Proceedings of ICAD2002
Second International Conference on Axiomatic Design
Cambridge, MA – June 10-11, 2002
ICAD 008

THE VALIDATION OF A MODULAR COMMERCIAL PRODUCT ARCHITECTURE

V. A. Lentz, Bruce Lerner
Otis Elevator, a UTC Company
5 Farm Springs
Farmington, Connecticut, USA
860.676.5287 / 860.676.6149
Virginia.Lentz@otis.com
Bruce.Lerner@otis.com

Clifford Whitcomb
MIT
77 Mass Ave
Cambridge, Massachusetts, USA
617.253.9309 voice
617.258.5730 fax
whitcomb@mit.edu

ABSTRACT

This paper addresses the process used by one commercial company, Otis Elevator, to validate the architecture of a new flagship product. The existing commercial product architecture is loosely derived through a coupling of the functional domain, as envisioned by corporate and customer needs, and the physical domain, which evolves over time to keep up with innovation and changes in the marketplace. As new technology is investigated, and sometimes subsequently inserted, the occasion arises to more formally reassess the structure of the functional architecture, with a need to continually validate the evolving physical architecture in light of new technology. The goal of the current effort in architecture validation is to create a modular architecture with standardized interfaces to enable significant decreases in time to market, and reduction in the cost of production and support. Modules must meet the need for independence from change with coherence of the system architecture. As the modules and interfaces are standardized, 'standard requirements' are available for reuse across products. The method employed to gain understanding and consistency of the new product architecture was the establishment of criteria for determining the boundaries between the functional entities and physical entities and the mapping between them. The principles of design, as advocated by Professor Nam Suh, were used to drive decisions by weighting using the Independence and Information Axioms. The methods and approach will subsequently be used with the other product lines. The use of the principles of design is discussed, with the business drivers, related to taking the time and resources to do the validation, and business plans, for evolving the engineering and operations within the company to realize the advantages sought, will be addressed.

VALIDATION DRIVERS

The physical architecture of the elevator has evolved over time to keep up with innovation and changes in the marketplace. As technology is investigated and sometimes subsequently inserted, the occasion arises to more formally reassess the functional architecture and validate the evolved physical architecture in regard to expanded functionality and new technology. Today, independent modifications in the various pieces of the product force substantial changes in the physical architecture.

Otis Elevator found two primary reasons to validate the architecture of the elevator: 1) The need to significantly reduce the time to market for new products and 2) the need to optimize the operations aspects for translation to business results. The Senior VP of Operations noticed that ad hoc interfaces that did not support an operational function of the elevator were costing the company a lot of money in terms of drawings, manufacturing processes and field processes. Since there was not an operational basis for the interface one might ask what was the driver for the connection? The answer is immediacy. The customer wanted a handrail. The most expedient solution was to connect the handrail to an existing structural member, which might not have been chosen if a 'clean sheet' design was developed. The solution for a one-time customer request was reused for the next request for the same feature, and this immediate fix was not just for handrails. There were other examples, and the costs were adding up. In addition the distributed nature of the product delivery operations around the globe meant that an expedient solution in one hemisphere, might be solved differently in another hemisphere. Rationalization of the various designs for global efficiencies was difficult if not impossible. The parts of the elevator that are visible to the public must accommodate variation in the aesthetics. The riding public sees only thirty percent of the elevator. The unseen seventy- percent is an opportunity for global design and perhaps distribution.

Final design of the elevator for any single installation takes place when the order is in hand. Hoistway dimensions vary from building to building, and the physical product must accommodate this variation. Regional and local regulatory agencies might require changes in the installation. Regional and local aesthetic tastes must be accommodated. Final assembly of the elevator takes place in the host building.

Today, the parts are engineered in x places, manufactured in y places, and need to fit together at one installation site. Installation teams are distributed among z regional companies. When the volume is sufficient, installation teams can focus on a particular product within the product line.

The pieces to implement the global designs are procured globally. For strategic pieces, there is one supplier providing pieces around the globe. The local manufacturing or logistic center consolidating shipments to customer sites obtains other pieces. The pieces arrive at the installation site in a compact form as a truck full of pallets containing boxes of elevator pieces. The pieces are installed by local mechanics, according to a global installation process and subsequently maintained by global service and repair processes.

