A Model for One-Off Systems Engineering

Mark S. Fox and Fil Salustri

Department of Industrial Engineering, University of Toronto
4 Taddle Creek Road, Toronto, Ontario CANADA M5S 1A4
tel: 1-416-978-6823; fax: 1-416-978-3453; internet: msf@ie.utoronto.ca

Appeared in: ***Proceedings of the AI and Systems Engineering Workshop*****, AAAI-94, Seattle WA.**

1.0 Introduction

In order to apply Artificial Intelligence to Systems Engineering, it is necessary to first have a model of what Systems Engineering is. This is not as simple as one might think. A review of the engineering literature (see section 5) shows that there exist few models of systems engineering, or what has come to be called, Concurrent Engineering.

This paper describes the "as-is" and "to-be" approaches to systems engineering at Spar Aerospace. Spar Aerospace is a Canadian aerospace company with 2,800 employees and revenues of about CA$400m in 1991.The Advanced Technology Systems Group (ATSG), with whom this study was conducted, is headquarted in Toronto Canada. It has over 600 employees located primarily in Toronto, with others in Carpinteria California. The product focus of ATSG includes:

- Space Manipulators - US space shuttle remote manipulator,
- Space Mechanisms,
- Solar Arrays - e.g., Olympus solar array,
- Nuclear Robotic Systems - e.g., CANDU Reactor servicing manipulator Systems, and
- Infrared Systems - e.g., AN/SAR-8 Program naval IR search and track deignation system.

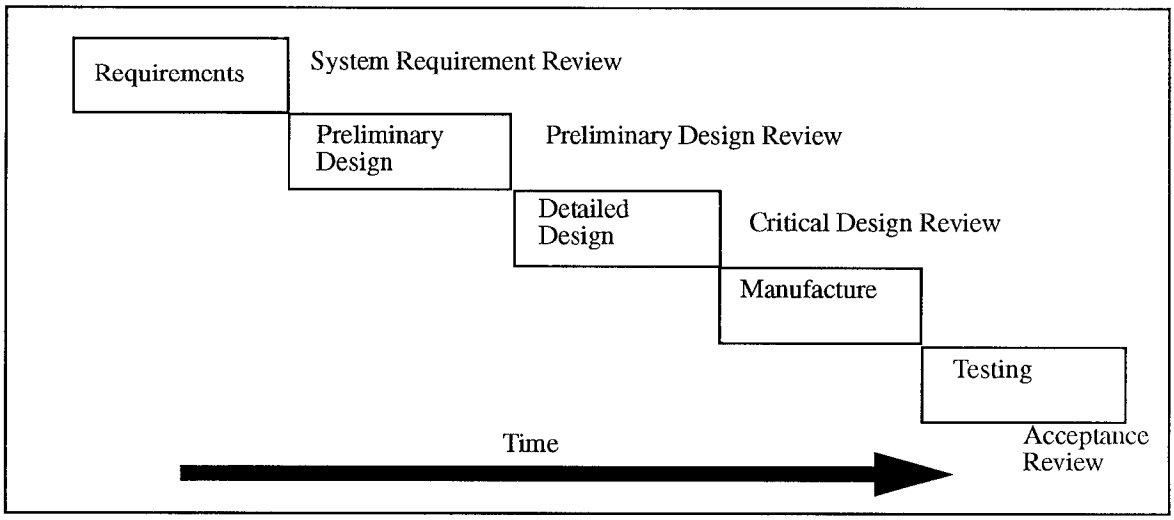
Spar designs, engineers, and manufactures complex, one-off systems. A *system* is a set of closely inter-related components which does something having real world relevance, and has definable external interfaces and interactions. Systems are organised hierarchically, that is, a specific system may be composed of smaller systems, and may itself be part of a larger system. The systems are *one-off* in the sense that they usually design and build only one or two of a particular artifact.

In the following sections, we review the Spar's current approach to systems engineering, the problems that arise from the approach, and Spar's new approach.

