On the Development of a Computing Infrastructure that Facilitates IPPD from a Decision-Based Design Perspective

Mark A. Hale*, James I. Craig†, Farrokh Mistree§, Daniel P. Schrage††

Georgia Institute of Technology
Atlanta, Georgia 30332-0140

Abstract

Integrated Product and Process Development (IPPD) embodies the simultaneous application of both system and quality engineering methods throughout an iterative design process. The use of IPPD results in the time-conscious, cost-saving development of engineering systems. Georgia Tech has proposed the development of an Integrated Design Engineering Simulator that will merge Integrated Product and Process Development with interdisciplinary analysis techniques and state-of-the-art computational technologies. To implement IPPD, a Decision-Based Design perspective is encapsulated in an approach that focuses on the role of the human designer in product development. The approach has two parts and is outlined in this paper. First, an architecture, called DREAMS, is being developed that facilitates design from a decision-based perspective. Second, a supporting computing infrastructure, called IMAGE, is being designed. The current status of development is given and future directions are outlined.

Glossary

IPPD ....**I**ntegrated **P**roduct and **P**rocess **D**evelopment. Embodies the simultaneous application of both system and quality engineering methods throughout an iterative design process.

DBD ....**D**ecision-**B**ased **D**esign. A paradigm that captures the notion that the principle role of a designer is to make decisions.

DSPT ....**D**ecision **S**upport **P**roblem **T**echnique. A technique for implementing Integrated Product and Process Development from a decision-based perspective. The technique facilitates the partitioning of a design problem through the use of Support Problems.

Support Problem ..A model that describes the transformation of information into knowledge. The model has a structure defined by keywords. Support Problems can be used to model an entire design timeline.

DREAMS ....**D**eveloping **R**obust **E**ngineering **A**nalysis **M**odels and **S**pecifications. An architecture that formally supports Decision-Based Design.

IMAGE ....**I**ntelligent **M**ultidisciplinary **A**ircraft **G**eneration **E**nvironment. A computing infrastructure that facilitates design from a decision-based perspective.

Funding for this paper is provided under the NASA MDO Grant (NAG-1-1564) and the Graduate Student Researchers Program (NGT-51250) directed by the NASA Langley High Performance Computing and Communications Program.

* Graduate Researcher, Student Member AIAA, Aerospace Systems Design Laboratory, Corresponding Author

† Professor, Member AIAA, Aerospace Systems Design Laboratory

§ Professor, Member AIAA, Systems Realization Laboratory

†† Professor, Member AIAA, Aerospace Systems Design Laboratory

1 Background and Motivation

Considerable time and effort has been invested in the development of new computing technologies and their associated methods for **I**ntegrated **P**roduct and **P**rocess **D**evelopment (IPPD). Unfortunately, these efforts have resulted in implementations that are disjoint in their application. A designer-centered approach is taken that focuses these efforts on the development of a structured decision support process that originates from the designer's perspective. A formal, structured technique for the embodiment of a decision-based perspective and a corresponding computer-based implementation scheme is taken in the approach.

Computer-based resources play a significant role in generating knowledge about a design. Considerable time and expense has been given to the development of the computing technologies required for better resource efficacy. These technologies have been applied in systems that emphasize modularity, interdisciplinary program utilization, resource collaboration, and distributed processing.[1-8] These systems have marked improvements in information processing. However, their applicability to aiding the designer in making decisions based on new design knowledge remains questionable. Furthermore, the applicability of these systems to a continuous, iterative design life-cycle has not been shown and is uncertain.

A two-step approach is being taken in the design of a new computing infrastructure for assisting a designer in making decisions. First, a coherent, systematic decision-making architecture that is used to structure, but not restrict, the means by which the designer solves the design problem (story) is formalized. Second, after the necessary components of the new technique are identified, an open computing infrastructure is designed to explicitly include support for these design related components. Then, specific tools for successful computing operation are identified independently of the design related activities.

The approach used here will result in a computing environment that will serve to implement IPPD. The resulting implementation will:

- *Facilitate designing from a decision-based perspective;*
- *Provide a means for both partitioning a design problem and of solving the resulting Support Problems; and*
- *Focus the application of new and existing computing technologies toward assisting a designer in applying engineering methods.*

The two elements of this approach are outlined in this paper. First, an architecture that facilitates design from a decision-based perspective is formalized. Secondly, a suitable computing infrastructure used to implement the design architecture is presented.

2 A Designer Makes Decisions

The basic premise set forth in the design of the computing infrastructure is that the framework exists to aid the designer in making decisions. Fundamentally, the principal role of the designer is to make decisions throughout the design process. A paradigm for capturing this perspective is ***D**ecision* ***B**ased* ***D**esign* (DBD). One embodiment of DBD is the ***D**ecision* ***S**upport* ***P**roblem* ***T**echnique* (DSP Technique).[9-11] The DSP Technique is used to implement IPPD from a decision-based perspective. Integrated Product and Process Development embodies the simultaneous application of system and quality engineering methods throughout the iterative design process. Other implementations focus on the identification and application of methods to be employed in IPPD. The DSP Technique facilitates IPPD by providing a designer a means for partitioning design processes into an organized solution scheme, utilizing Support Problems. Not only is the design problem defined, but an achievable solution process is known.

An idealized timeline is shown in Figure 1. The process of designing is encapsulated by *Phases*, *Events*, and *Information*.[9, 12] There are four design phases depicted in Figure 1. During those four phases, design events transform the Statement of Requirements into the Total Life-Cycle Design Knowledge. In fact, all design events serve to augment the knowledge about a design. A potential sequence of events for the Conceptual Design phase is illustrated at the bottom of Figure 1. Perhaps, one can think of the sequence as being exploded by double-clicking on the  icon. Design events may occur simultaneously, thus requiring the use of multidisciplinary and concurrent analysis techniques.

