

# Taming the Expanding System Complexity of a Commercial Product: Embracing Systems Engineering Principles in a Commercial Development Environment

Paul Miller,  
Fujitsu Australia Limited,  
607 St Kilda Road,  
Melbourne, Victoria, 3004,  
AUSTRALIA.

**Abstract.** The development momentum of complex commercial products are often choked by an interminable cycle of test and corrective action that devours and overwhelms valuable resources. The root cause of this class of problem is that traditional commercial development environments do not make use of sound Systems Engineering principles as their products mature into highly complex systems. This paper provides an account of how Fujitsu Australia's Telecommunications Research and Development (R&D) centre is exploiting the years of experience gained by the Aerospace and Defense industries, and adapting their highly tuned and proven Systems Engineering practices into the commercial development environment. The centre piece of this paper is the adopted generic Systems Engineering Model that has become the central focus of development capability maturation within the R&D centre.

## INTRODUCTION

The world is well and truly gripped by the Telecommunications revolution. With deregulation occurring throughout the world and new market opportunities opening up in the Asia Pacific area, the competition between suppliers is at it's most aggressive at a time when the complexity of their systems are at their greatest. The commercial pressures of meeting time to market are truly immense and must be achieved in order to consolidate market position.

Pushing against this pressure are development times that are increasing disproportionately in relation to the amount of expected effort required to include and verify new functionality. The area where the time seems to overrun the most is during the test and corrective action phase.

A prime reason for this is that the traditional fast-track processes being used within commercial development environments are no longer fit for the purpose of maintaining control over a product that is continually expanding in system complexity every

time new features are added. These levels of complexity are reaching those that have been commonly experienced in the Aerospace and Defence industries for decades. Commercial development environments need to exploit the lessons learned from these industries and tailor the principles of Systems Engineering into their development processes.

## THE PROBLEM DOMAIN

The development of a new product often starts out with a small development team within an environment suited to support small projects developing products of low to modest complexity and risk. Such projects can afford to be managed through a combination of good project experience, familiarity with what worked and didn't work in previous projects and positive team dynamics such as good morale and good aural communications. In small development teams, the responsibility of each team member tends to cross many functional lines, thus reducing potential human interpretation errors that can occur when individuals work within specialist functional areas, such as project management, hardware, software, test or customer support.

As the features of a product begin to expand to meet the insatiable and uncompromising demands of the market place, so will it's system complexity and so will the need to rapidly expand the associated development resources. There are numerous demands associated with managing the influx of new employees, the acquisition of more tools, the creation of new internal organisations and maintaining customer delivery demands. Such demands can obscure the impact associated with the transition towards the more specialist roles of staff and the management of a products expanding complexity. These impacts often go unnoticed until a products development progress starts to show signs of fatigue.

The experience of Fujitsu Australia's Telecommunications R&D centre was that expanding product complexity combined with a growing development team, started to stretch the ability of management, architects, designers and testers to

effectively communicate and maintain adequate comprehension and control of that complexity.

The flow on effect of this was an ever increasing level of effort required to maintain paper based specifications and an increased risk of not detecting design errors due to the size and complexity of the specifications. The structure and quality of the paper based specifications also became a problem. They tended to focus mostly on the design of the individual parts of the system rather than on the functionality of the system comprised of integrated parts. Further to this, testing became more difficult to plan and manage due to the structure and quality of the specifications as well as the distributed nature of these specifications.

The Quality System was also being stretched. Grown from a period when the group was much smaller, it's ability to handle larger more complex products was somewhat lacking. Research indicated that the centre needed a Quality System who's central theme should be based on the principles and practices of Systems Engineering.

### **SYSTEMS ENGINEERING**

The discipline of Systems Engineering evolved from the Aerospace and Defence industries where the increasing complexity of its programs required a more specialist approach towards managing complex and potentially life threatening systems. To this day, Systems Engineering tends to be a misunderstood discipline within commercial environments where it's name is often used as a misnomer to describe more field oriented engineering activities.