2.0 Current Approach to Systems Engineering

Today, much of ATSG practices an approach to systems engineering widely known as the "waterfall model." The waterfall model defines a sequence of activities, each distinctly different and each must be completed before the next one begins.

FIGURE 1.



Requirements Phase: Translates customer wishes into a an agreed upon requirements document. The document states, in technical terms, what the customer wants done, and should be free of designer introduced implementation details.

Preliminary Design Phase: Creates a design that will implement the requirements. The design is outlined in this phase, focusing on major components and their functions. Several options are considered, analysed, and the best one selected. The preliminary design results in a commitment to a particular design.

Detailed Design Phase: The design of major components is completed to the point where it can be implemented. It is verified against the preliminary design, and validated against the requirements.

Implementation Phase: Components are fabricated and the final product is assembled.

Testing Phase: The product is exercised thoroughly enough to convince both the company and the customer that it fulfils and will continue to fulfil the contractual requirements of the customer.

The sequentiallity of the model allows for the definition of clear milestones, i.e., at the end of each stage.

For decades, the waterfall model has been applied successfuly to the construction of hardware and software systems. But within the last tens years, limitations of the approach have been recognized; time-to-market pressures plus the increasing complexity of systems have uncovered a number of problems. At Spar, the application of the Waterfall model has been problematic for the following reasons:

- The customer, in this case NASA, is not able to provide a complete set of requirements at the outset of the contract. For example, in developing the Canadarm for the Space Shuttle, many of the operating characteristics of the arm had to be discovered during the project. The same is true for the Remote Maintenance System for the Space Station. Lack of complete requirements has resulted in a "backfilling" behaviour where the act of design generates information that defines or refines the requirements.
- The performance of testing as the last stage of the model, postpones until the end a significant aspect of the systems engineering. Given a hierarchical decomposition of the system, each level

of the hierarchy posses both design and integration problems. Given the complexity of the artifacts that Spar designs, one cannot assume that separately designed components will integrate successfuly. Therefore, at each level of the hierarchy, effort needs to be expended in integrating and testing components well before the engineering is complete. That is, testing has to be interleaved with engineering and manufacturing.

- The hiearchical decomposition of the system, ignores the importance of cross component systems, such as electrical, software, etc. These systems must be optimised across the artifact, but the decomposition model, coupled with the waterfall model does not enable it.
- Studies have shown that 80% of the cost of an artifact is committed in the design stage. Secondly, if changes are made after this point, the cost of redoing it grows exponentialy with each subsequent stage. Since the artifacts Spar engineers, are often one of a kind items, there does not exist sufficient prior experience to guarantee that a design will optimise fabrication, assembly, testing , distribution, etc., i.e., the product life cycle. Therefore it is necessary that knowledge of the impact of each design decision on the product life cycle be made known to design engineers.

