Layered Systems Engineering Engines

Julian C. Breidenthal[1]
Jet Propulsion Laboratory, California Institute of Technology, Pasadena, California 91109

and

Marvin J. Overman[2]
Business Management Solutions, Inc., Ralston, Nebraska 68127

A notation is described for depicting the relationships between multiple, contemporaneous systems engineering efforts undertaken within a multi-layer system-of-systems hierarchy. We combined the concepts of remoteness of activity from the end customer, depiction of activity on a timeline, and data flow to create a new kind of diagram which we call a "Layered Vee Diagram." This notation is an advance over previous notations because it is able to be simultaneously precise about activity, level of granularity, product exchanges, and timing; these advances provide systems engineering managers a significantly improved ability to express and understand the relationships between many systems engineering efforts. Using the new notation, we obtain a key insight into the relationship between project duration and the strategy selected for chaining the systems engineering effort between layers, as well as insights into the costs, opportunities, and risks associated with alternate chaining strategies.

Nomenclature

SE	=	Systems Engineering
SoS	=	System of Systems

I. Introduction

SYSTEMS of systems (SoS) invariably involve the contemporaneous operation of multiple systems engineering efforts, usually applicable to different degrees of granularity depending on their position within the system breakdown structure. A challenge arises when planning, exercising, or assessing multiple systems engineering engines, because work allocation and product scheduling between multiple efforts is much more complex than for a single effort. Effective management of this complexity requires visualization aids for the relationships between multiple efforts in different layers. Several existing notations are available, e.g. data flow diagram[1], structure chart[2], flow chart[3], activity diagram[4], sequence diagram[5], Gantt chart[6], PERT chart[7], and "Vee" diagrams[8], but all are unable to be simultaneously precise about activity, level of granularity, product exchanges, and timing. By combining the concepts of remoteness of the end product from the customer, depiction of activity on a timeline, and data flow, we devised a notation that is capable of expressing the necessary information efficiently. Specifically, the notation is capable of explicitly representing relationships between systems engineering activity, system granularity, degree of removal from the customer, product exchanges, and time. Using this notation, we obtain a key insight into the relationship between the strategy used for chaining SE efforts at different layers in the SoS, and the costs, durations, risks, and opportunities which result.

---

[1] Assistant Program Manager, Communications Systems Research Section, 4800 Oak Grove Drive Mailstop 238-420, Member.
[2] President, 5312 Miller Avenue, Non-member.

II. Description of New Notation

The notation described here represents relationships between systems engineering activity, system granularity, degree of removal from the customer, product exchanges, and time. When applied, the notation creates a Layered Systems Engineering "Vee" Diagram, hereafter referred to as a “Layered Vee Diagram”.

The Layered Vee Diagram is a cartesian plane containing a representation of systems engineering activities, as depicted in Figure 1. The activities are assumed to be bounded in time, to exchange products with each other at discrete times, and to be drawn from a set of activities defined by a systems engineering model, which individually are characterized by degrees of removal from a customer. The activity set is further assumed to allow multiple instantiations of the set in a hierarchy of granularity. Although the examples provided later in this paper are worked out in the context of the activity set comprising the NASA SE model, there is nothing special about this particular set; this notation is applicable to any other set of activities which satisfies the assumptions.

In the notation, one direction (usually horizontal) represents time, usually increasing from left to right. The orthogonal direction (usually vertical) represents two qualities simultaneously: levels of granularity in system decomposition, and degree of removal from a customer. The orthogonal direction is divided into major bands representing levels of granularity. Between major bands, the downward direction usually represents smaller granules. The major bands are further divided into minor bands representing degree of removal for activities. Within a major band, degree of removal usually increases downward. Horizontal and vertical, or the directions of increase, may be interchanged without changing the meaning of the diagram. Such interchanges may be advantageous in certain situations to cope with constraints of the medium used to express the diagram.

Rectangles represent activities. Each activity is placed on the diagram at a horizontal and vertical position corresponding to that activity's time bounds and degree of removal from the customer.

Circles, ellipses, or clipped rectangles represent products exchanged between activities. Each product is placed on the diagram at a convenient location with respect to legibility, but the position is not otherwise significant.

Solid-lined arrows or connected solid-line segments represent transition from operation of one activity to operation of the next in time sequence. Each end of a transition must be connected to exactly one activity; the point of attachment to the activity is significant with respect to time.

Dashed-line segments represent either associations between activities and products, or control gates, distinguished as follows:

1) Single dashed-line segments orthogonal to the time direction, with an accompanying text annotation, represent control gates. One end of a control gate line segment must be unconnected, the other end may be unconnected or connected to one activity. The position of the dashed-line segment is significant with respect to time.
2) Dashed-line segments or a series of connected dashed-line segments, which are connected at one terminal end to exactly one activity and also connected at the other terminal end to exactly one product, represent associations between activities and products. Among a set of activities associated with a product, the earliest one is the generator. Optionally an arrowhead may be used to indicate the generator, pointing away from the generator. The point of attachment of the dashed-line segment to the activity is significant with respect to time.

Solid-line segments orthogonal to the time direction, with an accompanying text annotation, represent an epoch of consideration. In many cases the epoch may represent "time now" but other meanings may be assigned. Both ends of an epoch line segment must be unconnected. The position of the solid-line segment is significant with respect to time.

III. Application to a Typical SE Model

The generic SE "Vee" model depicts SE activity using horizontal timeline, a descending band of activity departing from the customer's needs at the top, and an ascending band returning to the customer where an end-product is delivered. Thus the generic Vee contains the concept of remoteness from the customer as a function of time. There is also the concept of increasing technical resolution from top to bottom of the Vee, usually associated with a hierarchy of system decomposition. This is a general feature of systems engineering activities, layered or not, widely used to organize thinking about SE activity. What we would like to add to this model, however, is the ability to be precise about the level of granularity within the system decomposition, or to be specific about the activities or products resulting from the activities. Certainly, all SE efforts do eventually involve different levels of granularity, specific times of activity, and products produced along the way. Managers usually expect some specifics do be defined for the purposes of work definition, cost and schedule estimation, or progress evaluation. Our experience

