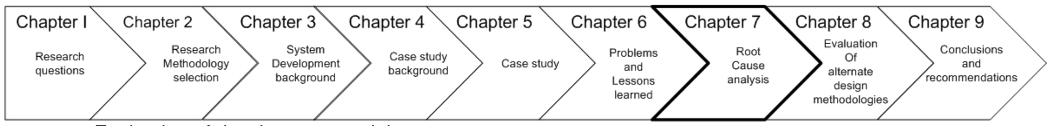
"We live in a complex world. People and organisations don't believe they have the time to perform the in-depth analysis required to solve problems. Instead they take remedial actions to make the problem less visible and implement a patchwork of ad hoc solutions they hope will prevent recurrence. Then when the problem returns, they get frustrated – the cycle repeats."
Duke Okes, (2009)


Chapter I
Research questions
Chapter 2
Research Methodology selection
Chapter 3
System Development background
Chapter 4
Case study background
Chapter 5
Case study
Chapter 6
Problems and Lessons learned
Chapter 7
Root Cause analysis
Chapter 8
Evaluation Of alternate design methodologies
Chapter 9
Conclusions and recommendations

Evaluation of development model
Finding the root cause
Determining theoretical background
Development of generalised impact equation
Impact of functional couplings
How can the development model be improved
What other models would be appropriate

Chapter 7 ROOT CAUSE ANALYSIS

7.1 Purpose and Outline of the Chapter

In the previous chapter the problems experienced using the IPS development model and the identification of the root causes for these problems on the case-study project have been identified and quantified. It was found that management related problems overshadowed the other problems for the case-study project. This result is surprising, since according to Roos, (2001), the IPS development model's primary management focus is on:

- Effective organization policy.
- Utilization of design practices for the different development disciplines.
- Development of low risk methodologies for transition from design to production.
- Interrelation between development disciplines in a multi-disciplinary environment.

The result of the case-study problems experienced was surprising, particularly in view of the combined experience of the project team. A number of advocates for the IPS development model used in the survey by Roos, (2001) were prominent players in the ZT3 Weapon System development project used for the case-study. Project cost and schedule overrun is a project quality issue and the negative outcome is confirmed by the research by Sanjay et al, (2000). Using 418 quality practices from multiple industries, Sanjay et al, (2000), concluded that management has a major impact on quality. Pretorius et al, (2007), investigated the role of design and design management in development project failures using two case studies and concluded that a systems view accompanied by "***proper design management***" will result in project success. The paper however does not detail "***proper design management***". For the case-study project of this research, the project and systems engineering management was highly experienced suggesting that other causal factors are at play as suggested by Christensen et al, (1998). The two top level causes have been identified and discussed in the previous chapter.

The purpose of this chapter is twofold:

- Evaluate the effectiveness of the IPS development model for a complex weapon system development project inclusive of all the logistic products.

- To determine whether there is a theoretical ground for the findings of the case-study.

7.1.1 Evaluation of the IPS Development Model

Discussed in Chapter 6, technically the development project was a success. The client operational, maintenance and munitions personnel were very pleased with the new weapons system. The client-contractor specialist workgroups to influence the ergonomics, maintainability and munitions safety design really proved their worth. It was announced in the internal company newsletter in October 2009 that the ZT3 Anti-Tank Weapons System project was selected for the Chairman's award. With small adaptations to the anti-tank weapons systems, the company also markets the system for universal vehicle mount as announced in a media release on 12 September 2008. From the company point of view the cost and schedule overruns were a worthwhile investment for a competitive marketable product range (ALRRT turret missile range, Denel Dynamics' brochure 0269, (2012)).

The question that must be further researched is what are the fundamental reasons for development projects of complex multi-disciplinary systems cost and schedule overruns?

Christensen et al, (1998) found that often, a contract is not fully defined when it is awarded, and changes to the contract occur as the project progresses. They termed this a "rubber baseline" or baseline instability. In a study using cost performance data from over 400 defence acquisition projects, Christensen et al, (1998), came to the surprising conclusion that there is no relationship between baseline instability and cost overruns. Their research also showed that the results were insensitive to the contract type. They suggested that other possible causal factors should be further researched.

Before a configuration item (CI) design is frozen and baselined[10] anticipated Iterations are planned for upfront and incorporated into the project plan. The cost and timescale is a function of the complexity and maturity of the technology for the design of that specific item. If it were a state-of-the-art design, invariably the planning would include the building and evaluation of various development models such as XDM[11], ADM[12] and other models depending on the complexity and technology maturity of the specific item. Apart for a project technical risk, this generally does not lead to unplanned iterative rework with their associated negative cost and schedule impacts.

Cost and schedule pressures however may force the premature release of a design CI for a system by the design review board. This may increase the technical risks at the subsequent levels of system integration. The real problem arises after the CI has been baselined and a latent design defect is found at the integration stage of the system. Holt, (2009), by using a systematic approach, developed maturity models to reduce system integration risk. A number of researchers propose the Stage Gate model to reduce the risk of design entering the next phase before the objectives of the first phase have been accomplished, (Markeset et al, 2003) and (Kleinsmann et al, 2005). If however during the integration process a problem is identified with the item, the impact from a project management point of view will be the following:

- Under the matrix organisational management structure, (Blanchard, 1998), the original design resources will not be available anymore since they would have been allocated to other projects. The waiting for resource availability will result in an unplanned project activity delay. If the activity lies on the critical path, the project will suffer an overall slippage.

- The additional cost for the unplanned activity is not planned for and must be financed from the project reserve. This reserve due to tender price competitiveness is generally kept to an absolute minimum. Unplanned costs can easily exhaust this reserve and lead to overall project cost overrun.

- In a concurrent engineering environment, all functional coupled items are also affected when one item in the system hierarchy is changed. The consequences of the functional couplings are that one unplanned change can result in a number of unplanned changes and resultant impact on the project.

[10] Baselined implies that the item’s configuration status moves from draft (revisions a, b, etc.) to revision 1.
[11] XDM Experimental Development Model sometimes called breadboard model.
[12] ADM Advanced Development Model evolved from the XDM

Generally problems with a CI identified at integration level result in a class I change of the item since this is generally the first time that the full interface of the item is comprehensively tested. It is very seldom that a latent design defect that can be fixed with a Class II change will surface at integration level due to the rigorous qualification testing of the CI prior to baselining and releasing the design.

Concurrent engineering is integral to the IPS development model. A class I change in the concurrent engineering environment underlying the IPS development model will result in a ripple effect throughout the system hierarchy. The extent of this ripple effect is a function of the functional couplings of the affected item with other CIs in the system as a result of the emergent properties, discussed in chapter 3. This finding is confirmed by the research of Smith et al, (1997), who state that engineering design involves a very complex set of relationships among a large number of coupled problems in a concurrent engineering environment. Concurrent engineering is the underlying basis of the IPS development model. In a concurrent engineering environment, the affected functionally coupled items that must also be changed can result in a change avalanche effect with very negative consequences for the project cost and schedule performance.

Summarising, the structured focus group of Narrative Inquiry into the case-study problems experienced, found that the IPS development model is a very effective model for the development of complex systems in a multi-disciplinary environment. It resulted in a very successful product for the client. Described in chapter 6, the problems, experienced on the case-study project could not be ascribed to the IPS model but appear to be rather as a result of causal factors in the SE and PM processes. This will now be further investigated.