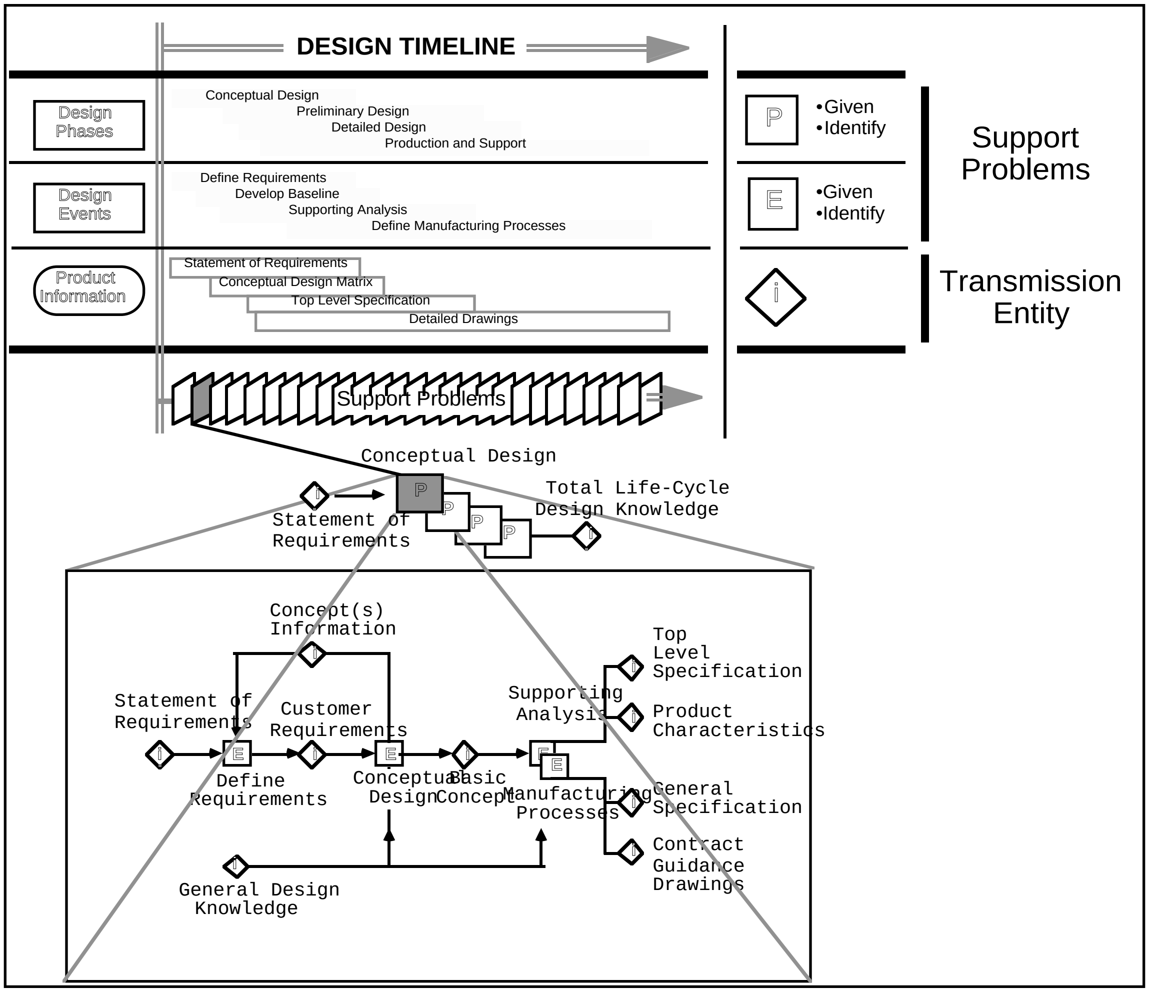

Figure 1. *A Design Timeline*

Support Problems model, with the aid of keywords, the design related activities that occur during Phases and Events. The result is that Support Problems govern the systematic creation of new design knowledge about a product. As shown in Figure 1, Support Problems are used continually and in a consistent fashion by the designer throughout a design's life-cycle. The corresponding Support Problem keywords and representative icons are illustrated to the right in Figure 1. Information is encapsulated as a transmission entity.

A second level of abstraction further clarifies the procedures that form design events. Events are embodied by additional *Events*, *Tasks*, *Decisions*, *System Parameters*, and *Information*, see Figure 2. These entities are coalesced into a design sequence similar to that in Figure 1. Notice that additional Support Problems are defined for the new entities.