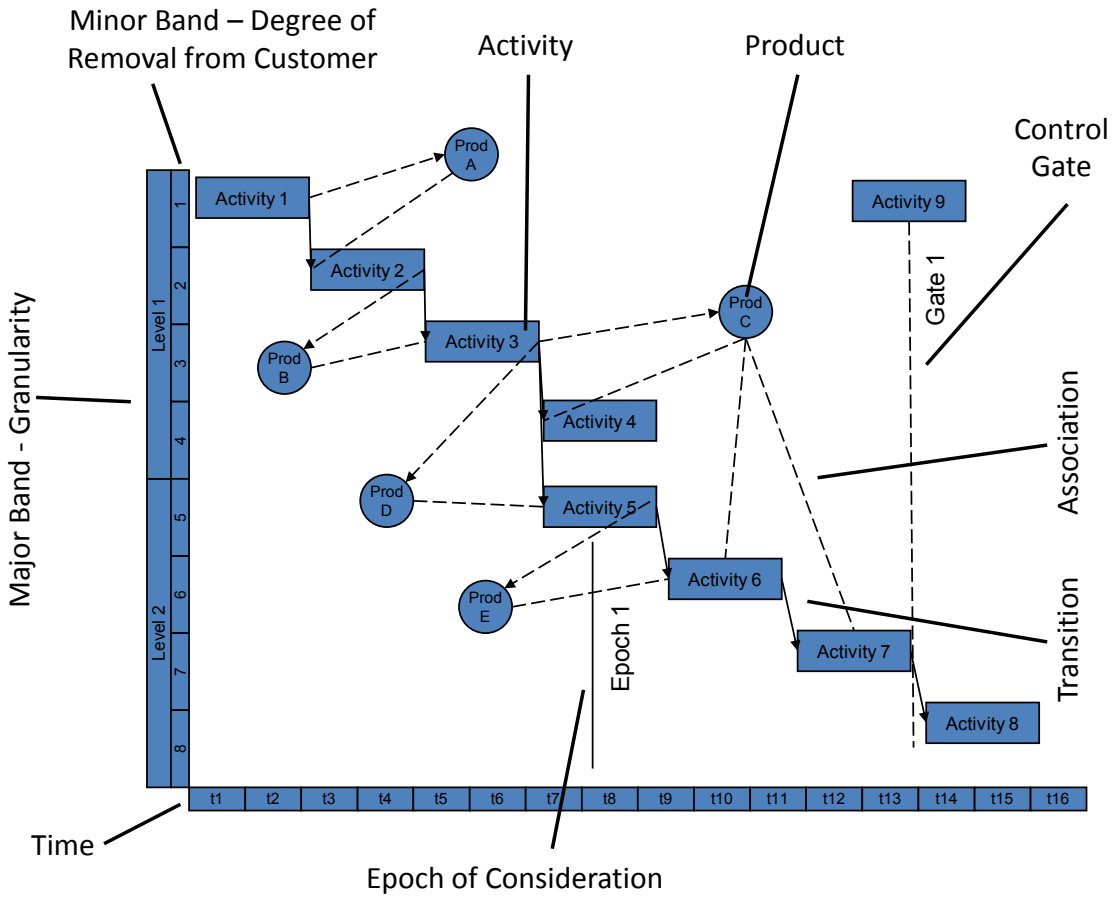

Figure 1. Graphical Elements of the Novel Notation.

has been that verbal (either oral or written) discussions of the specifics quickly become intractable as levels of granularity or iterations of SE activity are added. Thus, it would be helpful to have a visual aid to assist in clarifying the discussion.

For a concrete example, we selected the NASA SE model described in Ref. 9. This model contains a broadly-applicable, technology-independent description of the SE process that is capable of illustrating the issues involved in a layered SoS. The NASA model makes extensive use of the concept of a "systems engineering engine", referring to a set of technical processes that realizes products for a higher-level customer, and that manages the operation of the engine. Within the engine are nine technical development processes:

- Stakeholder Expectations Definition (SED)
- Technical Requirements Definition (TRD)
- Logical Decomposition (LD)
- Design Solution Definition (DSD)
- Product Implementation (PIMPL)
- Product Integration (PINT)
- Product Verification (PVER)
- Product Validation (PVAL)
- Product Transition (PT)

These processes convert initial system specifications from a customer or higher-level system into lower-level specifications, and then integrate lower-level products into an end-product delivered to the customer or higher-level system. The upward and downward transformations made by the engine are intended to enable construction of large, layered systems by recursive application at every node of the system hierarchy. There are also a number of

technical management processes, which are important and have some layer-to-layer interactions, but are not essential for understanding of the layered SoS issues discussed in this paper.

Figure 2 depicts the NASA SE process using the new notation. Note that the product implementation process applies only to the lowest tier in a system hierarchy, the level where products are made, purchased, or reused. In the figure a mid-tier system is assumed; to depict a lowest-tier system, the product integration activity is replaced by the product implementation activity.

It can be seen in the figure that the shape of the generic Vee model is readily discerned, but now it is enhanced with specifics of time, type of activity, and products exchanged. It can also be seen that the general features of a Gantt chart are available (although folded upward on the right-hand side to acknowledge increasing proximity to the customer), and that a complete product list is available. This features are readily translated into traditional planning products such as work assignments and schedules. If the activities are resource loaded, costs can also be estimated; with historical information relating resources needed to produce certain products, schedules can be estimated.