7.1.2 Finding the Root Cause

Addressing symptoms very often introduce unexpected other problems, particularly in complex multi-discipline real time control systems since the system's dynamic balance could be disturbed as discussed in chapter 3. It is therefore essential that the root cause/s of a problem be thoroughly understood before a corrective solution is developed and implemented.

7.1.3 Determining the theoretical ground

In this paragraph, the theoretical ground for the findings of the case-study will be determined. In chapter 6 the problems experienced on the case-study project have been discussed. This has led to the identification of the following top level root causes:

- Project management and the systems engineering processes have areas of incompatibility.
- Systems Engineering primarily develop "***WHAT***" data and insufficient "***HOW***" data. The development specification focuses on "***WHAT***" the design must do; the product specification confirms that the design complies with the "***WHAT***" requirements. There is no requirement in the specifications to describe "***HOW***" the specific design solution works.

The next paragraphs will discuss these causes to provide a better understanding and basis for deeper research into the fundamental causes. Discussed earlier once these causes are fully understood, only then will it be possible to develop mitigating solutions.

7.1.3.1 Project management and systems engineering processes

(PM and the SE processes have areas of incompatibility)

The cornerstone of the systems engineering process is iterations essential for design optimisation and the achievement of design maturity. Iterations are fundamental to the systems engineering process (INCOSE 2010), (NASA 2007). Iterations are an indeterminate process in so far that the number of iterations required cannot always be predicted upfront of the project to enable incorporation into the project planning. Project management cannot accommodate an indeterminate iteration process due to cost and schedule constraints. Under project management rules, a completed milestone cannot be revisited (PMBOK 2008). A new milestone should have been defined at the start of the project which is not possible since the number of iterations required for the design of a CI are not known at the start of the system development project.

This was the primary reason why development projects of complex systems very often suffer cost and schedule overruns and will be further researched in this chapter to find the theoretical reason for the apparent conflict between the project management and the systems engineering processes.

7.1.3.2 Systems Engineering Shortcomings

(Systems Engineering primarily develop "***WHAT***" data and insufficient "***HOW***" data)

A shortcoming in the current SE process is that the formal SE process is entirely focussed on the development of "***WHAT***" product data, (INCOSE, 2010). The process does not formally require to design engineers to develop "***HOW***" product data. The SE process output product data packs contain primarily "***WHAT***" product data against which the product is tested and qualified for compliance to specifications. This view is supported by the DoD Systems Engineering Management College, (2001).

Specification practises standard (Mil-Std-490A) used in the industry provides a guideline for prime item functional specifications. The design engineer must design the item to comply with this specification. Once the design has been qualified to comply with the functional specification, the designer must prepare a product specification describing the performance parameters of the product for its intended use, as well as the necessary interface and interchangeability characteristics. The performance parameters include all essential functional requirements under service environmental conditions or under conditions simulating the service Environment, (Mil-Std-490A). These specifications form part of the formal product data pack. There is no formal requirement for the design engineer to describe precisely *"**HOW**"* his design solution works.

The lack of *"**HOW**"* data leads to problems further downstream in the systems engineering process, particularly when the logistics package for the system is being developed, specifically the operator manuals, maintenance manuals and training system. In order to optimally deploy a system and be able to effectively diagnose any problem, a thorough understanding of "***HOW***" the system and subsystems work and "***HOW***" these different system components interact is essential. The formal system data pack under configuration control does not make provision for the "***HOW***" information to enable the technical authors and training specialists to proceed with the development of these logistic products.

The lack of "***HOW***" system data also affects the effectiveness of through-life engineering support of the system during the operational life cycle. It is highly unlikely that the original design engineers will still be available a number of years later when the system is in the operational phase and in need of a modification or upgrade. The effect at that stage is that the designs must be reverse engineered before changes can be implemented. This may lead to unexpected problems and generally lead to extensive re-qualification testing which could have been avoided. The classification of characteristics in accordance with DoD-Std-2101, (1979) to a certain extent reduces this risk but is generally limited to safety and critical performance characteristics of a specific CI.

This lack of "***HOW***" data is a contributing factor for the failure of development projects of complex systems. A work-around solution has been offered in Chapter 6. This problem area falls outside the scope of this research since it may ultimately result in a major change and adaptation of the current established systems engineering process. It is suggested that further research in this field be undertaken.

This research will focus on the first cause in the context of design influencing with the aim to better understand the fundamental mechanisms of design at the lowest level. Once these mechanisms are fully understood will it be possible to quantify the impact of a design change and find mitigating solutions.

7.1.4 Detailed Analysis of design iterations

Development projects of complex multi-disciplinary systems are an intimate and coordinated process of project management and systems engineering. A system cannot be developed using the systems engineering process by itself. It requires project management to coordinate and manage the schedule as well as the consumption of resources to ensure ultimate project success. For the development of complex systems, the one process cannot function without the other.

The first cause for development project failure discussed above, identified that the systems engineering and project management processes have areas of incompatibility. This is a paradox and poses a serious question on how any systems can be brought into being? It is also inconsistent with real life experience in that many successful systems have been brought into being and are being successfully deployed.

Before an answer can be provided, further analysis is required to fully understand and substantiate this finding. Literature abound discussing project management aspects and systems engineering aspects, refer to figure 1. There is however a dearth of literature discussing the interaction between the two processes. Eisner, (2002), is one of the few authors discussing both processes yet he fails to assess and discuss the individual interactions between the two processes.

7.1.4.1 Established design process

The NASA Systems Engineering Handbook, (2007) is one of the many published sources describing the systems engineering process with particular focus on complex systems. The successive design refinements are illustrated in figure 22.

According to NASA (2007), successive refinement involves a recursive and iterative design loop, driven by the set of stakeholder expectations where a draft architecture/design and derived requirements are developed. Each step also involves an assessment of potential capabilities and pitfalls identified through experience-based review of lessons learned from other projects.

The handbook however fails to state when the iteration process will no longer produce any more meaningful (desirable) changes (improvements) to the system design.

The Systems Engineering Handbook by INCOSE, (2004), view the number of iterations planned as part of the tailoring process but makes no mention when design iterations are complete. In the updated INCOSE Systems Engineering Handbook, V3, (2006), there is also no mention when design iteration ends. Similarly in the latest INCOSE Systems Engineering Handbook, V3.2, (2010), there is also no mention of when iterations ends. The systems engineering standard prepared by Pennell et al, (2005), confirms that iterations are part of the systems engineering process.

According to v/d Merwe, (2002), the spiraling process, such as the planning or any other activity phase must be "***completed***" before the next phase is entered and so on. He identifies the reason for the spiral process as a result of the focus shift from phase to phase during the process. The research by Ashton et al, (1998), found that for the development of an optimization model in the automotive industry, multiple levels of iterations are required in their model. At business level, the research by Asharayri et al, (1998), showed that multiple iterations are essential to re-engineer a business process.