Commercial environments have traditionally been sceptical and suspicious of any practices that have evolved from the Aerospace and Defence industries. Systems Engineering is often too easily dismissed as being paper bound, bureaucratic, pedantic and too costly. However, the proponents of Systems Engineering are not suggesting that commercial environments fully adopt practices that are relevant in the context of mission critical and life threatening systems. Most proponents argue along the lines of both adopting and adapting Systems Engineering practices to suit. That is, make these practices fit for the purpose and to a level of acceptable commercial risk. If all you ever did was just introduce the basic Systems Engineering practice of writing good quality requirements, then that would be a major advantage in itself (Parth 1998).

With a bit of foresight, commercial industries can use Systems Engineering to their advantage to give them a competitive edge. By making the simple observation that commercial systems will continue to ride on the crest of advanced, sophisticated and complex new technology, then it's own products can only become more and more complex themselves. By starting to adopt and adapt Systems Engineering practices now, then within a short period of time they

will have a competitive head start in terms of the maturing of its Systems Engineering capability. Competitors that wait or believe that they can defy gravity, will reach a capability saturation point where their development environments cannot progress due to the limits imposed by inappropriate practices.

One of the main skills required for Systems Engineering is to have the vision to see the whole system in it's entire project and product life-cycle, This starts from the moment the first customer requirement is captured, through to the final acceptance and sign-off of each deliverable item to the customer. It essentially requires a holistic perspective.

A major aspect of Systems Engineering is that it's now becoming inclusive of the discipline of Project Management which has to deal with a complex system of organisations, resources, information, money, business objectives, Quality processes etc. All of which must be also broken down into more manageable defined sub-systems (usually recorded in a Project and Quality Plan).

### **RULE NUMBER 1**

Even with the best of intentions, attempting to encourage the introduction of Systems Engineering principles by dropping copies of MIL-STD-499B (DoD 1994) on managements desk, or reciting abstract prophetic phrases from published Systems Engineering papers will not win over colleagues or management! From experience, this will do more to harm than benefit.

So, rule number 1: Steer clear of reciting abstract vernacular commonly found in various Systems Engineering standards and papers. Map any foreign vernacular into terms best understood by the company.

It's important to try and adapt the principles of Systems Engineering into the target commercial environment not to try and fully replicate them. Once again, make them fit for the purpose.

### **REQUIREMENTS AND TEST METHODOLOGY**

Of primary importance to any development environment is the ability to effectively communicate instructions to all staff. A badly articulated instruction will most likely end up being badly executed and have to be done again. Product requirements are a type of instruction. In it's most basic form, a requirement can be defined as a single measurable imperative instruction. A specification can be defined as a collective series or aggregate of related requirements. Poor quality or non existent requirements will raise the risk of a poor quality outcome.

A start was made towards the introduction of Systems Engineering practices by using training to raise the awareness of engineers and management about the characteristics of quality requirements. This

was also supported with basic training in the principles and relationships between requirements and testing.

There are a number of published papers that offer some excellent examples of good quality requirements (Hooks 1993) (Kar and Bailey 1996).

A brief description of some of the more salient characteristics and principles used in the awareness training are provided below:

**Correctness.** Requirements must be stated in a correct and unambiguous form as single imperative instructions. Requirements grouped together by similarity shall form a specification.

**Unique Identifier.** Requirements must be uniquely identifiable via the use of unique prefix's. Once allocated, a unique prefix must never be re-used.

**Measurable.** All requirements must be stated in measurable terms that can be verified.

**Formality.** Hierarchical organisational of specifications that reflect the architecture of the product from it's system level down to it's lowest managed component levels.

**Traceability.** Ability to trace all child requirements to parent requirements within a specification hierarchy and to trace all requirements to individual tests. No requirements can exist in isolation and all requirements must be verified.

**Verifiable.** Requirements must have a capacity to be verified at the same level in which the requirements are raised within the specification hierarchy. They must also be verifiable by one of the following methods:

- i.) Application of a test procedure.
- ii.) Analysis or calculation.
- iii.) Documented evidence.
- iv.) Visual inspection.

**Modifiable.** Within the bounds of tight change control procedures, individual requirements, specifications and test procedures must be able to be modified relatively easily and there must be an ability to trace the impact of changes.

It's always important to note that there will be exceptions to the above rules. It's very easy to find the 1 in 100 case that will ruin the whole principle ignoring the other 99 cases that support the principle. Exceptions can be managed by virtue of their infrequency and often, the similarities far out weigh the exceptions.