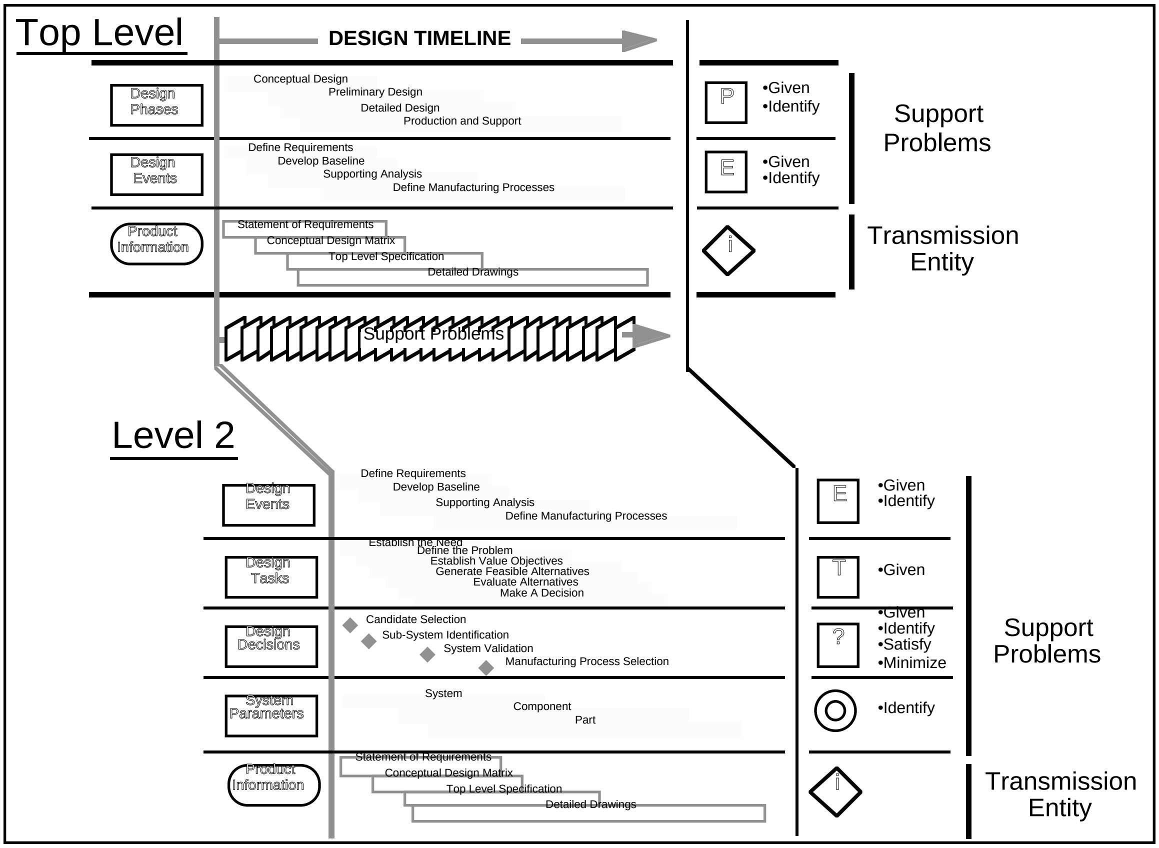

Figure 2. *Varying Levels of Design Abstraction*

3 DREAMS - An Architecture that Facilitates Decision-Based Design

The architecture that is being developed aids a designer in solving a design problem and is shown in Figure 3. Based on the DSP Technique described in the previous section, the architecture has three fundamental components:

- *A designer's perspective*, the role of the designer is to make decisions throughout a design's life-cycle;
- *Support Problem definition and solution*, knowledge about the design is used by the designer to make decisions and Support Problems model the transformation of information into knowledge; and
- *Design management*, knowledge about the design is appropriately managed so that it can be used in making decisions.

The implementation of this architecture will aid the designer in **D**eveloping **R**obust **E**ngineering **A**nalysis **M**odels and **S**pecifications. Thus, the architecture is named DREAMS.

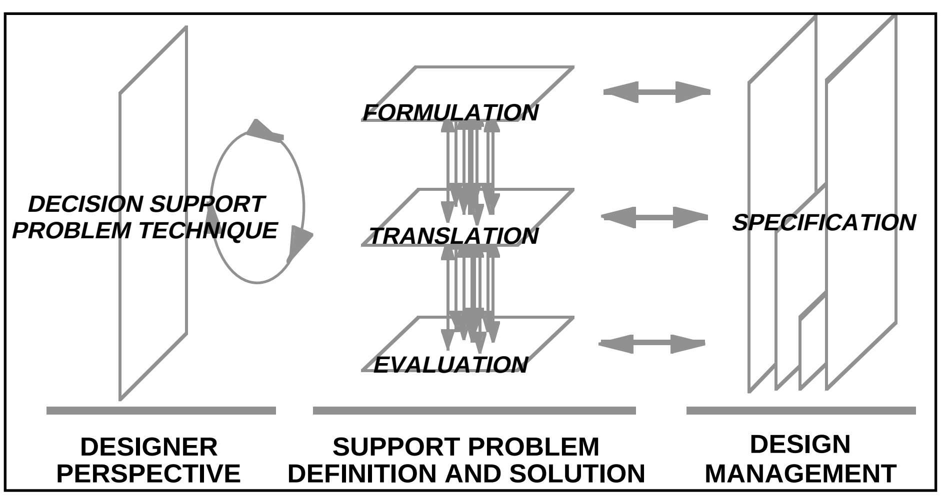

Figure 3. *DREAMS Architecture*

3.1 A Designer's Perspective

There are two phases in the DSP Technique:

- *The meta-design phase*, whereby the designer designs the design process with the aid of Support Problems; and
- *The actual design phase*, whereby Support Problems are exercised so that knowledge about the design can be generated and decisions can be made.

A meta-design phase allows a designer to explicitly model design process using Support Problems. Meta-design results in design sequences similar to that found in the bottom of Figure 1. Care should be taken when interpreting the use of the Decision Support Problem Technique. The DSP Technique is used to capture and focus designer decisions and should not restrict design possibilities. Decisions that have already been made are documented in the DSP Technique. Those decisions may be from earlier in the current design or may be from other designs. A designer can then use these previous decisions to plan future decision-making milestones. However, only near term decision-making should be exercised. Therefore, a designer maintains the flexibility to correct and append to future design activities.

After laying out the design process in meta-design, a designer can use the DSP Technique to generate knowledge used for decision-making through the use of Support Problems. For example, each icon in Figure 1 corresponds to an associated Support Problem. One kind of Support Problem is the Decision Support Problem and is used to explicitly declare a decision making process. A sub-class of DSP's is the Compromise DSP. The Compromise DSP has a linguistic statement comprising of the *Given*, *Satisfy*, *Find*, *Minimize* keywords.

3.2 Support Problem Definition and Solution

Within the DSP Technique, Support Problems are exercised by a designer to produce knowledge about a design so that decisions can be made based on that knowledge. Support Problems provide standard models for transforming design information into knowledge. There are three steps required in defining and solving Support Problems:

- *Formulation*, the structuring of the problem statement into specific Support Problem models;
- *Translation*, associating processes, that govern the generation of information into knowledge, with the Support Problems; and
- *Evaluation*, producing design knowledge through the solution of the Support Problems.

These three steps are illustrated with a Multidisciplinary Wing Integration Compromise Decision Support Problem in Figure 4. This problem is being investigated under the "New Approaches to HSCT Multidisciplinary Design and Optimization" research study at Georgia Tech and examines the flutter and buckling constraints imposed on the High Speed Civil Transport.[13]

3.2.1 Formulation - Support Problem Support Problems are defined when the design process is partitioned in meta-design. As shown in Figures 1 and 2, Support Problems have a defined structure given by keywords. The Compromise DSP has the form:[14]

Given: Feasible Design and Aspiration Space
Find: Values of Variables
Satisfy: Systems Constraints, Bounds, and Goals
Minimize: Deviation between "what I want" and "what I can have"

Support Problems are formulated as linguistic statements, a form natural to the designer and, hopefully, unambiguous in meaning.