Any architectural initiative needed to focus on understanding the functions performed by the physical elements, and provide the flexibility to accommodate the needs of technology, manufacturing, installation and maintenance. To support a product range from 2 to 110 floors or more, the architecture also had to accommodate scalable elements and the derivation of product platforms.

An early proof of concept activity for the module based architecture, indicated the basic physical architecture was sound, and any refinements would be evolutionary rather than revolutionary. The effort to standardize the interfaces across the product range resulted in the need to accommodate 'classes of interfaces'. While the modules for the mid and high range products might be the same, the interfaces for such items as bolt and welding patterns might need to be tailored for each class of products.

ARCHITECTURAL DRIVERS

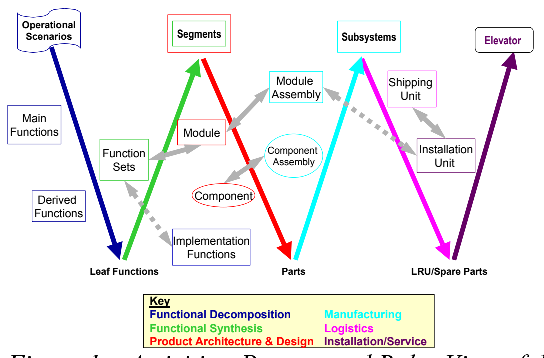


Figure 1. Activities, Process and Roles View of the Architectural Initiative

To realize the potential savings in Time to Market, and in the cost of production and support, more than the 'engineering drivers' needed to be applied to the development of the architecture. This approach shown in Figure 1 was initially described in Adifon (2001). It touches on three aspects of the validation initiative. It represents the activities in the primary order in which they were performed, providing a process overview for the execution of our architecture development. The products of each activity identify the deliverables and our terminology. The colors of the arrows identify the roles within the enterprise that are the primary stakeholder in the activities. Any standard architecture is required to address the needs of each of the enterprise participants. The team applied typical Systems Engineering methods of operational analysis, functional analysis and physical analysis resulting in descriptions of the various elements and their interfaces. Along the way, we stabilized the applied terminology, a hybrid of company legacy terms and 'new' globally consistent terms.

Functional Decomposition - discovering functions

Scenario Analysis.

Existing product knowledge was applied to develop scenarios that document the sequence of activities to use the product / system. The user might be the 'consumer' or the maintainer of the system / product. This activity identified the majority of the main and derived functions. The authors define a main function as a major function that directly contributes to an operational scenario. These functions are unique in that they are not duplicated, shared or reused in the functional hierarchy A derived function is a function that supports the functions at the next level up in the decomposition. They are directly derived from higher level functions and do not depend upon implementation decisions.

Reengineering.

It is often easier for legacy engineers to work backwards from the current implementation or a series of implementations to the function(s). Let them! Exploit the knowledge of these engineers and facilitate the sessions where they use existing product knowledge to analyze the product components to identify the functions performed by the component.

Reverse engineering is particularly useful to identify the implementation functions. We *define implementation functions as those functions in the functional hierarchy that support derived functions but only exist due to implementation decisions. They are not directly derived from higher level functions.*

Functional Synthesis - Function Sets and Segments

A small team individually grouped the resultant leaf functions (*lowest level functions from the decomposition*) based on their experience. These groupings were then compared and rationalized creating a single, agreed upon set of function sets to carry forward into the development process. This technique was reapplied to group the function sets to categories that provide breadth and options for design and implementation. While the initial 'intent' was to find categories that addressed both design and implementation, it was found that there was a predilection to address design or delivery issues based on the experience of the group member. The drivers associated with design or implementation were found to be orthogonal and we developed two groupings - 'segments' for design and 'subsystems' for delivery.

Design

The function sets needed to be grouped for implementation into the entities we called 'modules'. The modules compose both segments and subsystems and are the transition elements between the functional and physical spaces. The functional space transitions to the physical world in the segments, while the subsystems address issues in manufacturing and the field. Function sets were allocated to segments. After initial progress with our functional synthesis, the allocation decisions were made using *The Principals of Design* Suh (1990).