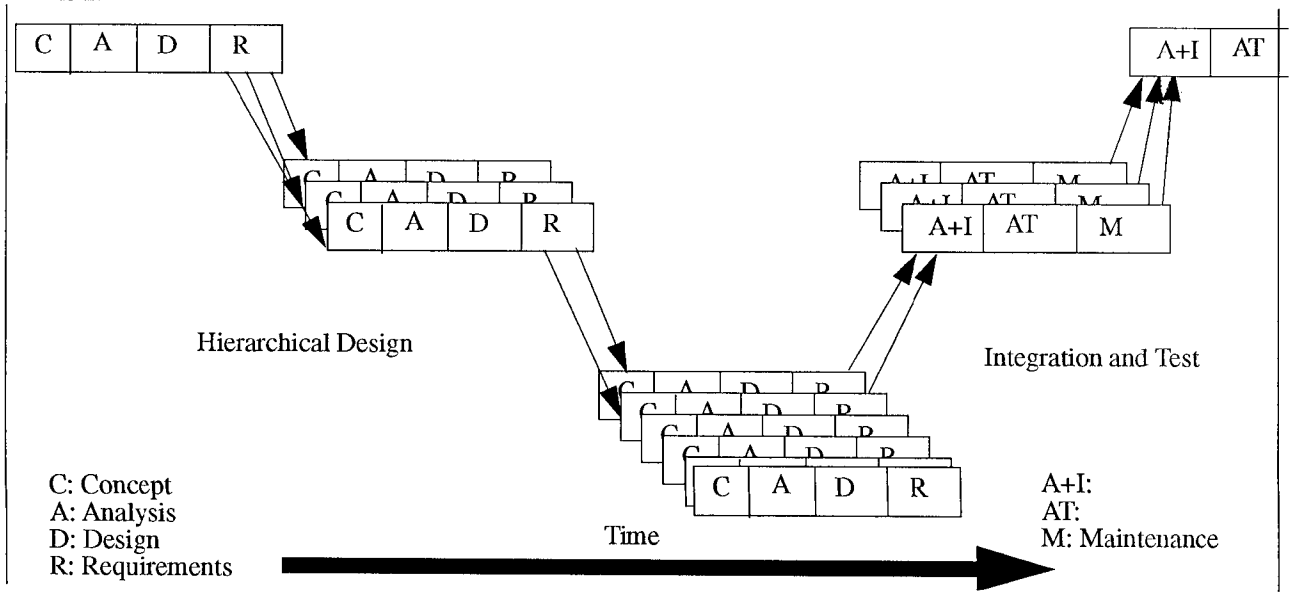
3.0 A Model for Systems Engineering

To address the above problems, Spar Aerospace is melding a number of engineering process concepts. In the following we will introduce the Spar model one concept at a time.

3.1V Model of Systems Design

The first concept is the replacement of the Waterfall Model with the *V model* of systems engineering. The V model highlights both the decomposition and integration aspects of system design. The left side of the V depicts the top-down, hierarchical approach to system design. Each box in the left arm of the V denotes a separate design/engineering task, whose requirements are determined at the next higher level. Each box may use the waterfall model internally. The right arm of the V depicts the bottom-up integration & testing of the subassemblies and final assembly.

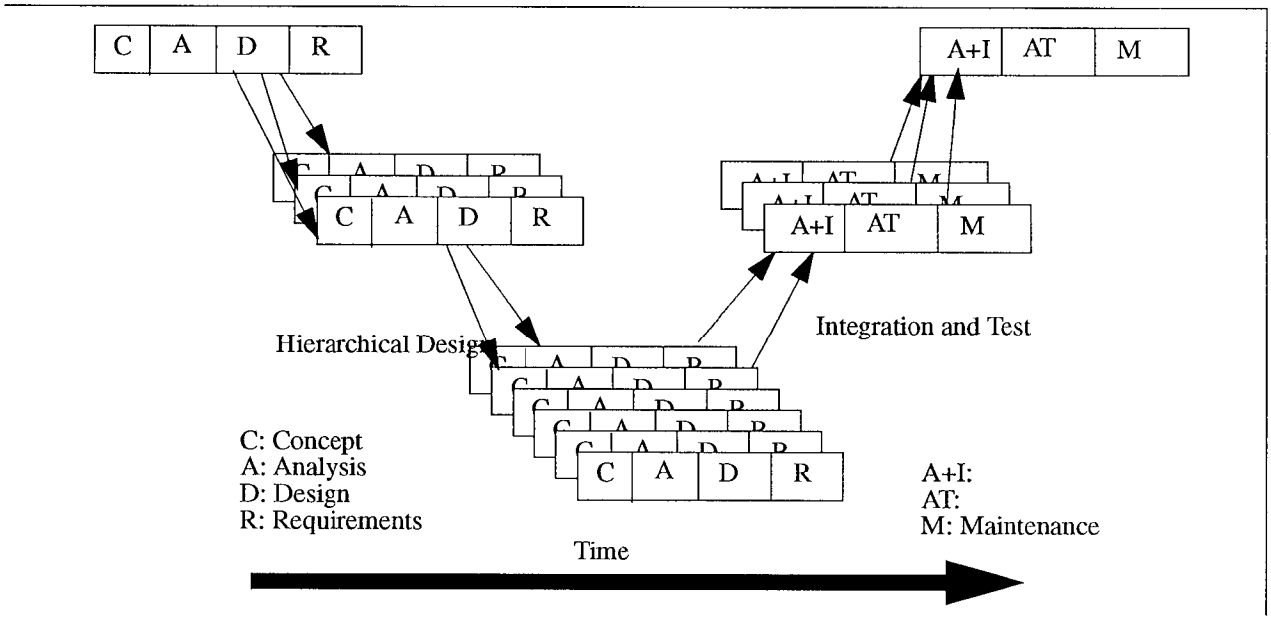
FIGURE 2.