3.2.2 Translation - Math Form. Once a Support Problem has been formulated, the problem is translated into an equivalent Math Form. The Math Form provides the process connectivity between forms and functions. For instance, the functions `lift` and `drag` are associated with the form `wing` through the relation:

$$\frac{L}{D} = \frac{C_l}{C_{D_o} + K\,C_l^2} \tag{1}$$

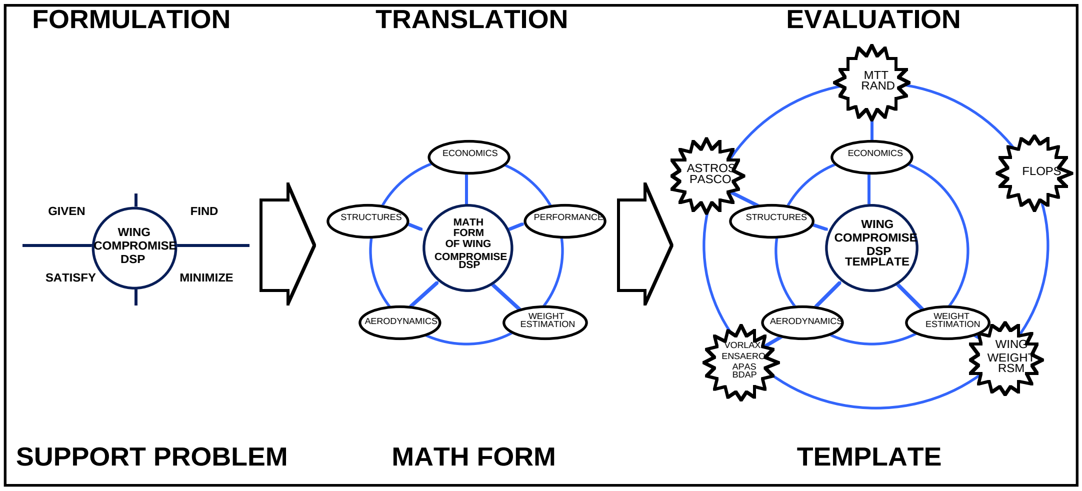

Figure 4. *Multidisciplinary Wing Integration Compromise Decision Support Problem*

As the Math Form becomes more complex, equations are typically grouped into engineering models. In turn, models are often grouped into disciplines. In Figure 4 some of the traditional aerospace disciplines that are present in the Multidisciplinary Wing Integration problem are shown. Notice that inter-disciplinary models do exist, as in the case of aeroelasticity, and must be accounted for. Looking again at Figure 4, the problem definition can be visualized as an expanding cone. The Compromise DSP forms the frustum of the cone and the cone expands as the problem is translated into the Math Form.

3.2.3 Evaluation - Template. Finally, the Math Form of the Support Problem can be solved. The Support Problem solution consists of three steps: pairing the Math Form with a suitable Design Operator, structuring a solution network, and solving the Problem. A Design Operator generates additional design information from the expressions found in the Math Form of a Support Problem. (The familiar computational counterpart to the Design Operator is the *Agent*.) As shown in Figure 4, Design Operators are typically engineering analysis codes. Other Design Operators include expert systems, hyper-media sources, virtual reality, and the human designer. The combination of the Math Form and the Design Operator is called a Design Event. The Design Event is important because during its instantiation knowledge about the design is generated. After the Math Form and Design Operators have been collected, the new form of the Support Problem is called the Support Problem Template. The SP Template forms the base of the expanding cone.

Continuing, a solution network must be generated for the Design Events. Figure 5 shows a solution network corresponding to the Multidisciplinary Wing Integration Problem shown in Figure 4.[15] Finally, the Support Problem must be solved. **D**ecision **S**upport **I**n the **D**esign of **E**ngineering **S**ystems (DSIDES) is a suite of tools used to solve Support Problems.[10] Tools in DSIDES can be used to solve Selection DSPs (SELECT) and multi-level, multi-goal Compromise DSPs (ALP).

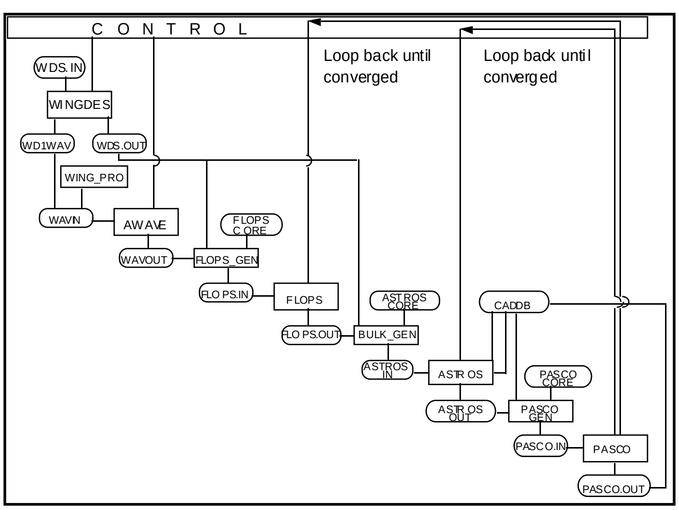

Figure 5. *Partial Solution Network for the Multidisciplinary Wing Integration Problem*

3.3 Design Management

Design management aids a designer in reviewing knowledge for decision-making and is included in the DREAMS architecture, see Figure 3. A suitable design management scheme is extremely important since information is used in decision-making throughout the life-cycle of design. The knowledge gained about the design must be accurate, accountable, and time-consistent. Further complicating matters, the amount of design information produced in a design is extremely large, and widely distributed. Successful design management requires three fundamental components:

- *The structure of information*, the means by which information is organized as a design progresses;
- *The measurement of information*, the ability to quantify the progression of a design; and
- *Information access*, large, distributed storage and retrieval schemes.

3.3.1 The Structure of Information. As information is collected, there is a tendency to structure the information to satisfy management and quantification concerns. The form-function-process-model / temporal paradigm was shown to characterize the design space: the *information hierarchy*. With the aid of this paradigm, the space spanned by the design can be quantified and, therefore, systematically explored.[16]

Equally important to the design representation is unstructured information, or loose information: the *information heterarchy*.[14] During the design process, some information will be structured from the heterarchy into the information hierarchy. Other information not found to be related to the design specification, but equally important to the design process, may remain unstructured. An example of unstructured information would include local program variables and process id's. The importance of heterarchical information is recognized and included in the DREAMS architecture.