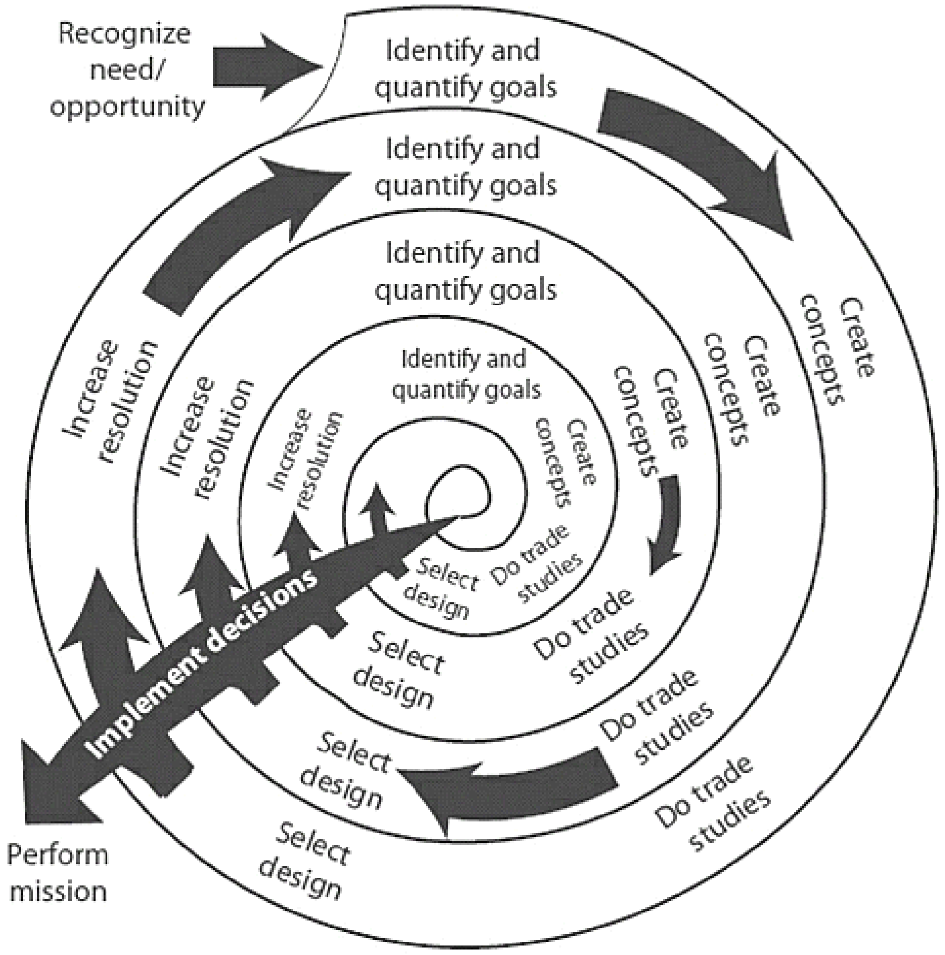


Figure 22: Successive design refinement

Source: NASA Systems Engineering handbook, (2007).

Most literature states that the design iterations ends when the design meets specification. This is a very broad statement since a large number of system requirements are often non-functional requirements or constraints that are difficult to quantify. Alexander, (2009), suggests a "***soft systems***" engineering approach for these requirements. My own view based on experience is that design ends when design maturity has been achieved. The question that now remains to be answered is "***what is design maturity*?**" The literature supplies many definitions and views of design maturity but none of these definitions actually fit into the context of design influencing and design refinement. Healy, (1989), provides the following definition: "*a system is mature when it performs its required function at specified performance levels at an optimum Life Cycle Cost for a stated period of time"*. The field of design maturity in the systems engineering context should be further researched.

For the purpose of this research, it is sufficient to accept that design refinement is an indeterminate process where the number of iterations required cannot be predicted with certainty at the start of a development project. This is in direct conflict with the fundamental project management principle where all activities and resource expenditure must be accurately planned and managed at the start of a project. This finding is supported by the research of Lu et al, (2001) that found that the Project Evaluation and Review technique (PERT) method does not support representation of iterations of the process. In an attempt to reduce design iterations, Torczon, (2007), developed a method using approximations to accelerate engineering design optimization. Li et al, (2008), using Reliability Based Design Optimization (RBDO) by decoupling the nested loops to reduce the computational workload, developed the d-RBDO model with the objective to make design based optimization deterministic.

Pritsker, (1966), developed the Graphical Evaluation and Review Technique (GERT) or conditional diagramming technique and systems dynamics models to allow for non-sequential activities such as loops in project management. This technique was taken up in the Guide to the Project Management Body of Knowledge (PMBOK 1996). However PMBOK (1996), cautions that the Precedence Diagramming Method (PDM) and the Arrow Diagramming Method (ADM) do not allow loops or conditional branches. The Project Management Institute (PMI) due to disuse discarded GERT from its third edition of PMBOK, (2004) onwards.

It can therefore be concluded that project management does not cater for iterative loops that is an essential part of the systems engineering process to enable design optimisation. To find the root cause of the management related case-study project problems, the quantitative interaction between the systems engineering and project management processes must now be determined.

7.1.4.2 Application of the SD-FD design influencing model

In Chapter 3 the structuring of the design teams in a DSR setting into ***Success Domain*** (**SD**) and ***Failure Domain*** (**FD**) teams was proposed. A design influencing model will now be developed to provide better insight of the design process at lowest level.

In preparation to find an answer to the research question: "*Can models be established to depict the success/failure domain interactions in a dynamic project management environment?*"

the proposed success frame and failure frame concepts discussed in chapter 3, were applied to the case-study design teams.

A systems engineer headed each team. One systems engineer was responsible for the development and architecture of the system and the other team was responsible for the logistics system engineering tasks and the subsequent development of the logistics products. The author had the responsibility for the logistics systems engineering and the development of the logistics products.

Systems engineering are also the custodians of the DOORS®[13] tool for requirements traceability and ensuring that all the requirements at each hierarchical level of the system have been addressed.

Eisner, (2002), states that if there is no coherent design, there is nothing to analyse. This implies that the SD team must first provide a concept design before it can be analysed by the FD team. Only when the Success Domain (SD) team makes a draft design available, can it be analysed by the Failure Domain (FD) team and feedback provided to the design review board (DRB). In practice this is an informal iterative process between the SD and FD teams with short iterative cycles.

Expanding figure 12 showing the interaction between the SD and FD teams, discussed in chapter 3, an unconstrained design influencing model can now be developed. Once the SD team has prepared a concept design, can it be analysed by the FD team and submitted to the DRB. The DRB identifies any deviations of the concept design from the specification and order another iteration until all design requirements have been satisfied. Once the design is acceptable, is the design baseline fixed and released for further integration into the system.

The DRB functions as a gate, similar to the Stage Gate model proposed by Markeset et al, (2003). This process effectively results in design iterations until the design is optimised and acceptable as illustrated in figure 23.

[13] DOORS® is supplied under licence by IBM® Rational® DOORS®; http://www-01.ibm.com/software/awdtools/doors/ (August 2010).