3.2 Activity Overlapping

The second concept is the reduction of design/engineering time through the overlapping of activities. The primary idea is that what were previously viewed as sequential activities, can now be performed in parrallel using incomplete information. That is, the requirements at one level do not have to be complete for the next level to begin its work. Activities can now be started earlier, using incomplete design information, so that the overal time scale of the project is significantly reduced.

FIGURE 3.

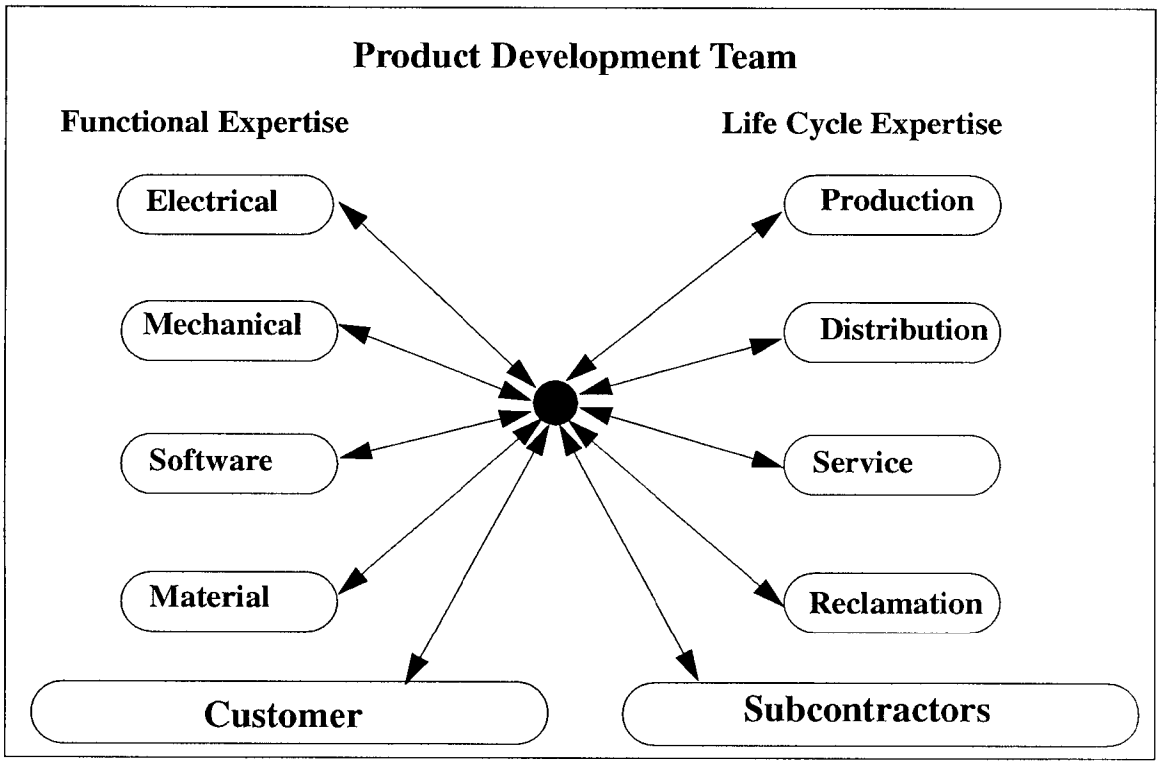


3.3 Life Cycle Design

The third concept is that each design task should consider all elements of the product life cycle from concept through disposal, including quality, cost schedule and user requirements. To accomplish this, each design task is performed by what is called a product development team (or "tiger team"). A product development team is both cross life cycle and cross functional in its membership, thereby guaranteeing that a component's design is optimised across the product life cycle and across functions.

