-system and system products.

Our IEDA approach involves component-based agent methodology utilizing a strict input/output, query/answer protocol. Anything from a shock, brakepad, tire, alternator, engine, to a propeller, turbine blade, etc. can be an IEDA. The IEDA system's purpose is to coordinate DA's with other DA's in a MAS making a decomposable collaborative

modeling environment. The MAS model composition of a system, sub-system, and component IEDA consists of a decentralized distributed tree structure. Through the agent topology, no matter what level of complexity is represented an equivalent agent functionality is retained. An IEDA system model made up of many sub-system Design Agents and those sub-systems made up of additional sub-system or component DA's operate in the same manner as any one of the comprising DA. Communication through the global ontology sets a strict input/output methodology for interaction. Recent modular modeling research has shown that modular model elements utilizing a bond graph method may be assembled into larger models through a strict power-based input/output communication (Byam, Radcliffe, 2000). It was also found, that through this systematic modeling method large models experimental performance verification is improved and equation reformulation is eliminated (Byam, Radcliffe, 1999). Bond graphs span a wide range of domains and would allow one way in which various domain ontology could interact. These results provide the fixed input-output causal structure needed for independent agent-based distributed modular models.

Internet Engineering Design Agent Structure

The structure of the DA has been defined through the environment it is used in. This method would allow for less problematic implementation in that the target use environment would not have to be completely remodeled around the IEDA. On the outside there is the illusion of a shell representing a system, subsystem or component. This perceived shell gives physical meaning or discrete representation of its various internal qualities. It also defines the virtual boundary for interaction with other agents. The real boundary behind the illusion is in the agents' architecture (Figure 2).

An individual Design Agent (Figure 2) includes a network communication protocol, a query handler, a knowledge-based system and the resources necessary to that knowledge base. Queries are received via the communication protocol and parsed by the query handler into a form suitable for processing by the agent's knowledge base. The knowledge base then utilizes internal resources to assemble a response.

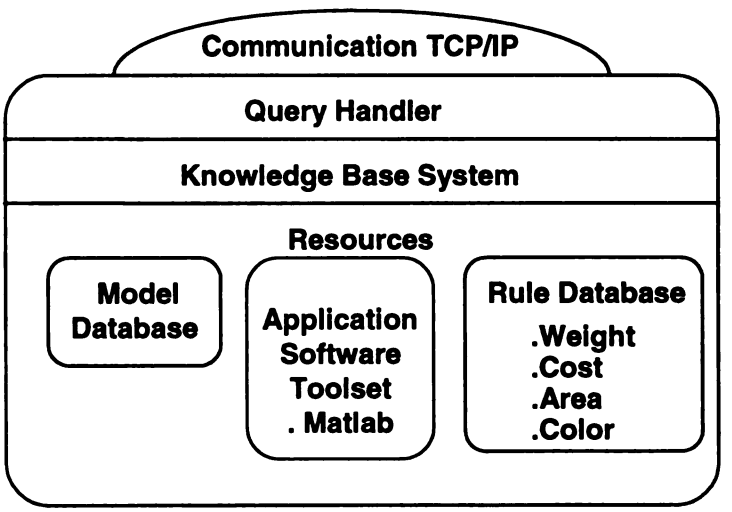


Figure 2. An IEDA Framework

The query handler of the DA defines a DA. The strict query and answer protocol (Figure 3) enables the Design Agent to communicate with the outside world without comprising its internal nature. The query handler converses in a manner void of contextual inferences. Because the queries, along with the answers, are structured in a standard way, query and answer context do not convey any information not desired to be communicated. Any standard query is valid to the query handler, yet not all queries are appropriate to each DA. As a result, the answers can range from this query cannot be answered to a specific non-implicative answer. The query handler is the pinnacle of the IEDA and its actions to a query are the result of interaction with the Knowledge Base System (KBS).

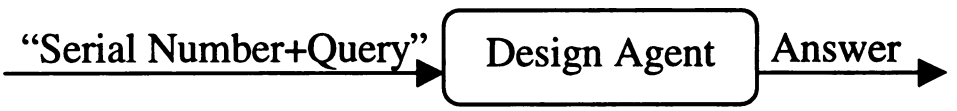


Figure 3. Standard Communication Protocol

A basic definition of a KBS (Tasso, Oliveira, 1998) is a software system able to explicitly represent the knowledge of a given domain and capable of high-level problem solving through reasoning mechanisms upon the knowledge base. A KBS is built up from the declarative knowledge of a domain (Dymm, Levitt, 1991). The domain may be any of the disciplines coordinating on a product. The declarative knowledge may include the information databases, past experience, and most importantly expert knowledge of a particular task. The abstract view of a KBS consists of a central kernel along with a collection of special purpose modules. The kernel here is the problem solving capability of the IEDA to return to the caller information on a part's properties (e.g., color, cost, weight, ...) and model responses (static response, dynamic response) without revealing

the details of proprietary engineering designs that generate those properties and model responses. The KBS may facilitate any individual in any discipline regarding the product design. The special purpose modules of the KBS are any entity of the KBS that do not fall into the kernel of the KBS.

The Resources of the topology fall under the kernel or into the special purpose modules. The Rule Database is inherent to the kernel in representing the problem solving capabilities and the knowledge base. Here perception, cognition, and appropriate action take place. The Model Database, depending upon KBS construction, could fall under either subset. It may be viewed as a part of the knowledge base or as a module that assists the kernel. The Application software toolset (e.g., MatLab, Excel, ProSolids, etc.) is a collection of special purpose modules. Again, these provide support in realizing a solution so that an appropriate answer may be communicated. Local in-house capabilities of the company providing the agent are supported in the architecture through the Resources. In many cases, Design Agents from other corporate organizations may be an important part of the resources used by another DA.

A Design Agent within the IEDA system may either be a component, subsystem, or system. Although these are different virtual representations of varying entities, the agent architecture remains the same. The level of complexity does not alter the architecture. Each levels environment is made up of similar characteristics: geometry, cost, material properties, performance, etc. Allowing for a generalized architecture to competently span the component, sub-system, and system concepts (Wooldridge, Jennings, 1998). The agent framework is no more than an extension of a company's product and capabilities

able to freely communicate with the outside world, sub-system, and system.**An IEDA Implementation**

The IEDA prototype consists of distributed Design Agents representing various companies collaborating to provide product information. This implementation lacks an Agent Registry and Global Ontology. These two aspects of the IEDA system are comprised within the Client/User software and DA's. The user interface of the Client Software (Figure 4) may remotely access the network and connect to any one of the various companies. The network consists of 5 computers or nodes. On these 5 computers various agents representing 8 companies stand-alone or collaborate with each other to serve the Client/User software. The software (Figure 4) allows for the User to choose a company of interest. There are 3 bar companies, 3 truss companies, and 2 span companies providing DA's to choose from. The DA representing the bar companies are resources to the DA of the truss and span companies, and the DA of the truss companies are resources to span companies' DA. Once a company is chosen the company's list of available product serial numbers may be accessed and a product chosen. The software prototype allows the user to choose from 4 (Table 1) different queries. The Design Agents through the standard communication protocol respond to the query, assemble the solution, and return an answer.

