01 Jan 2005

# A Generic, Adaptive Systems Engineering Information Model I

Ann K. Miller
*Missouri University of Science and Technology*

Joseph J. Simpson

Scott Erwin Grasman
*Missouri University of Science and Technology*

Cihan H. Dagli
*Missouri University of Science and Technology*, dagli@mst.edu

# A Generic, Adaptive Systems Engineering Information Model

**Joseph J. Simpson**
**Boeing Company**
**6400 32nd NW # 9**
**Seattle WA, 98107**
jjs-sbw@eskimo.com

**Dr. Scott Grasman**
**UM – Rolla**
**219 Engineering Management and Systems Engineering Department,**
**Rolla MO 65409**
grasmans@umr.edu

**Dr. Ann Miller**
**UM – Rolla**
**Department of Electrical and Computer Engineering,**
**Rolla MO 65409**
annmiller@ieee.org

**Dr. Cihan Dagli**
**UM – Rolla**
**229 Engineering Management and Systems Engineering Department,**
**Rolla MO 65409**
**dagli@umr.edu**

## Abstract

Systems engineering tasks generate large volumes of data and information that must be available over the lifecycle of the system. This paper outlines an information model designed to support existing systems engineering methods and practices as well newly developed techniques. Specific methods and models used for the capture, encoding and persistence of systems engineering information and design artefacts were given special attention during the evaluation, analysis and model design phases. A generic systems engineering meta-model was then developed and used as a basis for the systems engineering information model that was developed and is presented in this work

## Introduction

Systems Engineering is a structured technical design and management discipline. This professional discipline is used to control the design, development, production and operation of large-scale complex systems produced by large distributed teams and organizations. The systems engineering discipline has matured from an operational birth in the governmental procurement process during World War II based on systems acquisitions, operational analysis and value engineering, to a multifaceted commercial discipline that can be used to address any type of complex system development.(Hughes 1998)

The fundamental tenets of an ordered, structured, scientific problem solving process prove the foundation of systems engineering principles and practices. An organization's ability to encode, communicate and organize information has continued to increase from the 1940s up through today. The analytical, computational and organizational tools that are used to manage systems engineering information have also grown more and more powerful over this ensuing time period.

Prior to the mid 1980s, most systems engineering techniques were based on the production of written (textual) problem and solution statements as well as analytical (graphical and computational) techniques used to explore the problem at hand. The great proliferation of computing power and information exchange capability in the last 15 to 20 years has facilitated the translation of the textual and graphical techniques to computer-based tool sets. However, many of these systems engineering artefacts are still based on static, non-executable, document centric information types. Executable models, real-time communication and collaboration systems enable the work of distributed teams and produce a large volume of video, graphic

and computer executable information types. This paper outlines an information model designed to effectively manage the integration of static and dynamic systems engineering information types.

## Systems Engineering Process Models

The Electronic Industries Association (EIA) 632 and The Institute of Electrical and Electronic Engineers (IEEE) 1220 standards detail two different models of standard systems engineering processes. In addition to EIA 632 and IEEE 1220, EIA published a companion standard, EIA 731 that outlines the activities an organization must perform to evaluate its capability to accomplish effective systems engineering tasks and process steps. (Lake 1998)

Starting in the 1980s, significant efforts have been made to transfer system engineering processes and practices to computer-based systems. The International Council On Systems Engineering (INCOSE) was established in the early 1990s to provide a forum for the further development and publication of systems engineering practices and processes. The INCOSE Tools Database Working Group (TDWG) has mapped the typical systems engineering tools to the areas of the standard systems engineering process where they are typically used.(INCOSE 2003) Figure 1 shows a typical three-tier computing system and a typical distribution of operations computing components across the three tiers.

The generic, adaptive systems engineering model presented in this paper is designed to be deployed at the database tier. When this model is developed and deployed on standards compliant, open source, freely available software packages and systems, the organization obtains a robust, adaptable, long term information management asset. These types of information management assets will reduce the probability that the systems and information formats will become obsolete and unsupportable in the long term.

