Why Johnny Can't Write Requirements

by Ivy Hooks

Abstract

Program cost overruns and schedule delays are attributed, in large part, to problems associated with requirements. These problems range from incomplete, inconsistent, and incomprehensible requirements to the complexities of the change management process. These problems stem from an inability to write good requirements and a lack of understanding about the importance of requirements. They are compounded by the scope, complexity, and long life cycle of major programs. Education of the people involved and automation and integration of the requirements management process are needed to combat the problem.

The Problem with Requirements

Requirements problems have been identified as major contributors to program cost overruns and schedule slips. This "problem with requirements" can be attributed to two major factors:

1. The initial requirements were incomplete, inaccurate, and/or misunderstood by the designers/developers.
2. Circumstances changed which resulted in changes to the requirements.

There are several reasons why problems exist with requirements:

1. The program total life cycle, including operations, was not well defined at the time that many of the requirements were written.
2. "Johnny can't write requirements"
3. Few people understand the requirements process.
4. Management has not focused sufficient attention on the problem.
5. Changes are required to correct poor assumptions or poorly written requirements or to correct for inconsistencies or mission requirements.
6. Changes are required in response to external factors over which the program has no control.

All of the reasons for requirements problems stated above contribute to operations costs and schedules. From an operations perspective, the problem is manifested by the fact that the requirements focus is often on what is to be built without consideration for how it will be operated. Operations costs are directly driven by program definition and the requirements imposed by all elements of the program. Frequently, the requirements do not consider the operations concepts, constraints, or plan. When operations requirements are finally considered, additional requirements will be needed to overcome this initial oversight.

The challenge to operations management is to:

1. Have the operations aspects of the program defined early in the program.
2. Have quality requirements written across the program - including the specific operations requirements.
3. Have an educated work force that understands the requirements process.
4. Increase management attention on the requirements process.

Operational Concepts

A prelude to defining requirements is to define the program objective and operational concepts. Standard specifications (functional requirements documents) provide a section to describe operational scenarios. In order to develop valid operational scenarios, experienced operations personnel must be involved at the beginning of the program. These people need to have the skills necessary to write realistic scenarios and functional level requirements.

Operations personnel must also be able to analyze other system requirements to identify drivers to operations. Consider the NASA Assured Crew Return Vehicle (ACRV). This vehicle is in the early stages of design. The Level A functional system requirements were developed with a team of personnel which included both Kennedy Space Center (KSC) launch operations personnel and Johnson Space Center (JSC) crew training and flight operations personnel. These groups added insight into the operational aspects of the requirements.

The contractors performing the design trade studies leading to segment requirements will be guided by the operational scenarios and the requirements defined in the Level A document. They should also have, as a part of their team, operations experts to provide guidance during their design and trade study activities.

Requirements based on operational concepts must flow down through the requirements documents. The Level A requirements specify that the vehicle will be checked out and installed in the shuttle Orbiter at KSC. They also specify the conditions for removing the crew from a return flight. The return vehicle could be designed for a water landing or a runway landing. The recovery operations will be quite different depending upon the design chosen. The recovery operations cost must be a serious consideration in the trade studies.

The segment requirements will include those for the flight segment, the ground segment, and the mission operations segment. Many of these requirements will be tightly coupled. How the flight vehicle (part of the flight segment) lands will be directly related to the recovery and refurbishment elements (part of the ground segment). Fully developed operational scenarios are needed prior to writing specific requirements for either element.

Only by integrating the operations expertise with the vehicle design expertise early in a program can operations costs and schedules be contained. Waiting until a vehicle is designed to consider how it will be operated will ensure added costs and longer schedules.

Why Johnny Can't Write Requirements

Any contractor who has tried to respond to a set of requirements knows the difficulty of the task. Many requirement documents contain statements that are not requirements, but are unverifiable goals or objectives. For example, "minimize costs" or "provide adequate margins" are statements of design goals or objectives and should be stated as such. Other statements found in requirements documents are actually statement of work items, such as, "perform trade studies". Some requirements are so grammatically incorrect that they defy interpretation. Conflict and inconsistency between requirements are not unusual. Sets of requirements are often incomplete.

The problem is that most engineers who are assigned the tasks of writing requirements do not know how to do the job. Colleges do not normally provide training in this aspect of engineering. Everyone gets "on-the-job" training, but often this is without any guidance. Most engineers assigned such a task simply obtain an existing requirement document and use it as an example. Often this example is flawed, so the engineer starts with bad information. Over several generations of documents, the problem will compound to the point that any resemblance to valid requirements is purely coincidental.

The reasons "Why Johnny Can't Write Requirements" are described in the following paragraphs.

He doesn't know what to do.

He doesn't understand how requirements fit into specifications. He looks for an existing specification for an example and gets bad examples.

He doesn't have related information that he needs. Even if he is writing requirements that respond to high-level functional requirements, he probably does not understand why those requirements exist or what trade studies and analysis took place to arrive at those requirements.

He cannot write. Engineers and computer scientists can obtain college degrees with little emphasis on writing. Asking them to write a concise and clear statement is asking them to perform a task beyond their skill levels.