The awareness training focused heavily on the fundamental principle that you test requirements and only requirements. If you're testing anything else, it's either an undocumented requirement which runs the risk of being inappropriately or inadequately tested, or it needs to be brought to the design teams attention in case it's something they may have missed in the

design.

The point to get across here is that the process of test can only be successful when it is based on the same requirements used by the developers to implement their design. If the requirements used by developers are of a poor quality, not up to date or non existent, the test process will not be effective as it will not have an ability to verify that the design conforms to requirements.

Testing all to often is viewed as "the bit you do at the end" when in actual fact, the management, application and life-cycle of testing must be included and well articulated within any Project and Quality Plan.

As Test Engineers need to have a similar level of understanding of the requirements to that of the Design Engineers, they must be included early in the process of reviewing specifications.

Test procedures must be written and managed so that when non-conformance's are detected, they can be easily replicated by re-applying the test steps and set-up information documented in the Test Procedure. Often, even a small variation in the way a test is carried out can be the difference between recording a pass and a fail or being able to fully replicate a previously reported non conformance. There's nothing more frustrating and time consuming than trying to replicate a reported incident only to find that it cannot be replicated because there was no recorded procedure applied.

It was found that once these basic characteristics and principles for requirements and test were understood, it brought about an immediate change in the critical analysis of requirements. However, the next step was to bind these principles into a generic holistic model that would act as focal point of reference for all development processes being undertaken.

## SYSTEMS ENGINEERING MODEL

The basic principles and relationships between requirements and test methodology were extended to produce a Systems Engineering model (SE model). The SE model adapted for the Telecommunications R&D centre is shown in Figure 1. The model is based on what is commonly referred to as a "V" diagram.

The model should not be confused with being a sequential life-cycle process flow chart. Whilst some degree of process can be derived from the model, it's main purpose is to act as a point of reference and to focus attention on the structure and relationships between critical elements of information that define (Specification Hierarchy), produce (Component Implementation), verify (Test Hierarchy) and manage (Project/System Control Mechanisms) a complex system. Structuring helps to simplify a complex concept. Without structure, a complex concept becomes nebulous and chaotic.