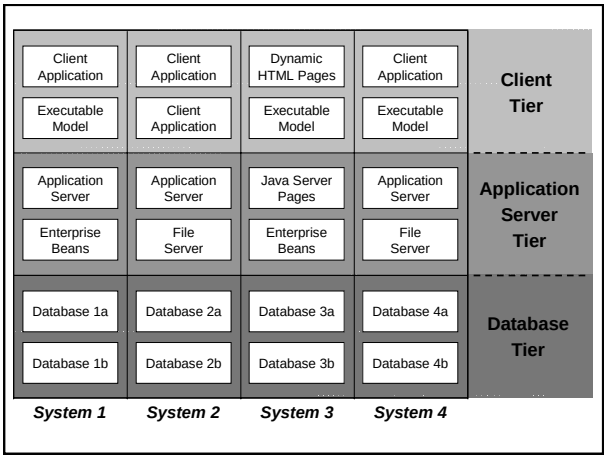


**Figure 1. Three-Tier Computing Systems.**

A core set of high level systems engineering practices and processes were developed by Oliver. (Oliver 1997) This "Model Based Systems Engineering" approach is abstracted into a six layer model. These six layers are: the systems engineering process layer, the information representation layer, the tool layer, the changes layer, the staffing layer, and the external visibility and review layer. The systems engineering process layer contains a core set of seven process steps that map directly to the general scientific problem solving process. This core systems engineering process flow model is shown in Figure 2.

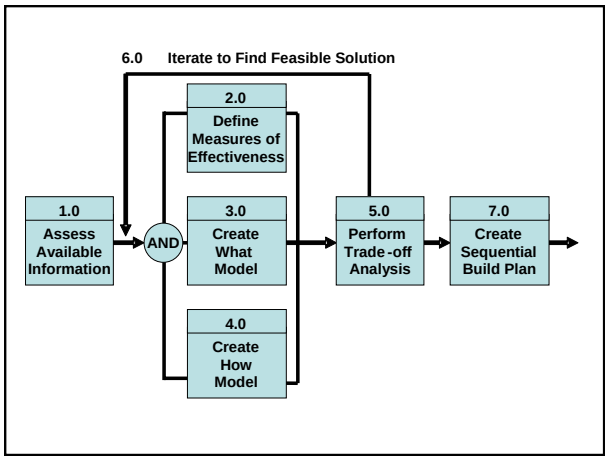


**Figure 2. Core Process Flow Model.**

A series of seven system views were used by Oliver as the basis to produce a set of information models that represent the systems engineering process. These information models are: system behaviour, system input and output, system structure and behaviour, system requirements, effectiveness measure creation, build and test plan and sell as text requirements, behaviour and content.

A review of the systems engineering literature indicates that representing a system in a series of views is a widely used and accepted technique. Mar developed the four view definition of any system. (Mar 1992) The function, requirement, architecture and text (FRAT) approach states that all systems are represented by four views, the function view, the requirement view, the architecture view and the test view. Maier discusses the importance of systems views in the systems architecting process. (Maier 1998) Maier suggests that the conceptual mismatch between hardware systems and software systems can be addressed using five system views. These five views are the logical view, process view, physical view, development view and scenario view. These views are used to organize the systems requirements and facilitate a structured bottom-up development process. Other systems engineering authors have reported using systems views as an organizing principle but these system views are all quite different in content and organizing principles.

## Systems Engineering Requirements Models

Systems engineering has two basic foundations: a systems engineering process and requirements management activities. Just as there are many different "standard" systems engineering processes, there are many different interpretations for specific requirements management activities. Grady discussed the connection between poorly written and understood systems engineering standards and the confusion associated with the sequence of requirements analysis and functional analysis. (Grady 1995) The key area of confusion, according to Grady, is the relationship between the current level of system definition activity and the higher-level system need or function that produced the lower-level system requirements activity. This fundamental semantic relationship between the specified need and the specified solution, as it appears in the program lifecycle, has been documented by the following authors: Forsberg and Mooz (1991), Mar and Morais (2002), and Simpson and Simpson (2001).