He doesn't know what he wants. Since a requirement is a statement of a want or a need, not understanding what is needed or wanted guarantees bad requirements.

He doesn't understand why he should do it.

He doesn't understand the impact of the requirements. Those assigned to write requirements do not relate their requirements to cost, schedules, or resources that are driven by the requirements.

His attitude is "it's just documentation". This may also be management's attitude. Hence, the most knowledgeable people are not assigned to the tasks - they have too many important things to do and cannot be spared for this "documentation" effort.

He would rather be doing something else.

He would rather do design. There is a strong tendency to jump from the highest statement of functional requirements to a design, without further definition of intermediate levels of requirements. Design work is much more fun than writing requirements.

He doesn't have enough time and he believes the review process will catch the problem. He really does not have enough time to do a good job - there is always a tight deadline. Unfortunately, the reviewers may not have enough time either, and the right people may not participate in the review process.

He sees no reward.

He doesn't care. No matter what requirements he writes, he will be battered during reviews by those who could not or would not take the time to participate in writing requirements. There is no reward for doing a good job.

All the automation is the world will not solve some of these problems, but automation and integration can provide a means to solve many of them.

The Requirement Management Process

The requirement management process is not "just a front-end" to building a system, it is a life-cycle process.

Table 1 describes the major steps in the requirements management process.

Requirement Definition begins with an overall objective and operational concepts and scenarios. With this information, requirements can be developed. These requirements will flow down into the next level of requirements, shown in the table as segments. The flow down continues and at each level design analysis, trade studies, and operational scenarios are performed to support the development of the requirements.

Requirement Verification should begin with the definition of the first requirement. How the requirement will be verified should be understood. That the requirement is verifiable is mandatory. It should be verified that all requirements above have been flowed down to the appropriate next level as each level of requirement is developed. It should be verified that no "gold plating" has crept into the requirements at lower levels. ("Gold plating" is the addition of requirements at levels that are not in response to higher-level requirements, but are requirements for something that would be nice to have.)
As the levels of requirements are defined, the verification processes must also be defined. At the highest level, integrated test and analysis may be the methods of verification, whereas at the parts level, benchmark testing may be the process used. Finally, the verification process must certify that all requirements are met.

Requirement Change Management must be performed at all levels. Change impact assessment must consider not only the obvious impact of the change to the system against which it is proposed but also to any other systems that are related. It must consider the impact of the

change to all lower level requirements. As a program develops, the impact to change must consider the impact to verification and documentation as well. The cost of a change is not just the cost of altering the design; it encompasses the change to the design documentation, to any test plans, and to operational procedures.