Table 1. Available User Queries

Cost	Weight	Area	Color

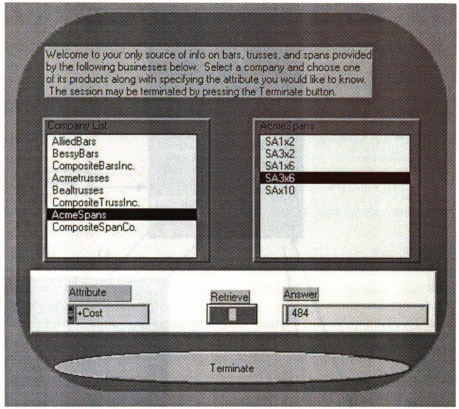


Figure 4. The user interface for the client software

In the example scenario of (Figure 4), the user is interested in the cost of a span from the AcmeSpans Company. The location of the AcmeSpans Agent has been internally registered into the user software specifying the path for the query to travel. The cost query to the AcmeSpans SA3x6 results in generating a query string sent via TCP/IP (Transmission Control Protocol/Internet Protocol) to the AcmeSpans Agent. The agent operates on a remote computer and initiates the process to produce the answer. Future software may include a location agent that answers queries specifying locations of agents such as the AcmeSpans Agent. The user software would call the location agent to identify where the AcmeSpans agent is located. The user software would only need the TCP/IP address of the location agent internally registered rather than all the participating Design Agents.

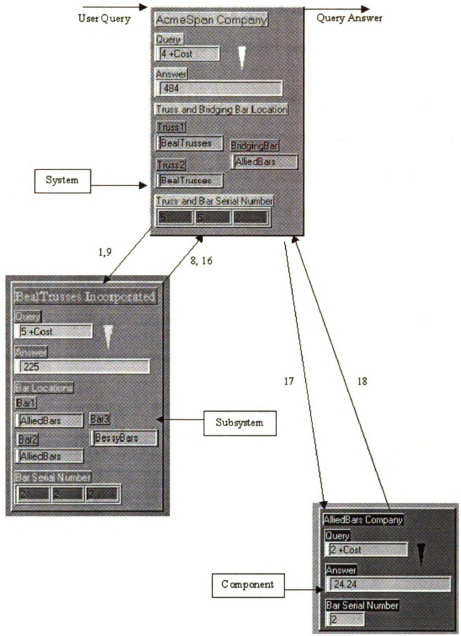


Figure 5. AcmeSpans agent interaction for SA3x6 Cost solution

The generation of a answer to the cost query sent to the AcmeSpans SA3x6 system begins with the receipt of the User Query string (Figure 5). The client software has sent a query that includes the SA3x6 serial number attribute 4 plus the Cost query. The query handler of the AcmeSpans Agent receives the query string and breaks it into segments

that its resources may use. For each agent, Figure 5 shows both query input and answer generated. The figure also shows the internal model used to generate answers to queries.

The AcmeSpans Agent uses its resources to answer the cost query for span SA3x6. Its resources utilize the serial number to know the subsystems' or components' serial numbers comprising span SA3x6: 5, 5, and 2 (Figure 5). The locations of these subsystems and components are internally registered as part of the resources. Again a location agent could be used for this task, reducing the agent size and increasing the ease of product expandability. The AcmeSpans Agent uses its internal client to query the subsystem and components comprising SA3x6. BealTrusses and AlliedBars companies provide these subsystem and components located on remote nodes.

Table 2. TCP/IP network arrow description

Network Call	Sending Agent	Type	String	Receiving Agent
User Query	Client User	query	"4+Cost"	AcmeSpans
1	AcmeSpans	query	"5+Cost"	BealTrusses
2	BealTrusses	query	"2+Cost"	AlliedBars
3	AlliedBars	answer	"24.24"	BealTrusses
4	BealTrusses	query	"2+Cost"	AlliedBars
5	AlliedBars	answer	"24.24"	BealTrusses
6	BealTrusses	query	"2+Cost"	BessyBars
7	BessyBars	answer	"172.68"	BealTrusses
8	BealTrusses	answer	"225"	AcmeSpans
9	AcmeSpans	query	"5+Cost"	BealTrusses
10	BealTrusses	query	"2+Cost"	AlliedBars
11	AlliedBars	answer	"24.24"	BealTrusses
12	BealTrusses	query	"2+Cost"	AlliedBars
13	AlliedBars	answer	"24.24"	BealTrusses
14	BealTrusses	query	"2+Cost"	BessyBars
15	BessyBars	answer	"172.68"	BealTrusses
16	BealTrusses	answer	"225"	AcmeSpans
17	AcmeSpans	query	"2+Cost"	AlliedBars
18	AlliedBars	answer	"24.24"	AcmeSpans
Query Answer	AcmeSpans	answer	"484"	Client Software

The dissemination of the cost query through the agents required to generate an answer is shown in Table 2, and is the result of the standard communication. This process begins at query 1 (Figure 5, 6 and Table 2) network call. The query handler of the BealTrusses Agent receives the query string of a serial number plus the desired query from the AcmeSpans Agent. Since span SA3x6 is comprised of two subsystem trusses from BealTrusses, this also occurs at network 9. The AcmeSpans Agent receives the answers to the query from BealTrusses in networks 8 and 16 for 1 and 9, respectively. The gap between network calls 1 and 8 results from the BealTrusses Agent performing network calls to obtain the answer supplied at network call 8 (Table 2). In an identical process, the AlliedBars Agent query handler receives a query string at network call 17 and returns an answer at network call 18. The AcmeSpans Agent then assembles the answer information using a set of rules developed by the AcmeSpans Company and sends the Query Answer back to the Client Software.

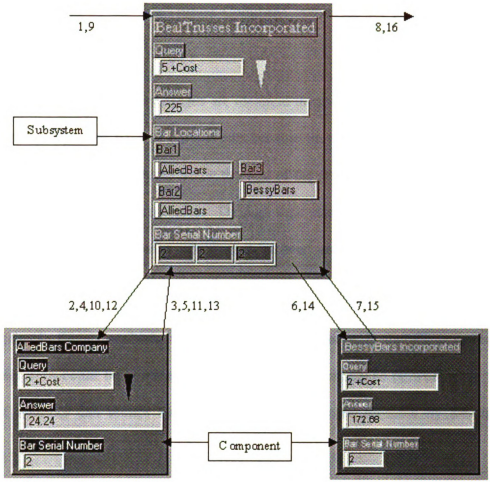


Figure 6. BealTrusses agent interaction for SA3x6 Cost solution

The BealTrusses Agent (Figure 6) is identical in structure to the AcmeSpans Agent and uses its resources to answer the cost query of network calls 1 and 9. The resources utilize the serial number obtained from the query handler to know the components that comprise this truss. The locations of these components are internally registered as part of the resources. The BealTrusses Agent uses it internal client to query these components: AlliedBars and BessyBars Agents. These agents query handlers receive the network calls 2, 4, 10, 12, 6, and 14. The answer is returned in network calls 3, 5, 11, 13, 7, and 15 (Table 2 and Figure 6). The BealTrusses Agent, then assembles this information using a

set of rules developed by the BealTrusses Company, and sends the subsystem answer back to the AcmeSpans Agent.

The BessyBars and AlliedBars Agents are identical in structure to any of the Truss or Span Agents, however they have no components because they are at the component level. These Agents through their resources know the cost of the serial number bar component. They request service from their resources. Through their internal set of rules, BessyBars and AlliedBars Agents generate a component answer and this information to the appropriate subsystem, system, or client software.

The Agents in this example do not have a working memory. In the above example, network calls 9-16 could be eliminated because they are identical to network calls 1-8. The use of memory can reduce the quantity of network calls required by the agent system.

Conclusions

The IEDA approach illustrated here is different then current approaches. The agent methodology of facilitating interaction between physical components rather than applications allows for various platforms to be seamlessly connected. The decomposable nature allows for unlimited size system formulation. This approach addresses local computation away from a centralized modeling platform. The standard ontology based communication protocol allows for proprietary security. The viable set of queries represents the services provided by the agent, and hence the corporation. The approach is system based and models global collaborative engineering as a system wherein the system is realized through coordinated interaction of the entities comprising the system.