THE PRINCIPLES OF DESIGN

The Principles of Design describes a design process that maps the objectives in the functional space captured as specific requirements ➔ Functional Requirements (FRs) to the solution in the physical space characterized with Design Parameters (DPs). This mapping process begins with the high level FRs and iterates between initial DPs and lower level FRs creating a hierarchy of both FRs and DPs. Particular DPs indicate design decisions which constrain the FRs at the next level of detail. The treatment of the design process as one that is rooted in a differentiation of the functional and physical domains matched the goal of our project - to find a stable architecture from which to instantiate product families. Although the axiomatic framework provides substantial mathematical treatment , we found value in a more qualitative approach that provided decision criteria for allocation of functionality in both the functional and physical architectures.

Axioms and Corollaries

The two main axioms, the Independence Axiom and the Information Axiom, are applied using corollaries that are derived to aid in making design decisions. In this effort, the team cast the corollaries as decision criteria for use in decision matrices, exemplified in Figure 2.

		Alternative 1			Alternative 2		
Criteria	Wt		Sc			Sc	
Min. external functional dependency	10						
Max. opportunity to innovate/improve	10						
Max. testing independence	8						
Min. information scope	6						
Max. functions with similar 'information' needs	6						
Min. sources of requirement	5						
Min. differing customer sets	5						
TOTAL							

Figure 2. Function to Function Set Decision Matrix

The specific points of application of the criteria in this architecture are in the:

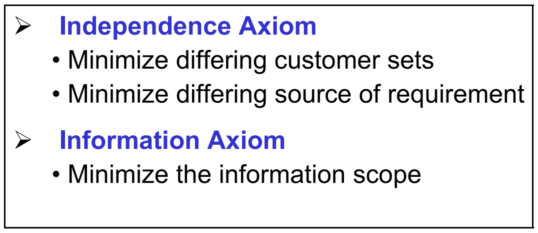
1. Aggregation of Functions to Function Sets
2. Development of the Segments (allocation of Function Sets to Segments)
3. Allocation of Function Sets to Modules
4. Separation of function sets to facilitate product delivery. (This application was needed because the team determined that function sets would not be split across modules).

Group Functions to Function Sets.

The authors define *function sets as sets of functions (or function sets) grouped using defined design principles and trade-offs to support system implementation goals.* The functions are grouped using criteria based on the information and independence axioms as originally postulated. We found that both grouping and separation criteria were needed to overcome the tendency to group to a monolithic set. Our Function to Function Set grouping criteria:

- ➢ **Independence Axiom**
  - • Minimize external functional dependencies
  - • Maximize testing independence (greater coupling within than without)
  - • Maximize opportunity to innovate/improve within the Segment without impacting other Segments
- ➢ **Information Axiom**
  - • Maximize functions with similar data needs

Figure 3. Function to Function Set grouping criteria



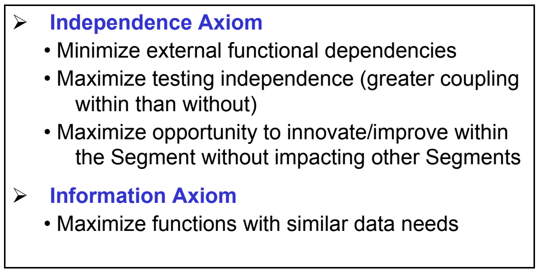
- **Independence Axiom**
  - Minimize differing customer sets
  - Minimize differing source of requirement
- **Information Axiom**
  - Minimize the information scope

Figure 4. Function to Function Set separation criteria

In doing the functional synthesis, we want a stable set of modules, however, we often end up wanting to group things together based on what is in a scenario. The team's initial experience indicates that slavishly follow the functional or operational architecture is terribly inefficient and decreased the ability to 'share' implementation functions. Use of the independence and information axioms helps alleviate this inefficiency.

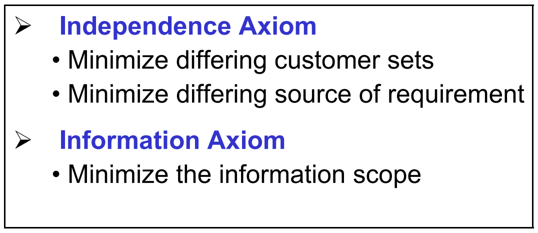
Group Function sets to Segments.