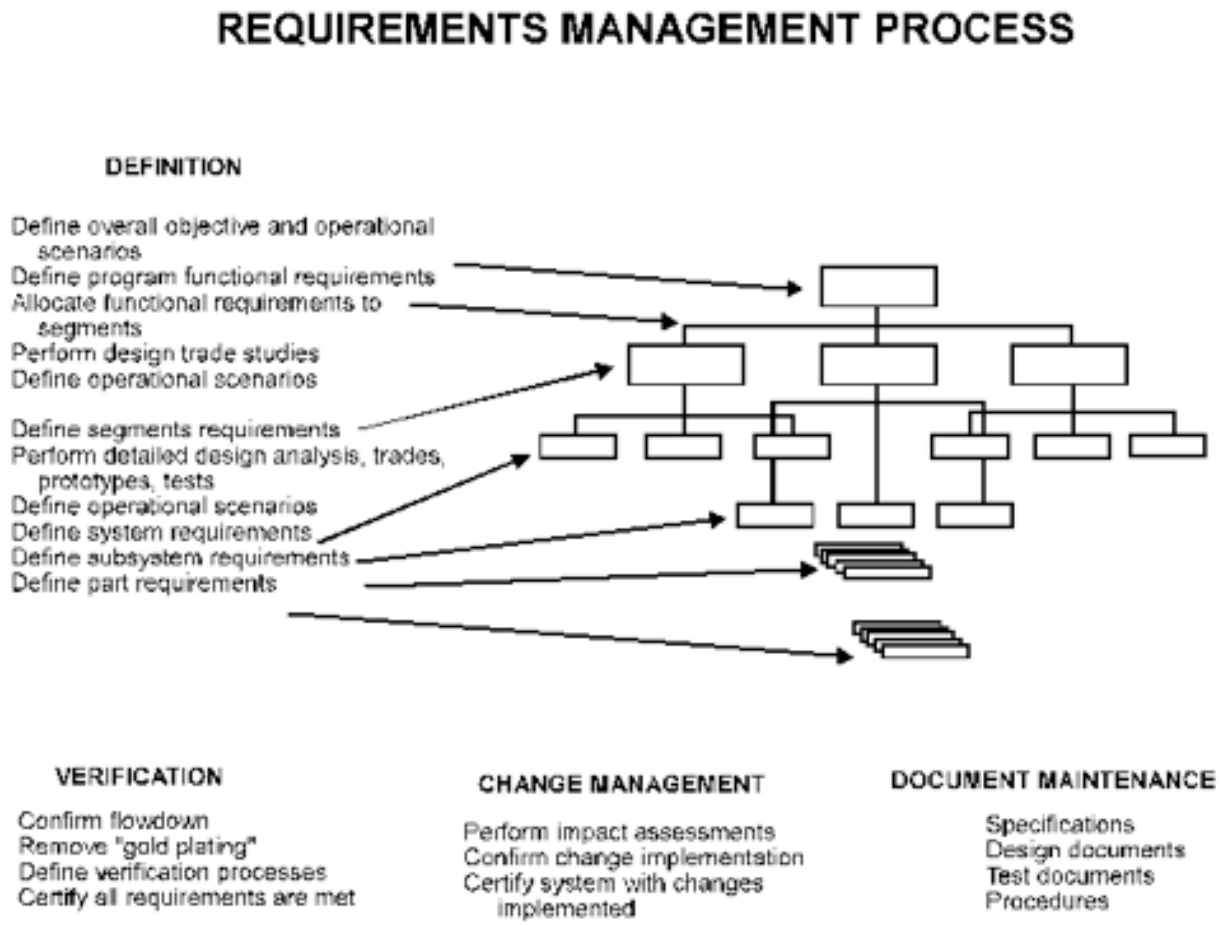

Table 1. The Management Requirements Process

Change management must also assure that the change is implemented - in the specifications, design, and in any and all documentation. And change management must ensure that the verification processes have been put in place to certify that the implemented change works properly and that the system, with the change incorporated, meets all of its other requirements.

Requirement Document Maintenance must be responsive to the initial requirement definition process and to the change process. The requirements documents need to reflect the current approved baseline and changes. Design and other documents such as computer software code, need to be updated in a timely manner. Test documents (plans, procedures, processes) must track the requirement baseline.

Real-World Example

The requirement management process has more meaning if viewed from a real-world example. The example is from the Space Station Program - a very large and very complex program, within which reside requirements for many sub-programs requiring many types of hardware, software, training, and operations.

From Johnny's Viewpoint

Figure 1 [page 7] illustrates a small portion of the requirements documents (specification) that must be generated for the Space Station program. (*This figure is for illustration purposes and does not represent actual Space Station Program division of documents.*) The documents are denoted by boxes. Arrows denote flow down of requirements or flow of information that drives requirements.

For purposes of illustration, we have identified a Launch Processing Center (LPC). The LPC would be responsible for launch related activities for all space station hardware, consumables, and payloads. The responsibilities would include receiving hardware and payloads, launch processing, and installation in the Orbiter for delivery. It would include maintaining consumable supplies, packaging consumables for delivery, and installing these in the Orbiter for delivery. It would also include removing, processing, and shipping hardware, consumable containers, and payloads returned by the Shuttle from the Space Station.