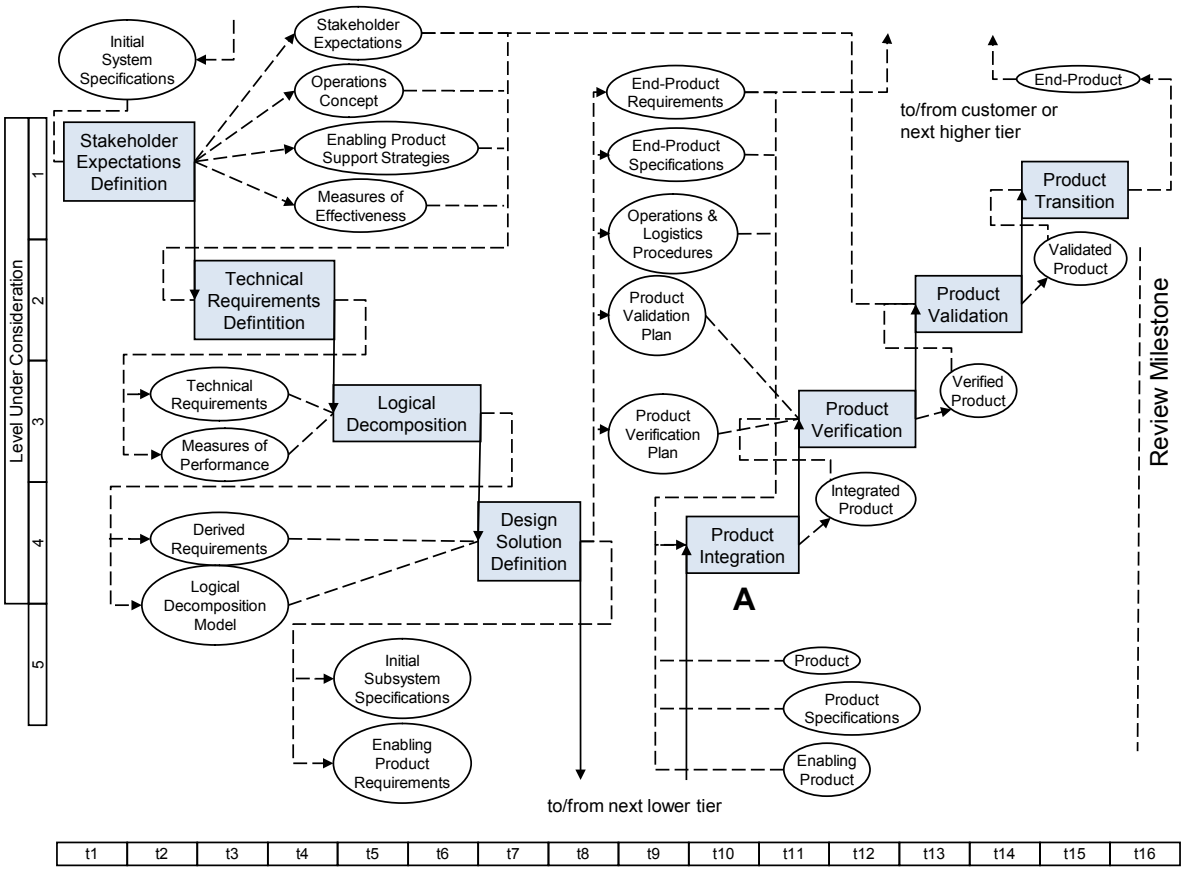

Figure 2. Application of the Notation to NASA SE Model.

The technical development processes for a single tier in a system hierarchy is shown. The cycle shown is iterated once for each review milestone (system requirements review, preliminary design review, etc.). A tier with subsystems is assumed, that is, a mid- or top-tier other than the bottom. For a bottom-tier system, the process at point A is replaced with Product Implementation connected directly to Design Solution Definition.

IV. Application of the Notation to Layered or Iterated SE Effort

When a SoS is developed, multiple systems engineering efforts in different degrees of granularity are needed to complete the entire SoS. This implies that higher-level efforts must be connected to lower ones in some fashion, and probably multiples of those. At any given time, there will probably be many activities occurring simultaneously. This situation can be made clear using the new notation, as shown in Figure 3.

Iteration of the SE activity may occur either because a successive-refinement strategy is in effect, or incremental development of capabilities is desired. This situation can be represented as shown in Figure 4.

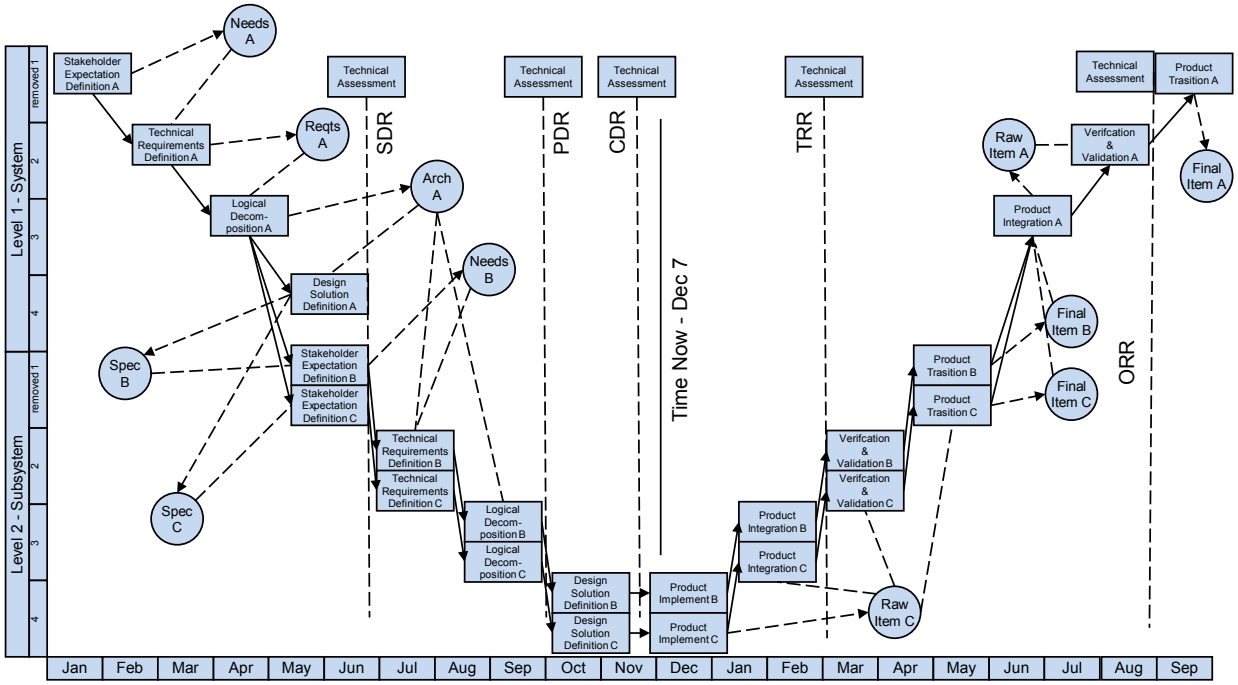

Figure 3. Application of the Notation to Layered SE Effort.
A system "A" containing two separately-managed subsystems "B" and "C" is assumed, with both subsystems developed simultaneously. One management process, Technical Assessment, is introduced to show how system-wide control gates could be represented.

V. Options for Chaining of Layered SE Effort

In the foregoing example an arbitrary time lag was used between iterations of the SE effort, and the iterations were left unconnected as to products (and therefore information). However, engineers in real SE organizations would usually exchange information between the layers or iterations as the effort progressed. We will show in a following section that it is possible to chain the layers and iterations in such a way as to control cost, mitigate risk, or improve schedule. By "chain" we mean to connect the outputs of one process to the inputs of another in time sequence according to a particular strategy. There are a very large number of possible ways of connecting such efforts, but we will explore a few illustrative cases that can form a basis for more complex arrangements. For concreteness we will continue to work these examples out in the context of the NASA SE Model, but the concepts involved would be applicable to any model which recognized layering or iteration of the SE effort.