3.3.2 The Measurement of Information. The ability to quantify information content allows the designer to follow a design's progression through its life-cycle. If information content can be measured, two design states can be compared and the later state has a higher degree of design knowledge.[17] Moreover, Suh's second axiom says that a "better" design has the least information content associated with it.[18] Though the measure of information content gives us insight into the temporal aspects of design, the measurement would be more beneficial as a gauge of design fidelity and knowledge evolution. In doing so, methods can be developed that can assist in facilitating IPPD from a Decision-Based Design perspective.

3.3.3 Information Access. A vast amount of information is accumulated during the design of complex systems. Techniques must be developed for distributed, simultaneous-access storage in a multi-user environment. The UNIX file system provides a good model for simultaneous, multi-user access. However, the files (information) locations are known directly or indirectly (links and mounts). This model must be expanded to include support for distributed database management. A virtual storage system must be developed that allows transparent access to data that will be found in files, databases (relational and object-oriented), memory, and storage media. The notion of "smart" files can be expanded and ideas borrowed from World-Wide-Web implementations to provide virtual storage capabilities.[19] The end result will allow designers suitable access to knowledge for decision-making.

4 IMAGE - A Computing Infrastructure

Having captured a designer's perspective in an architecture that supports Decision-Based Design, a computing infrastructure is being developed that provides a coherent implementation of the architecture. The computing infrastructure is designed in two parts:

- *Explicit entities are used to directly implement the DREAMS architecture; and*
- *An environment combines these entities with supporting computational tools.*

The resulting infrastructure is called IMAGE, an **I**ntelligent **M**ultidisciplinary **A**ircraft **G**eneration **E**nvironment. The infrastructure began as a special project that recognized the lack of designer support in traditional frameworks.[20] This fact resulted in frameworks that are difficult to implement and has lead to their limited use.

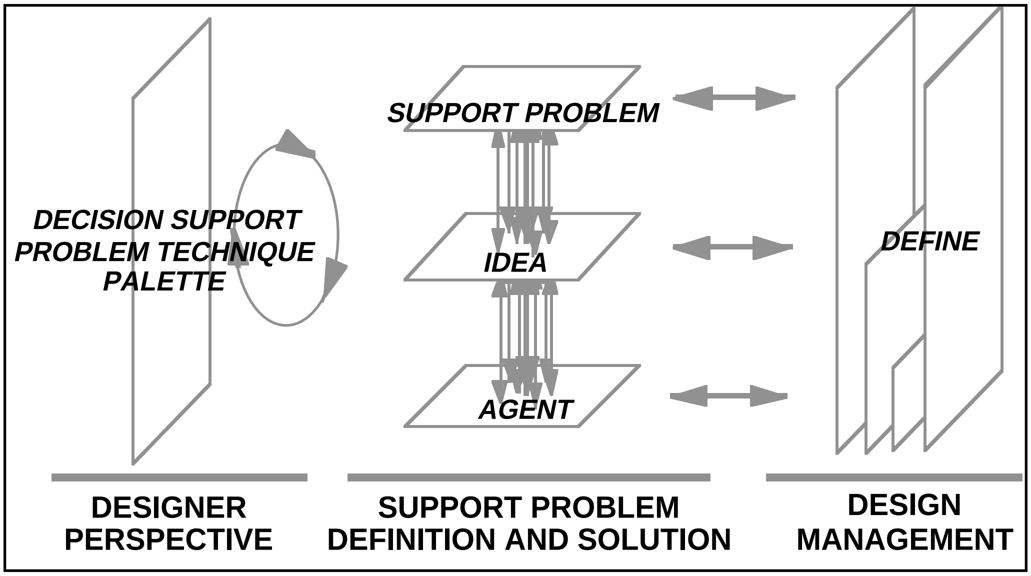

Figure 6. *IMAGE Entities Corresponding to DREAMS Architecture*

The entities used to implement the DREAMS architecture are shown in Figure 6. Notice that there is a specific computing entity corresponding to each DREAMS entity, see Figure 3. The entities are shown in Table 1.

Table 1: *DREAMS Architecture and IMAGE Infrastructure Correspondence*

DREAMS Architecture		IMAGE Infrastructure
Designer's Perspective	→	DSPT Palette
Formulation	→	Support Problem
Translation	→	Idea
Evaluation	→	Agent
Design Management	→	DEFINE

4.1 The Designer's Perspective → DSPT Palette

Bras and co-workers have developed a system that implements the DSP Technique on the computer and is called the `DSPT Palette`.[14] The basic `DSPT Palette` presents a number of Support Problem icons, see Figure 7, that are used in meta-design to construct the design decision-making processes, see Figures 1 and 2. Methods are in place within the `DSPT Palette` that aid the designer in structuring the problem statement from natural language statements in the meta-design phase.

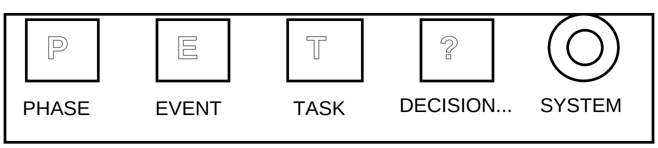

Figure 7. *`DSPT Palette` Support Problem Entities*

4.2 Support Problem Definition and Solution

The DREAMS architecture employs the use of Support Problems throughout the design process. Support Problems are used to model the transformation of information into knowledge throughout a design timeline. Computational support is provided for the Formulation, Translation, and Evaluation of Support Problems.

4.2.1 Formulation → Support Problem. `Support Problems` are formulated using the `DSPT Palette`. Various `Support Problem` icons are shown in Figure 7. As discussed previously, `Support Problems` are formulated linguistically, requiring the use of a natural language parser.

4.2.2 Translation → Idea. `Support Problems` are translated into an equivalent Math Form. The `Idea` is the computational entity used to instantiate the Math Form, see Figure 8. The `Idea` is used to implement parse tree instances: form, function, and process relationships. There are one or more `Ideas` for each `Support Problem`. `Ideas` are stored and used as an independent transmission entities in the computing infrastructure.