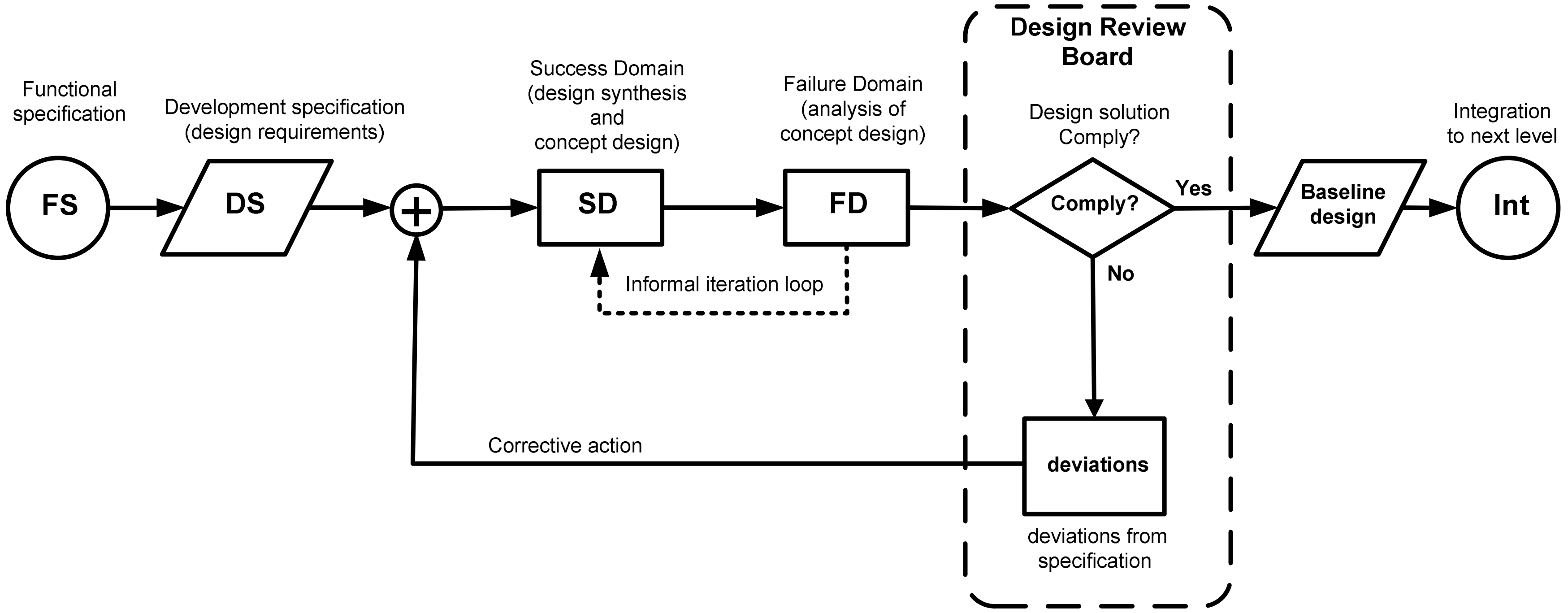

Design Review Board
Functional specification
Development specification (design requirements)
Success Domain (design synthesis and concept design)
Failure Domain (analysis of concept design)
Design solution Comply?
Integration to next level
FS
DS
SD
FD
Comply?
Yes
No
Baseline design
Int
Informal iteration loop
Corrective action
deviations
deviations from specification

Figure 23: Unconstrained "*effect-to-cause*" design influencing model

Expanding SD block in figure 23, the design engineer as part of the SD team, by means of synthesis of the requirements and constraints, produces a draft design. Expanding the FD block in figure 23, the logistic engineering analysts as part of the FD team analyse this draft design for the "***-ility***" performance against the requirements. The Design Review Board (DRB) refers any shortcomings or deviations from the requirements back to the SD team for another design iteration. This iterative design process continues until the design complies with all the requirements and the design configuration is frozen and placed under configuration control in preparation for the next level of system integration. The number of iterations required is generally determined by the maturity of the technology selected and the technical complexity of the design (Smith et al, 1997). The FD team can only perform the analysis ***after*** a concept design has been provided by the SD team. In other words design influencing is an "***effect-to-cause"*** process.

This process, although at CI level, agrees with the systems level process by NASA illustrated in figure 22. Again the question remains "***when is the design acceptable****?*" This question is not trivial since a number of the design requirements such as reliability can only be verified after extensive qualification (TAAF) testing, Mil-Hdbk-189, (1981). Experienced design review teams normally take a calculated risk based on past experience with similar technologies and designs to expedite the release and baseline of a design.

7.1.4.3 Real world design influencing model

Discussed in Chapter 3, the Systems Engineering process by itself cannot bring a system into being. It requires the Project Management process to structure and manage the systems engineering activities and the consumption of resources, thereby ensuring within budget and on time delivery of the system to the client. The two processes therefore cannot be separated and must function in an integrated harmonious manner.

In a DSR setting, the developed unconstrained design influencing model shown in figure 23 will be expanded to incorporate the influence of project management.

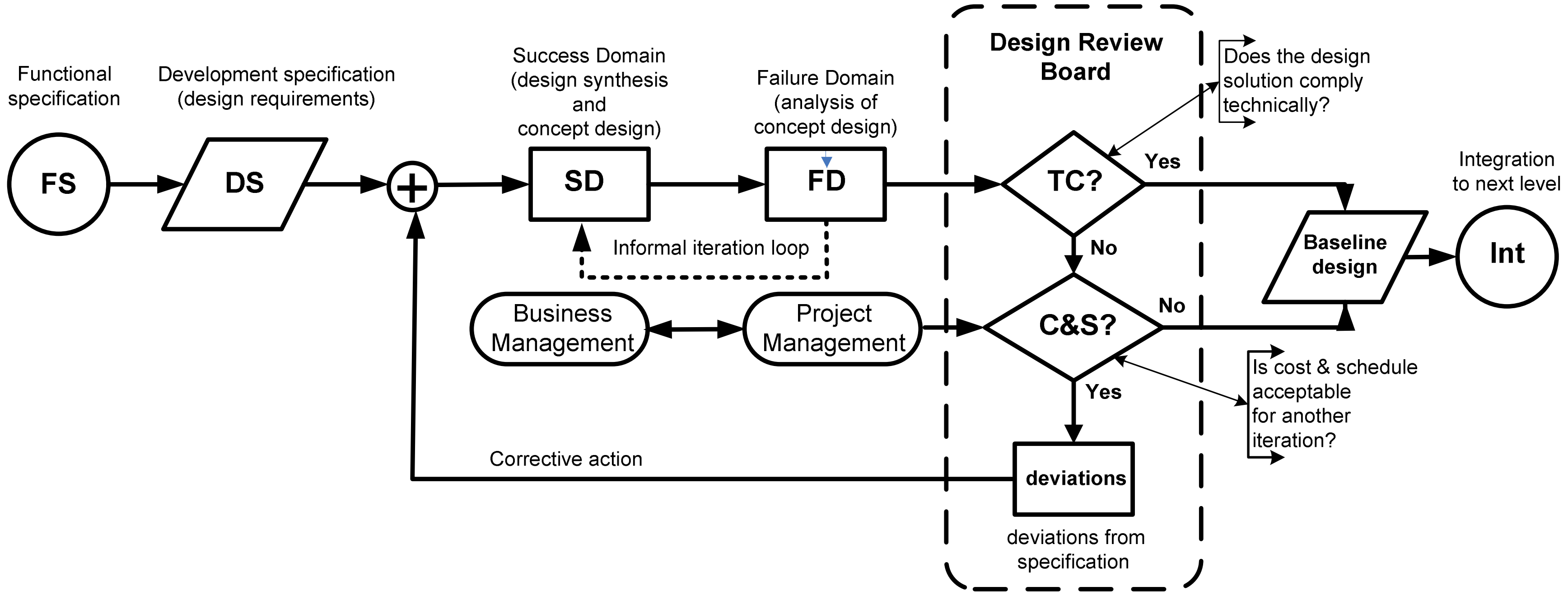

Functional specification
Development specification (design requirements)
Success Domain (design synthesis and concept design)
Failure Domain (analysis of concept design)
Design Review Board
Does the design solution comply technically?
Integration to next level
FS
DS
SD
FD
TC?
Yes
No
Baseline design
Int
Informal iteration loop
Business Management
Project Management
C&S?
No
Yes
Is cost & schedule acceptable for another iteration?
Corrective action
deviations
deviations from specification