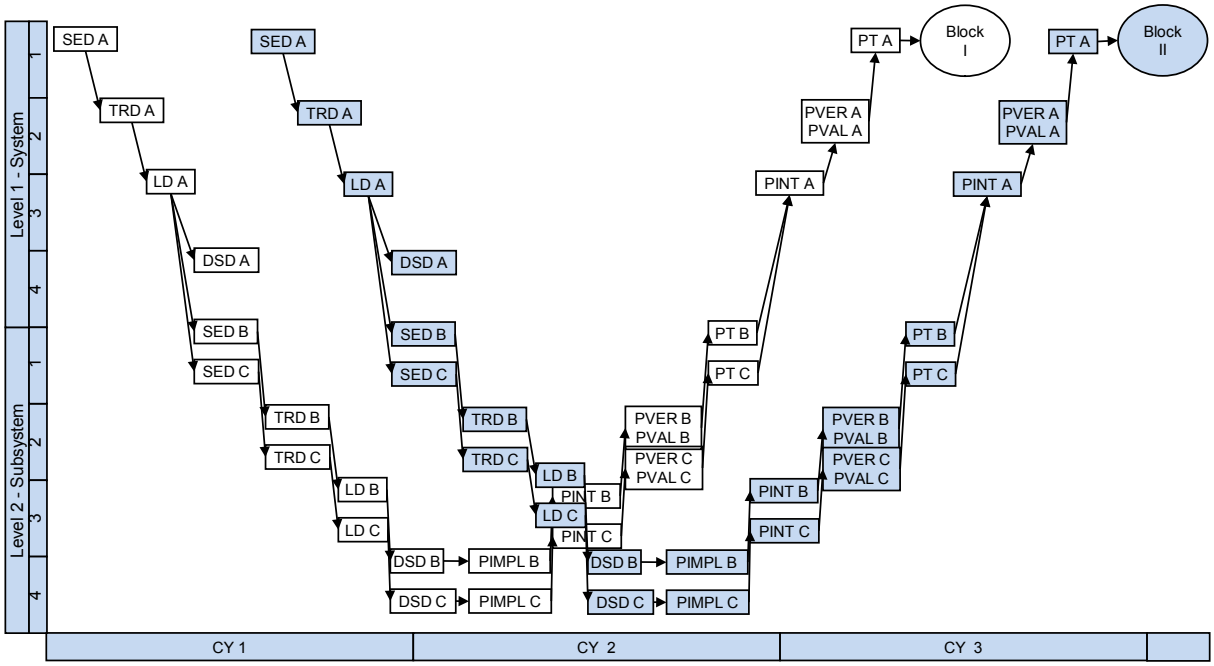

Figure 4. Application of the Notation to Iterated SE Effort.
The same systems and subsystems are assumed, but are delivered in two increments Block I and Block II. Intermediate products are suppressed for clarity.

A. End-Product Chaining - Latest Possible

A common strategy in SoS development is to chain efforts at lower levels in series with the higher levels as shown in Figure 5, by connecting the descending specifications of one effort to the input of the next lower layer, and the ascending products of the next lower layer to the integration process of the next higher layer. This is in fact the chaining method described in the NASA Systems Engineering Handbook at Figure 2.2-1, Section 2.3.2, and implicit in the recursive structure of the SE Engine. In this "latest possible" scheduling strategy, the higher layer completes its system design activity, triggers the next lower layer to begin its system design activity, then waits while the lower layer or layers proceed to a low-level product that can be implemented directly. This accounts for the general downward trend of activities on the left half of the figure. Then, the lower layers execute their product realization activity, and when complete trigger the integration activity of higher layer engines until the end product is delivered to the customer. This accounts for the general upward trend of activities on the right half of the figure. Note the ordering of the milestone reviews, which is from lowest granularity to highest; this is a simple test which can reveal that end-product chaining is in use.

B. Final-Design Chaining

Another strategy for chaining is shown in Figure 6, in which the next-lower tier of specification from one effort is connected to the integration process of the next higher layer. In this strategy, the higher layer completes its system design activity, triggers the next lower layer to begin its system design activity, which then triggers the higher integration activity *as soon as a final design is available*. Note the changed ordering of the milestone reviews, which is now from highest to lowest; this is a test which can reveal that final-design chaining, or one of the more aggressive variants described below, is in use.

It may be observed that the product-realization activities are "dangling" in Figure 6. That is, the end products of lower layers are not connected for integration at higher layers at the end of a single iteration of the SE process.

These connections do of course need to be made, the means of which can be seen when one considers that multiple iterations of the process usually occur. For example, a program will often have a sequence of milestone reviews such as preliminary design review (PDR), critical design review (CDR), and system acceptance review (SAR). These iterations provide an opportunity to connect the "dangling" products from one iteration into a later iteration, a general technique which applies to all chaining strategies. One example of how this could be done for final-design chaining appears in Figure 7. Connection to subsequent iterations resolves all the dangling products except those for the last iteration, which can only be resolved by a final end-product-chained cycle. Optionally, extra product-realization (integration-verification-validation-transition) cycles could be added during that final cycle if desired, along with additional "checkpoints" shown in the figure at the point marked "B". The extra cycles may be useful for management purposes but are not essential to resolving the dangling products.

A notable feature of iteration-to-iteration connections is that it may be possible in some cases to insert the *information* resulting from a lower-level iteration into a higher-level design process, sooner than the actual product can be inserted into the higher-level integration process. Such a situation occurs at the point marked "A" in the figure.