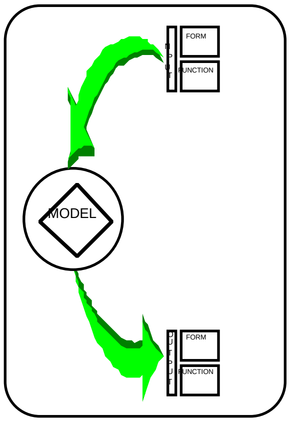

Figure 8. *`Idea` Icon*

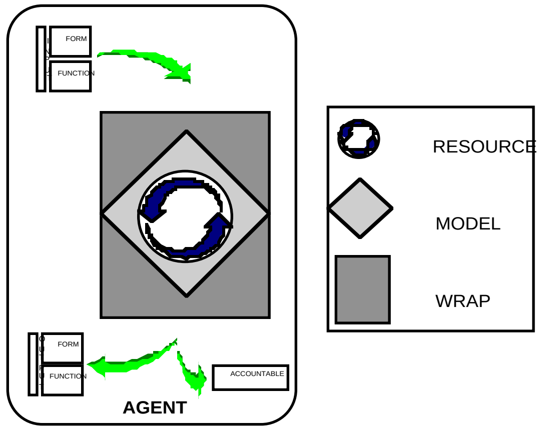

Figure 9. *An `Agent`*

4.2.3 Evaluation → Agent. An `Agent` is the computational implementation of the Design Operator, see Figure 9. `Agents` provide a mechanism for resource use in a distributed, heterogeneous computing environment. `Agents` transform information into knowledge based upon known models. An `Agent` has three components: the resource, model, and wrap. Based on these three components, standardized implementations can be formulated and used to encapsulate a variety of design resources found throughout the life-cycle of design. `Agents` will be coupled with the `Idea` into a `Design Event` and structured with other `Design Events` into solution networks, see Figure 5.

4.3 Design Management → DEFINE

A user interface called `DEFINE` has been developed in LEGEND (Laboratory Environment for the Generation, Evaluation, and Navigation of Design) for the creation and modification of design specifications.[16] The user interface is shown in Figure 10. The interface displays the form, function, process, and model hierarchies. Also, an enhanced QFD interface is presented as a means of creating a specification instance, the design protocol. A protocol element includes one or more forms, functions, and models. These instances result in `Agent` definitions. There are one or more agents for each instance.

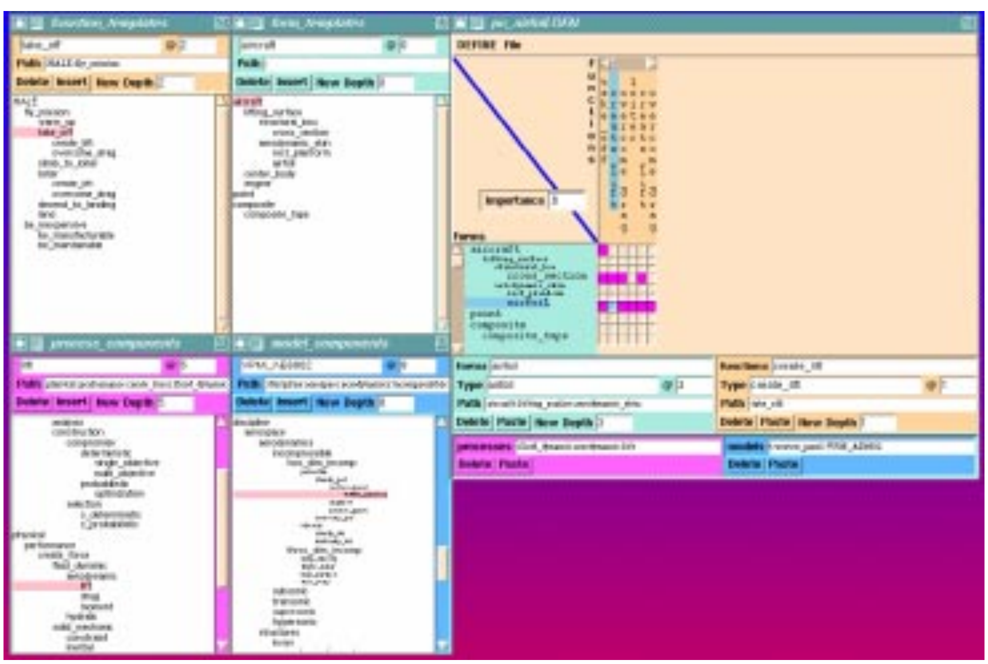

Figure 10. *`DEFINE` Interface for Creating Design Protocols*

5 IMAGE

Having identified the fundamental entities required to support design decision-making, see Table 1, the tools required to implement IMAGE are formalized. IMAGE is a loosely configured, agent/tool-based federation. The environment is constructed using the same agent technologies used to implement Design Operators assuring design support and maximum architecture flexibility. An agent environment will support multiple platforms, operating systems, and users. IMAGE combines the services of both agents and tools. The resulting IMAGE framework is shown in Figure 11. Tools can be used in place of agents when a design expert is present to utilize the resources and are required for operations that have no model (for example operating system level services).

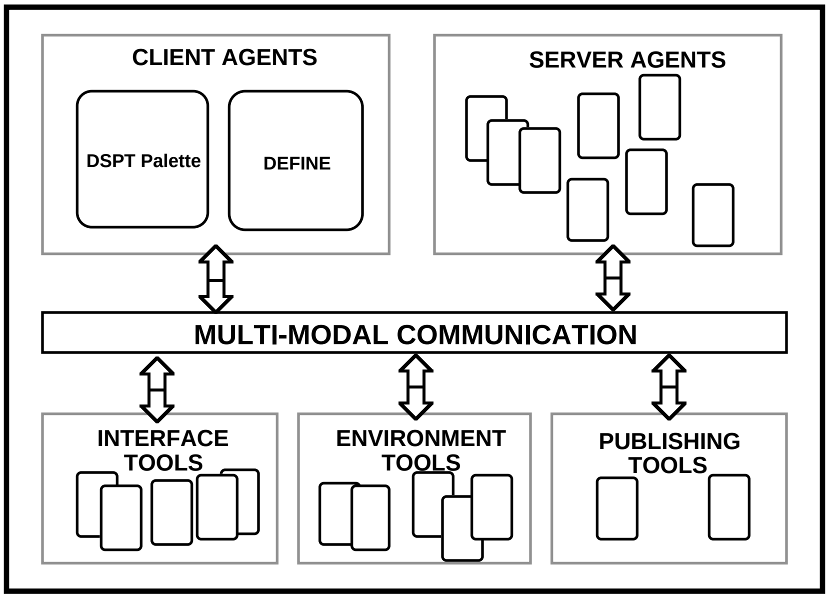

Figure 11. *IMAGE Infrastructure*

5.1 Agents

There are two types of agents that exist in the environment: meta-agents and agents. Meta-agents are those agents that provide user services for directing design. The meta-agents have already been introduced in the previous section, see Table 2. Basic agents are used to implement design operators. Table 3 summarizes a few of the agents that can be used for aircraft design. Specific agents are being developed as example problems become better defined.