Figure 24: Constrained "effect-to-cause" design influencing model

Expanding figure 23 and introducing another gate, the project management gate, a constrained design influencing model can now be developed. The model adds project management to the DRB. Project management is now formally represented on the DRB and can apply its influence to the design process.

Whereas the systems team review a concept design from a pure requirements and technical perspective, the project management team review a proposed design also from a project cost and schedule perspective.

Again in the constrained design influencing model, the SD team prepares a concept design, to be analysed by the FD team and submitted to the DRB. The DRB identifies any deviations of the concept design from the specification and if acceptable, the design baseline is fixed and released for further integration into the system similar to the unconstraint design influencing model in figure 23.

Discussed in chapter 3, project management constraints are different from those of systems engineering and as such can influence the design process. If the DRB identifies any deviations of the concept design from the specification, project management has the final decision to allow another design iteration or to force a release of the design for the next level of integration.

Design influencing in the real world is constrained by project management as shown in figure 24. The iterative design for constrained "***effect-to-cause"*** process design influencing model is identical to the unconstrained design process with the addition of a gate in the iterative design process by the project manager. The project manager, depending on his constraints, generally cost and schedule, can allow another design iteration or force a premature design release. The design is therefore not fully optimised and mature to the satisfaction of the SD and FD teams. This increases the risk that problems may occur at the next level of integration of the system as a result of the prematurely released design. Design review checklists to a large extent mitigate these risks, (INCOSE 2010).

Design review checklists must be dynamic and must be regularly updated from company MIS sources such as PRACAS. The checklists must be universal and not project specific. The checklists must be developed to incorporate the lessons learned from not only the present system under development but also other systems under developed as well experience from fielding data.

The NASA FTA handbook, (2002), suggests that fault tree analysis (FTA) with the purpose of design influencing should take place as early as possible in the program to avoid costly changes later on. From the research of Markeset et al, (2003), the "**Stage Gate**" model was developed to reduce development program risk. The gates ensure that the next phase of the program is not entered before the objectives of the first one have been achieved, confirming the validity of the developed models shown in figures 23 and 24. According to INCOSE (2010), the gate ensures that the next step is achievable and the risk of proceeding is acceptable. This also agrees with the findings by v/d Merwe, (2002).

Underfunding and applying too stringent and unrealistic schedules to a development project exacerbate the project risk. NAVSO P-6071, (1986), NAVSO P-3686, (1998), NASA System Engineering Manual, (2004) as well as NASA Systems Engineering Handbook, (2007), support the view that apart from minimising the technical risks, ensuring that a project is not under budget and realistic timescales have been set, can reduce the risks of a complex multi-disciplinary development project. They also caution against project underfunding. The rationale for this caution can be deduced from the developed constrained design influencing model shown figure 24 where a project manager under severe pressure may be forced to take very high risks and release an otherwise unacceptable design.

In practice all that happens is that the problem is shifted to the next level of integration, where the resources required for corrective action becomes considerably more primarily, due to the ripple effect of the corrective action throughout the system hierarchy discussed earlier. The NAVSO Best Practices Manual (1986) cautions that underfunding and unrealistic timescales may sometimes lead to the total failure of an otherwise promising project.

Summarising, from a DSR setting, a model has been developed to better understand why design iterations are fundamental to design. This model has been expanded to a constrained design influencing model that provides a better understanding of the influence of project management in the design process.

The model agrees with the discussed literature. This model shows that the project manager, particularly if he is under unrealistic constraints, can force a premature design release for integration to the next system level. The developed model provides a fundamental understanding of the design process.

The question now remains what happens when a premature design is released to the next level of system integration. What would the impact of such a premature design be at system level?

In the next paragraph the impact of a design change at the system integration level will be studied.

7.1.5 The Generalised Design Change Impact Equation

(Development of the generalised design change impact equation)

In this paragraph the research question: "*What is the impact of functional couplings between system components of a concurrent engineering design?*" posed in chapter 1 will be discussed.

Before the ripple effects of design changes can be studied in the context of a concurrent design process, it is necessary to have a clear understanding of a typical complex multi-disciplinary system. Designs can be uncoupled, decoupled or coupled. In an uncoupled design each functional requirement is satisfied by one design parameter. This is considered the best design. The next best design is a decoupled design where functional requirement independence can only be achieved if the design parameters are arranged in a proper sequence. The least ideal design is the coupled design. Here the functional requirements are more than the design parameters selected to satisfy the functional requirements. An everyday example of a coupled design is a bathroom water faucet. The two functional requirements are *control-the-temperature* and *control-the-flow rate*. The two design parameters are the hot- and cold-water tap handles. This design is coupled because it is impossible to adjust either design parameter without affecting the other functional requirement: Each handle affects both temperature and flow rate, (Gumus 2005).

Real physical complex multi-disciplinary systems, typically have a multi-dimensional hierarchal structure, of which the individual system functional elements may be coupled or uncoupled as discussed above. Real systems generally have numerous functional couplings between the different system structural elements (Smith et al, 1997), e.g. the logistic system PBS lies actually behind the operational system PBS with functional couplings between them. This presents a practical problem of presenting the complete system on a two-dimensional sheet of paper. For convenience and simplicity, as part of the system decomposition process, it is customary for complex multi-disciplinary systems to present each discipline on its own PBS such as the logistic system, software system, hydraulic system, pneumatic system, optical system, (NASA, 2007).

This creates the misconception that these product breakdown structures are separate and independent when in actual fact they are not since they are part of the system functional breakdown structure (FBS), (NASA, 2007). Generally there are numerous functional couplings between the different system PBS elements as confirmed by Smith et al, (1997). This implies that for the analysis of the ripple effect of a change, the system must be viewed as a whole multi-dimensional entity. The hierarchical system in figure 6 (Hitchins 1992), can be redrawn to also show the functional couplings between system elements that result in the emergent properties of the system.

Figure 25 illustrates a simplified hypothetical multi-level system showing possible functional couplings between elements.

To avoid obscuring the illustration, figure 25 reflects a very simplified multi-level system of i Levels, each level consisting of CIs pertaining to that specific level. The numbering of each CI identifies it to its level and to its position in the hierarchy. Possible functional coupling are illustrated by the double ended arrows.

The ripple effect of a change on one CI in say the hydraulic system may manifest in the electrical, software, logistical system elements depending on the individual functional couplings. The functional couplings between parents and children are as a result of the emergent properties discussed in Chapter 3.

