2007

A task model of software intensive acquisitions: an integrated tactical avionics system case study

Ricardo Peculis
University of Wollongong, peculis@uow.edu.au

Derek Rogers
University of South Australia, derek.rogers@drderekrogers.com

Peter Campbell
University of South Australia, pcampbel@uow.edu.au

A task model of software intensive acquisitions: an integrated tactical avionics system case study

Abstract

The acquisition of complex software intensive systems is fraught with significant risks and often incurs schedule delays, cost overruns and reduced functionality when the product is finally delivered. This paper presents a model to assess the effectiveness of software intensive acquisitions, founded on the premise that the solution depends on a series of transformations that transform input products in one domain into output products in another domain. Transformations are performed by people, and require knowledge and skills pertinent to both input and output domains. Ideally, products should be transformed without distortions resulting into the desirable solution. In reality, distortions occur because people are limited by their own knowledge, skills, cognition, emotions and are moved by personal and corporate goals and bounded by their roles in the social organisation. The proposed model represents the products of a domain as vectors in a vector space; and tasks as transformations that change products from one domain into another. The Analytic Hierarchy Process (AHP) is used to determine the input values for the model, to quantify a physical situation so that it can be manipulated by vector mathematics. Transformations require a nominal level of knowledge, skills and effort to be executed without distortions, which occur when the person executing the transformation possesses less than the required knowledge and skills, or has those skills subverted by other factors, reducing the effectiveness of the acquisition. The effect of 'Chinese Whispers' is easily demonstrated with the model and is an analogy of the type of distortion that can occur. A case study based on the acquisition of an integrated tactical and avionics system for the Super Seasprite SH-2G(A) Helicopter is presented to demonstrate how the model proposed in this paper can be applied to a defence software intensive acquisition. The paper concludes by showing how the proposed task model provides the basis for a more comprehensive model that can be applied to understand, explore and design complex acquisitions.

Keywords

software, intensive, acquisitions:, integrated, tactical, avionics, system, task, case, model, study

Disciplines

Engineering | Physical Sciences and Mathematics

Publication Details

Peculis, R., Rogers, D. & Campbell, P. (2007). A task model of software intensive acquisitions: an integrated tactical avionics system case study. Twelfth Australian Aeronautical Conference (pp. 1-12).

A Task Model of Software Intensive Acquisitions: Integrated Tactical Avionics System for a Maritime Helicopter Case Study

Ricardo Peculis
Senior Systems Engineer,
Computer Sciences Corporation
PhD Student,
Systems Engineering Evaluation Centre, University of South Australia

Dr Derek Rogers
Adjunct Senior Research Fellow,
Systems Engineering Evaluation Centre, University of South Australia

Prof Peter Campbell
Professor of Systems Modelling and Simulation
Systems Engineering Evaluation Centre, University of South Australia

ABSTRACT

The acquisition of complex software intensive systems is fraught with significant risks and often incurs schedule delays, cost overruns and reduced functionality when the product is finally delivered. This paper presents a model to assess the effectiveness of software intensive acquisitions, founded on the premise that the solution depends on a series of transformations that transform input products in one domain into output products in another domain. Transformations are performed by people, and require knowledge and skills pertinent to both input and output domains. Ideally, products should be transformed without distortions resulting into the desirable solution. In reality, distortions occur because people are limited by their own knowledge, skills, cognition, emotions and are moved by personal and corporate goals and bounded by their roles in the social organisation. The proposed model represents the products of a domain as vectors in a vector space; and tasks as transformations that change products from one domain into another. The Analytic Hierarchy Process (AHP) is used to determine the input values for the model, to quantify a physical situation so that it can be manipulated by vector mathematics. Transformations require a nominal level of knowledge, skills and effort to be executed without distortions, which occur when the person executing the transformation possesses less than the required knowledge and skills, or has those skills subverted by other factors, reducing the effectiveness of the acquisition. The effect of 'Chinese Whispers' is easily demonstrated with the model and is an analogy of the type of distortion that can occur. A case study based on the acquisition of an integrated tactical and avionics system for the Super Seasprite SH-2G(A) Helicopter is presented to demonstrate how the model proposed in this paper can be applied to a defence software intensive acquisition. The paper concludes by showing how the proposed task model provides the basis for a more comprehensive model that can be applied to understand, explore and design complex acquisitions.