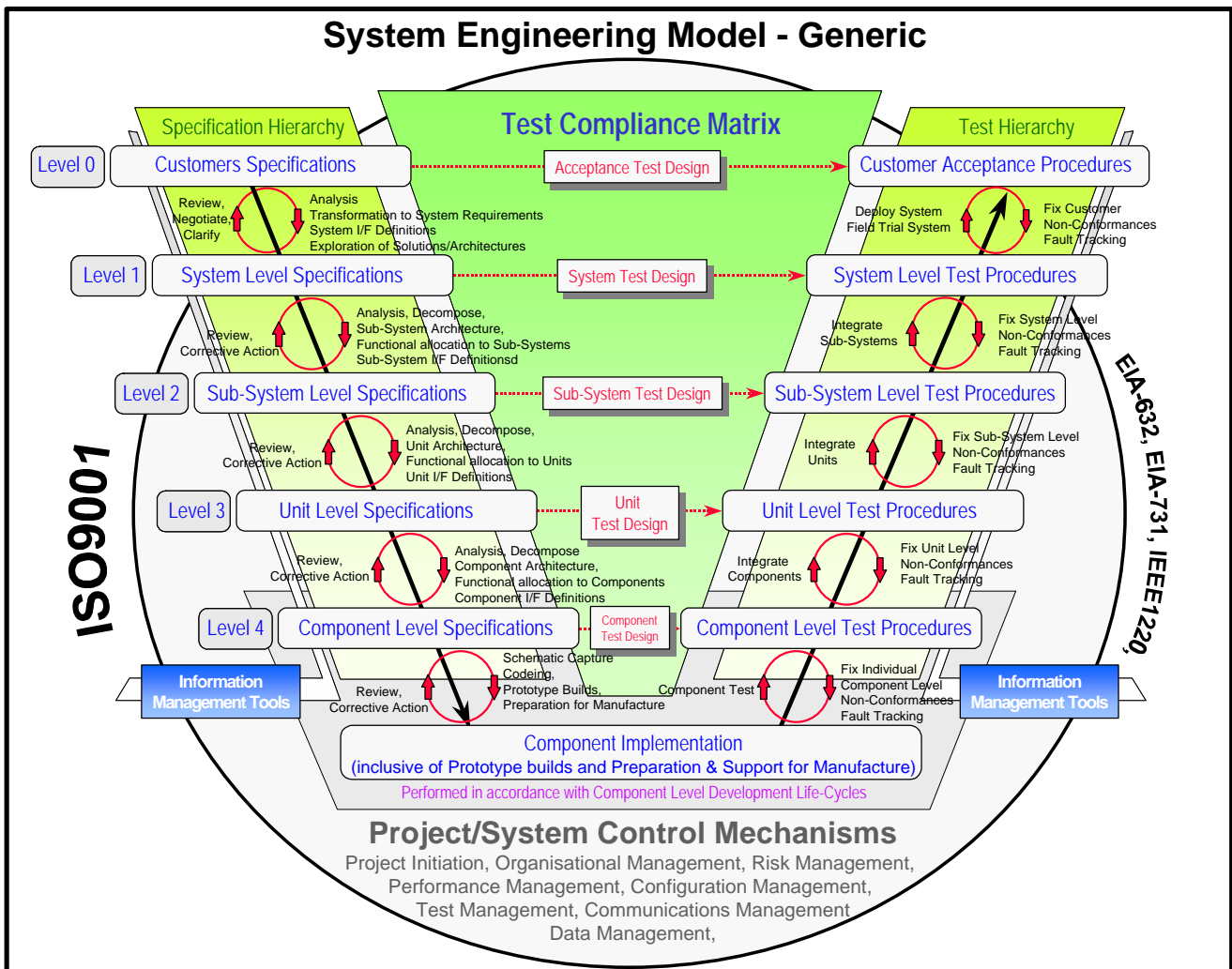


Figure 1: Systems Engineering Model

An important high level requirement of this model is that its implementation into the Quality System must satisfy all of the requirements of ISO9001.

The maturity of the SE model is following an evolutionary path and is being modified as needs arise and lessons are learned. The model itself is being tailored from reference standards such as EIA-632 (EIA, 1999) EIA/IS-731 (EIA, 1998) and IEEE1220 (IEEE, 1998).

A detailed description of every element within this model would be outside of the scope of this paper, however, descriptions of each major structural aspect depicted in the model are provided in the following sections.

### SPECIFICATION HIERARCHY

The specification hierarchy depicts the critical elements of information that define a complex system.

Each level depicted in this model has been labelled with two examples of generally used label identifiers.

1. Each level could be labelled simply as "Levels", starting at "Level 0"

2. Alternatively, more tangible labels could be used such as "Customer", "System", "Sub-Systems", "Units", and "Components". Some care must be taken when using labels such as these as they can be the source of confusion unless explicitly defined.

The hierarchy shown in Figure 1 depicts a generic series of five specification levels. Depending on the complexity of a product, there could be more levels. For example, there could be more than 1 level of Sub-Systems between the System and the Unit Levels. There could also be multiple sub levels of components that make up a Unit level. There may even be only a single component level after the system level, it all depends on the complexity of the product.

The specification hierarchy starts with the mapping and transformation of a Customers requirements into System requirements.

It's important to keep these two levels of requirements separate so that you can always maintain a separate view and focus on the customers defined needs and problems separate from the suppliers engineering solution.

Blending the two together runs the risk of the customer's needs being swamped by engineering speak. This often results in the delivery of a system that is not really in accordance with what the customer was expecting but more in accordance with how the Engineers wanted to solve the problem and designed it for their own understanding. Man Machine Interfaces (MMI's) such as Graphical User Interfaces (GUI's) can be prime examples of such a problem. If the MMI cannot be understood by the customer or is not what they asked for, then it may prejudice the customer's perspective of the whole system.

If the customer is essentially a generic market or pool of possible customers, then Marketing investigations must be performed to initiate a profile of what the market requires and the features necessary to support product differentiation. This information can then act as the Customer Level requirements. As contracts are sought and won, the Customer Level requirements can then be updated with individual customer needs.

The System level takes the customer level requirements and then decomposes them along major functional lines and in terms best understood by the development environment. The System level adds more detailed information such as operational scenarios, constraints (aka non functional requirements), inherent support functions etc. During the System level analysis, it will be necessary to perform feasibility and trade off studies by exploring possible architectures down to the component level. There will often be a need to meet with the customer to clarify, negotiate and change the customer's requirements based on the outcomes of these studies.