The work presented by this paper is an initial step toward a valid global collaborative engineering design tool. It is a foundation of knowledge from which to build. Insight has been gained into the evolution of the IEDA through the implementation of the simple prototype example. This prototype, although limited, demonstrates the very basic nature of the IEDA. The limitations of the prototype arose in its sequential methodology, internally registered agents, a lack of global ontology, a lack of a working memory, which in turn created unneeded network overhead, and a lack of response granularity or parametric dependencies. Yet these shortcomings allow for the IEDA to grow. The topology of the IEDA is not complete. The model database must not only include its own models, but also a location of a registry agent containing the location of the agents comprising the sub-system or system. As the prototype showed, memory is an essential aspect needed to be included in the resources of the IEDA. The topology augmented with

memory, closely resembles a basic KBS structure (Dym, Levitt, 1991). The missing component is a knowledge acquisition facility. Incorporating this into the IEDA topology presupposes communication in two separate methods: non-proprietary vs. strict input/output proprietary communication. At one extreme a product may be fully developed, physically and an IEDA, and incorporated into a greater system in which the only communication may be of the ontology, proprietary based form. At the other extreme a product may be in the early stages of the design process where desired aspects of the product may be communicated both externally between companies and in-house. This communication through knowledge acquisition changes the ontology and allows for the current proprietary information of the changed product to be communicated safely. Also, this allows for the utilization of any current available technologies (Wooldridge, Jennings, 1998). Providing an un-exhaustive package for concurrent distributed design. The knowledge gained from this work helps extend into successful future developments.

The next step for the IEDA is to redesign, iterate, and expand from the existing prototype and proposed topology into a more complex field test. This entails more extensive performance capabilities, utilizing various modeling techniques, such as modular modeling, incorporating a greater amount of AI, detailing and defining the minimal requirements for a system of IEDA, and exploiting existing technologies. The appropriate architecture for efficiently representing the domains of discourse of the Global Ontology along with the governing body, Michigan State University, needs to be realized. The viable process of corporate DA published to the Agent registry and query submission to the Global Ontology. The required communication and action of the governing body of the Agent Registry and Global Ontology, as well as participating

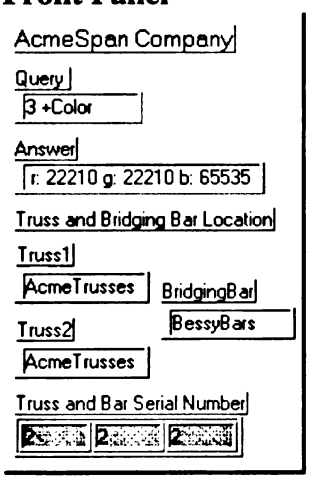
Corporations through DA adoption, for realizing a dynamic system of IEDA. It is a challenge of great scope that must be addressed by not only a variety of academic disciplines but along with industry as well. Through the cooperation of all the various engineering domains, the business domain, and industrial partners the project challenge can be met successfully. The result will be a tool of integrated computer related technologies for efficient and effective global collaborative engineering design.

A-1. Acme Span Software Documentation

The AcmeSpans.vi is the DA representing the Acme Spans Company and its available products. This DA answers queries through the collaboration of AcmeTrusses, BealTrusses, BessyBars, and AcmeBars Design Agents. It supplies cost, wieght, area, and color information for 5 products that the Acme Spans Company provides. It communicates via TCP/IP and waits for queries to arrive. Once recieved it uses its resources to provide an answer.

Connector Pane

Front Panel



Controls and Indicators

 port TCP port

 stop

 mode

 Answer 3

 Answer 2

 Answer 1

 Answer

 Truss and Bar Serial Number

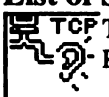
 Truss1

 Truss2

 BridgingBar

 Query

List of SubVIs



TCP Listen.vi
R:\control-cjr\Gosciak\My Documentsp3\CompositeTrussInc..llb\TCP Listen.vi

General Error Handler.vi
C:\Program Files\National Instruments\LabVIEW 6\vi.lib\Utility\error.llb\General Error Handler.vi

No EOC Error.vi


R:\control-cjr\Gosciak\My Documentsp3\CompositeTrussInc..llb\No EOC Error.vi

No Time Out Error.vi


R:\control-cjr\Gosciak\My Documentsp3\CompositeTrussInc..llb\No Time Out Error.vi

StringParcer2.vi
R:\control-cjr\Gosciak\My Documentsp3\CompositeTrussInc..llb\StringParcer2.vi

Scaller.vi
R:\control-cjr\Gosciak\My Documentsp3\AcmeSpans.llb\Scaller.vi
Spancost.vi
R:\control-cjr\Gosciak\My Documentsp3\AcmeSpans.llb\Spancost.vi
Spanarea.vi
R:\control-cjr\Gosciak\My Documentsp3\AcmeSpans.llb\Spanarea.vi
Spanweight.vi
R:\control-cjr\Gosciak\My Documentsp3\AcmeSpans.llb\Spanweight.vi
ColruleS.vi
R:\control-cjr\Gosciak\My Documentsp3\AcmeSpans.llb\ColruleS.vi
SAserialL.vi
R:\control-cjr\Gosciak\My Documentsp3\AcmeSpans.llb\SAserialL.vi
TandBLoc.vi
R:\control-cjr\Gosciak\My Documentsp3\AcmeSpans.llb\TandBLoc.vi

History

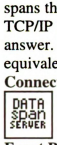
"AcmeSpans.vi History"

Current Revision: 60

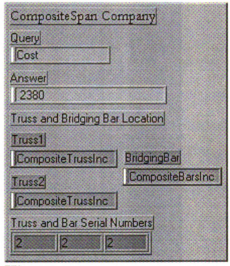
A-2. Composite Span Software Documentation

The CompositeSpanCo.vi is the DA representing the Composite Span Company. The DA answers queries through collaboration of the CompositeTruss and CompositeBar Design Agents. It supplies cost, weight, area, and color information for 3 composite spans the Composite Span Company provides to the User Software. It communicates via TCP/IP and waits for queries to arrive. Once received it uses its resources to provide an answer. It is located on computer eb2553p2.egr.msu.edu. The programming structure is equivalent to that of the Acme Span software structure.

Connector Pane



Front Panel



History

"CompositeSpanCo.vi History"

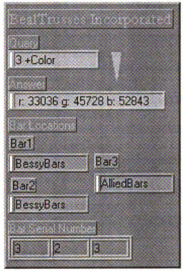
Current Revision: 60

B-1. Beal Trusses Software Documentation

The BealTrusses.vi is the DA representing the Beal Trusses Company and the products it provides. It communicates via TCP/IP and waits for queries to arrive. Once received it uses its resources to provide an answer of cost, weight, color, and area to either the AcmeSpans DA or the User Query. It provides 5 trusses through the collaboration of AlliedBars and BessyBars Design Agents. It is located on computer eb2553p2.egr.msu.edu.