Most requirements management practices are based on managing a collection of text documents and sets of linkages between and among the documents in this collection. The arrangement, content, and relationships between the documents in the collection are program and project specific, with many of the documents carrying a contractual connection. In individual systems engineering projects, the "natural systems development" order and information flow varies across the aspects of system design, development and production. These variations create activity and requirement gaps that must be evaluated and addressed by the systems engineering staff. Systems engineering requirements models have been used to organize the various aspects of the systems engineering phases, tasks and activities. It is clear that both textual and executable types of data and models are required to effectively address the numerous types and forms of requirements information encountered in a system development program.

Requirements traceability has been mandated by government process standards and is specified for use by many government system acquisition programs. Even though the practice of requirements traceability was

mandated, no common model or specific set of practices were specified to guide the implementation of traceability practices. The Naval Postgraduate School conducted a series of studies on traceability practices. A primary outcome of these studies was a set of conceptual information models that address the basic areas found in systems requirements development. (Ramesh 1993) Four conceptual models were developed during this work. These four models are: requirements management model (Figure 3), design allocation model (Figure 4), design/implementation decision making model (Figure 5) and compliance verification model (Figure 6).

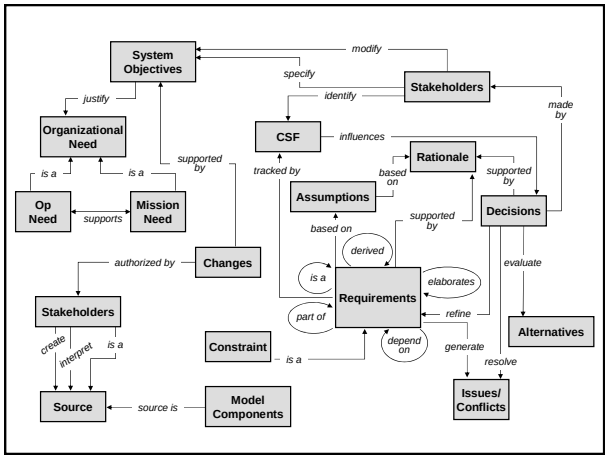


**Figure 3. Requirements Management Model.**

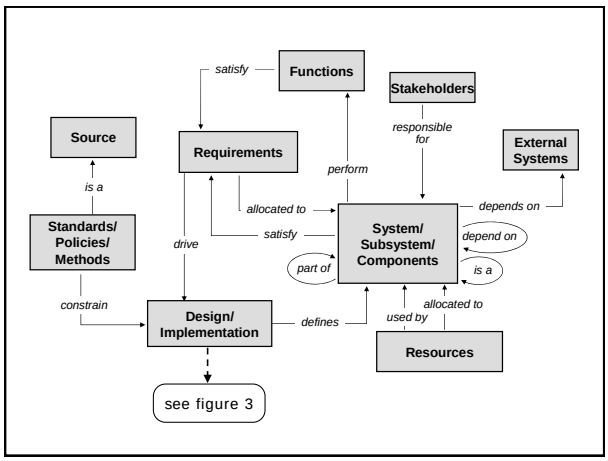


**Figure 4. Design Allocation Model.**

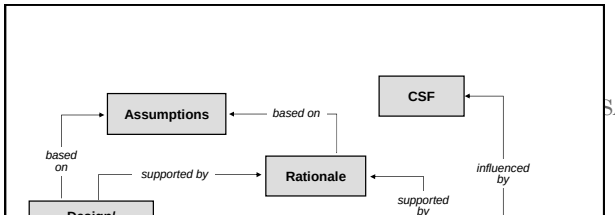


**Figure 5. Design/Implementation Decision Making Model.**

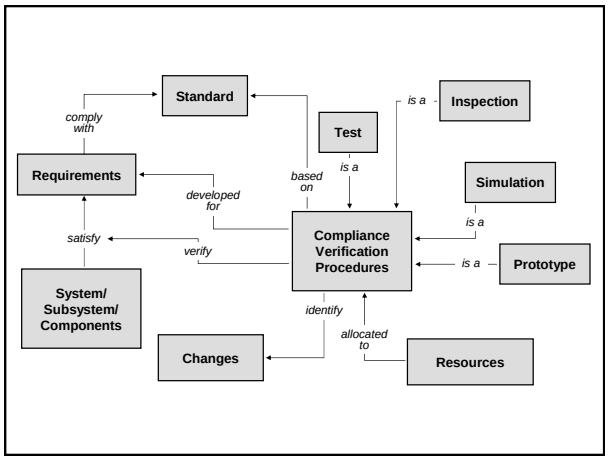


**Figure 6. Compliance Verification Model.**

The requirements traceability model is composed of three of these four models: requirements management, design/ implementation and allocation as well as compliance verification. The fourth model, design/implementation decision making is used only for the decision process during the design and allocation phases. All of the models focus on the types of requirements links that should be used to implement the mandated requirements linking activity in an intelligent and useful manner.

These information models are solely designed to link text-based system descriptions and do not address executable

system models. The subject of the models is "the system under development." A standard approach to the modelling of the system production process and deployment environment is not provided. A clear distinction needs to be made between the product system and the system that produces and maintains the product system.