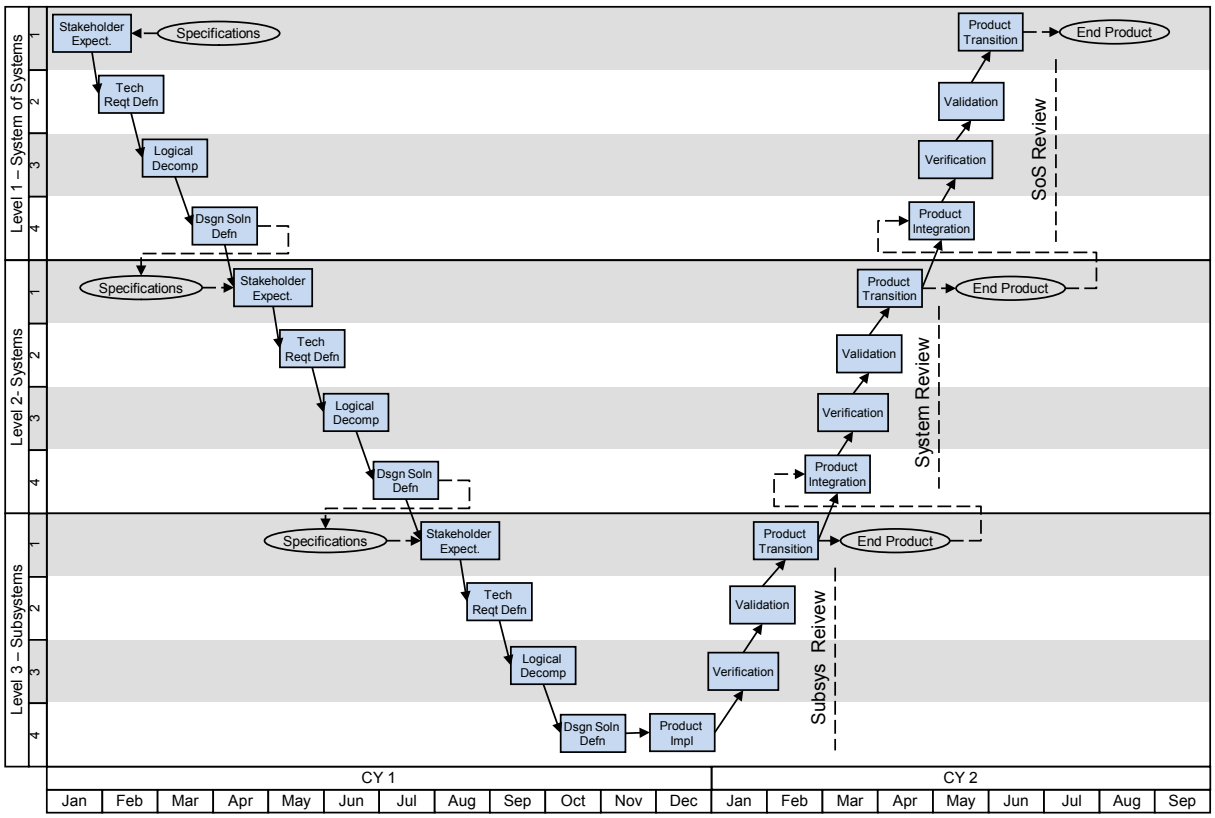

Figure 5. End-Product Chaining - Latest Possible.

End products of a lower tier are connected to the integration function of the next-higher tier. This and subsequent figures use an abbreviated form of the notation in which most of the products and the multiplicity of lower-level activities are suppressed for clarity .

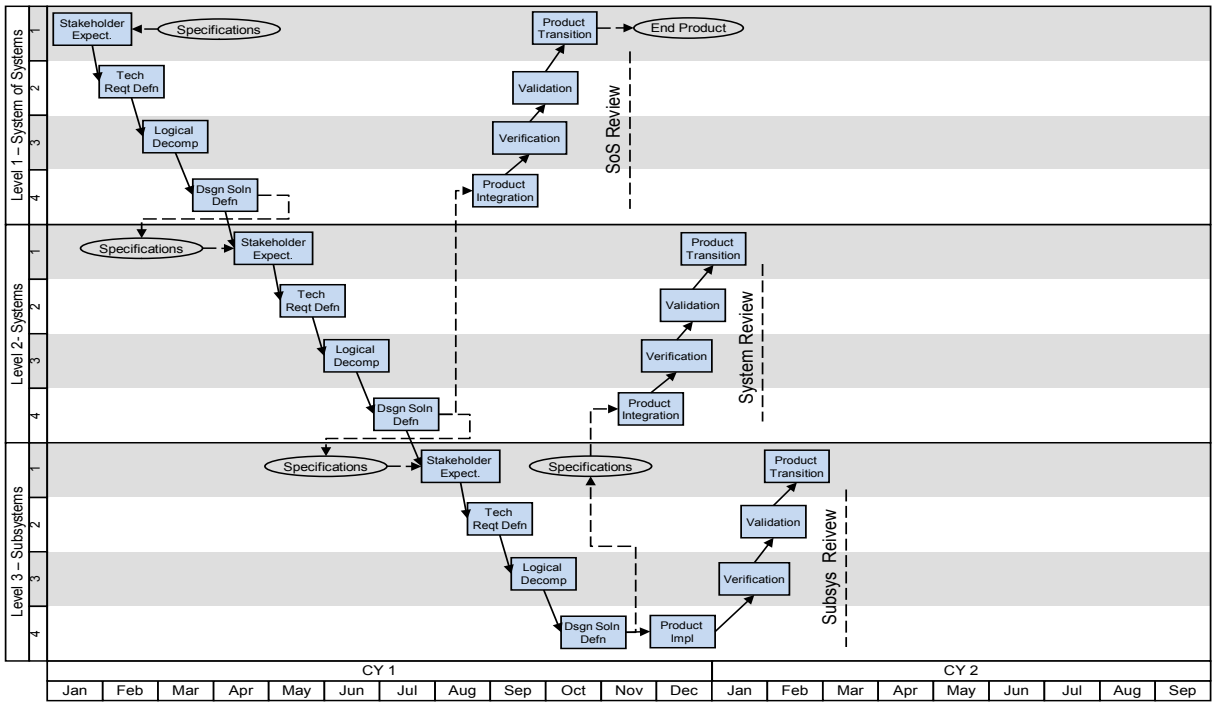

Figure 6. Final-Design Chaining.
Final design specifications resulting from a lower tier are connected to the integration function of the next higher tier.