KEY WORDS

Software Intensive, Acquisition, Social Organisation, Analytic Hierarchy Process, Integrated Tactical Avionics, Case Study, Helicopter

INTRODUCTION

The acquisition of complex software intensive systems is fraught with significant risks and often incurs schedule delays, cost overruns and reduced functionality when the product is finally delivered. Software intensive systems have a strong dependency on specific and innovative software products to execute their functions. Without software, a software intensive system becomes inoperative.

This paper presents a model to assess the effectiveness of software intensive acquisitions by comparing the actual results against the ideal. The model is founded on the premise that the solution depends on a series of transformations that transform input products in one domain into output products in another domain. Product transformations are performed by people and require knowledge and skills pertinent to both input and output domains.

The model defines three domains in the acquisition space: Situation Domain, Problem Domain and Solution Domain. A physical situation identifies a need expressed by a set of products in the Situation Domain. The need leads to the definition of the problem as another set of products in the Problem Domain. The solution that resolves the problem comprises a set of products in the Solution Domain.

Ideally, the need should be accurately expressed, the problem well defined and the solution correctly implemented to satisfactorily address the need. In reality, the expressed need diverges from the real need; the expressed problem incurs further distortions and carries the sum of all these distortions on to the solution. The resulting implemented solution is unlikely to satisfy the real need. Distortions occur because people are limited by their own knowledge, skills, cognition and emotions and are moved by personal and corporate goals and bounded by their roles in the social organisation.

The model proposed in this paper represents the products of a domain as vectors in a vector space; and tasks as transformations that change products from one domain into another. The Analytic Hierarchy Process (AHP) is used to determine the input values for the model, to quantify a physical situation so that it can be manipulated by vector mathematics. Transformations require a nominal level of knowledge, skills and effort to be executed without distortions. If the person executing the transformation possesses less than the required knowledge and skills, or less than the nominal effort is applied, distortions occur and reduce the effectiveness of the acquisition. The effect of ‘Chinese Whispers’ is easily demonstrated with the model and is an analogy of the type of distortion that can occur.

A case study based on the acquisition of an integrated tactical and avionics system for the Super Seasprite SH-2G(A) Helicopter being acquired by the Commonwealth of Australia for the Royal Australian Navy (RAN) is presented to demonstrate how the model proposed in this paper can be applied to a defence software intensive acquisition. This task model provides the basis for a more comprehensive model that can be applied to understand complex software intensive acquisitions. Aspects that influence the effectiveness of the acquisition can be explored by the model such as: variations in input parameters and resources, the propagation of lack of knowledge and skills and how the development of software is likely to be affected; how troubled projects are likely to get worse; cause and effect of tension between engineering and project management; individual and corporate motivation; staff morale; learning and cooperation. Finally, the application of the model using agent-based simulations is presented as further work.

THE ACQUISITION MODEL

The aim of the acquisition is to engineer and produce the system that will satisfy the need. The system comprises of “end-products” and “enabling-products” (EIA, 1999). End or operational products are the elements that will be delivered to the ultimate user and that will be applied as the solution that meets the need, being hardware, software, facilities, people and services. The

processes required to enable operational products are performed by enabling-products (e.g. development products, test products, training products, production products, support products, etc). Enabling-products are of value during the development of the operational products and are discarded when the latter enter in operation.

The model is founded on the premise that the result of the acquisition depends on the sum of all operational products, plus the cost and time utilised to produce all operational and enabling-products. The quality of the solution is determined by how well the operational products meet the need when put into operation.

The acquisition model will include products, the tasks to produce these products and the people that will execute the tasks. Products, whether operational or enabling, can be physical or functional in nature. Physical products have physical properties, while functional products are usually ideas, concepts and specification of products that do not exist physically (Aslaksen, 1996). Products can be inputs to tasks to produce other products and both physical and functional products are produced by tasks executed by people.

Modelling the Acquisition through Domain Spaces

Products are described by functional and performance attributes associated with the domain space in which they exist. The acquisition domain space comprises of all operational and enabling-products. Products can be grouped logically in accordance with sub-domains of the acquisition space. Three main sub-domain spaces within the acquisition space are defined: Situation Domain, Problem Domain and Solution Domain.