Connector Pane

Front Panel

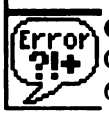


Controls and Indicators

U16	port TCP port
TF	stop
◆	mode
abc	Answer 3
abc	Answer 2
abc	Answer 1
DBL	Bar Serial Number
DBL	
abc	Answer
abc	Bar1
abc	Bar2
abc	Bar3
abc	Query

List of SubVIs

TCP Listen.vi
R:\control-cjr\Gosciak\My Documentsp2\CompositeSpanCo.llb\TCP Listen.vi



General Error Handler.vi
C:\PROGRAM FILES\NATIONAL INSTRUMENTS\LABVIEW 6\vi.lib\Utility\error.llb\General Error Handler.vi

No EOC Error.vi
R:\control-cjr\Gosciak\My Documentsp2\CompositeSpanCo.llb\No EOC Error.vi

No Time Out Error.vi
R:\control-cjr\Gosciak\My Documentsp2\CompositeSpanCo.llb\No Time Out Error.vi

StringParcer2.vi
R:\control-cjr\Gosciak\My Documentsp2\CompositeSpanCo.llb\StringParcer2.vi
Tcaller.vi
R:\control-cjr\Gosciak\My Documentsp2\BealTrusses.llb\Tcaller.vi
Trusscost.vi
R:\control-cjr\Gosciak\My Documentsp2\BealTrusses.llb\Trusscost.vi
Trussweight.vi
R:\control-cjr\Gosciak\My Documentsp2\BealTrusses.llb\Trussweight.vi
Trussarea.vi
R:\control-cjr\Gosciak\My Documentsp2\BealTrusses.llb\Trussarea.vi
Colrule.vi
R:\control-cjr\Gosciak\My Documentsp2\BealTrusses.llb\Colrule.vi
TBserialL.vi
R:\control-cjr\Gosciak\My Documentsp2\BealTrusses.llb\TBserialL.vi
BarDisplay.vi
R:\control-cjr\Gosciak\My Documentsp2\BealTrusses.llb\BarDisplay.vi

History

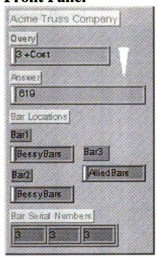
"BealTrusses.vi History"

Current Revision: 67

B-2. Acme Trusses Software Documentation

The AcmeTrusses.vi is the DA representing the Acme Trusses Company. It provides 3 trusses to the Acme Spans DA and the User Software.. This DA answers queries through the collaboration of the AlliedBars and BessyBars Design Agents. It supplies cost, area, weight, and color information. It communicates via TCP/IP and waits for queries to arrive in which it utilizes its resources to provide the answer. It is located on computer eb2553p1.egr.msu.edu. The programming structure is equivalent to that of the Beal Trusses software structure.**Connector Pane**

Front Panel



History

"Acmetrusses.vi History"

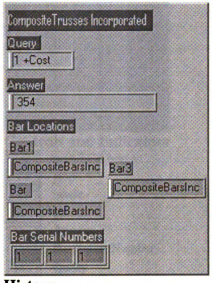
Current Revision: 62

B-3. Composite Trusses Software Documentation

The CompositeTrusses.vi is the DA representing the Composite Trusses Company and the products it provides. It communicates via TCP/IP and waits for queries to arrive. Once received it uses its resources to provide an answer of cost, weight, color, and area to either the Composite Spans DA or the User Query software. It provides 6 trusses through the collaboration of AlliedBars and BessyBars Design Agents. It is located on computer eb2553p3.egr.msu.edu. The programming structure is equivalent to that of the Beal Trusses software structure.

Connector Pane

Front Panel



History

"CompositeTrussInc.vi History"

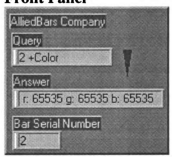
Current Revision: 67

C-1. Allied Bars Software Documentation

The AlliedBars.vi is the DA representing the Allied Bars Company. This company provides 3 bars to the AcmeTrusses, BealTrusses, and AcmeSpans Design Agents. It is a base component from which the above trusses and Spans are developed. It provides properties of cost, weight, color, and area. It communicates via TCP/IP and waits to provide service for upper level Design Agents as well as the User Software. It is located on computer eb2553p4.egr.msu.edu.

Connector Pane

Front Panel



Controls and Indicators

U16	port TCP port
TF	stop
◆	mode
abc	Serial# 4
abc	Answer
abc	Bar Serial Number
abc	Query

List of SubVIs

TCP Listen.vi
C:\WINDOWS\Profiles\Gosciak\Desktop\AlliedBars.llb\TCP Listen.vi

General Error Handler.vi
C:\Program Files\National Instruments\LabVIEW 6\vi.lib\Utility\error.llb\General Error Handler.vi

No EOC Error.vi
C:\WINDOWS\Profiles\Gosciak\Desktop\AlliedBars.llb\No EOC Error.vi

No Time Out Error.vi
C:\WINDOWS\Profiles\Gosciak\Desktop\AlliedBars.llb\No Time Out Error.vi

StringParcer2.vi
C:\WINDOWS\Profiles\Gosciak\Desktop\AlliedBars.llb\StringParcer2.vi
Barweight.vi
C:\WINDOWS\Profiles\Gosciak\Desktop\AlliedBars.llb\Barweight.vi
Barcosta.vi
C:\WINDOWS\Profiles\Gosciak\Desktop\AlliedBars.llb\Barcosta.vi
Bararea.vi
C:\WINDOWS\Profiles\Gosciak\Desktop\AlliedBars.llb\Bararea.vi

History

"AlliedBars.vi History"

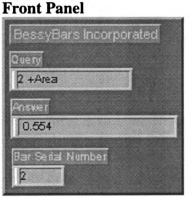
Current Revision: 40

C-2. Bessy Bars Software Documentation

The BessyBars.vi is the DA representing the Bessy Bars Company. This company provides 3 bars to the AcmeTrusses, BealTrusses, and AcmeSpans Design Agents. It is a base component from which the above trusses and Spans are developed. It provides properties of cost, color, are, and weight. It communicates via TCP/IP and waits to provide service for upper level Design Agents as well as the User Software. It is located on computer eb2553p1.egr.msu.edu. The programming structure is equivalent to that of the Allied Bars software structure.

Connector Pane

Front Panel



History

"BessyBars.vi History"

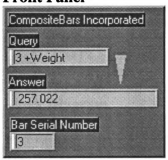
Current Revision: 42

C-3. Composite Bars Software Documentation

The CompositeBarsInc.vi is the DA representing the Composite Bars Company. This company supplies 3 Composite bars to both the Composite Truss Company and the Composite Span Company Design Agents. It is a base component from which the trusses and spans are developed. It provides answers for the properties of cost, weight, area, and color. It communicates via TCP/IP and weights to provide service for upper level Design Agents. It is located on computer eb2553p5.egr.msu.edu.

Connector Pane

Front Panel



History

"CompositeBars.vi History"

Current Revision: 42

D-1. Span Cost Software Documentation

The Spancost.vi is the general cost calculating vi for spans. It accepts the answers retrieved from the Scaller.vi. It is here where the span company can determine the required cost of the specific span. The cost could have been made specific to Serial Number. Its output is the cost of the span.

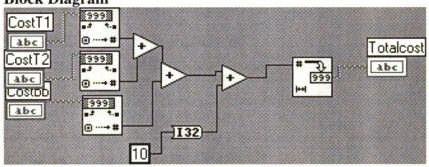
Connector Pane

Front Panel

Controls and Indicators

Costbb
CostT2
CostT1
Totalcost

Block Diagram



History

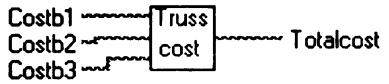
"Spancost.vi History"

Current Revision: 5

D-2. Truss Cost Software Documentation

The Trusscost.vi is the general cost calculating vi for the truss. It accepts the answers retrieved from the Tcaller.vi. It is here where the truss company can determine the required cost of the specific truss. The cost could have been made specific to Serial Number. Its output is the cost of the truss. Its software structure is equivalent to Spancost software structure.

Connector Pane



Front Panel

History

"Trusscost.vi History"

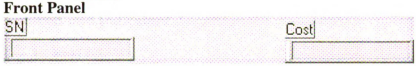
Current Revision: 4

D-3. Bar Cost Software Documentation

The Barcosta.vi is represented of the VI at the bar level which determine the cost of a bar depicted by the incoming Serial Number (SN). It uses a Hi-Q script to call a Matlab mfile script. The VI script has been pointed to an MFiles file located in the same file that holds the VI library this VI is a part of. The MFiles file contains Matlab script files utilized by the Various DA's as a resource. Matlab runs the script along side LabView. The cost of the bar is its output.