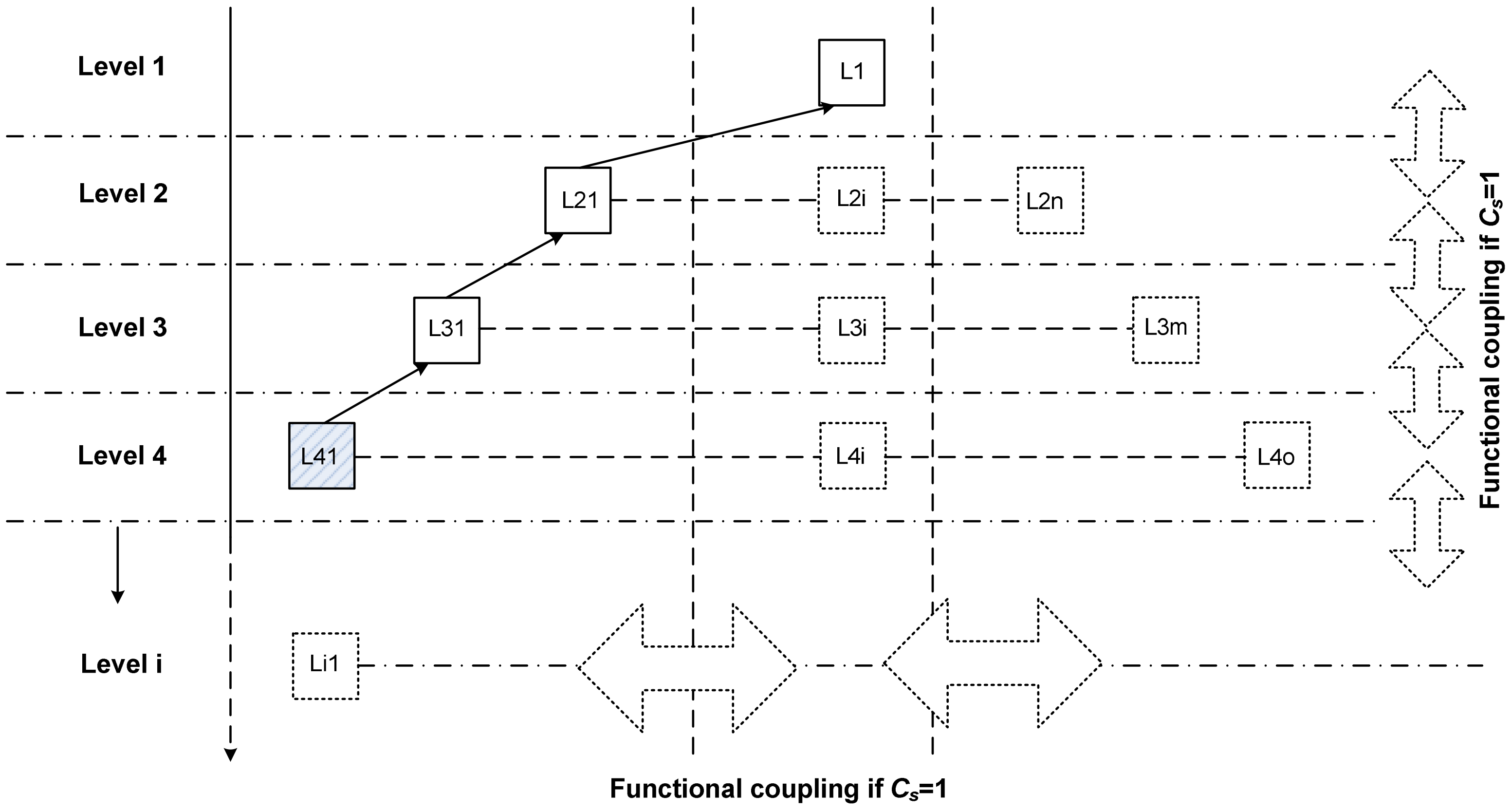


Figure 25: Multi-level system showing possible functional couplings

7.1.5.1 Impact of functional couplings

A mathematical model will be developed to quantify the impact of a design change in a multi-hierarchical system in a concurrent engineering environment as a result of functional couplings between the system functional elements, (Wessels et al, 2011). In Chapter 6 the following categories of configuration control have been identified:

- Class I modification
- Class II modification
- Rework

It is only the Class I modification that will result in a ripple effect to other functionally coupled system elements in the system hierarchy since the interfaces to the outside world or FFF is affected.

The Class II modifications and rework categories are contained within the CI and do not affect the FFF of the item. The interfaces to the outside world are not affected and therefore these categories of modification will not cause a ripple effect. These two categories however will result in unplanned expenditure of resources to correct the affected CI. Apart of unplanned expenditure of resources to correct the design, a Class II change or Rework causes no ripple effect to other system elements.

The ripple effect of a Class I change of a CI occurs by forcing changes to other CI's under development, are as a result of the functional couplings between functional elements in the system hierarchy.

In practice, it is often only during integration of the system that a latent design defect of a CI is identified that may result in a Class I change. At this stage of the system development project other CI designs are already completed or nearing completion. Apart from the corrective design change to the affected CI, the forced design changes to all other functionally coupled CIs, generally impact adversely on the development project cost and schedule.

It can thus be deduced that the root cause for the ripple effect is due to a class I change of a CI as a result of the functional couplings between functional elements in the system hierarchy. Class I changes of a CI only impact other functionally coupled system element at the subsequent levels of integration of the system.

7.1.5.2 Development of the mathematical model

A mathematical model of the hypothetical multi-level system in figure 25 will be developed. From this mathematical model it will be more convenient to study the implications of different system hierarchies in particular the effects of the functional couplings between system elements.

a) Functional coupling rules

Functional coupling rules must be defined as a prerequisite to the development of the impact equation. The dotted lines between system elements shown in figure 25, illustrate possible functional couplings. A coupling constant C_s is defined as follows:

- If there is functional coupling between affected CIs, the coupling constant C_s=1.
- If there is no functional coupling between affected CIs, the coupling constant C_s=0.
- There is ***always*** a functional coupling between an affected CI and its own parent as a result of the emergent properties, in that case C_s=1.
- There is ***always*** a functional coupling between an affected CI and its own children as a result of the emergent properties, in that case C_s=1.
- There may be functional coupling between the affected CI and its peers, other parents and children, in all those cases C_s=1.
- If there is no functional coupling between the affected CI and any other CI in the system hierarchy then, C_s=0.

b) Impact of CI design change

(Impact of a design change of an affected CI on its parents and children)

Using the system hierarchy in figure 25, assume that the affected CI is L41.

Then L31, L21 and L1 are the parents of CI L41. System emergent properties dictate that functional couplings must exist between child and parent. Therefore C_s = 1 for these instances (Hitchins 1992).

Similarly, L51 to Li1 are the children of the affected CI, L41 where i is the i_{th} level in the system hierarchy.

System emergent properties dictate that functional couplings must exist between parent and children. Therefore $C_s = 1$ for these instances.