The acquisition originates in the Situation Domain, where a need exists and can be expressed by a set of products in the Situation Domain. The need leads to the definition of the problem as another set of products in the Problem Domain. The solution that resolves the problem comprises a set of products in the Solution Domain. The transformations of products from need to problem and than to solution require human resources with knowledge and skills specific of each domain.

Both the situation and solution domains contain physical and functional products. The need is perceived and expressed in the form of specifications of functional products. The solution is defined also as functional products, such as specification and design, and implemented by physical products, such as equipment and operational procedures. The problem domain is functional in nature. Ideally, the process should occur as shown on Figure 1, where the need is accurately expressed, the problem is well defined, the solution is correctly implemented and addresses satisfactorily the need.

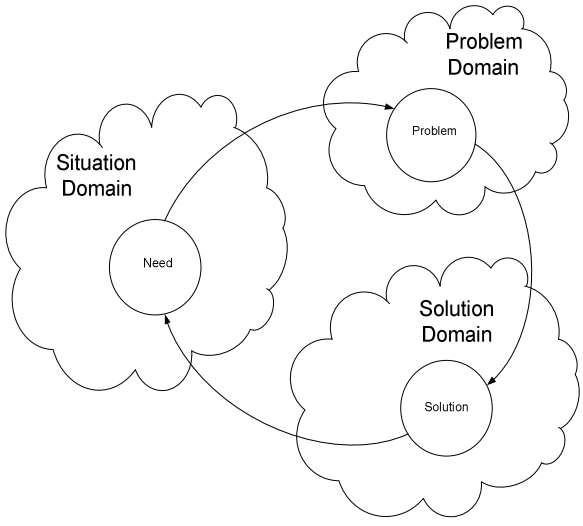

Figure 1 – Transformations between Domain Spaces.

The execution of the tasks that produces the products require knowledge associated with the domain space of the input and output products. The quality of the task execution will depend on people's knowledge and skills and their motivation to execute the task.

Figure 2 shows the real case. Within the Situation Domain, as the "Real Need" is interpreted into the "Perceived Need" they diverge. As the latter becomes the "Expressed Need" it incurs further distortion. Thus, within the Problem Domain, the derived "Expressed Problem" carries the sum of all these distortions on to the Solution Domain. The resulting "Solution Implemented" is quite unlikely to satisfy the "Real Need".

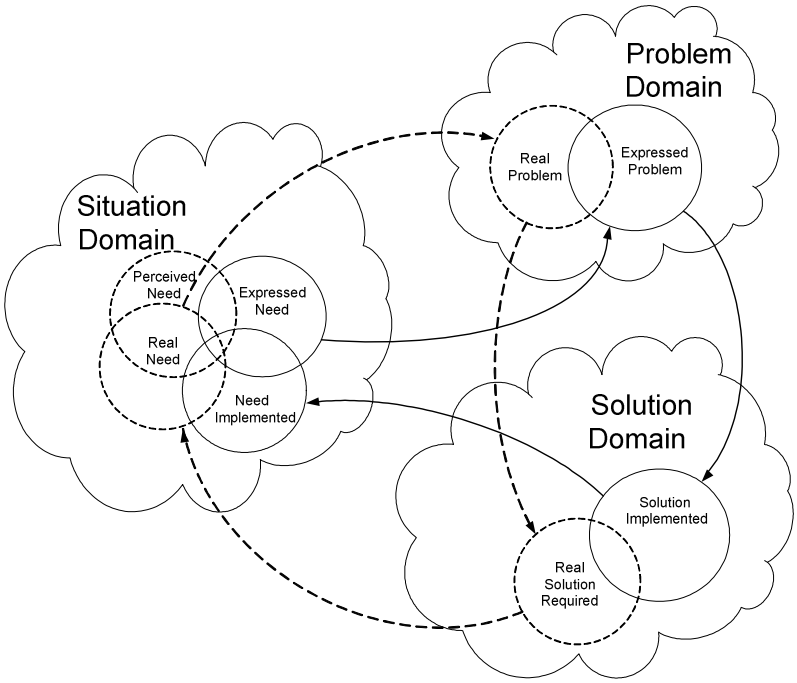

Figure 2 – Actual transformations between Domain Spaces.

Distortions occur because people within social organisations, limited by their own knowledge, skills, cognition, emotions, moved by personal and corporate goals and bounded by their roles in the social organisation, are responsible for the execution of transformations between and within domains.