From the System level downwards there is an iterative cycle of analysing parent requirements and applying various methods of architectural analysis and detail design to elicit new child requirements for the next level below. Traceability between parent and child requirements become essential during this activity. Poor or non-existent traceability will allow potentially unnecessary and incompatible requirements to creep into the specifications.

As the specification structure starts to expand, it begins to reveal the configuration breakdown of the system there-by identifying all of the specifications and implementation elements, such as hardware and software components, that must be configuration managed. The importance of protecting the build integrity of a system that is made up of numerous individual components cannot be emphasised enough. This can only be done by the application of configuration management from the first customer requirement to the final acceptance of the system. Inadequate configuration management for complex systems results in the wrong functions being implemented, the wrong combination of components being tested and ultimately the wrong product being

built.

## **COMPONENT IMPLEMENTATION**

The model dictates that requirements feed eventually into component implementation. Component Implementation involves activities such as schematic entry, PCB layout, software coding, preparation of production drawings, prototyping etc. All too often, there is a mad rush to start effort at the Component Implementation level too early as this is where there is a perception of real work being done and real progress is being made. The truth is that there's really only an illusion of progress, characterised by the endless "90% complete" syndrome.

Component implementation is where most organisations have at least a convention or modicum of operating procedures that describe component level development and manufacturing life cycles. For software this might be the adoption of the Object Oriented Analysis and Design techniques and life-cycles, for hardware there would be life cycles for the selection of components, schematic entry, PCB design, building of prototypes, creation of manufacturing documentation etc.

## **TEST HIERARCHY**

The Test Hierarchy tracks the same levels as the Specification Hierarchy there-by promoting the concept that tests on requirements should be applied at the same level as they appear in the Specification Hierarchy.

Each level in the Test hierarchy consists of a managed suite of test procedures that are updated as requirements are changed.

Each Test level represents the integration and testing of the system elements from the level below.

## **TEST COMPLIANCE MATRIX**

The Test Compliance Matrix represents the horizontal mapping and traceability between requirements in the Specification hierarchy and points of verification within the Test Procedures of the Test Hierarchy. It essentially provides the means to ensure that each requirement is tested at least once.

The compliance matrix is also used to determine the scope of regression testing when requirements are modified, new requirements are added or when non-conformances to a requirement have been detected and a retest is required.

## **TEST DESIGN**

Test Design at each level involves the analysis of all of the requirements at that level and designing an architecture of efficient test systems and sets of test methods that will verify those requirements.

The major aspects of Test Design are to analyse the baseline requirements for their verifiability, group all requirements by test similarity, creation of an outline list of candidate test procedures and their

constituent test cases that will verify all requirements, identify the constituent elements that need to be integrated from the level below, design test system configurations and work out an appropriate order of test procedure execution.

Once the Test Design is complete, it will provide documented rationale and a road map for the creation of test procedures at each level.

### **PROJECT SYSTEM CONTROL MECHANISMS**

The application of Project and System Control Mechanisms help to bind each of the elements in the SE model together. This involves the creation, execution and management of various sub-plans that will support and manage the critical “V” path from concept to completion. These plans include Project Initiation, Organisational Management, Risk Management, Configuration Management, Project Performance Management (time and cost), Data Management, Communications Management and Test Management. Combined together, these plans represent the framework for effective Project Management practices and should be described as a part of a Project and Quality Plan. It cannot be emphasised how important it is to have an up to date and relevant Project and Quality Plan. Attempting to ad lib your way through a complex project will invite failure.

### **REFOCUSING THE QUALITY SYSTEM**

As a general observation, a lot of quality systems that have been modified to cope with large development projects tend to evolve into a confusing road map of life cycle process flows containing multiple levels of nested conditional branching, numerous official forms and numerous operating procedures. Quite often fragmented over many pages of a quality document or spread over an enormous wall chart, such systems start to become bit of a maze making it very difficult to visualise the critical path of information and process.

The solution for the Telecommunications R&D centre has been to use the SE model as a single point of reference for the overhauling and re-development of the Quality system. This has allowed a more homogeneous system of operating procedures and processes to be designed, all of which facilitate and satisfy the needs of the SE model. Another way of viewing this is that by applying the same concept of structure to help simplify the understanding of a complex product, the complexity of the Quality System was greatly simplified by structuring it into two levels. That is, the high level SE model acts as a parent requirement for the Quality System, and the lower level child requirements present the Quality System solution in terms of a more detailed quality road map that can be traced back to elements of the SE model and beyond to ISO9001.