## Generic Information Model Development

Any generic systems engineering information model must be able to support a wide range of applications. In essence, the generic information model must be based on fundamental systems engineering processes and patterns that are widely recognized and reduce domain complexity for the human users. At the same time, the generic model must provide a powerful, extensible construct upon which the design of large, distributed systems can be effectively based. The primary design goal is the reduction of system complexity, both in system production and system use. Therefore, the generic model must be based on the natural information flow found in the systems engineering domain. From the engineering standpoint, the model must address the activity of problem solving. At the same time, from the systems aspect, the model must address basic system concepts. The most important systems concepts that must be addressed are system components, system boundaries and the system universe of environment. The most important engineering concepts that must be supported by the generic model are problem solving and structured system solution development.

The generic information model, named CCFRAT, presented here was developed from the FRAT system model by encapsulating the four FRAT views in a system concept view and system context view. The system concept and context view are added to the model to provide a mechanism where system designers can detail specific global aspects of the current system design and how these aspects map to other systems in the environment. The context view is used to detail relationships that are important between and among the current system and other systems existing in the environment. The concept view provides a mechanism to detail controlling concepts that directly relate to the current system design and design abstraction process. Figure 7 shows the view relationships.

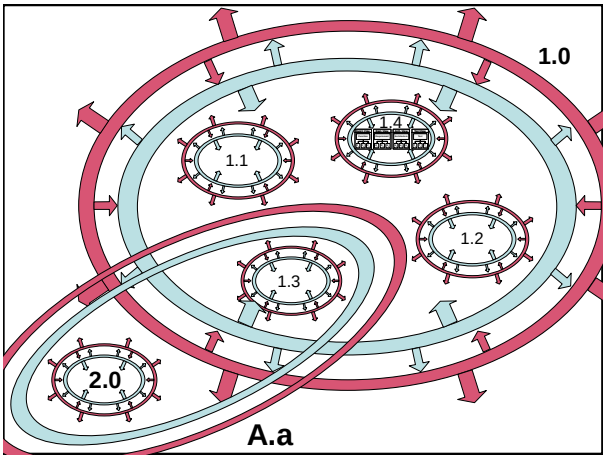


**Figure 7. Generic System Views.**

## Logical Generic Model Relation Development

The standard database systems development cycle was followed to produce a set of database design models. First a set of six conceptual data models were developed, one for each of the CCFRAT views. Next, data and process analysis was accomplished to prepare for the development of the logical database models and the entity database tables. The generic database will contain the following main entity tables: system table, context table, concept table, function table, requirements table, architecture table, test table, decision table, type table, document table, model table, database model table,

fr_link table, fa_link table, and the at_link table.

The primary database entity relationship (ER) structures are presented next. The system database model table is shown in Figure 8.