Let R_p be the total impact of a CI as a result of the parent and children functional couplings.
Then

$$R_p = C_sL_1 + C_sL_{21} + C_sl_{31} + C_sL_{41} + - + - + C_sL_{i1}$$

$$R_p = \sum_{i-1}^{l} C_sL_{i1} \qquad (1)$$

Where l is the total parent and children CIs and i is a real integer reflecting the parent or child CI.

Note

As a result of the system emergent properties, equation (1) ≥ 1

c) General impact of CI change in the system hierarchy

Equation (1) can be generalized to incorporate the whole system hierarchy.

Let R_c be the impact of a specific CI change due to other system functional couplings in the system structure:

Assuming

- *m* represents the total configuration items in the system structure not related to the affected CI structure.
- *j* is an integer reflecting the j_{th} configuration item in the system structure.
- C_s is the functional coupling (C_s=0 if functionally decoupled or C_s=1 if functionally coupled).
- Where *n* is the total number of configuration items in the system.
- C_s=1 for all the configuration items where a functional coupling exist and the affected configuration item.
- C_s=0 for those configuration items where no functional coupling exist with the affected configuration item.

Then $$R_c = R_p + \sum_{j=1}^{m} C_s L_j$$

From equation (1) $$R_c = \sum_{i=1}^{l} C_s L_{i1} + \sum_{j=1}^{m} C_s L_j$$

Since $l+m = n$

$$R_c = \sum_{k=1}^{n} C_s L_k \qquad (2)$$

Equation (2) has been derived from figure 25.

As a result of the system emergent properties, there will always be functional couplings between the affected CI, its parent and children in the system hierarchy. Therefore equation (1) is always) ≥ 1 and can never be zero in a real system. Since equation (1) ≥ 1, equation (2) is also always ≥ 1 in real systems.

Inspection of equation (2) shows that the impact R_c increases substantially with each functional element coupling in the system hierarchy. A complete derivation and relative illustrated examples are provided in appendix C.

- **Implications of equation (2)**

  Equation (2) states that the impact of a change to the design of a CI is a function of the sum of all functional coupled items to that CI.

  Equation (2) for a specific system hierarchy will have a different value for each system CI.

d) Mathematical implications of the model

Equation (2) shows that the impact or ripple effect of a design change of one CI in a concurrent engineering environment escalates as a result of functional couplings between CIs and the size of the system hierarchy. The following conclusion can be drawn from the mathematical model:

- From equation (2), it can be deduced that in order to reduce the ripple effect of a design change to a configuration item, a design objective should be to minimize equation (2).

- To minimize the risk of a design change of one CI on the rest of the system, the system hierarchy must be optimized in such a way that the functional couplings between system elements (CIs) are minimum.

- Avoiding unnecessary functional couplings between configuration items.

- Ensuring that each functional requirement links to only one design parameter.

- Since equation (1) is always) ≥ 1 it precludes equation (2) from ever becoming zero.

- Totally decoupled designs can only be found in very simple single hierarchical level systems such as components or simple products.

- The value of R_c is an indication of a development project's cost and schedule risk should a design change be required.

A simple 3 level system hierarchy structure with 9 CIs at the lowest level was considered as a case-study, refer to appendix C. The summarized findings for the case-study examples in appendix C are:

- A simple 3 level system hierarchy structure with 9 CIs at the lowest level and with minimum functional couplings. This is considered a best-case system design of only functional couplings between parent and children and no peer functional couplings. If these remaining functional couplings for example were to be removed there would be no system but only a collection of CIs without any emergent properties.

- Using the same simple 3 level system hierarchy as above, but this time all the CIs are functionally coupled to one another providing a worst-case system hierarchy design.

- The impact for a design change in this case-study example was a cost increase of 213% and a schedule penalty of 300%.

The case-study examples are hypothetical for illustrative purposes only. In reality, real systems are much more intricate with multiple level system hierarchies and numerous different discipline CIs. Computer simulation is required to analyze these systems. Such a model would provide quantified CI design change impact information to enable design review boards to make informed decisions. Computer modeling development falls outside the scope of this research.

From the analysis it can be concluded that a design change of a CI in a complex multi-component, multi-hierarchical system during system integration in a concurrent engineering environment invariably has a detrimental effect on project cost and schedule. In practice, design modifications/changes during system integration of a complex multi-hierarchical system are virtually unavoidable. The impact of forced changes can only be improved by optimizing the system architecture to keep the system data content or functional couplings to a minimum.

e) Conclusions of the case-study examples

The hypothetical case-study examples provided in the appendices above clearly demonstrate the escalating cost and schedule impact of a design change on a concurrent systems engineering development project. This impact is a function of the number CIs and functional couplings in system hierarchy. Design changes in a concurrent engineering development project have the following consequences:

- Design changes in coupled designs are generally not feasible due to the detrimental project cost and schedule impact. Design changes, discussed above are invariably Class I changes and result in a ripple effect due to the functional couplings throughout the system hierarchy.

- Limited design changes for uncoupled and decoupled designs may be possible for simpler systems.

- The adverse project impact in terms of cost and schedule is generally too high to implement any design changes of a CI for complex multi-level systems.

- ***Effect-to-Cause*** design changes place a severe system optimization constraint on the system designer.

- Design changes are a major development project constraint in a concurrent engineering environment. Very often a band aid fix is the only practical non project intrusive way of solving the problem, which can lead to a non optimal design. This will be discussed further below.

- Further research into techniques and design processes should be performed to reduce design changes for optimal system design.

From the case-study results, it can be concluded that unplanned design changes of even a single CI in a concurrent engineering environment can be a major contributor to development project cost and schedule overrun.

From equation 2, the actual impact of a design change in a real system under development, can be calculated in order to assess the feasibility of allowing the design change, or to rather look for alternative lesser development project intrusive solutions to the problem at hand.

From the analysis it can also be concluded that the system architecture plays a very important role in reducing the system development risks. To achieve this, early system engineering is mandatory to optimize the system architecture with the objective of reducing the system information content amongst others.

Early systems engineering efforts can substantially reduce system development project risks (Honour 1994). With increased demand for "*systems of systems*", systems integration practices have steadily become more formalised and more specialised in recognition of the improved technical, cost, and schedule outcomes that can be achieved by the control and incremental validation of system interfaces throughout the developmental programme.

7.2 Summary of the impact of change

From the above model and analysis, it can be construed that due to the functional couplings, design changes in a complex multi-disciplinary system in a concurrent engineering development environment generally impact negatively on development project cost and schedule. Also design iterations should be curtailed due to project cost and schedule constraints.

In practice the impact of a class I change and associated ripple effect as a result of functional couplings in the system hierarchy is very often curtailed by doing a class I change on another lesser impact CI. This is risky unless the root cause mechanism for the failure is fully understood since a "*Band-Aid"* fix can easily result in a host of other unexpected problems. The preferred candidate by failure review and design review boards for this "*surrogate*" modification to mask the problem is software provided Human-Machine Interface (HMI) is not affected. This is generally very effective but leads to distortion, fragmentation and logical flow of the software. Also unless meticulously documented and kept under configuration control, can severely hamper through-life engineering support of the system. A simple example of such a surrogate fix could be the masking of contact bounce and resulting glitches caused by a relay or switch by means of software. This route may be far cheaper and less disruptive than to source another component but such a fix introduces an extra time delay that might affect system stability.