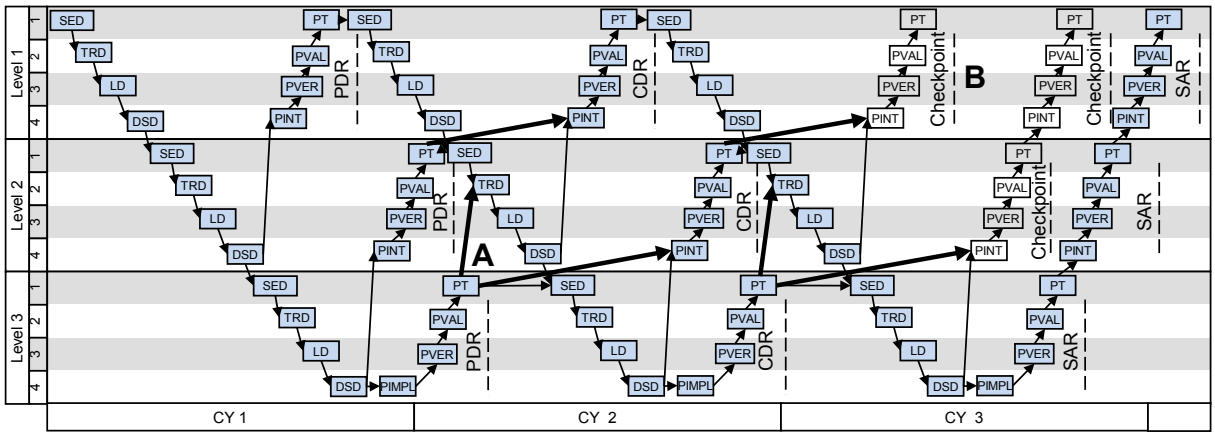

Figure 7. Resolution of Dangling Products.
Products left unresolved at the end of a single iteration are connected to the next iteration at the appropriate level, as indicated by the bold arrows. Point A marks a situation where information can be forwarded to a higher-level design process sooner than the actual product can be forwarded to the corresponding product-integration process. A complete end-product-chained cycle is required at the end to fully resolve the final end product; point B marks optional integrations and checkpoints that may be enabled in the final cycle if desired. Products are completely suppressed in this diagram, but may be assumed to be the same as in Figure 5.

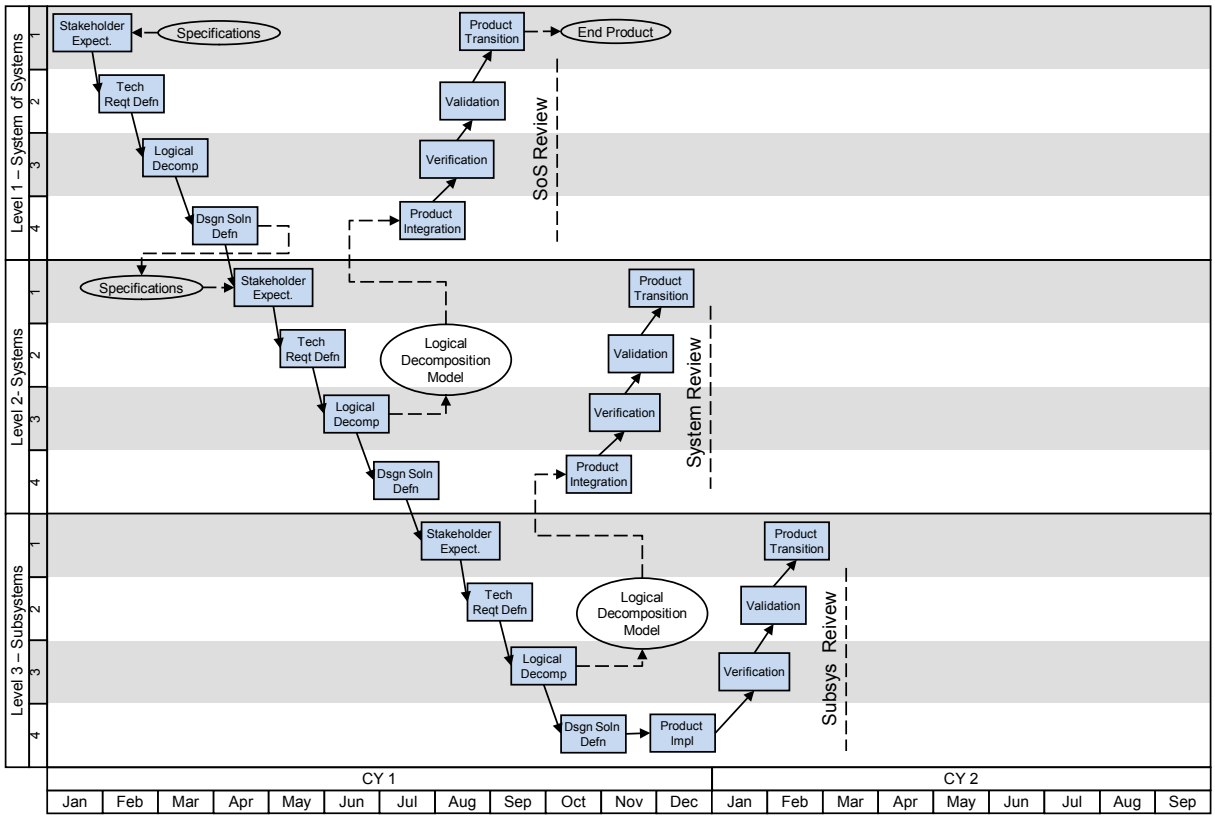

Figure 8. Product-Model Chaining.
Product Models resulting from a lower tier are connected to the integration function of the next higher tier.

C. Product-Model Chaining

The next strategy for chaining is shown in Figure 8, where the product model of a lower-tier effort is connected to the integration process of the next higher layer. In this strategy, the higher layer completes its system design activity, triggers the next lower layer to begin its system design activity, which then triggers the higher integration activity *as soon as a product model is available*.

D. Technical-Requirements-Chaining - Earliest Possible

The final, and most aggressive, strategy for chaining is shown in Figure 9, where the technical requirements of a lower-tier effort are connected to the integration process of the next higher layer. In this strategy, the higher layer completes its system design activity, triggers the next lower layer to begin its system design activity, which then triggers the higher integration activity *as soon as technical requirements are available*.

VI. Costs, Durations, Opportunities, and Risks of Chaining Options

We will now examine the ramifications of the two extremes of the chaining strategies with respect to costs, durations, opportunities, and risks. These ramifications are summarized in Table 1. At the present time we are not aware of any qualitative differences (other than degree of the ramifications) for the intermediate options.