Table 2. *Meta-Agents*

Agent	**Model**
DSPT Palette	Support Problems
DEFINE	Design Specification

Table 3. *Agents*

Agent	**Model**
Geometry	Solids geometry construction
Convergence	Weight convergence, fuel balance
Aerodynamics	Newton-Euler, Navier-Stokes
Structures	Energy methods
Propulsion	Thermodynamics
Weight and Balance	Mass properties, Newton's Second Law
Controls	Control laws
Costing	Historical CERs

5.2 Tools

There is a suite of tools that can be used to assist in the design process and general environment services. These tools have been categorized as interface (both human and inter-agent), monitoring, and publishing tools. These tools transparently coordinate inter-agent, agent-tool, and inter-tool efforts. The services of these tools are summarized in the following tables:

Table 4. *Human and Inter-Agent Interface Tools*

Tool	Service
Database	Relational and object-oriented data management.
Heterarchy Editor	Loose information management
File	Distributed file management
Application Defaults	User-defined (graphical) user interface defaults
Security	User and process security
Recovery	Environment recovery
Lookup	Agent/tool name server
Dictionary	Cross-agent ontology correlation
Accumulator	Quantify design knowledge
Annotator	Record design events
Review	Review design decisions and history
Units	Agent independent standard units and conversions
Expert System	Distributed, rule-based inferencing capabilities
Communications	Data routing through unsynchronized, unlike channels
Language Processor	Natural language processing services. May grow to include pictures, movies, and sound.

Table 5. *Environment Monitors*

Tool	Service
Software	Software versioning and synchronization
Process	Resource execution status
Data	Data storage/retrieval utilities
User	User and process management
Project	Distributed project capabilities

Table 6. *Knowledge Publishing*

Tool	Service
Printing	Local and remote spooling
WWW	Electronic documentation

The IMAGE infrastructure provides the computational support required to exercise the DREAMS architecture. Specific, design-related tools are being developed for the computing environment. The use of these tools guarantees a systematic mechanism for generating knowledge used for decision-making. Additional tools provide the resources required for collaboration in open, multi-user computing systems.

6 Status of Development

Significant, preliminary work has been done in the development of the DREAMS architecture and the IMAGE infrastructure. The work has been well received and continuous to grow coherently.

6.1 DREAMS Architecture

The DREAMS architecture is the foundation for the NASA MDO contract (NAG-1-1564). The development of an **I**ntegrated **D**esign **E**ngineering **S**imulator (IDES) that will merge Integrated Product and Process Development with interdisciplinary analysis techniques and state-of-the-art computational technologies is outlined by the contract proposal. As a result, the IDES can be used to develop and test virtual design prototypes. The first year's efforts in developing the DREAMS architecture brings together a number of research efforts in the areas of:

- Methodology
- Computer Integration Strategies
- Robustness and Compromise Decision Support Problems
- Heuristic Decision Support Problems
- Design Management
- Problem Decomposition and Solution

Under the contract, Rockwell's active flexible wing technology and Lockheed's lean aircraft manufacturing and advanced structures and materials developments will also be incorporated. Currently, application research efforts are focusing on the Multidisciplinary Wing Integration problem illustrated in Figure 4.

The MDO contract work can be found at the http site `http://www.cad.gatech.edu/mdo`.

6.2 IMAGE Infrastructure

Work is underway in the development of agent technologies.[21] Over the past two years, a suite of wrapping techniques was explored and developed that allows for agent design independent of resource implementation. Therefore, agents can be extended to include proprietary resources. In terms of technology, the most sophisticated tool developed under the research is a forms-based WWW interface that has real-time access to CATIA.[22] Through the interface, parametrically-defined geometry can be automatically generated on a remote server running CATIA and graphics returned real-time.

IMAGE research can be found at the http site `http://www.cad.gatech.edu/image`.

7 Further Development

The foundation is being placed for a computing infrastructure that facilitates design from a decision-based perspective. Several areas are being investigated as the new infrastructure matures. These areas are:

7.1 DREAMS Architecture

- *Implement other Integrated Product and Process Development (IPPD) techniques through DREAMS.* The architecture used in DREAMS can be used to implement Georgia Tech's IPPD Trade-off process and Rockwell's Affordable Systems Optimization Plan (ASOP).
- *Model the progression of Support Problems throughout a design's life-cycle.* Support Problems model the transformation of information into knowledge throughout a design timeline. Initially, little is known about a design and the exploration of a design region is desired: the satisficing model. As a design progresses, design regions become sufficiently refined and an single-point, optimization model can be used. The transition between the two models is still unclear. Presently, the notion that the transition can be identified by quantifying the design ambiguity along the design timeline is being investigated.
- *Integrate Support Problem definition and solution with the Design Specification.* The art of capturing the designer's perspective has been traditionally removed from a design's specification. As depicted in Figure 3, the design specification, which can be quantified, and Support Problems are intimately coupled in varying degrees as Support Problems are formulated, translated, and evaluated.