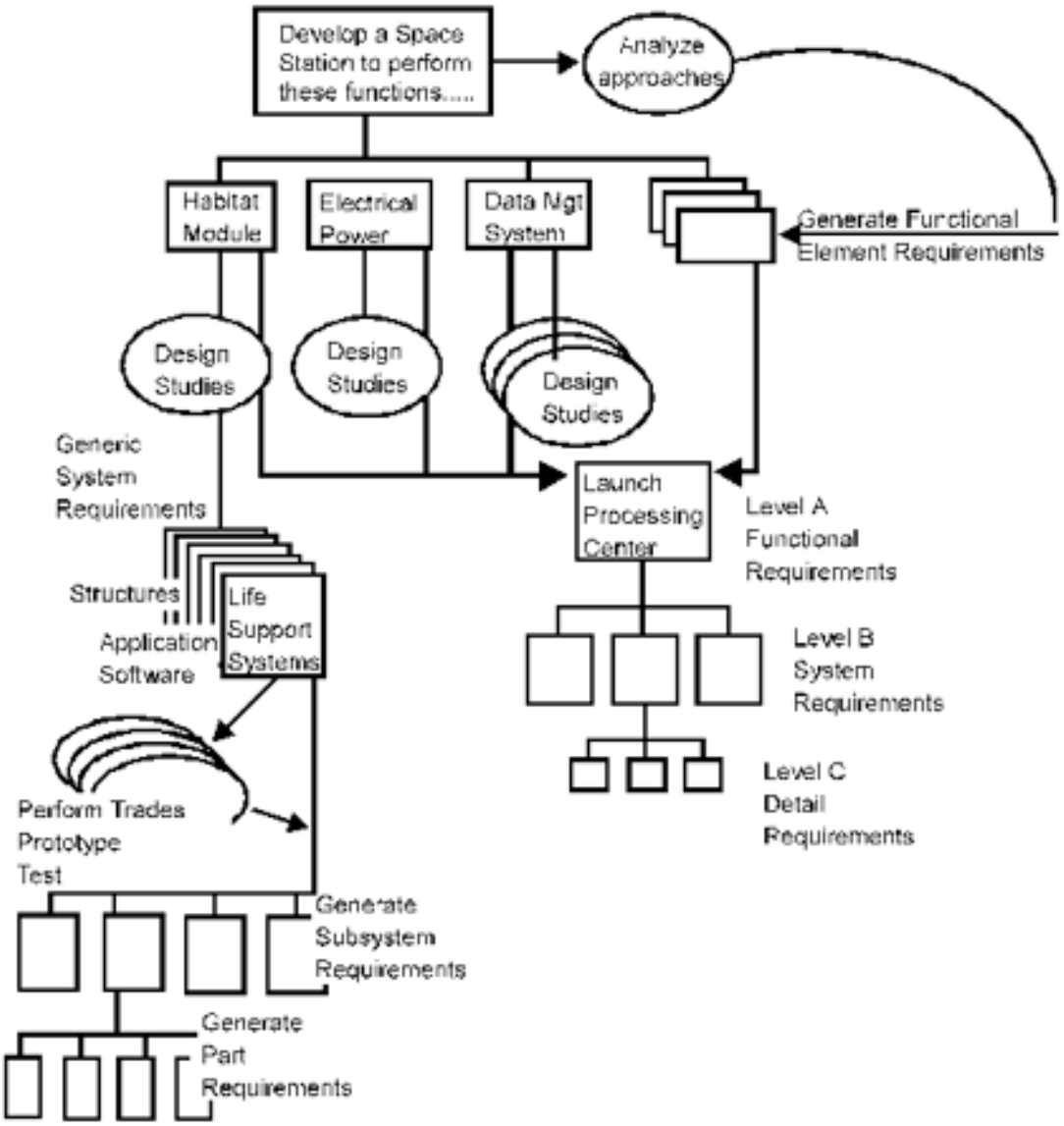

Figure 1. Requirements Development Flowdown

The LPC functional requirements (shown in the center of Figure 1) are subservient to both the Space Station functional requirements and the Element functional requirements. The Space Station requirements include those that will require replacement units and consumables.

Each Space Station element has its own unique set of requirements. The LPC requirements must take into account the design of these systems in order to be able to interface with the flight system.

Also, note that the LPC requirements are "system" level and hence will drive lower levels of requirements to define the facilities, ground equipment, hardware, and software that comprise the system. The total system includes: facilities, transportation, ground equipment, power,

checkout equipment, computer hardware, and software. The requirements must reflect safety and security considerations.

To develop the requirements for the LPC requires an understanding of the Space Station architecture and of the operational concepts for launch processing. It requires an approach that combines knowledge of past launch processing operations and designed with new concepts and new technology.

Figure 2 illustrates the challenge of writing the LPC functional requirements. The author of a set of these requirements must be knowledgeable about those higher-level requirements that affect his design. He also needs to know more - where does this parent requirement fit into the overall architecture of the Space Station, what is the rationale behind the requirement, and what is the status of this requirement - is it firm or is it likely to change.