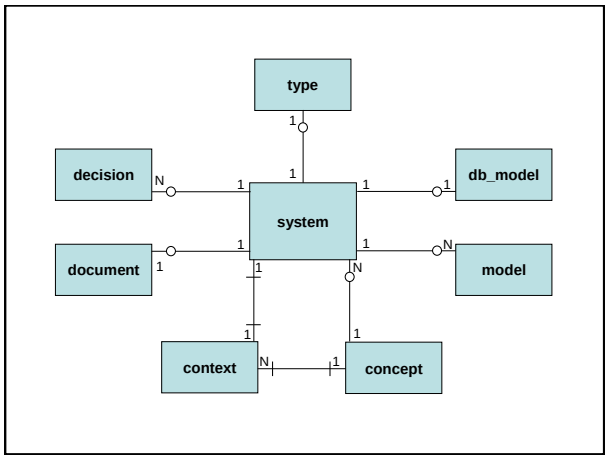


**Figure 8. System Logical ER Model.**

The system table contains a row for each system in the model. The minimum number of system table entries is three, one for the environmental system, one for the process system and one for the product system. In any real system development activity the data table would contain tens if not hundreds of entries.

The context table contains a row for each system context view in the system model. The context view is focused on the outward looking view from the system of interest, and details the important connections and interactions in the system environment. Since systems can reside inside other systems, there are three general types of connections that can be made by starting at the context boundary and traveling outward. The first type of connection is a "**context to context**" connection. The second type of connection is a "**context to concept**" connection that goes between two systems. The third connection type is a "**context to concept**" connection which is made on the same system. The context database model table is shown in Figure 9

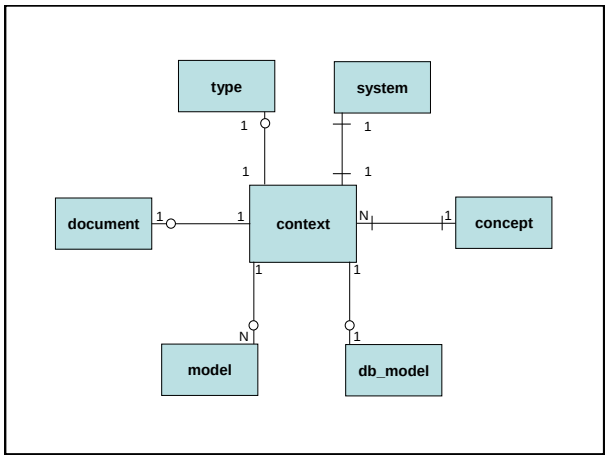


**Figure 9. Context Logical ER Model.**

The concept model contains one row for each system concept view represented in the model. The concept view is focused on an inward view of the system and details the concepts associated with entities that make up the interior system structure. The concept view can be of two general types: one type that contains other system context views and another type that contains only function, requirement, architecture and test views. The concept database model table is shown in Figure 10.

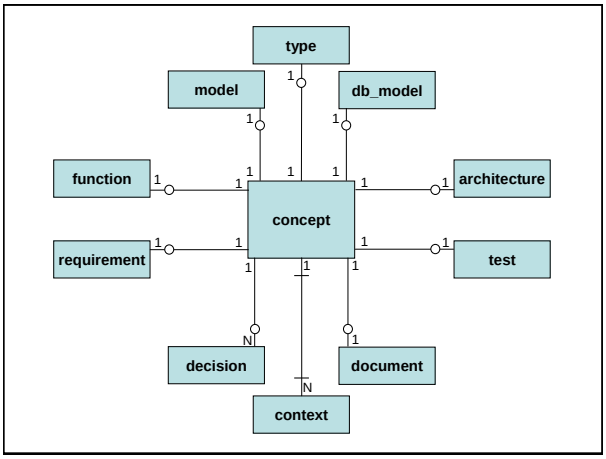


**Figure 10. Concept Logical ER Model.**

The function table contains one row for each system function view represented in the system model. The system functions are

determined and modelled in a manner described in the associated concept view. The function database table model is shown in Figure 11