A. Costs

To a first approximation, the chaining options have the same cost per iteration of the SE cycle. This is because the chaining options differ only in the timing and connection of product exchanges, absent any adjustments that might be made to capitalize on (or recover from) opportunity or risk factors described later. There may be some additional costs if the option to have extra integrations and checkpoints (point B in Figure 7) is exercised. These costs would depend on strategy in the following way: the earlier the return to the higher-level integration process, the more steps must be repeated at the end to resolve dangling products; and the more layers or more subsystems per layer, the greater would be the cost of the optional integrations. If many iterations are planned, or the system has few layers, this might not be a significant penalty compared to the cost of the overall effort; if only a single iteration is planned, or the system has a large number of layers, the penalty is proportionally more serious.

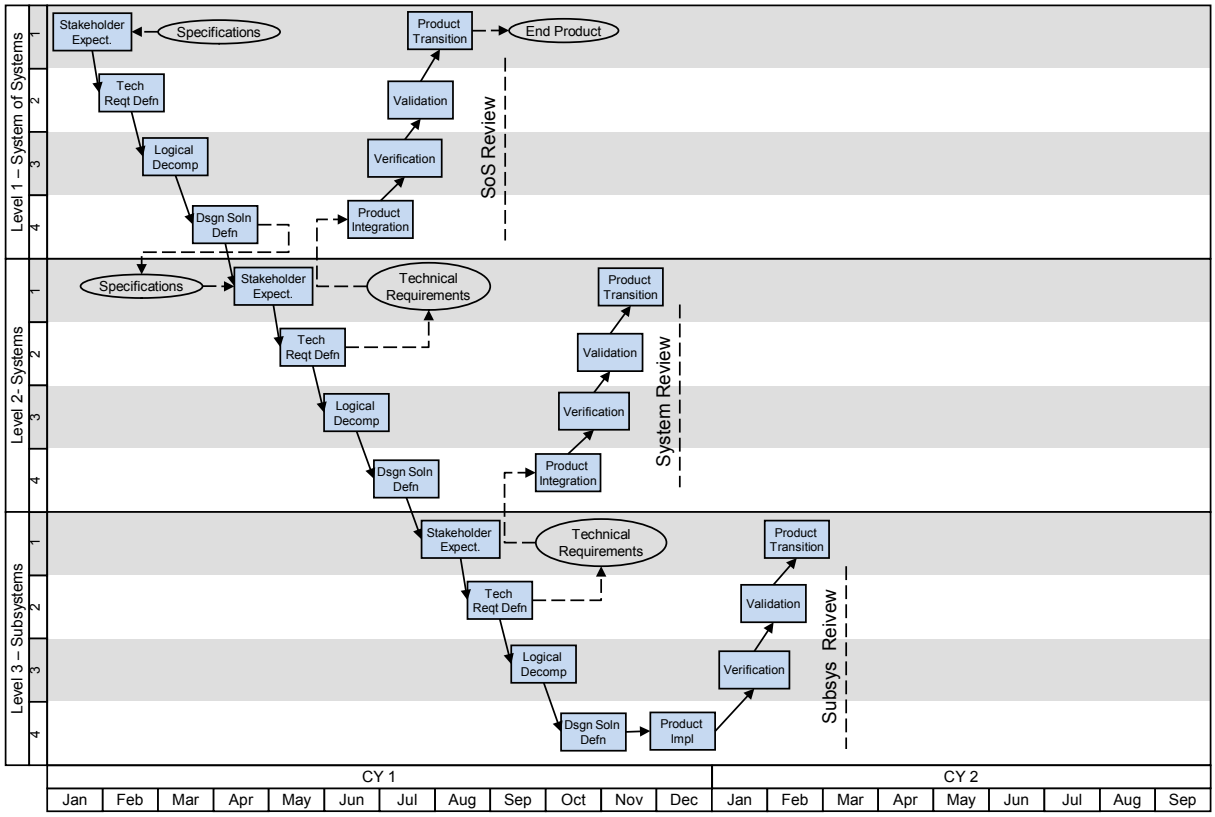

Figure 9. Technical-Requirements Chaining - Earliest Possible.
Technical requirements resulting from a lower tier are connected to the integration function of the next higher tier.

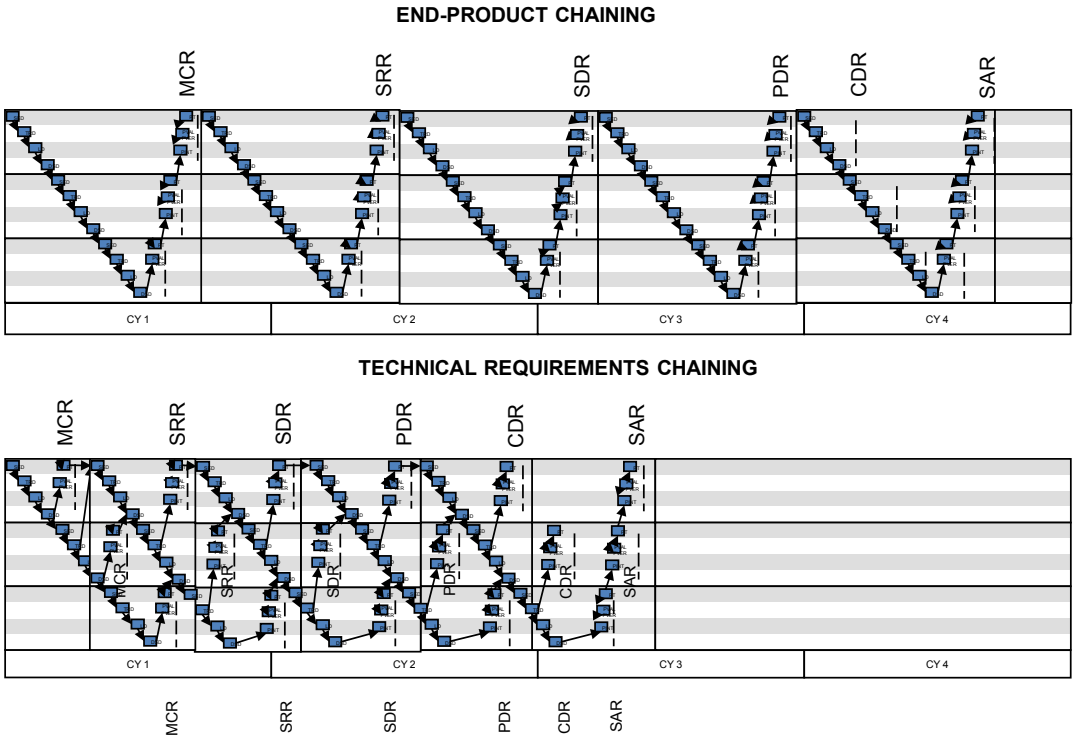

Figure 10. Comparison of Durations between End-Product and Technical-Requirements Chaining. *The specified iterations of the NASA SE process are used, with a final end-product-chained cycle to resolve dangling products.*