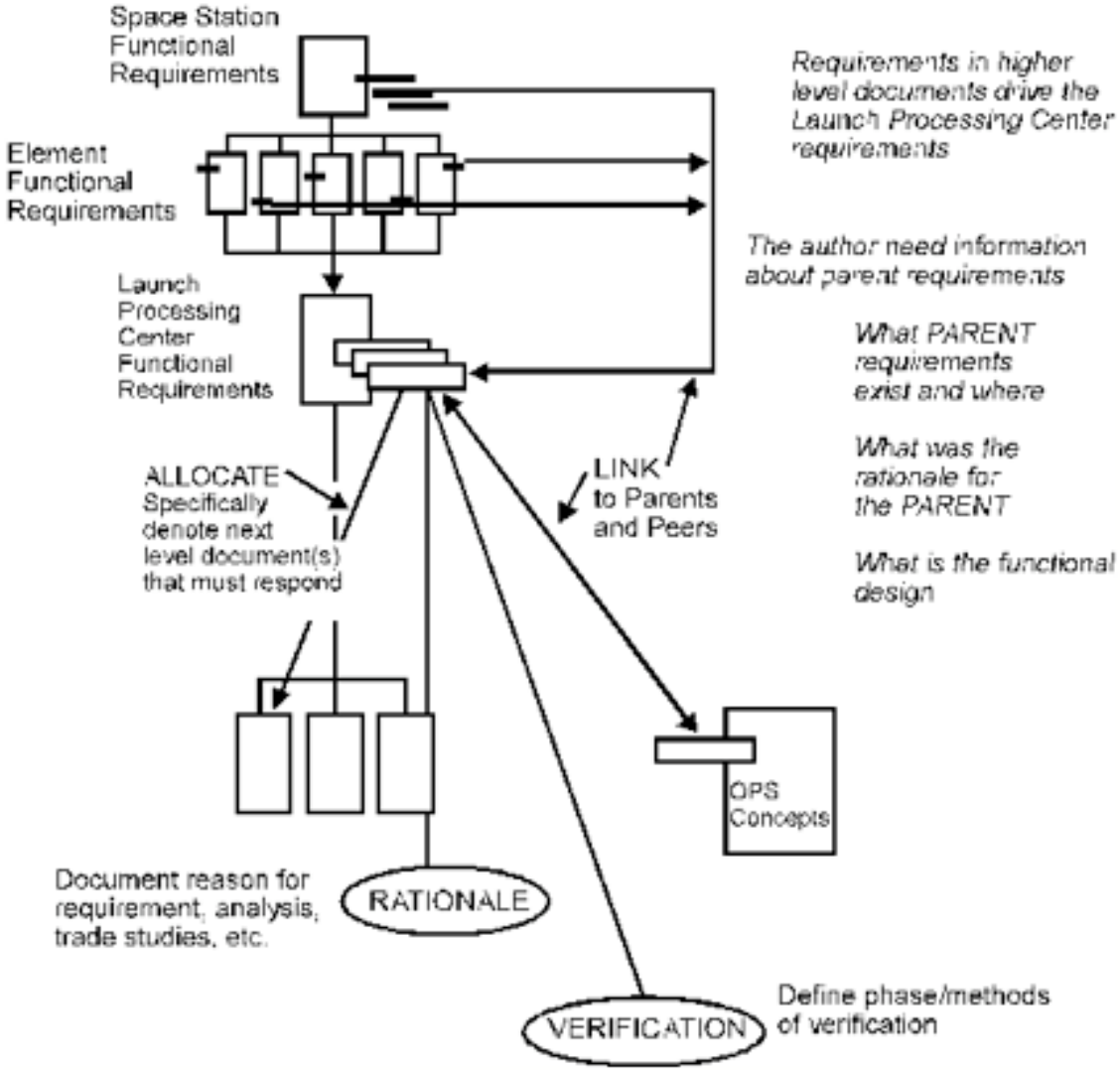

Figure 2. Information related to requirements.

As the author develops his requirements, he needs to link them back to their parents so that he can be notified if a parent requirement changes.

He needs to allocate his requirement to the next level. The specification that contains his requirement should clearly define the next level of documentation - such as facilities, hardware, and software. The author of the requirement should possess the best insight into which of these are to respond to his requirement. By allocating, he can provide the information to the authors at the next level, rather than have them guess at what applies.

The author also needs to define the rationale for each of his requirements. This information provides a "corporate history" for use in future change impact analyses and for educating those persons who will enter the program at a later time.

Other documents may exist which contain related information, such as an Operational Concepts document. By linking his requirements to the correct sections of the Operational Concepts document, he can be alerted if the concepts change. He may also link to other sections in other documents for the same traceability.

For a requirement to be valid, it must be verifiable. Specifying the phase and methods of verification is integral to defining requirements.

From Management's Viewpoint

Management is interested in how much the system is going to cost, how long it is going to take to build, what resources will be needed to build and maintain it, and will it work when it is delivered. Each of these items depends upon the requirements.

If the Space Station requirement is "to be able to receive, process, and deliver payloads", then the LPC must develop a system that can do this. Exactly, *how much* and *what* will be defined by a high level organization representing the users. Suppose the requirement is "20 payloads of type A, B, and C for each launch". This will drive requirements for many subsystems - facilities, transportation, ground processing equipment, check-out equipment, and even the management information system.

Management needs a view of the lower level requirements that are derived from this user-defined requirement. Management needs information about how these requirements are driving the design of the system, and the design sensitivity to the payload types, mix, and launch rate. This is illustrated in Figure 3.