An important extension to the tiger team concept is the inclusion of customer and subcontractor representatives. Since many of the customer requriements are incomplete, it is necessary to include them in the design/engineering process so that they may participate in decisions that affect the product. Secondly, since many components of the artifact are produced by subcontractors, their involvement is necessary in order to assure that the components are producable.

FIGURE 4.



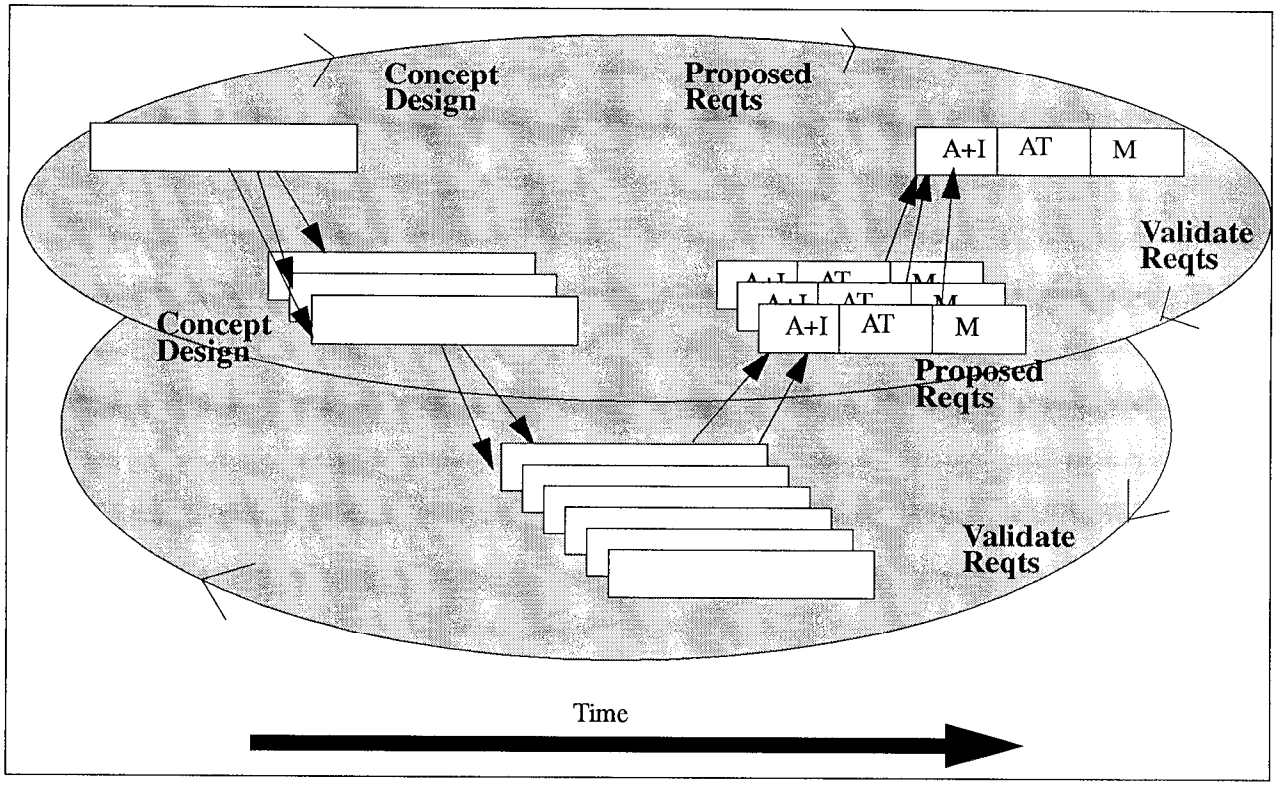
Factors leading to successful teams include:

- Full team membership and commitment at each stage of the process.
- Permanent, core, col-located team takes product from requirements through the life cycle.
- Other team members kept fully informed through regular meetings with core.
- Team member are cross trained and multi-skilled.
- Each team has life cycle cost allocation which they track.