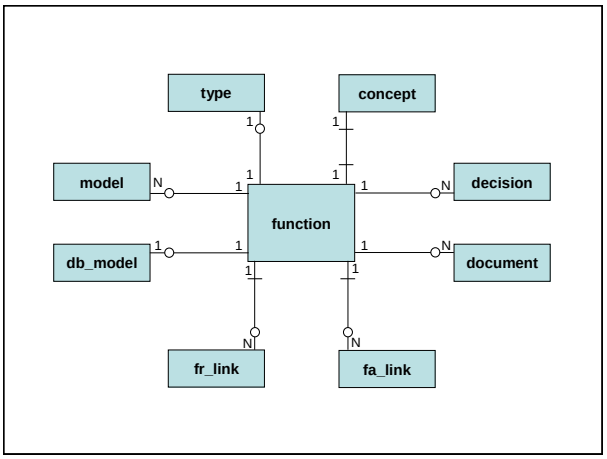


**Figure 11. Function Logical ER Model.**

The requirement table contains one row for each system requirement view. Each requirement view is directly connected to one or more function views to create the basic problem statement. The requirement database model table is shown in Figure 12.

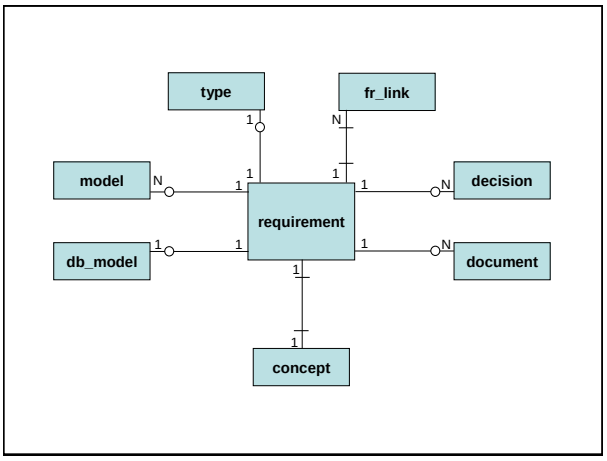


**Figure 12. Requirement Logical ER Model.**

The architecture table contains one row for each candidate architecture solution considered for inclusion into the system solution. The architecture is the system solution that answers the "function and requirement" problem statement. The architecture database model table is shown in Figure 13

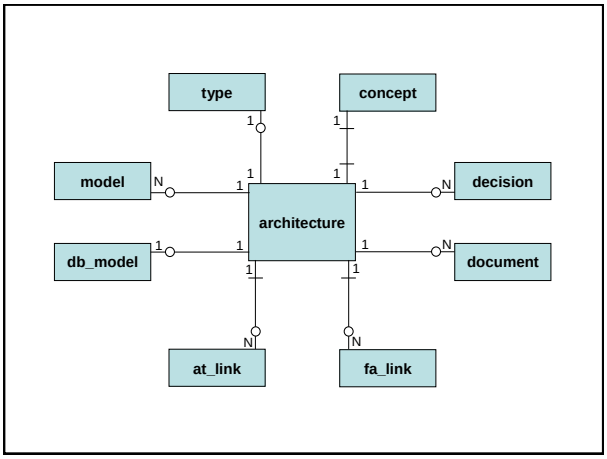


**Figure 13. Architecture Logical ER Model.**

The test table contains one row for each test view represented in the model. The test view focuses on recording the tests and procedures used to assure that the selected architecture will perform the required function as well as the requirement states. The test database model table is shown in Figure 14.

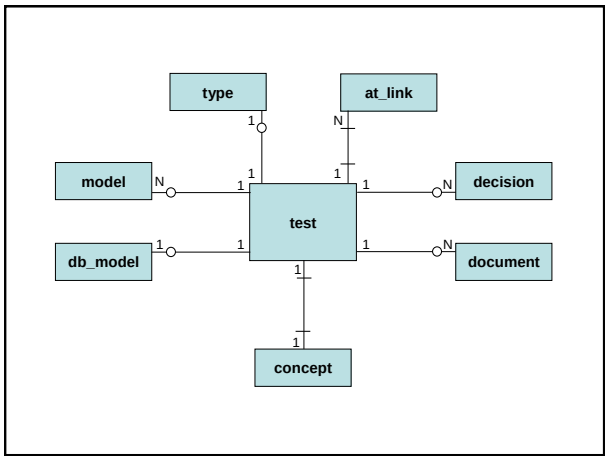


**Figure 14. Test Logical ER Model.**

The decision database table contains one row for each decision made during the system

development process. Decisions and decision types range from the recording of expert judgement about a subject, to fully developed trade studies involving many people and experts.