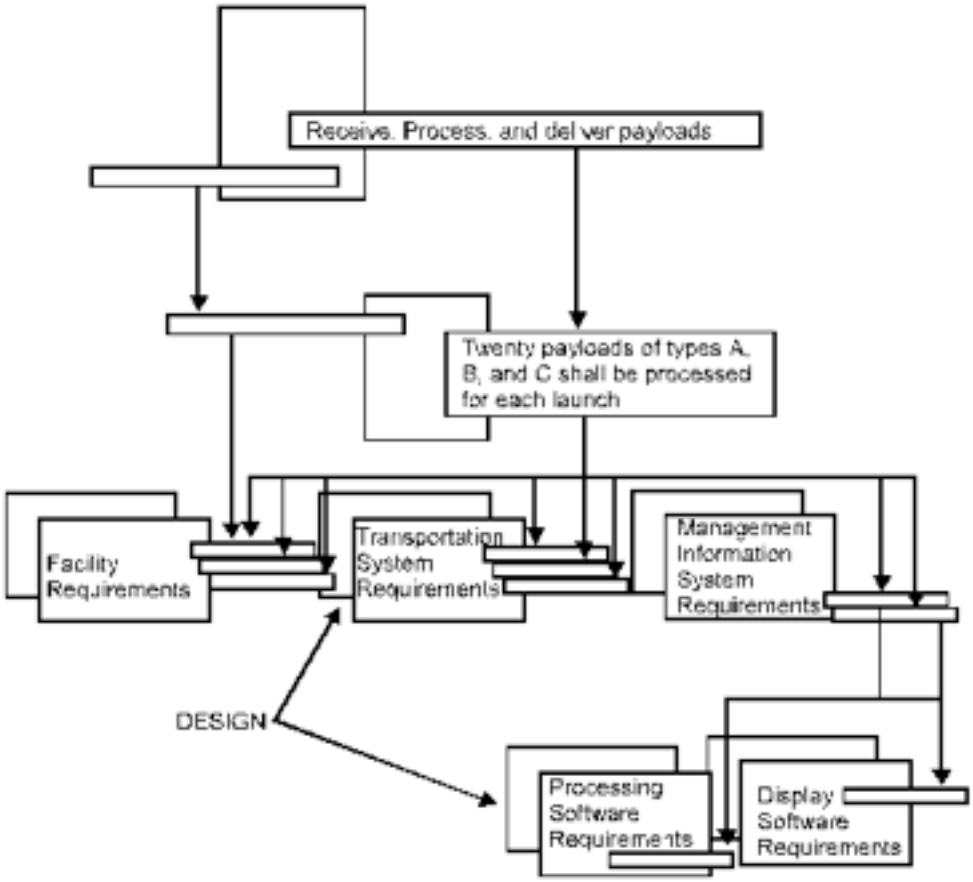

Figure 3. Requirements must be traced between levels.

This figure shows how the higher-level payload requirements have flowed to the subsystems and their design. Also, shown are other requirements that are related to the lower level requirements. If a change to the number of payloads is proposed, then all the subsystem requirements that may change must be viewed with respect to other driving requirements. In this example, we have shown other parent requirements related to a Facilities requirement (shadowed). The proposed change may not be a problem. Or, it may not be possible to make the change due to the other higher-level requirement. Management needs this interrelated information and only management can resolve the issues.

Management also needs to assure that all requirements are verified. The requirements definition process and the verification process are illustrated in Figure 4. The term "test" is used at each step, but verification may be in other forms, such as analysis, simulation, or inspection.

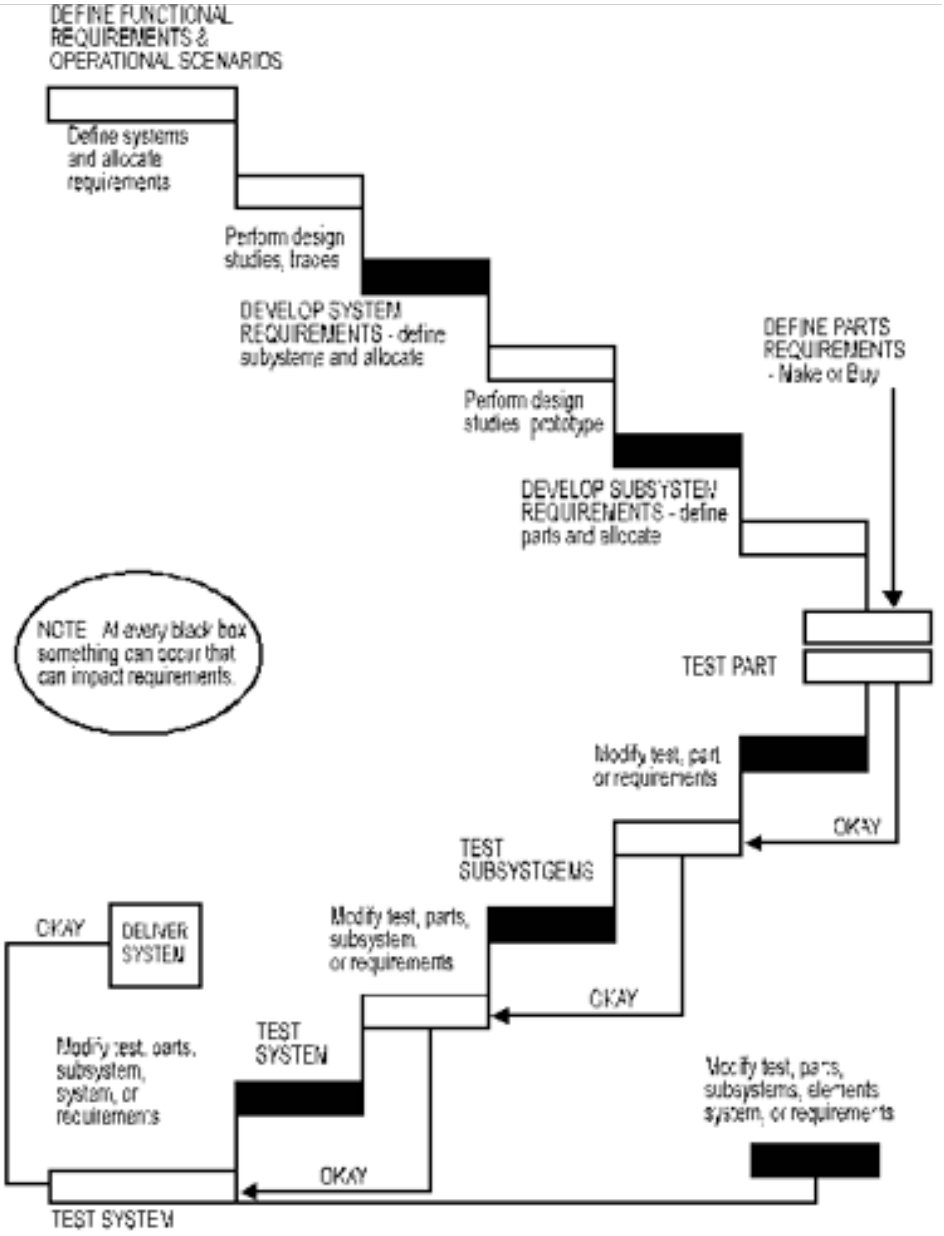

Figure 4. Requirements / Verification

At each level of design and verification there is the possibility that requirements will change. Change may be driven by a number of factors, such as, requirements exceed budget or requirements are incompatible between two parts that must interface. Again, the relationship between all requirements must be known so that a change to fix one problem will not result in many more problems.