3.4 Iterative Design

The fourth major concept is design as an *iterative* process. When we introduced the V model, we stated that the the stages within a level conformed to the waterfall model. The waterfall model assumes that requirements are completely known prior to design. Such is not the case in the "one-of" domain. Because of the uniqueness of the artifact, much additional knowledge is learned at all levels of the design/engineering V. Decisions made at higher levels may be revised in light of new knowledge discovered or created by lower levels in the V. Consequently, there is a need for timely feedback from lower levels to the upper levels of the V. This feedback results in an iterative design process where knowledge generated at lower levels of the V may affect decisions made at higher levels, which in turn may affect decisions at lower levels. This process of iterating through a few stages has been called the "spiral model" and uses "rapid prototyping" for early testing of design concepts.

FIGURE 5.



3.5 Management of Decisions

The fifth major concept is the *management of decisions.* Design decisions need to be made continuously in a project. In many cases, they are either postponed because complete information does not exist, they are forgotten - "drop through the crack" -, or people are unwilling to risk making a decision. Secondly, the implication of a decision on life cycle costs is seldom considered. The concept of managing decisions requires that:

- decisions be made continuously and in a timely manner,
- the life cycle cost of a decision be determined, and
- that risk of alternative decisions be assessed continuously.

One tool for managing decisions is the formal review. The waterfall model, as depicted in figure 2, provides a well defined set of review points at the end of each stage. With the advent of the spiral model, the review points are less clear, nor is the information complete. Given the need to make decisions in a timely manner and assess risk, review meetings are no longer points where decisions are made, but instead become risk assessments. Of course, major stop/go decisions do occur at these meetings. The following defines a set of reviews and the point at which they occur:

System Requirements Review: We assume that the requirements are incomplete, and have yet to be proven feasible or compatabile with system interfaces.The SRR is performed after the first pass once a decomposition of the initial design concept is explored using rapidly produced prototypes. Two or three levels of decomposition may be explored before this review occurs.

System Design Review: The design is reviewed once additional layers of the decomposition have been explored, usually down to the hardware/software level. Sufficient feedback has been given to the upper levels for the design to converge.

Critical Design Review: At this point, no system has been constructed, except for engineering models. Only those portions of the system considered to be of high risk are constructed and tested. All levels of the decomposition side of the V have been explored, along with the corresponding integration and test activities.

Acceptance Review: Once the first system is made, called a qualification model, testing is performed, sometimes destructive, to see if it conforms to the requirements.

3.6 Overhead Reduction

The sixth major concept is *overhead reduction*. The focus is on removing things that increase the overhead of the project. For example:

Documentation: Every project document entails its own life cycle cost. Projects at Spar are swimming in documentation that few read and understand. If a document does not add real value to the project, then it should not be produced.

Process Complexity: There is a trend to overly complicate design through the imposition of large numbers of requirements. Each requirement entails a life cycle process and may overly constrain and complicate the design. Therefore, only requirements that are absolutely necessary should be considered.

Precision: Like complexity, the requirement for excessive precision overly complicates design and engineering. Its requirement should be used sparingly.

Information Reduction: Information created at one level of the project and suitable for that level only is hidden from lower levels.

4.0 Existing Models

Concurrent engineering (CE) is the subject of vigorous research, very often taking advantage of systems engineering concepts as foundational elements. Indeed, Chapman et al. [Chapman et al. 92] write "Systems engineering ... *is* concurrent engineering."