Each type of classifier used in the system model is recorded in the type table. These classifiers or types are used for grouping aspects of the system concepts in convenient workable units. Every document or document fragment developed during the system development activity will be recorded in a row of the document table.

The model table provides a mechanism to track all executable models associated with the system acquisition task. The model table contains one row for each executable model used in the process. Each row in the database model table contains an entry for the database models, scripts, functions, and processes used in the systems development process.

The preceding twelve database tables are used as the core basis of the systems engineering information model presented in this paper. The primary system concepts and data entities are represented directly by using one or more of these tables. A set of link record tables are presented next to facilitate the "many-to-many" relationships found in the generic model. The link record table is used to break a many-to-many relationship into two "one-to-many" relationships between two entities or tables. The link record is the third table used in this connection type.

The function, requirement, architecture and test tables have many-to-many relationships. These many-to-many relationships are: function table to requirements table, function table to architecture table and architecture to test table. The following link record tables are used to create the required one-to-many relationships: fr_link table, fa_link table and the at_link table.

This database model provides a systems engineering information model that may be used as a common foundation for the construction of a set of standard systems engineering databases. A standard generic database model provides needed semantic continuity and application. Many systems engineering and management tasks have their own specialized computed based tools and models that are already deployed and used in the organization. The information model presented here provides for the inclusion of these other data sources by either linking them as a specific system support model or including them as a complete system.

The systems engineering information model presented in this paper places a strong emphasis on modelling and tracking of the incremental, sequential development of customer requirements and needs. The complete system development lifecycle is modelled in a manner that enhances technical communication by establishing a core set of system concepts and models that can be used for any type of system development. Generic systems development patterns and associated information model patterns are used to communicate complex sets of technical data.

## Conclusions

Distributed systems engineering tools are readily available in most large systems development efforts to support the efforts of the distributed engineering teams on the program. This paper presents a generic, adaptable core systems engineering information model that can be used as the common data storage mechanism in distributed systems engineering tool sets. This model is designed to reduce complexity by selecting a core group of concepts that can be applied recursively at every level of system decomposition or system abstraction.

The establishment of this generic model and application pattern provides the foundation for complexity and cost reduction in any system development program.

## References


Forsberg, K., Mooze H., The Relationship of Systems Engineering to the Product Life Cycle, *Proceedings of the First Annual International Symposium of the National Council on Systems Engineering*, 1991 pp57-68.

Grady, J.O., "The Necessity of Logical Continuity" *Proceedings of the Fifth Annual International Symposium of the International Council on Systems Engineering,* CDROM 1995.

Hughes, Thomas, *Rescuing Prometheus*. Vintage Books, New York, 1998

INCOSE Tools Database Working Group, 2003, *IEEE-1220 Process to SE Tools Mapping.* http://www.incose.org/tool/ieee1220top.html (10 Aug. 2003)

Lake, J.G., and Sheard, S.A., *Systems Engineering Standards and Models Compared.* http://www.software.org/pub/externalpapers/9804-2.html (10 Jan. 2005)

Maier, M.W., "Reconciling Systems and Software Architecture" *Proceedings of the Eighth Annual International Symposium of the International Council on Systems Engineering,* CDROM 1998

Mar, B.W., "Back to Basics" *Proceedings of the Second Annual International Symposium of the International Council on Systems Engineering,* pp 37-43 1992.

Mar, B.W., and Morais, B.G., "FRAT – A Basic Framework for Systems Engineering" *Proceedings of the Eleventh Annual International Symposium of the International Council on Systems Engineering,* CDROM 2002.