Managers are very concerned with change. Change is certain. Change will result in higher costs and longer schedules. To understand the impact of change, a manager needs to know everything that is impacted - other requirements, design, test cases, and documentation. Figure

5 shows how the change to one requirement can impact numerous documents and related activities.

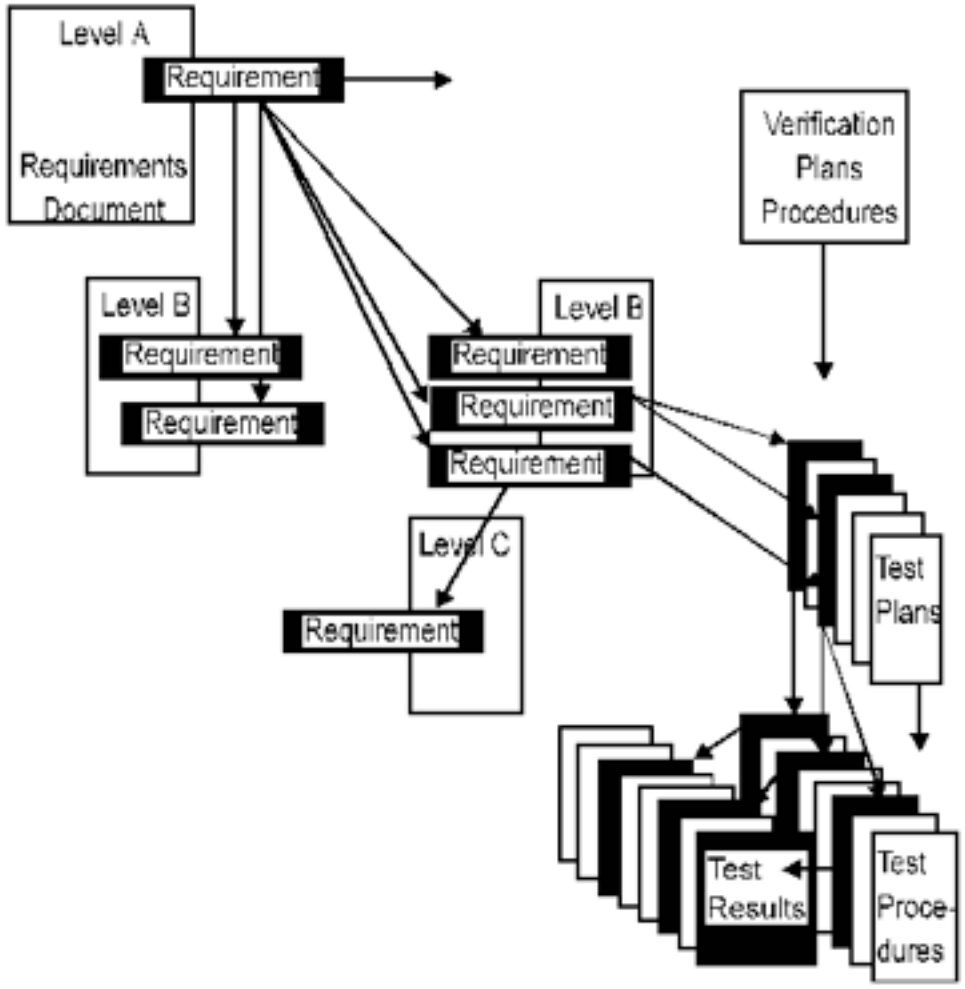

Figure 5. Effect of Change.

If the information is linked, them impact assessments can be made throughout the system. Otherwise, management may find that it has approved a change that was supposed to "save" $100,000 but will actually result in "costs" of millions.

Current Environment

To a large extent, the requirements management process is the same as it was 25 years ago. Automation has been applied only to automate a paper process - using a word processor instead of a typewriter; or putting a matrix into a spreadsheet that was once hand-written or typed. Little has been done to integrate the process using the technologies that are now available.

In the Department of Defense (DoD) and other government agencies, there is concern for requirements traceability. Aerospace companies have faced this traceability problem for many years. The solution that many have adopted is to create their specifications on word processors or in text publishing tools and in parallel, develop databases to handle requirements.

Verification has always been an issue and is generally handled by creating a matrix of requirements versus phase and method of testing and to use clipboards to check off that testing is complete. This process may be minimally automated by placing the data in a database or a spreadsheet. Whenever a requirement is changed, this information must also be changed.

There is now concern about capturing a "corporate history" in order to maintain the systems over the life of the project. This information may be captured in many forms, but is frequently separate from the basic requirement information.