B. Durations

The chaining options with the earlier returns to the higher-level process have a shorter duration per iteration for all but the last iteration, but the need to resolve dangling products requires least one final end-product chain at the end and therefore prevents the shorter duration from being used there. If there are many iterations, the advantage of shorter iterations is multiplied by the number of non-terminal iterations. This situation is depicted in Figure 10, which compares end-product and technical requirements chaining over the specified iterations of the NASA SE process: mission concept review (MCR), system requirements review (SRR), system definition review (SDR), preliminary design review (PDR), critical design review (CDR), and system acceptance review (SAR). It is readily seen that all other durations being equal, preliminary-design chaining -- the fastest product development cycle -- is faster by 1.5 to 2 iterations, or about 30%. The largest improvement results from the step from end-product to final-design chaining; incremental improvements from more aggressive chaining are roughly one-fourth as large within the NASA SE process.

C. Opportunities

End-product chaining provides simple engineering operations, with only one layer of activity operating at a given time. This feature limits the management resources needed to coordinate the layers of SE effort. End-product chaining has potentially lower peak staff requirements if personnel can move between layers; or, personnel can be released to other activities during the dormant periods of the higher layers.

Technical-requirements chaining provides the fastest possible detection of errors in specification at higher layers, because the impact of those specifications is quickly fed back to the higher layer after they are analyzed by the lower layer. Technical-requirements chaining also provides the earliest possible validation of the higher-layer end products. This provides the lowest risk of mistargeting due to either misunderstanding of, or changes in, stakeholder needs.

D. Risks

End-product chaining has the longest time until the first validation occurs. Therefore there is increased risk of mistargeting of the entire system effort due to misunderstandings or stakeholder need changes of stakeholder needs. In fact, the longest delay applies to the highest-risk product: the overall SoS. End-product chaining also exhibits potential 'bathtubs' of employment if personnel cannot move between layers, or cannot be released and retrieved from other activities. This chaining option also has a longer time to detect errors in a lower-layer's implementations.

Technical-requirements chaining engenders an increased risk because early products are integrated. These early products are inherently approximations to the end product and therefore have lower authenticity or technical resolution. Lower authenticity is only an issue in the early iterations; the final iteration should always be an end-product-chained one which is inherently free from the issue. This risk has to be balanced against the risks of going without the early, approximate information. Also, since activity is continuous at all layers, employment can be steady in all layers.

Any of the chaining options other than end-product requires the organization to work and think in many layers simultaneously. This may be an insurmountable hurdle for organizations lacking practices or sufficient numbers of personnel able to address the parallelism involved in the more aggressive chaining options. Also, the technical management processes (such as change management, or technical assessment) must perform to a much faster tempo than would be permissible for end-product chaining.

Table 1. Comparison of Chaining Options.

Chaining Strategy	Costs	Durations	Opportunities	Risks
End Product	Normal	Longest	Simple engineering operations Lower peak staff if personnel can move between layers	Longest time to first validation. Bathtubs of employment. Longest time to detect lower-level implementation errors.
Technical Requirements	Normal	Shortest if there are many iteration	Fastest detection of errors. Earliest validation of end product.	Lower authenticity of early products (correctable by the last iteration). Requires organization to work on many layers simultaneously.

VII. Conclusion

We developed a new notation for describing systems engineering activity. The value of this novel notation is that it provides a way to unambiguously relate activities, time, level of system decomposition, process, and product relationships. This helps systems engineering managers to express and understand the relationships between many systems engineering engines, a task which normally is a formidable challenge. The particular representation selected is very easy to translate to schedules, costing, work agreements, and decision-making activities. Through a set of elementary examples we showed that the notation also makes it possible to compare alternative chaining strategies, and that the chaining strategy can influence the overall performance of the organization with respect to durations, opportunities, and risk. Duration can be improved by about 30% of the total duration if there are many iterations. To first order, there is no difference as to total cost. The more aggressive chaining options have lower risk of misunderstanding or mistargeting, lower risk of early implementation errors, but higher risk of early but temporary errors. Opportunities afforded by the more aggressive options are earlier detection of implementation errors, and earlier validation of the end product.

Acknowledgments

The research described in this publication was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration. J. C. Breidenthal would like to thank David Petri and Jeff Schutter at the Johnson Space Center, Terry Morris at the Langley Research Center, and Dan Dvorak of the Jet Propulsion Laboratory for helpful discussion in development of this paper.

References

[1]De Marco, T., *Structured Analysis and System Specification*, Fig. 1, Prentice-Hall, New Jersey, 1978.

[2]Ibid., Fig. 99.

[3]International Organization for Standardization, "Information processing -- Documentation symbols and conventions for data, program and system flowcharts, program network charts and system resources charts," ISO 5807:1985, Geneva, Switzerland, 1985.

[4] Object Management Group, "Unified Modeling Language Specification Version 1.4.2," Part 10, 2005, URL: http://www.omg.org/cgi-bin/doc?formal/05-04-01 [cited 9 March 2009]; also available from International Organization for Standardization as "Information technology -- Open Distributed Processing -- Unified Modeling Language (UML) Version 1.4.2," ISO/IEC 19501:2005, Geneva, Switzerland, 2005.

[5]Ibid., Part 7.

[6]Gantt, H. L., *Work, Wages and Profit*, published by *The Engineering Magazine*, New York, 1910; republished as *Work, Wages and Profits*, Easton, Pennsylvania, Hive Publishing Company, 1974.

[7]Kerzner, Harold, *Project Management: A Systems Approach to Planning, Scheduling, and Controlling (8th Ed.)*. Wiley, Hoboken, New Jersey, 2003.

[8] Forsberg, K., and Mooz, H., "The Relationship of Systems Engineering to the Project Cycle," *First Annual Symposium of the National Council On Systems Engineering (NCOSE)*, October, 1991; available at URL: http://www.csm.com/repository/model/rep/o/pdf/Relationship%20of%20SE%20to%20Proj%20Cycle.pdf [cited 11 March 2009].

[9]Office of the Chief Engineer, "NASA Systems Engineering Processes and Requirements," NASA NPR 7123.1A, Washington, DC, 2007.