7.2 IMAGE Infrastructure

- *Harness computing technologies that utilize existing engineering practices.* Many of the new computing technologies offer significant advance in computing power but have seen limited application in engineering programs. The new technologies are not backward compatible with existing code and they require constructs not found in traditional programming languages.
- *Develop a generic agentization scheme.* A generic agentization scheme will enable designers to consistently integrate design resources into the IMAGE infrastructure. Three components have been identified for the agent:
    - Resource - Typically engineering analysis codes but also includes the designer and domain knowledge sources.
    - Model - An idealization of a process as well as the agent implementation characteristics.
    - Wrap - Facilitates bi-directional agent collaboration in an open, distributed, homogeneous computing framework.
- *Develop the agents and tools required to implement the IMAGE infrastructure.* Agents and tools required to implement the IMAGE computing infrastructure are outlined in Tables 2-6. The agents/tools will be developed as research progresses and demonstration problems mature.

When in place, the new architecture, supported by the computing infrastructure, will result in a system that facilitates designing from a decision-based perspective. Thus, a designer will have the capability to producing better designs while expending fewer resources. A designer will have more knowledge, that is complete and structured, available during decision-making processes.

8 References

[1] "ACSYNT Overview and Installation Manual," ACSYNT Institute, Virginia Polytechnic Institute and State University, May 1992.

[2] Cutkosky, M. R., R. S. Engelmore, R. E. Fikes, M. R. Genesereth, T. R. Gruber, W. S. Mark, J. M. Tenenbaum and J. C. Weber, "PACT: An Experiment in Integrating Concurrent Engineering Systems," IEEE Computer, pp. 28-37, January, 1993.

[3] Dovi, A. R., G. A. Wrenn, J.-F. M. Barthelemy, P. G. Coen and L. E. Hall, "Multidisciplinary Design Integration System for a Supersonic Transport Aircraft," Fourth AIAA / USAF / NASA /

OAI Symposium on Multidisciplinary Analysis and Optimization, Cleveland, OH, September 21-23, 1992. AIAA-92-4841.

[4] Hughes, D., "Generic Command Center Speeds Systems Design," Aviation Week & Space Technology, pp. 52-53, March 8, 1993.

[5] Jones, K. H., D. P. Randall and C. K. Cronin, "Information Management for a Large Multidisciplinary Project," Fourth AIAA / USAF / NASA / OAI Symposium on Multidisciplinary Analysis and Optimization, Cleveland, OH, September 21-23, 1992. AIAA-92-4720.

[6] Gage, P. and I. Kroo, "Development of the Quasi-Procedural Method for Use in Aircraft Configuration Optimization," Fourth AIAA / USAF / NASA / OAI Symposium on Multidisciplinary Analysis and Optimization, Cleveland, OH, September 21-23, 1992. AIAA-92-4693.

[7] Kroo, I. and M. Takai, "A Quasi-Procedural, Knowledge-Based System for Aircraft Design," AIAA / AHS / ASEE Aircraft Design, Systems and Operations Meeting, Atlanta, GA, September 7-9, 1988. AIAA-88-4428.

[8] McCullers, L. A., "FLight OPtimization System, User's Guide, Version 5.41," NASA Langley Research Center, December, 1993.

[9] Bras, B. A. and F. Mistree, "Desiging Design Processes in Decision-Based Concurrent Engineering," SAE Transactions Journal of Materials & Manufacturing, pp. 451-458, Warrendale, PA, SAE International, 1991.

[10] F. Mistree, O. F. Hughes and B. A. Bras, *The Compromise Decision Support Problem and the Adaptive Linear Programming Algorithm*, Ed. Kamat, M. P., Structural Optimization: Status and Promise, Washington, DC, (pp. 247-286), AIAA.

[11] Mistree, F., W. F. Smith, B. A. Bras, J. K. Allen and D. Muster, "Decision-Based Design: A Contemporary Paradigm for Ship Design," Transactions, Society of Naval Architects and Marine Engineers, Jersey City, New Jersey, pp. 565-597, 1990.

[12] F. Mistree, W. F. Smith and B. A. Bras, *A Decision-Based Approach to Concurrent Engineering*, Ed. Paresai, H. R. and W. Sullivan, Handbook of Concurrent Engineering, Chapman & Hall, New York, 1993. (pp. 127-158).

[13] "New Approaches to HSCT Multidisciplinary Design and Optimization," Georgia Institute of Technology, Lockheed Aeronautical Systems Company, Rockwell-North American Aircraft, September,1994.

[14] B. A. Bras, W. F. Smith and F. Mistree, *The Development of a Design Guidance System for the Early Stages of Design*, Ed. Oortmerssen, G. V., CFD and CAD in Ship Design, Elsevier Science Publishers B.V., Wageningen, The Netherlands, (pp. 221-231).

[15] Röhl, P., D. P. Schrage and D. N. Mavris, "A Multilevel Wing Design Procedure Centered on the ASTROS Structural Optimization System," AIAA / NASA / USAF / ISSMO Symposium on Multidisciplinary Analysis and Optimization, Panama City, Florida, September 7-9, 1994. AIAA-94-4411.

[16] Stephens, E., "LEGEND: Laboratory Environment for the Generation, Evaluation, and Navigation of Design," Doctoral Dissertation, Georgia Institute of Technology, School of Aerospace Engineering, September 1993.

[17] Vadde, S., J. K. Allen, T. Lucas and F. Mistree, "On Modeling Design Evolution Along a Design Time-Line," AIAA / NASA / USAF / ISSMO Symposium on Multidisciplinary Analysis and Optimization, Panama City, Florida, 1474-1482, September 7-9, 1994. AIAA-94-4313-CP.

[18] Suh, N. P., Principles of Design, Oxford University Press, Oxford, U.K., 1990.

[19] Rosendale, J. V., "Smart Files: An OO Approach to File System Interoperability," Institute for Computer Applications in Science and Engineering, July, 1994.

[20] Hale, M. A., "IMAGE: An Intelligent Multidisciplinary Aircraft Generation Environment," Masters Program Special Project, Georgia Institute of Technology, School of Aerospace Engineering, September, 1992.

[21] Hale, M. A. and J. I. Craig, "Preliminary Development of Agent Technologies for a Design Integration Framework," AIAA / NASA / USAF / ISSMO Symposium on Multidisciplinary Analysis and Optimization, Panama City, Florida, September 7-9, 1994. AIAA-94-4297.

[22] Hale, M. A., "Demonstrating Preliminary Agent Technologies With CATIA," CATIA Operators Exchange, Dallas, TX, October 9-13, 1994.