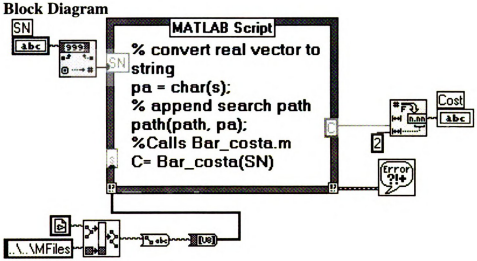
Connector Pane

Front Panel



Controls and Indicators

Block Diagram



List of SubVIs

General Error Handler.vi
C:\Program Files\National Instruments\LabVIEW 6\vi.lib\Utility\error.llb\General Error Handler.vi

History

"Barcosta.vi History"

Current Revision: 15

E-1. Span Area Software Documentation

The Spanarea.vi is the general area calculating vi for spans. It accepts the answers retrieved from the Scaller.vi. It is here where the span company can determine the required area of the specific span. Its output is the area of the span.

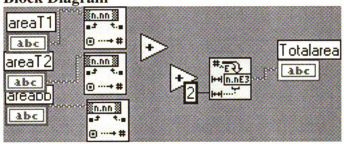
Connector Pane

Front Panel

Controls and Indicators

- areabb
- areaT2
- areaT1
- Totalarea

Block Diagram



History

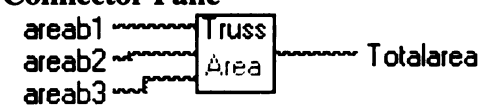
"Spanarea.vi History"

Current Revision: 8

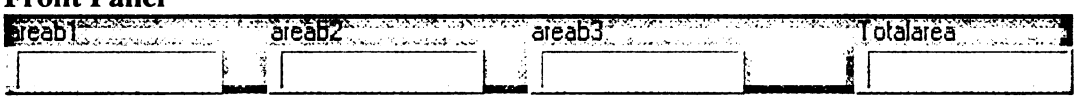
E-2. Truss Area Software Documentation

The Trussarea.vi is the general area calculating vi for trusses. It accepts the answers retrieved from the Tcaller.vi. It is here where the truss company can determine the required area of the specific truss. Its output is the area of the truss. Its software structure is equivalent to Spanarea software structure.

Connector Pane



Front Panel



History

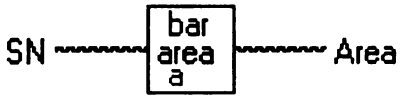
"Trussarea.vi History"

Current Revision: 8

E-3. Bar Area Software Documentation

The Bararea.vi is represented of the VI at the bar level which determine the area of a bar depicted by the incoming Serial Number (SN). It uses a Hi-Q script to call a Matlab mfile script. The VI script has been pointed to an MFiles file located in the same file that holds the VI library this VI is a part of. The MFiles file contains Matlab script files utilized by the Various DA's as a resource. Matlab runs the script along side LabView. The area of the bar is its output. The programming structure is equivalent to bar cost programming structure with the exception of what mfile is called.

Connector Pane



Front Panel

History

"Bararea.vi History"

Current Revision: 12

F-1. Span Weight Software Documentation

The Spanweight.vi is the general weight calculating vi for spans. It accepts the answers retrieved from the Scaller.vi. It is here where the span company can determine the required weight of the specific span. The weight could have been made specific to Serial Number. Its output is the weight of the span.

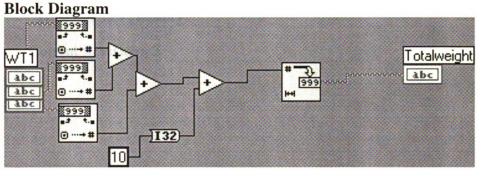
Connector Pane

Front Panel

Controls and Indicators

- Wbb
- WT2
- WT1
- Totalweight

Block Diagram



History

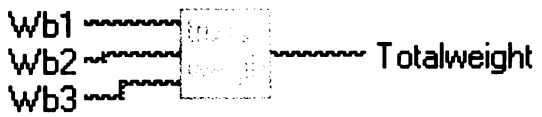
"Spanweight.vi History"

Current Revision: 3

F-2. Truss Weight Software Documentation

The Trussweight.vi is the general weight calculating vi for trusses. It accepts the answers retrieved from the Tcaller.vi. It is here where the truss company can determine the required weight of the specific truss, ie proprietary assembly technique allows for less weight then competitors. The weight could have been made specific to Serial Number. Its output is the weight of the truss. Its software structure is equivalent to Spanweight software structure.

Connector Pane



Front Panel

History

"Trussweight.vi History"

Current Revision: 3

F-3. Bar Weight Software Documentation

The Barweight.vi is represented of the VI at the bar level which determine the weight of a bar depicted by the incoming Serial Number (SN). It uses a Hi-Q script to call a Matlab mfile script. The VI script has been pointed to an MFiles file located in the same file that holds the VI library this VI is a part of. The MFiles file contains Matlab script files utilized by the Various DA's as a resource. Matlab runs the script along side LabView. The weight of the bar is its output. The programming structure is equivalent to bar cost programming structure with the exception of what mfile is called.

Connector Pane

Front Panel

History

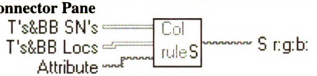
"Barweight.vi History"

Current Revision: 9

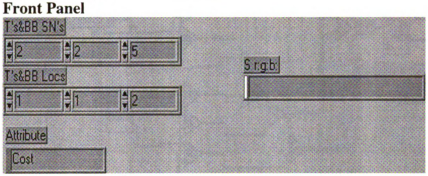
G-1. ColruleS Software Documentation

The ColruleS.vi accepts the string of component serial number, locations, and color query. Through its internal vi's of ColareaS and RGBParcer a calculation of the r:g:b: color scale is determined for the specific span of choice.

Connector Pane



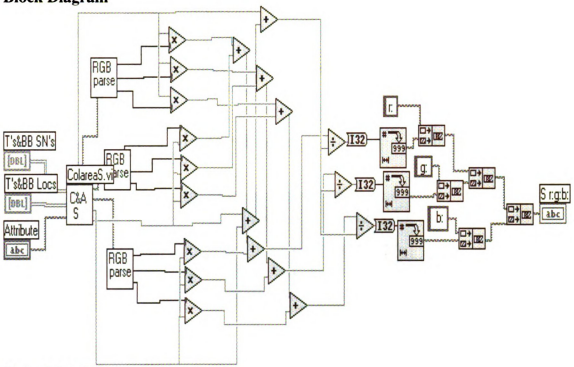
Front Panel



Controls and Indicators

- Attribute
- T's&BB SN's
- T's&BB Locs
- S r:g:b:

Block Diagram



List of SubVIs

RGB parse	**RGBParcer.vi** R:\control-cjr\Gosciak\My Documentsp3\AcmeSpans.llb\RGBParcer.vi
C&A S	**ColareaS.vi** R:\control-cjr\Gosciak\My Documentsp3\AcmeSpans.llb\ColareaS.vi

History

"ColruleS.vi History"

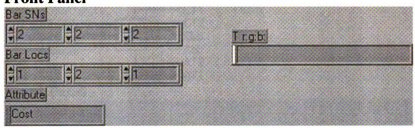
Current Revision: 9

G-2. Colrule Software Documentation

The Colrule.vi accepts the string of bar SN, bar locations, and cost query. Through its internal vi's of Colarea and RGBParcer a calculation of the r:g:b: color scale is determined for the specific truss of choice. The programming structure is equivalent to that of the ColruleS software structure.