Oliver, D. W., Kelliher, T.P., and Keegan, J.G., *Engineering Complex Systems with Models and Objects*. McGraw-Hill, New York, 1997

Simpson, J.J, and Simpson, M.J.,"U.S. DoD Legacy SE & Implications for Future SE Implementation" *Proceedings of the Eleventh Annual International Symposium of the International Council on Systems Engineering,* CDROM 2002.

Ramesh, B., .Powers, T, Stubbs, C., and Edwards, M, "A Study of Current Practices of Requirements Traceability in Systems Development." *Monterey: Naval Postgraduate School,* 1993.

Ramesh, B., .Harrington, G,. Rondeau, K., and Edwards, M, "A Model of Requirements Traceability to Support Systems Development." *Monterey: Naval Postgraduate School,* 1993.


## Biography

**Joseph J. Simpson**'s interests are centered in the area of complex systems including system description, design, control and management. Joseph has professional experience in several domain areas including environmental restoration, commercial aerospace and information systems In the aerospace domain, Joseph has participated in a number of system development activities including; satellite based IP network design and deployment, real-time synchronous computing network test and evaluation, as well as future combat systems communications network design.

Joseph Simpson has a BSCE and MSCE from the University of Washington, an MSSE from the University of Missouri-Rolla, is a member of INCOSE, IEEE, and ACM.

Currently Joseph is enrolled in a system engineering doctorate program at the University of Missouri-Rolla.

**Dr. Cihan H Dagli** is a Professor of Engineering Management and Systems Engineering as well as the director of the System Engineering graduate program at the University of Missouri-Rolla. He received BS and MS degrees in Industrial Engineering from the Middle East Technical University and a Ph.D. from the School of Manufacturing and Mechanical Engineering at the University

of Birmingham, United Kingdom, where from 1976 to 1979 he was a British Council Fellow. His research interests are in the areas of Systems Architecting, Systems Engineering, and Smart Engineering Systems Design through the use of Artificial Neural Networks, Fuzzy Logic, and Evolutionary Programming.

He is the founder of the Artificial Neural Networks in Engineering (ANNIE) conference being held in St. Louis, Missouri since 1991. He provided the conduit to the dissemination of neural networks applications in engineering and decision making through these conferences for the last fourteen years. He is the Area editor for Intelligent Systems of the International Journal of General Systems, published by Taylor and Francis, and Informa Inc.

**Dr. Ann Miller** is the Cynthia Tang Missouri Distinguished Professor of Computer Engineering at the University of Missouri – Rolla. Previously, she was the Deputy Assistant Secretary of the Navy for Command, Control, Communications, Computing, Intelligence, Electronic Warfare, and Space for the U. S. Department of the Navy. For a portion of that time, she had additional responsibilities as Department of the Navy Chief Information Officer (CIO). She also served as Director for Information Technologies, Department of Defense Research and Engineering. Prior to that, Dr. Miller served for over 12 years with Motorola, Inc. where she held a variety of technical and managerial positions. She holds one U. S. patent in satellite communications, has co-authored three books on the programming language Pascal, and is the author of more than five dozen journal articles and monographs. Dr. Miller chairs the NATO Information Systems Technology Panel and is a Senior Member of IEEE. Dr. Miller's research areas include reliability and security of computer-based systems, with an emphasis on networked large-scale systems.

**Dr. Scott Grasman** is an Assistant Professor in the Engineering Management and Systems Engineering Department at the University of Missouri – Rolla. Prior to joining UMR, he was an Adjunct Assistant Professor at the University of Michigan, where he received his B.S.E., M.S.E., and Ph.D. degrees in Industrial and Operations Engineering. His primary research interests relate to the application of quantitative models to manufacturing and service systems, focusing on the design and development of supply chain and logistics models, including business process design, logistics, and enterprise collaboration. He has received research funding from, among others, the National Science Foundation, US Department of State, and SAP America. He is a member of American Society for Engineering Education (ASEE), American Society for Engineering Management (ASEM), Decision Sciences Institute (DSI), Institute of Industrial Engineers (IIE), Institute for Operations Research and Management Science (INFORMS), and has served in various roles with these and other organizations.