Equation (2) can be used as the mathematical basis to model a system under development. Such a model will assist design review boards and project management by accurately and quickly determining the impact of a proposed change on the project prior to approval of the change.

Real life reality is that cost and schedule are the primary constraints of any development project. The IPS development model and likewise all the other development models allow only "***effect-to-cause***" design influencing by the FD team. The FD team can only start the design analysis once a coherent design has been made available by the SD team. This leads to design iterations that are in conflict with the constraints of project management due to the severe impact on cost and schedules. It appears that in practice systems engineers are very seldom allowed to fully optimise a system under development.

This conclusion is in conflict with real life experience in that there have been many successful complex systems developed and deployed. The case-study of the upgrade of the ZT3 anti-tank weapons system is a case in point. The cost and schedule overruns were there but not catastrophic to the project. From the cold theoretical analysis it is surprising that a functional system at all was developed and deployed. The logical deduction is that there must be other factors at play. This will be further discussed in the next paragraph.

7.2.1 What other factors are at play?

The analysis discussed so far covered the "***hard***" systems engineering, (Alexander et al, 2009). The "***hard***" systems engineering can be quantified as shown in equation (2) and Appendix C. Checkland, (2001), discusses the "***soft***" system engineering that involves not only technical but also social, political and emotional issues and the relationships between them. According to Alexander et al, (2009), hard systems engineering addresses well defined problems whilst the soft systems engineering addresses vague ill structured problem situations that are difficult if not impossible to quantify.

It is this author's view supported by the case-study that the primary "***soft***" factor at play is the development team's interpersonal skills. An autocratic domineering project manager can very quickly sink an otherwise promising development project. It suppresses team members' creativity and initiative. The author has experienced promising development projects being prematurely terminated primarily because of poor leadership style. This view agrees with the findings by Roos, (2001), for the IPS development model and his recommendation for the team approach.

It can be concluded qualitatively that the success of the anti-tank weapons system project was due primarily to the good leadership and interpersonal relations of the management team as well as the full support from the company's top management.

7.3 How can the IPS model be improved?

Once a design's baseline is frozen, any class I change can have severe ripple effects throughout the system hierarchy. To some extent there is a natural tendency for SD and FD teams to work in isolation, therefore, any improvements to the IPS development model can only be achieved by enabling continuous interaction between these teams. The downside of any continuous SD and FD team interactions are that both teams will be continuously interrupted leading to wasted man-hour resources whilst one team is waiting for the output of the other team. Also in a concurrent engineering environment, a number of these teams will be active at any stage of the development project. This creates a project management difficulty, since such a model will result in joint accountability between the two teams. Therefore PM must be aware of, and be ready to react to any possible overrun of cost or schedule which usually cannot be easily pinpointed.

The concurrent engineering development environment of the IPS model, Roos, (2001), finds that the biggest problem with design teams is that none of them were taught at home or in school to solve problems in a group environment and states that one of the most important challenges for concurrent development is to get engineers to work well in teams. Teamwork is the success recipe to concurrent engineering.

The word "*team*" must be construed in the broader sense to be the development team, the project management team as well as company's top management. If all participants work as a team, the "***hard***" systems engineering system limitations can be mitigated through "***soft***" systems engineering as advocated by Checkland, (2001).

7.4 What other models would be appropriate?

The case-study showed that the IPS development model is a good model for the development of complex multi-disciplinary systems as it combines the best of the other development models discussed by Roos, (2001). The limitation comes in that for the development of a complex system in a concurrent engineering environment, the model must interface seamlessly with project management to manage the resources and schedules. From the case-study and subsequent theoretical analysis this has proved not to be possible, in that project management cannot accommodate indeterminate iteration processes needed to fully mature the developed CIs of the system prior to integration at the next level. Design iteration is an established design optimisation tool and is primarily as a result of the ***"effect-to-cause"*** design influencing process discussed earlier. By forcing a premature design release, the likelihood of a class I change of a system CI at integration level and the associated ripple effect to all other functionally coupled CIs in the system hierarchy is increased.

A more structured systems engineering process may have the potential to reduce the number of design iterations thereby mitigating the risk of a premature design release. This will be further discussed in chapter 8.

7.5 Chapter Summary

Root cause analysis provides a better understanding of the fundamental mechanisms. Once these are fully understood, one is in a better position to develop mitigating solutions.

The purpose of this chapter was to assess the effectiveness for the case-study project of the IPS development model, for a complex weapon system development project inclusive of all the logistic products. The structured focus group during the Narrative Inquiry work sessions was unanimous that the IPS model is very effective.

The development of the anti-tank weapons system using the IPS development model combined with the **SD-FD** design influencing approach has resulted in a superior technical product. However the ***"effect-to-cause"*** design influencing process also resulted in design iterations in order to optimize and mature the design.

From a management perspective system and design engineers are measured according to design excellence in terms of the overall customer requirements, whilst project managers are measured according to project schedules and resource expenditure performance. The two performance measuring metrics are different resulting in conflict within the development team. Goldratt, (1992), Goldratt, (1997) and Goldratt, (2006), confirm this with the following statement: "***tell me how you measure me, and I will tell you how I will behave. If you measure me in an illogical way … do not complain about illogical behaviour.***"

It is now possible to confirm the validity of the high level root cause of the problems experience on the case-study project:

> "***Project management and the systems engineering processes have areas of incompatibility***"

Delving deeper, it was found that the root cause of this incompatibility is in fact the unexpected and unplanned design iterations by the systems engineering process. Design iterations are fundamental to the systems engineering process. Very often design iterations cannot be foreseen and planned for at the start of a project, and are therefore from a project point of view indeterminate. Indeterminate design iterations cannot be accommodated under the project management discipline as confirmed in PMBOK, (2008).

The purpose of this chapter also was to determine whether there is a theoretical ground for the findings of the case-study. A model illustrating the mechanism of ***"effect-to-cause"*** design influencing in an unconstrained and project management constrained environment has been developed. It has been shown that the project management constrained environment may lead to premature design release that can lead to later modifications during system integration. These modifications result in forced modifications to all functionally coupled CIs in the system hierarchy that impact negatively on the project performance management parameters of cost and schedule

Root cause analysis confirmed that the systems engineering and project management methodologies conflict with one another, in that the project management process cannot accommodate unplanned iterations, the corner stone of the systems engineering process. Planned design iterations within a project plan activity generally presents no problem to PM provided the activity is within cost and schedule constraints. Normally, however, once the activity has been completed, PM cannot accommodate a revisit of the activity for design optimisation later in the project.

The above discussion identified design iterations as a consequence of the "***effect-to-cause"*** design influencing process. The fundamental systems engineering process and the project management process are in conflict as a result of the unpredictability of the number of design iterations required. The consequences of this conflict may result in premature design release, increasing the risk of subsequent forced design changes during system integration. Any design change at this stage invariably result in a ripple effect of forced design changes to other functionally coupled CIs that are concurrently being developed.

One approach to mitigate this conflict is to investigate ways to reduce design iterations. In the next chapter ***"cause-to-effect"*** design influencing will be investigated, with the objective of reducing design iterations that are the natural outcome of ***"effect-to-cause"*** design influencing. Employing such a methodology may have benefits by reducing or eliminating iterations. This will make the systems engineering process for the development of complex multi-disciplinary systems more structured and therefore more compatible with the structured project management process.