The current literature makes evident that little is known about CE. Vos [Vos 93] writes that our understanding of CE is still "immature" and in a state of "evolution". Clausing, in [Clausing 93b], emphasizes that the CE capabilities that have been established so far, though they are a good start, are not enough; he proposes that a true CE system will include "basic" CE technologies (founded on multi-disciplinary and multi-functional teams) integrated seamlessly with Enhanced Quality Function Deployment and with Taguchi Methods for robust design. Furthermore, Stephens recently wrote, "To develop this understanding [of reliable design processes and products], design must be studied so that in the future it can become a reliable science." [Stephens 93]

A number of common problems are acknowledged by all these researchers, including: lack of understanding of the basic relationships between heretofore sequential phases of engineering; a general

failure of cooperation between team members; lack of integrated handling of quality issues; and so on. We summarize all these important points as follows: current research has focused on the individual tools needed to support CE; what is missing is the *systems perspective* needed to integrate these efforts seamlessly and effectively.

Whenever a tool is developed, there *must* be some underlying conceptual model of the application domain, even though it may be only implicitly and vaguely. The tools can be roughly categorized by the aspect of CE that their underlying models seek to address. We have found four such categories: *structural representation*, including knowledge-based representations [Kott et al. 92, Rosen & Peters 92], shared (or global) data models [Sriram et al. 91], and constraint-management [Serrano 91]; process models of collaboration and negotiation [Sycara & Lewis 91, Polat et al. 93, Khedro et al. 93]; the search for formal foundations for CE [Billatos & Grigely 93, Favela et al. 93] emphasizing Suh's Axiomatic Theory of Design [Suh 90]; and empirical, experimental research into the nature of interaction, collaboration, and negotiation between engineers [Tang & Leifer 91, Bradley & Agogino 91].

Recently, however, some in-roads have been made by various researchers towards constructing actual models of CE. Due to the relative infancy of these efforts, it is not surprising that the models are still quite general. Stephens [Stephens 93] introduces a new research project to study CE. His basic model is a state transition system including a component to manage compromise.

Kannapan et al. have reported their model of product development teams [Kannapan et al. 93]. In their model, a distinction is made between team members and "agents", the latter being components of a computerized environment giving access to various product models in a systematic manner. The number of team members, "aspects" (of product models), and "perspectives" (provided by agents) is allowed to vary over the course of a development project. Coordination scheduling is performed by an agent in accordance with requirements of selected negotiation protocols.

Krishnan et al. [Krishnan et al. 93] have produced a constraint-based mathematical model of the interaction between two simultaneous activities, which they refer to as "overlapping iteration". It estimates how, when, and for how long each iteration of the "downstream" activity should be carried out with respect to the "upstream" activity. The mathematical flavor of this model makes it quite compact and easy to manipulate. However, three important aspects are not covered: the time/resources required to incorporate data from one function into another at each iteration; the requirement for a shared representation of information in order to make transfer between activities effective and efficient; and the role of conflict and negotiation as information is passed between activities. Nonetheless, their model is quite interesting for its potential for future development.

It is evident from this review that the problems faced by CE researchers have not all been solved. Individual tools are quickly reaching maturity; what remains is the integration of these efforts into seamless and effective systems. The efforts of the current authors, presented in this paper, provide more of the much needed data that will drive future research and advancement.

5.0 Acknowledgements

The material in this document was acquired through interviews with Pat Cross and Mike Parfitt. Portions were extracted from Spar's "System Engineering Manual, Version A.0" and the presentation "Concurrent Engineering at Spar Aerospace" by Mike Parfitt. This research was supported in part by Industry Canada, Natutral Science and Research Council of Canada, Carnegie Group Inc., Digital Equipment Corp., Micro Electronics and Computer Research Corp., Quintus Corp., and Spar Aerospace.

6.0 References

[Billatos & Grigely 93] Samir B. Billatos and Lawrence J. Grigely. Functional Requirement Mapping as a Framework for Concurrent Engineering . *Concurrent Engineering: Research and Applications* 1:171-178, 1993.

[Bradley & Agogino 91] Stephen R. Bradley and Alice M. Agogino. Design Capture and Information Management for Concurrent Design . *International Journal of Systems Automation: Research and Applications* 1(2):117-141, 1991.