### **REDEFINING THE PRODUCT**

The following is an account of how the Telecommunications R&D centre has successfully taken on the task to redefine and structure the requirements of an existing product that evolved from being a straight forward system solution, into a system solution of immense system complexity.

Because the content and structure of the products original paper based specifications did not exhibit the characteristics of good quality requirements, and were not designed with a unified formal structure in mind, they could not be easily transformed into the specification and test structure demanded by the SE model.

To get around this problem, it was necessary to re-document the products known interfaces, functionality and general configuration breakdown in the form of an architecture or road map of desired specifications, structured in accordance with the SE model.

A short abstract of expected content was also documented against each identified desired specification. This desired specification road map then acted as an authoring guide for all of the specifications that would be needed to adequately define the product.

What was discovered was that some of the identified desired specifications in the road map either had no equivalent or could not be made up from the content of the existing paper based specifications. In other words, the process had identified missing or inadequate requirements. This didn't necessarily mean that these requirements had not been implemented, but it did indicate that if they were, they were not adequately documented, and by virtue may not have been adequately tested.

Once this specification road map was agreed upon, the next task was to start authoring all of the specifications themselves.

This is a daunting task. Resources must be acquired and trained. The minds of designers and architects would have to be interrogated. Existing trails of paper based specifications and test procedures would have to be traced and content audited. Agreements would need to be sought on how existing specifications will be restructured. The Quality System needed to be updated to reflect the new processes and most of all, there was to be minimal interruption to current customer delivery demands.

An important lesson that was learnt from this exercise was that you can't attempt a task of this size without formulating plans and seeking management commitment to support the plans, approve the acquisition of tools and to provide resources. Without this support, you're exposed and on your own, a challenge of almost futile proportions.

## **REQUIREMENTS MANAGEMENT TOOL**

The magnitude of the task to totally redefine the products requirements and manage them effectively required the acquisition of a powerful Requirements Management Tool (RMT). A number of available RMT's are very powerful as they're capable of providing all of the authoring, linking, structuring and configuration management features required for Specifications and Test procedures at all levels.

Studies performed by the Requirements Working Group of the INCOSE were used as a source of information about available RMT tools and determination of evaluation criteria.

After an exhaustive investigation into several RMT products which were evaluated against published performance and feature criteria (Raymond 1997), the Dynamic Object Oriented Requirements System (DOORS™) RMT product, designed by Quality Systems and Software (QSS) in the UK, was acquired.

## **THE IMPLEMENTATION PLAN**

The introduction of Systems Engineering principles and it's underlying Requirements, Test and Project Management methodologies as well as the introduction of a major new tool was going to require a significant change of practice within the Telecommunications R&D centre. It needed to be handled sensitively with regards to the existing product development and delivery pressures placed on staff.

An Implementation Plan was devised which identified four key phases.

### **Phase I: Commissioning.**

- i.) Training in Requirements and Test methodologies, training and installation of the RMT.
- ii.) Create a pilot project within the RMT and identify all it's environmental needs and constraints.
- iii.) Update the Quality System and author new operating procedures as necessary.

### **Phase II: Product Specification Structure.**

- i.) Creation of a road map of desired specifications.
- ii.) Restructuring, preparation and transfer of paper based specifications into the RMT.
- iii.) Authoring of new specifications as per the road map.
- iv.) Linking between parent and child requirements at each level.

### **Phase III: Product Test Structure.**

- i.) Test Design.
- ii.) Authoring of Test Procedures.
- iii.) Linking verification points within Test Procedures to individual requirements.

### **Phase IV: Capability Maturity.**

- i.) Use the RMT's extensibility features to customise and automate processes.
- ii.) Interface the RMT to other tools such as problem tracking and configuration management tools.

In order to minimise the impact on currently active projects, it was decided that the redefinition of the product and the introduction of the RMT should be targeted for the next major development. As there was already a small team selected to start work on the next major development, they were given the additional task of implementing the first 3 phases.