The ideal transformations should accurately move from one domain to another without distortions, leading to a solution that satisfactorily improves the situation. Success thus depends on how well the transformations occur.

Effectiveness and Efficiency of Software Intensive Acquisitions

Effectiveness indicates how well the acquisition achieves its goals and efficiency represents how much waste is produced in the process. Hitchins (1992) suggests that effectiveness should be assessed comparing the real system against the ideal. From Figure 2, effectiveness indicates how close the implemented solution came to fulfil the real need, i.e. the unbroken line circles coinciding with the doted circles, and efficiency shows how much effort, in the form of enabling tasks and rework, is required to bring the implemented solution to meet the real need.

In the ideal system people possess knowledge and skills required to execute the transformation tasks, while in the real system this condition will be less than ideal. Lack of knowledge and skills, or lack of applying knowledge and skills available, are some of the causes of low effectiveness and efficiency in software-intensive acquisitions.

THE TASK MODEL

The ACTS theory (Carley and Prietula, 1994) sets the context for the task model. In accordance with the ACTS theory, organisations are collections of intelligent agents cognitively restricted, task oriented and socially situated. In the acquisition of complex systems, people are oriented to apply their knowledge, skills and perceptions of the situation in a social environment to perform the task of expressing the need, formulating the problem, establishing constraints and finding and implement a solution that satisfactorily meets the need.

The task model comprises of products and tasks. Products are functional or physical objects that together constitute the whole. Tasks are the transformations applied to products to produce other products within the same or in other domains. The execution of transformation tasks requires a nominal level of knowledge and skills associated with input and output products and their respective domains.

A nominal effort is associated to the execution of a task. If an ideal agent applies less than the nominal effort, the distortions on the output product will be proportional to the ratio between the actual and nominal task effort.

Although the example adopts linear transformations, the system is not expected to be linear. Software-intensive acquisitions are complex adaptive systems and as people learn and adapt, knowledge and skills are not constant throughout the course of the acquisition. Learning and cooperation can improve useful knowledge. Learning and the application of available knowledge and skills are fuelled by motivation. Divergent motivation can produce non-linear unwanted effects, which in turn creates tension between engineering management and project management (Aslaksen,1996), and are likely to worsen the effectiveness of the system.

Products and Tasks in the Acquisition Space

The need identifies capabilities that are realised by products organised hierarchically. A process of analysis is a transformation task that identifies the products to fulfil the desired capabilities.

The need identifies the problem requiring a solution and another transformation, the process of engineering, identifies the solution to the problem. The need is communicated through the concept of execution and functional performance documents, which drives the specification, design and construction of the products that will form the solution. The solution is also realised by transformation tasks in the form of the application of engineering knowledge, processes and skills. To execute transformation tasks requires knowledge of input and output products. Lack of knowledge will lead to distortions and omissions in output products.

The solution is likely to include products realised by hardware, software and operational procedures. As the interest of the model is in the implications of the development of software, the model will focus on software products. The development of software, like any physical product, requires specification, design and construction.

The specification and design of software components depend on systems specification and design documents. If the software specification and design are complete, the software development activity will not require any additional information from previous domains. The need for missing information would otherwise flow into the software domain. Lack of information, knowledge and expertise on capabilities and products identified in previous domains while the software is in production is one of the most common reasons of software delays and cost increases, although not always recognised.

Amongst the root causes for software project failure are inaccurate estimation and changing requirements (Jones 1995, 2006; Standish 1998). Lack of knowledge and skills are drivers of poor specification and estimation and can be associated with the root causes of software project failure. There is a lack of engineering knowledge and skills to specify software components and lack of management knowledge and skills to recognise, estimate, plan and manage this deficiency.

Virtual Products and Tasks in the Acquisition Space

In a real acquisition, products and tasks are defined by their functional and physical attributes in the form of requirements, specifications, conditions of operation and processes, expressed in ways to be manipulated by people, that is, textually, graphically and verbally.

The proposed model is intended for simulations in a computational environment, where virtual agents, representing people, teams and organizations, manipulate virtual products and tasks. It would be difficult, if possible at all, to implement agents that would be able to discuss, analyse, write, interpret, implement and verify textual requirements. For that reason, products and tasks will be represented numerically.