Connector Pane

Front Panel



History

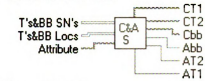
"Colrule.vi History"

Current Revision: 10

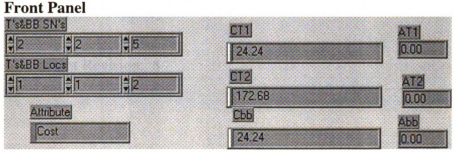
G-3. ColareaS Software Documentation

The ColareaS.vi based upon the component Serial Number and location retrieves the color and area of the components making up the Span. It uses the Scaller.vi twice to accomplish this, which uses an internal client, user software. It gives this information to the ColruleS.vi.

Connector Pane



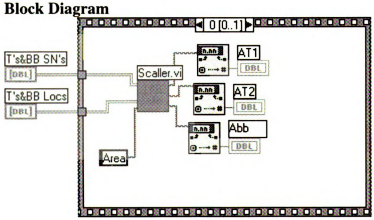
Front Panel

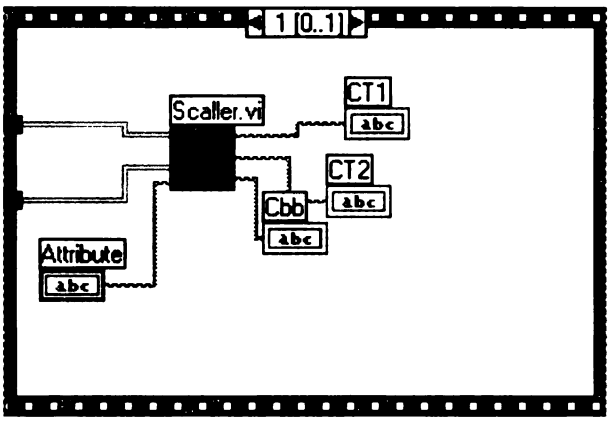


Controls and Indicators

- Attribute
- T's&BB SN's
- T's&BB Locs
- CT1
- CT2
- Cbb
- AT1
- AT2
- Abb

Block Diagram





List of SubVIs

Scaller.vi
R:\control-cjr\Gosciak\My Documentsp3\AcmeSpans.llb\Scaller.vi

History

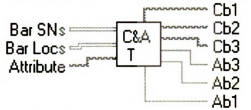
"ColareaS.vi History"

Current Revision: 8

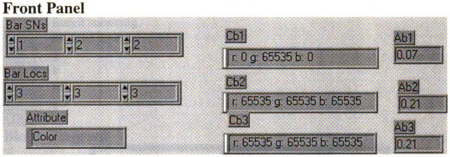
G-4. Colarea Software Documentation

The Colarea.vi based upon the bar Serial Number and bar location retrieves the color and area of the bars making up the truss. It uses the Tcaller.vi twice to accomplish this, which uses an internal client, user software. It gives this information to the Colrule.vi. The programming structure is equivalent to that of the ColareaS software structure.

Connector Pane



Front Panel



History

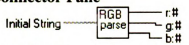
"Colarea.vi History"

Current Revision: 8

G-5. RGBParcer Software Documentation

The RGBParcer takes an incoming r:g:b color string and parces it into individual r, g, and b scales.

Connector Pane

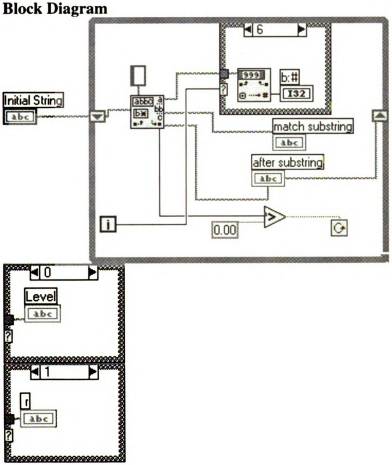


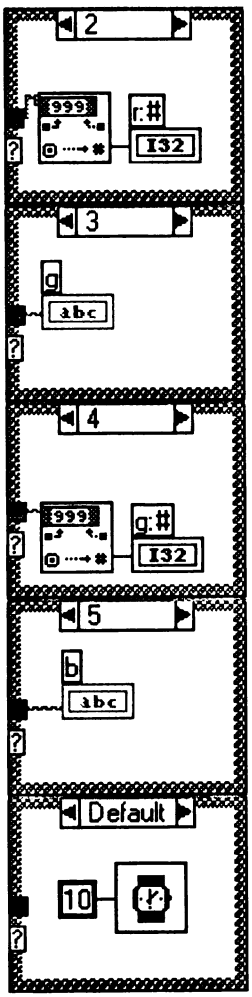
Front Panel

Controls and Indicators

- Initial String
- Level
- r
- match substring
- after substring
- g
- b
- r:#
- g:#
- b:#

Block Diagram





History

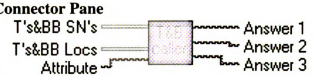
"RGBParcer.vi History"

Current Revision: 12

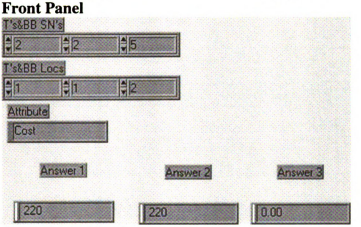
H-1. Scaller Software Documentation

The Scaller.vi accepts the component Serial Number and location strings and attribute. This VI is the general Span level location identifier and internal client. The location string is utilized to access an array, internal registry, to generate a location string, which along with the appropriate Serial Number and attribute is given to the internal client or user software. It outputs lower level answers.

Connector Pane



Front Panel



Controls and Indicators

- T's&BB SN's
- T's&BB Locs
- Attribute
- T's&BB Location
- String
- Answer 3
- Answer 1
- Answer 2

List of SubVIs

StringParcer2.vi
R:\control-cjr\Gosciak\My Documentsp3\CompositeTrussInc..llb\StringParcer2.vi
SInternalClient.vi
R:\control-cjr\Gosciak\My Documentsp3\AcmeSpans.llb\SInternalClient.vi

History

"Scaller.vi History"

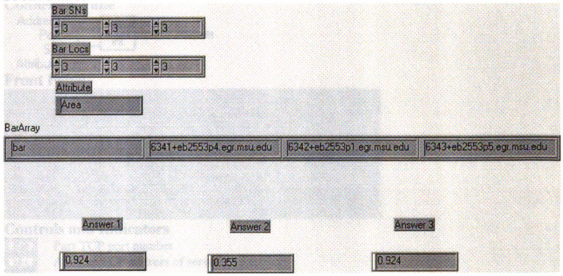
Current Revision: 23

H-2. Tcaller Software Documentation

The Tcaller.vi accepts the bar Serial Number and bar location strings and attribute. This VI is the typical truss level location identifier and internal client and equivalent in structure to the Scaller vi. The location string is utilized to access an array, internal registry, to generate a location string, which along with the appropriate Serial Number and attribute is given to the internal client or user software. It outputs lower level answers.