[Chapman et al. 92] William Al. Chapman, A. Terry Bahill and A. Wayne Wymore. *Engineering Modeling and Design*. CRC Press, Boca Raton, 1992.

[Clausing 93] Don P. Clausing.World-Class Concurrent Engineering . In Edward J. Haug (editor), *NATO ASI Series F: Computer and Systems Sciences. Volume 108: Concurrent Engineering: Tools and Technologies for Mechanical System Design*, pages 3-40. Springer-Verlag, Berlin, 1993.

[Favela et al. 93] J. Favela and A. Wong and A Chakravarthy. Supporting Collaborative Engineering Design. *Engineering with Computers* 9(3):125-132, 1993.

[Kannapan et al. 93] Srikanth M. Kannapan and David G. Bell and Dean L. Taylor. Structuring Information and Coordinating Teams in Product Development . In T. K. Hight and L. A. Stauffer (editor), *Proceedings of Design Theory and Methodology -- DTM 93*, pages 233-242. ASME, New York, 1993.

[Khedro et al. 93] Taha Khedro and Micheal R. Genesereth and Paul M. Teicholz. Agent-Based Framework for Integrated Facility Engineering . *Engineering with Computers* 9(2):94-107, 1993.

[Kott et al. 92] Alexander Kott and Charles Kollar and Alf Cederquist. Role of Product Modeling in Concurrent Engineering Environment . *International Journal of Systems Automation: Research and Applications* 2(1):1-16, 1992.

[Krishnan et al. 93] Viswanathan Krishnan and Steven D. Eppinger and Daniel E. Whitney. Iterative Overlapping: Accelerating Product Development by Preliminary Information Exchange . In T. K. Hight and L. A. Stauffer (editor), *Proceedings of Design Theory and Methodology -- DTM 93*, pages 223-231. ASME, New York, 1993.

[Polat et al. 93] Faruk Polat and Shashi Shekhar and H. Altay Guvenir. A Negotiation Platform for Cooperating Multi-agent Systems. *Concurrent Engineering: Research and Applications* 1:179-187, 1993.

[Rosen & Peters 92] Davis W. Rosen and Thomas J. Peters. Topological Properties That Model Feature-Based Representation Conversions Within Concurrent Engineering. *Research in Engineering Design* 4(3):147-158, 1992.

[Serrano 91] David Serrano. Constraint-Based Concurrent Design. *International Journal of Systems Automation: Research and Applications* 1(3):287-304, 1991.

[Sriram et al. 91] D. Sriram and R. Logcher and A. Wong and S. Ahmed. Computer-Aided Cooperative Product Development: A Case Study. *International Journal of Systems Automation: Research and Applications* 1(1):89-112, 1991.

[Stephens 93] Eric R. Stephens. Presentation and Evaluation of New Design Formulations for Concurrent Engineering. In Edward J. Haug (editor), *NATO ASI Series F: Computer and Systems Sciences. Volume 108: Concurrent Engineering: Tools and Technologies for Mechanical System Design*, pages 129-137. Springer-Verlag, Berlin, 1993.

[Suh 90] Nam P. Suh. *The Principles of Design*. Oxford University Press, New York, 1990.

[Sycara & Lewis 91] Katia P. Sycara and C. Micheal Lewis. Modeling Group Decision Making and Negotiation in Concurrent Product Design . *International Journal of Systems Automation: Research and Applications* 1(3):217-238, 1991.

[Tang & Leifer 91] John C. Tang and Larry J. Leifer. An Observational Methodology for Studying Group Design Activity . *Research in Engineering Design* 2(4):209-219, 1991.

[Vos 93] Robert G. Vos. The Emerging Basis for Multidisciplinary Concurrent Engineering . In Edward J. Haug (editor), *NATO ASI Series F: Computer and Systems Sciences. Volume 108: Concurrent Engineering: Tools and Technologies for Mechanical System Design*, pages 111-127. Springer-Verlag, Berlin, 1993.