Segments provide a grouping of the function sets that facilitates product development. For the purpose of not confusing this grouping for product development with the subsystem terminology for product delivery, we chose the word 'segment'. This provides for the optimization of development entities around broad functions. The 7 plus or minus 2 rule applies to the results of the grouping. One driver for the development of a new architectural layer, the segment, was an initial tendency to group things according to the engineering disciplines of mechanical, electrical and software. This has a natural extension to the capabilities of the factories that deliver the product elements. The result of carrying forward this particular set of constraints would have been a sub optimization and a conflict of the axiomatic design principles. The argument against this grouping was the effective because we were able to point to existing product development problems that were caused by using the discipline as the grouping criteria. The information and independence axioms as postulated by Nam Suh, were used to group the function sets into segments.



- **Independence Axiom**
  - Minimize external functional dependencies
  - Maximize testing independence (greater coupling within than without)
  - Maximize opportunity to innovate/improve within the Segment without impacting other Segments
- **Information Axiom**
  - Maximize functions with similar data needs

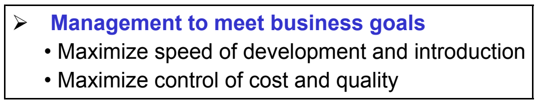
Figure 5. Function Set to Segment grouping criteria



- **Independence Axiom**
  - Minimize differing customer sets
  - Minimize differing source of requirement
- **Information Axiom**
  - Minimize the information scope

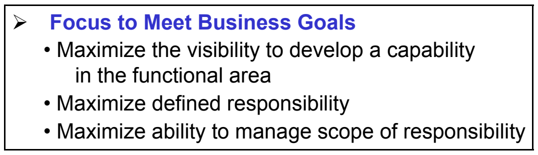
Figure 6. Function Set to Segment grouping criteria

In addition to Nam Suh's axioms, additional criteria were applied to address business goals for the system. To further combine and separate the functions in a function set we added:



- **Management to meet business goals**
  - Maximize speed of development and introduction
  - Maximize control of cost and quality

Figure 7. Function Set to Segment grouping criteria



- **Focus to Meet Business Goals**
  - Maximize the visibility to develop a capability in the functional area
  - Maximize defined responsibility
  - Maximize ability to manage scope of responsibility

Figure 8. Function Set to Segment separation criteria

These business criteria reflect company priorities and may need to evolve to address current technologies, external regulations etc.

The other set of needs to be addressed for grouping function sets to segments are design constraints that were represented as gray arrows on Figure 1. The methods for shipping and installation of a product and their associated constraints (size, manufacturing techniques and locations) that are needed to ship and install a product will affect the allocation of functions to modules. The capabilities of existing factories and suppliers were another consideration.

Allocate function sets to physical modules.

Again, applying criteria derived from the information and independence axioms, along with a few additional guidelines, one or more function sets were allocated to top level physical entities (modules). Here we developed Module Guidelines rather than more formal decision criteria:

- Elements in a Segment that are in different Subsystems must be in different Modules
- Elements in different Segments must be in different Modules
- Maximize the flexibility to 'scale' implemented Modules to cover the product range.

Figure 9. Module Guidelines

Additionally, product delivery considerations e.g. manufacturing sources, installation processes are applied to further refine the allocation of one or more function sets to physical entities. At this level, manufacturing processes may constrain the mapping. For elevators we have another step in that final assembly of the product is in the building elevator shaft. Thus we also need to consider the shipping packages and installation units when mapping to the physical architecture.

The Result

As we roll out the standardized interfaces we will have a robust architecture that will provide effective research and development that facilitates the reuse of existing supply chains while providing addition capability to the customer, and returns to the enterprise, and the technology matures to do so. The enterprise is now in a position to perform up front planning of generation changes at the module / component level. The release of a new product can then become the integration of proven components.

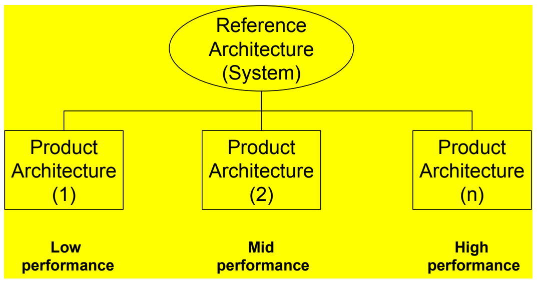


Figure 10. Architecture Hierarchy

The reference architecture is based in the functional domain with extension into the physical domain using constraints (how well, under what conditions) on the functions that are specific enough to define a 'family' of solutions but general enough to allow product differentiation at the design level.
The application of standard interfaces allow for consistency with flexibility as we migrate module assemblies up and down the product families and the flexibility to apply 'optional' modules as appropriate in the families.