This approach results in requirements that are documented in text, database, and spreadsheet programs. Thus, multiple sets of information must be maintained identically - a nearly impossible task and much of the data Johnny or management needs is not readily available. Moreover, Johnny will be handed large quantities of documents containing vast quantities of data - some related to him and much more not related to him. He will be overwhelmed.

Management must wait long periods of time to get answers for its questions concerning completeness or change impact and may never get all of the data.

Unless a single integrated set of information is maintained the following problems can occur:

1. Developers looking at a requirements document may not be viewing the latest version. They will be building something that has already been officially changed.
2. Those performing verification may have out-of-date information. Their tests will not include all items that need to be tested and will have to be repeated.
3. Much of the documentation will be inconsistent because of time lags in determining that updates have been implemented in one document, but have not found their way into related documents. Persons using multiple documents will waste a great deal of time trying to determine what is "right",

Helping Johnny Write Better Requirements

Automation cannot cure all of the problems stated above. In fact, automation without management commitment and a disciplined process cannot cure any of the problems.

What to do

If we can provide Johnny with more information about what he is to do, he will be better prepared to write requirements. Automation can provide Johnny with on-line information so he will have a better idea of what to do.

On-line outlines and help. Provide Johnny with an outline for his document and instructions, so that at any step in that outline he can get other information about what belongs in the sections. This will work better than giving him a book and telling him to read it. In this automated process,

he only looks at the part related to his task and is taken into details related to his task - the process is less overwhelming.

Examples. Provide Johnny with examples of good requirements of the types that go into certain sections of specifications and make these accessible from the outline as "examples of good requirements". Other tutorial data about why this is good and where it is applicable could also be added.

Access to related data. Provide Johnny on-line access to his parent requirements (those from higher level document(s)) so that he does not have to plow through a large document trying to interpret what applies to him. Provide him access to other information about the parent - its rationale, its verification, and its other children to peers - so that he has the knowledge of the parent requirement author.

Writing assistance. Provide Johnny with a spell checker and expert systems to check grammar and to check "the quality of the requirement".

Attach other data. Provide Johnny an easy means to attach other information to his requirement. Let him document his rationale, so that even if his requirement is not clear, a better writer may be able to correct it based on his rationale.

Why do it

If all Johnny sees is a set of big documents, he feels no ownership and he has no perspective of how a requirement that he writes fits into the system. If we can give Johnny a different perspective than "only a document" we may motivate him to do a better job of writing requirements. Automation can link information and display information from many perspectives.

Different perspective. Provide Johnny with different views of his requirements - not just as text in a document, but also as a requirement linked to other requirements - parents, children, peers. Let him see his requirement as part of a set of requirements to be verified. Let him see his requirement as part of a product or a delivery of a system.

Ownership. Attach Johnny's name to the requirements. Give him a sense of responsibility and make his work visible to management and his peers.

Impact. Provide Johnny with information that shows how his requirement related to cost and schedules.

Life-Cycle Effects

Johnny is interested in doing his job, not writing requirements. If he can better see the relationship between his job and the requirements, he will be more motivated to get the requirements right.

Design. Provide a means to link the requirements to the design so that he can see the flow from requirements to design.

Verification. Provide a means to link the requirements to test plans so that he can see the flow to verification.

Change Impact. Provide a means to identify all requirements linked to a requirement that is to change. Provide a means to view the design and test plan that may be impacted by change to the requirement.

Assisting Management

By helping Johnny write better requirements, we are assisting management. By linking requirements and providing traceability, we are giving management the information needed to assess change impacts. By providing rationale, we are providing management with a means to train new personnel.

Integration of verification with the requirements will provide management with the ability to plan and control the verification process and reduce some of the overhead costs.

Providing automated and integrated change management and configuration management will provide management with the ability to reflect change across multiple documents and will ensure that the correct change is implemented.

Providing management with a means to tag requirements as to the source, implementation, and other related information will allow sorting of requirements into manageable groups. Providing analysis tools to "count" the occurrences of requirements and associated information will enable managers to focus on problem areas.

Automation and integration of the requirements management process can greatly facilitate management visibility and control.

Summary

The requirements management process is a large and complex job. Lack of control and visibility can result in projects overrunning costs and slipping schedules. Automation and integration are essential to improving the effectiveness of both writing requirements and managing the process.

Education and training about the importance of the process and the relationship of requirements to the program life cycle are needed. Involvement of operations personnel is critical throughout the requirements definition phase and in change impact assessments.

About the Author, Ivy Hooks

The CEO and a founder of Compliance Automation Inc., Ivy Hooks is an internationally recognized expert on the subjects of requirements engineering and requirement management. Ivy has published papers in a number of journals and proceedings and has spoken to diverse audiences worldwide. She has also provided training and consulting to multiple government and commercial organizations for over 20 years.

Ivy's NASA career took her from the Apollo program through the Space Shuttle Program. She was a member of the initial Shuttle design team and managed the development of the separation systems and the verification of the flight software.

You might also find interesting:

- Ivy's August 2004 article in *Crosstalk, The Journal of Defense Software Engineering:* "Managing Requirements for a System of Systems"
- Ivy's book: *Customer-Centered Products: Creating Successful Products Through Smart Requirements Management*