The Pilot Project identified as a part of phase I was an invaluable exercise. Even though a demonstration copy of the RMT product was obtained for evaluation, it wasn't going to be possible to fully cover every feature and uncover every constraint until a real application was put into practice. The Pilot Project used the specification road map as the basis for the implementation of a fragment of it's structure. This unearthed a wealth of knowledge and saved potential chaos and wasted time should've we decided to go live with the RMT immediately. Amongst many other things, the Pilot Project helped to improve general familiarity with the tool, identify all of the import constraints on paper based specifications authored originally in a word processor, unearthed numerous time saving features hiding away in the bowels of nested menu structures, helped to identify which areas of access to the RMT required operating procedures and provided some indication as to what functions will need to be enhanced and customised as a part of phase IV. The Pilot Project exercise is essentially applicable to the introduction of any major new tools.

## **CURRENT STATUS**

The task of re-building product specifications in the RMT in accordance with the SE model and specification road maps are well under way. Of primary importance though has been the more immediate benefits of raising the awareness and gaining acceptance of basic Systems Engineering principles and practices such as Requirements and Test methodologies.

Design Engineers are now talking in terms of quality requirements. Test Engineers are writing tests against managed requirements and informing specification authors if they discover a critical scenario or point of verification they feel is necessary but cannot be mapped to any requirements. The impact of new requirements can now be more readily understood and quantified by tracing through to those lower level requirements that will be effected by any change to higher level requirements.

## **CONCLUSION**

The redefinition of an existing complex product helped to identify inconsistencies across common

interfaces and locate missing requirements. This exercise provided invaluable visibility of previously missing information that had most likely been a contributing factor towards the product spending greater than expected times in test and corrective maintenance. The information is now being routed back into the development to consolidate the baseline design and allow the product to move ahead with the addition of new more complex and highly competitive features.

Whilst there's still some work to be done, the process of embracing the principles and practices of Systems Engineering, has provided the framework for a development environment that will be able to consolidate a competitive advantage by being able to stay in control of the expanding complexity of its products, and release high quality features to the market place. The long term rewards for Fujitsu Australia's Telecommunications R&D centre will be the maturing of its development environment capability which can then be used to attract both customers and high quality skilled staff.

#### **ACKNOWLEDGEMENTS**

Fabian Musci BAppSc, GrdDip(CompSci), GrdDip(DigElec), Managing Director, EuroCyber Pty Ltd.

#### **REFERENCES**

- Raymond P., "INCOSE Requirements Management Tool Survey on DOORS", INCOSE Requirements Working Group Report, Published 22/4/97.
- Hooks I., "Writing Good Requirements", Proceedings of the Third International Symposium on the NCOSE - Volume 2, Published 1993.
- Kar P., Bailey M., "Characteristics of Good Requirements", INCOSE Requirements Working Group Report, Published 1996
- DoD, "MIL-STD-499B, Systems Engineering", Draft, 6<sup>th</sup> May 1994, never published.
- Parth F., "Systems Engineering Drivers in Defence and in Commercial Practice", INCOSE Systems Engineering Journal, Volume 1, Number 1, Published 1998
- Electronic Industries Alliance, "EIA-632 Processes for Engineering a System", January 7th 1999.
- Electronic Industries Alliance, "EIA/IS-731 Systems Engineering Capability Model", December 1998
- Institute of Electrical and Electronic Engineers, "IEEE1220 IEEE Standard for Application and Management of the Systems Engineering Process", 8th December 1998.

#### **BIOGRAPHY**

Paul Miller is a Senior Project Leader at Fujitsu Australia's Telecommunications Research and Development centre. He is responsible for managing teams of Hardware, Software and Test Engineers

working on the development of complex telecommunications products for the international market. He currently manages the R&D centres Tools and Process Technology group.

He graduated at the end of 1988 with a Bachelor of Engineering with Distinction in the discipline of Electronics Engineering. He began his career working for NEC Australia modifying PABX and telephony products to meet Australian regulatory standards. He has also worked for Rockwell Systems Australia, where he acquired valuable training and a higher awareness of professional systems engineering practices.

Paul is a member of the Systems Engineering Society Of Australia (SESA) which is a chapter of the International Council of Systems Engineering (INCOSE).

Telephone: ... +61 3 9520 8809 (Desk)

+61 3 9520 8520 (Reception)

Fax: ..... +61 3 9520 8800

Email: ..... paul.r.miller@fujitsu.com.au