Connector Pane

Front Panel



History

"Tcaller.vi History"

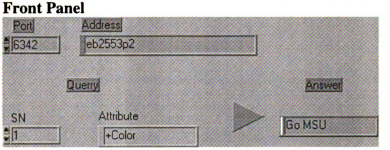
Current Revision: 23

H-3. SInternalClient Software Documentation

This is a standard client vi. It sends a string to the registered agents. Specifying the port and address of where they have been registered out on a TCP/IP network accesses the registered agents. As of now, all agents: bar, truss, span will be in the egr.msu.edu domain. This could make the address for now as localhost, eb2553p1, or eb2553p2. ie If on computer p1 must specify eb2553p2 to access BarAgentA at port 6341. If on p2 then address is localhost. BarAgentA is @ port 6341 BarAgentB is @ port 6342. BarAgentC is @port 6343 and is an Agent for composite bars. The port numbers for truss level and span level agents increases by 10. The string is a form of request, query, and the solution, answer, is returned. The answer is the data given by the agent request made by this client. A garbage level can be chosen to show effective bad string case structure. And not crash the requested agent/server/client.

Connector Pane

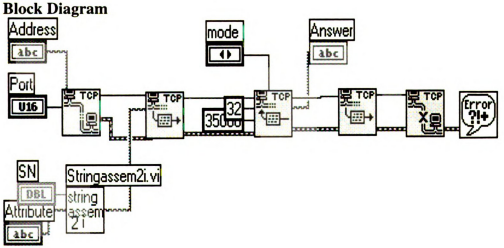
Front Panel



Controls and Indicators

- U16 — Port TCP port number
- abc — Address TCP address of server
- mode
- abc — Attribute
- DBL — SN
- abc — Answer

Block Diagram



List of SubVIs

General Error Handler.vi
C:\Program Files\National Instruments\LabVIEW 6\vi.lib\Utility\error.llb\General Error Handler.vi

Stringassem2i.vi
R:\control-cjr\Gosciak\My Documentsp3\AcmeSpans.llb\Stringassem2i.vi

History

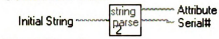
"SInternalClient.vi History"

Current Revision: 28

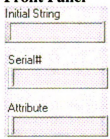
H-4. StringParcer2 Software Documentation

The StringParcer2.vi takes the incoming query string and parses it into usable segments for the DA. The string is broken into a Serial Number and the desired query.

Connector Pane



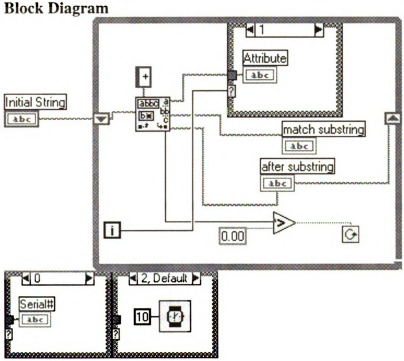
Front Panel



Controls and Indicators

- Initial String
- Serial#
- Attribute
- match substring
- after substring

Block Diagram



History

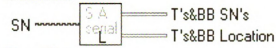
"StringParcer2.vi History"

Current Revision: 7

H-5. SAserialL Software Documentation

The SAserialL.vi take the incoming Serial number for the Span and uses this as an input for a Matlab script call. This VI is the general Span level vi for determining the required Serial number and locations of the components comprising the specified span. The output of the VI is a string of component Serial numbers and locations.

Connector Pane

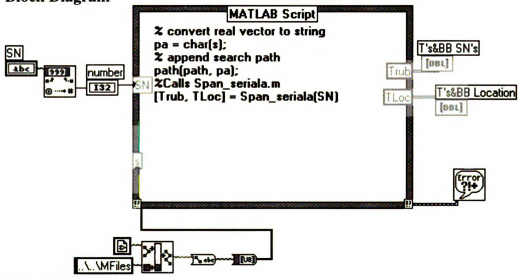


Front Panel

Controls and Indicators

- SN
- T's&BB Location
- T's&BB SN's
- number

Block Diagram



List of SubVIs

General Error Handler.vi
C:\Program Files\National Instruments\LabVIEW 6\vi.lib\Utility\error.llb\General Error Handler.vi

History

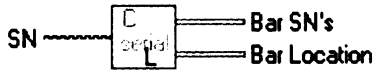
"SAserialL.vi History"

Current Revision: 19

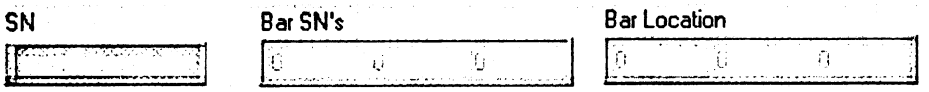
H-6. TCserialL Software Documentation

The TCserialL.vi take the incoming Serial number for the truss and uses this as an input for the Matlab script call. This VI is the typical truss level vi for determining the required Serial numbers and locations of the bars comprising the specified truss. The output of the VI is a string of bar Serial numbers and bar locations. The programming structure is equivalent to that of the SAserialL software structure.

Connector Pane



Front Panel



History

"TCserialL.vi History"

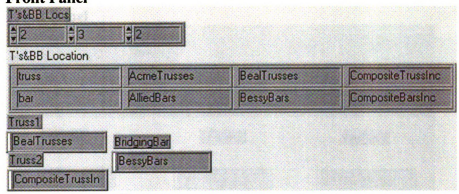
Current Revision: 19

I-1. TandBLoc Software Documentation

The TandBLoc.vi is used only for display purposes of the DA location of the components comprising the specified span.

Connector Pane

Front Panel



Controls and Indicators

[DBL] T's&BB Locs

[DBL]

[abc] T's&BB Location

[abc] String

[abc] BridgingBar

[abc] Truss1

[abc] Truss2

History

"TandBLoc.vi History"

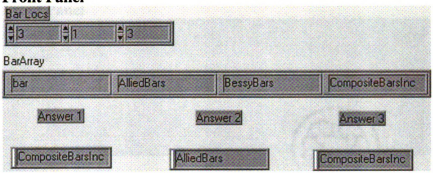
Current Revision: 25

I-2. BarDisplay Software Documentation

The BarDisplay.vi is used only for display purposes of the DA location of the components comprising the specified truss. The programming structure is equivalent to TanbBLoc software programming.

Connector Pane

Front Panel



History

"BarDisplay.vi History"

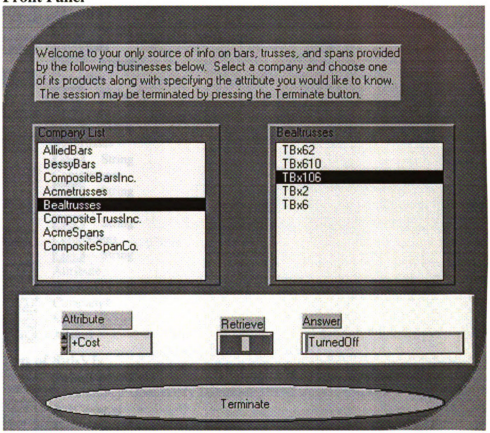
Current Revision: 24

I-3. FrontL Software Documentation

This VI is the directory that allows a user to scroll through the 8 various company's Design Agents along with an ability to choose what bar, truss, or span serial number the user is interested in. The user can choose from cost, color, area, and weight. This vi needs to run in a contiunuous mode for it to operate more correctly.