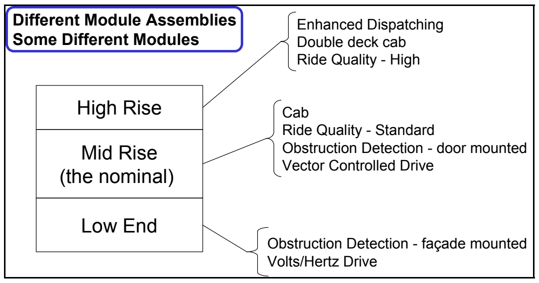


Figure 11. Application to Product Families

THE DELIVERABLES

The direct deliverables of the reference architecture design activity in our environment include the function list, the list of function sets and the list of modules with the associated allocations of the constituent elements and the rationale for the allocation.

In addition, an overall mapping from the requirements (both domain and product) to the architecture and design specifications has emerged.

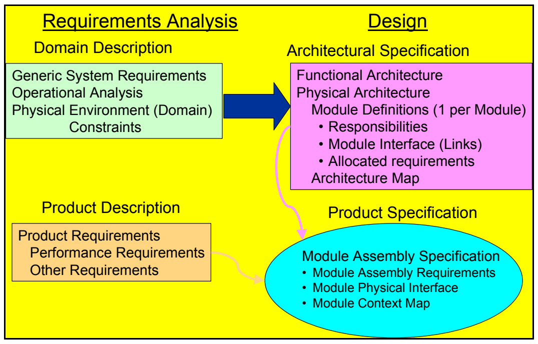


Figure 12. Architecture to Product Mapping

As important as the direct result is the captured rationale that allows for assessment and internalization by the next broader groups of stakeholders. The use of a defined alternatives analysis matrix helped us validate the criteria and defuse opinion driven design discussions.

REFERENCES

Adifon et al, "Validating a Commercial Product Architecture" *Proceedings of the 11th Annual International Symposium of INCOSE*, Melbourne, Australia, 2001.

DeGregorio, Gary, "An Information Architecture for Systems Engineering - Progress, Examples and Directions", *Proceedings of the 7th Annual International Symposium of INCOSE*, Los Angeles, California, 1997.

Fitch, John, "Structured Decision-Making & Risk Management", Course Notes, Systems Process, Inc., February, 1999.

Suh, Nam P., *The Principles of Design*, Oxford University Press, 1990

BIOGRAPHIES

V. A. Lentz is responsible for Systems Process and Infrastructure for Otis Elevator Company. She has been with United Technologies since 1996. Her focus is the use of Systems Engineering in commercial enterprises. Prior to UTC, she spent 30 years at IBM Federal Systems, LORAL, Lockheed Martin building large, unprecedented computer-based systems such as Global Positioning System Control Segment. She was also responsible for Systems Engineering Technology, Process and Training. Ms. Lentz was President of INCOSE in 1996, a recipient of the Founders Award in 1999, and represents UTC on the INCOSE Corporate Advisory Board.

Bruce Lerner is a Principal Systems Engineer at Otis Elevator, a United Technologies Company. He has 14 years experience developing embedded Software for communications and control systems and facilitating Software Process Improvement activities. His last five years have been applied to Systems Analysis and Architecting and the Systems Engineering process. Mr. Lerner is a member of INCOSE and the IEEE Computer Society.

Dr. Clifford A. Whitcomb is in the Engineering Systems Department (ESD), and Director of Learning in the System Design and Management (SDM) program at MIT. He is a member of the INCOSE Board of Directors, serving as Director, Region IV. He teaches and performs research in the areas of systems engineering, multidisciplinary design optimization, and multiple criteria decision making as they apply to product development and in particular, naval combatant ship design. He is currently conducting research in technology development at the MIT Center for Innovation in Product Development (CIPD), developing systems engineering frameworks for new product development funded by the U. S. Navy Office of Naval Research (ONR). He has practical experience managing engineering design teams for the New Attack Submarine Cost and Operational Effectiveness Analysis, and supervising ship construction of nuclear attack submarines.