Connector Pane

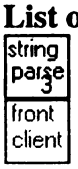
Front Panel



Controls and Indicators

- Company List
- BarCorporationA
- BarCorporationB
- CompositeBarsInc.
- Acmetrusses
- Bealtrusses
- CompositeTrussInc.
- AcmeSpans
- CompositeSpanCo.
- Retrieve
- AlliedBars
  - String
- BessyBars
  - String
- CompBar
  - String
- Acmetruss
  - String
- Bealtrusses
  - String
- Comptrusses
  - String
- AcmeSpans
  - String
- CompSpans
  - String
- Attribute
- Boolean
- Company#
- String
- Answer

List of SubVIs



StringParcer3.vi
R:\control-cjr\Gosciak\My Documentsp3\ClientUser.llb\StringParcer3.vi

3rdClient.vi
R:\control-cjr\Gosciak\My Documentsp3\ClientUser.llb\3rdClient.vi

History

"FrontL.vi History"

Current Revision: 81

J-1. MatLab Script Documentation

Bar Area mfile

```
function A= Bar_areaa(SN)
%so far just SN gravity will be assumed to be earths
%Bar property table as a row is [L0 A E kg/m]
D= [ 2 0.00142 2.07E11 8.5
    6 0.00142 2.07E11 8.5
    10 0.00142 5.00E9  8.5];
 %The width is calculated by taking the xarea A dividing
 %by value a little bit greater than its square.
 if SN > 3 | SN <= 0
    A=0;
 else
    L=D(SN,1);w=(D(SN,2))/.04;
    A=L*w;
 end
return
```

Bar Cost mfile

```
function C= Bar_costa(SN)
%so far just SN
%Bar property table as a row is [L0 A E kg/m]
D= [8.08
    24.24
    40.40];
 %The mass is calculated by entry 1 and 4 of the rows
 if SN > 3 | SN <= 0
    C=0;
 else
    C=D(SN,1);
 end
return
```

Bar Weight mfile

```
function W= Bar_weighta(SN)
%so far just SN Bar property table as a row is [L0 A E kg/m]
D= [ 2 0.00142 2.07E11 8.5
    6 0.00142 2.07E11 8.5
    10 0.00142 5.00E9  8.5];
 %The mass is calculated by entry 1 and 4 of the rows
 if SN > 3 | SN <= 0
    W=0;
 else
    m=D(SN,1)*D(SN,4);
    W=m*9.81;
 end
return
```

Truss Serial mfile

```
function [Bars, Loc]= Truss_seriala(SN)
%SNL is the Serial Number and Location of the bars contained in specified truss. The
%first 3 and last three columns represent the bar SN and location of where registered:
%column 1 and 4 represent bar 1.  A location of 1 represents bar agent A
SNL= [ 1 1 1 1 1 2
     2 2 2 1 2 1
          3 3 3 2 2 1];
Bars=SNL(SN,1:3);
Loc=SNL(SN,4:6);
Return
```

Span Serial mfile

```
function [Trub, TLoc]= Span_seriala(SN)
%SNL is the Serial Number and Location of the Trusses and Bridging Bar contained in
%specified Span.  The first 3 and last three columns represent the Truss,Bar SN and
%location of where registered: column 1 and 4 represent Truss1.  A location of 1
%represents TrussAgent A. The spans are comprised of trusses found at different servers
%to verify the connection various entities.
SNL= [ 1 1 1 1 1 1
     4 4 1 2 2 1
     2 2 2 1 1 2
     5 5 2 2 2 1
     3 3 3 1 1 1];
Trub=SNL(SN,1:3);
TLoc=SNL(SN,4:6);
return
```

REFERENCES

AEI, 2000a, "Redesigning Work Processes And Computing Environments", Automotive Engineering International, SAE International, Brimfield, OH, July, pp. 147-149.

AEI, 2000b, "Behind GM's Global Design Success", Automotive Engineering International, SAE International, Brimfield, OH, December, pp. 74-75.

Bokulich, F., ed., AEI 1999, "CAD software integration", Automotive Engineering International, SAE International, Brimfield, OH, April, pp. 52.

Bradshaw, J., ed., 1997, *Software Agents*, AAAI Press, Menlo Park, CA., ISBN 0-262-52234-9, pp. 1-46.

Brenner, W., Zarnekow, R., and Wittig, H, 1998, *Intelligent Software Agents Foundations and Applications*, Springer-Verlag, Berlin Heidelberg New York, ISBN 3-540-63411-8, pp. 1-86.

Bucholz, K., ed., AEI 1998, "SAE Global Vehicle Development Conference", Automotive Engineering International, SAE International, Brimfield, OH, November, pp. 105.

Byam, B. P. and Radcliffe, C. J., 1999, Nov. 14-19, Nashville, Tenn., "Modular Modeling of Engineering Systems Using Fixed Input-Output Structure", *Proc. of the ASME Dynamic Systems and Control Division, 1999 ASME Intl. Mech. Engin. Conf and Exposition,* DSC-Vol. 67, ISBN 0-7918-1634-6, pp. 545-551.

Byam, B. P. and Radcliffe, C. J., 2000, Sept. 10-13, Baltimore, Maryland, "Direct Insertion Realization of Linear Modular Models of Engineering Systems Using A Fixed Input-Output Structure", *Proc. ASME Design Engineering Technical Conferences, 26th Design Automation Conference,* DETC2000/DAC-14236, ASME

Dorf, R. C. and Kusiak, A., ed., 1994, *Handbook of Design, Manufacturing and Automation*, John Wiley & Sons, New York, ISBN 0-471-55218-6, pp. 25-33, 977-990.

Dym, C. L. and Levitt, R. E., 1991, *Knowledge Based Systems in Engineering*, McGraw-Hill, Inc., ISBN 0-07-018563-8, pp. 1-80.

Esterman, M. and Ishii, K., 1999, Sept.12-15, Las Vegas, NV, "Challenges in Robust Concurrent Product Development Across the Supply Chain", *Proc. ASME Design Engineering Technical Conferences*, DETC99/DFM-8920, ASME

Murch, R., Johnson, T., 1999, *Intelligent Software Agents*, Prentice Hall, Inc., Upper Saddle River, New Jersey, ISBN 0-13-011021, pp. 1-45.

Rosenman, M. and Fujun W., 1999, "CADOM: A Component Agent-based Design-Orientated Model for Collaborative Design", Springer-Verlag London Limited, *Research in Engineering Design*, pp. 193-205.

Tasso, C., Oliveira, E., ed., 1998, *Development of Knowledge-Based Systems For Engineering*, Springer-Verlag, Wien New York, ISBN 3-211-82916-4, pp. 11-35, 201-209.

Tecuci, G., 1998, *Building Intelligent Agents*, Academic Press, San Diego, Cal., ISBN 0-12-685125-5, pp. 1-12.

Wallace, D., Borland, N., Senin, N., and Abrahamson, S., 1999, Sept. 12-15, Las Vegas, NV, "Integrated Engineering, Geometric, And Customer Modeling: LCD Projector Design Case Study," *Proc. of the ASME Design Engineering Technical Conferences*, DETC99/CIE-9084, ASME

Wooldridge, M. and Jennings, N. R., 1998, "Pitfalls of Agent-Orientated Development," *Proceedings of the Second International Conference on Autonomous Agents*, pp. 385-391.

Wooldridge, M. and Jennings, N. R., ed., 1998, *Agent Technology Foudnations, Applications, and Market*, Springer-Verlag, Berlin Heidelberg New York, ISBN 3-540-63591-2, pp. 1-44.

GENERAL REFERENCES

Crabb, H. C., 1998, The Virtual Engineer, Howard C. Crabb and the Society of Manufacturing Engineers, ISBN 0-7918-0066-0

Cutkosky, M., Engelmore, R., Fikes, R., Genesereth, M., Gruber, T., Mark, W., Tenenbaum, J., and Weber, J., "PACT: An Experiment in Integrating Concurrent Engineering Systems", 1998, Readings In Agents, Morgan Kaufmann Publishers, Inc., USA, ISBN 1-55860-495-2, pp. 46-55.

Hisup, P. and Cutkosky, M. R., 1999, "Framework for Modeling Dependencies in Collaborative Engineering Processes", Springer-Verlag London Limited, *Research in Engineering Design*, pp. 84-102.

Thompson, B. S., 1997, *Creative Engineering Design*, 4th ed., Okemos Press, Okemos, Mi., ISBN 0-9630471-8-3.