Products will be defined on a functional hierarchy and expressed in numerical form. The Analytic Hierarchy Process (AHP) (Saaty, 2000) is used to determine the numeric values based on the contribution of each product to the whole.

Tasks transform products from one level to other products in the next level down. The execution of the transformation tasks requires a nominal level of human resources in the form of knowledge, skills and effort. The nominal knowledge and skills depend on the parent product and its derived sub-products, and effort determines the duration of the task executed by a single agent that possesses at least the nominal knowledge and skills. When an ideal agent applies the nominal effort, the task is executed without distortions and the product is correctly produced, otherwise distortions occur.

Analytic Hierarchy Process

The Analytic Hierarchy Process (AHP) is a framework of multi-valued logic based on the innate human ability to use information and experience to construct ratio scales through paired comparison (Saaty, 2000). The object of the analysis is arranged on a hierarchic network structure that brakes down the whole into its smaller parts thus allowing paired comparison. Paired comparison is done using "The Fundamental Scale" of nine levels 1-9, indicated in Table 2.

Table 2 – The Fundamental Scale of AHP

Intensity of Importance	Definition	Explanation
1	**Equal importance**	**The two components contribute equally to the objective**
2	Weak importance	
3	**Moderate importance**	**Experience and judgement slightly favour one component over the other**
4	Moderate plus importance	
5	**Strong importance**	**Experience and judgement strongly favour one component over the other**
6	Strong plus importance	
7	**Very strong importance**	**One component is favoured very strongly over another; its dominance demonstrated in practice**
8	Very, very strong importance	
9	**Extreme importance**	**The evidence favouring one component over another is of the highest possible order of affirmation**

The objective of the AHP is to pair compare all components in the system of interest to determine the weight of importance or contribution of each component to the whole.

If the system of interest has "n" components, the pair comparison is obtained by an n x n square matrix called the Priority Matrix. The weigh of importance of each of the "n" components is given by the normalised principal eigenvector, obtained from the maximum eigenvalue of the Priority Matrix.

The Acquisition as Vectors in a Vector Space

Products are modelled as vectors in a Euclidean Vector Space, while tasks are transformations that create other products as vectors within the same or into other vector spaces. The numeric value of a product represents its contribution to the whole. The complexity of a product is not determined by the absolute value of its numeric representation, but it is associated with its relationship with other products and by the knowledge, skills and effort required to produce the product.

The number of linearly independent products required to represent the acquisition determines the dimension of the acquisition space. The acquisition space adopts the standard base, comprising of unitary orthogonal vectors corresponding to the dimensions of the acquisition space.

Products are transformed into other products through tasks executed by the agents. Tasks are modelled as linear transformations matrices, plus the knowledge and effort required to performing the task.

Useful Knowledge

The effectiveness of the execution of the tasks depends of the agent's useful knowledge and is proportional to the projection of the agent's knowledge/skills vector onto the task's knowledge/skills vector, shown on Figure 3. The normalised value of 1.0 represents the required knowledge/skills.

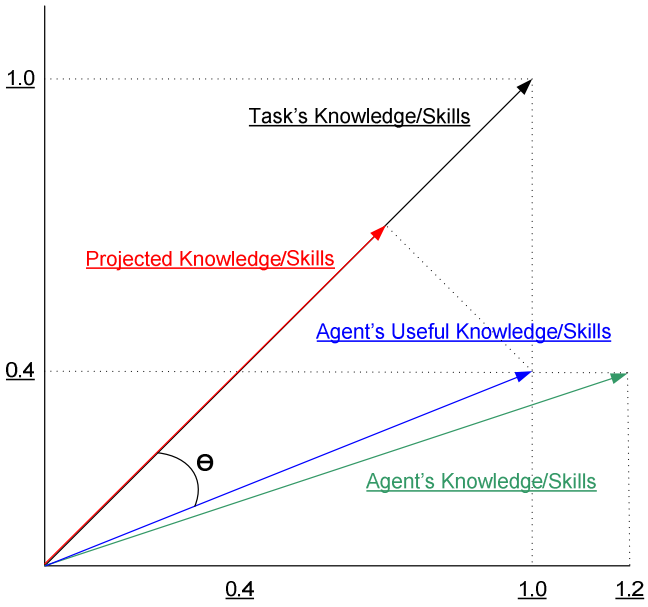

Figure 3 – Useful Knowledge.

Effective Knowledge = | Agent's Useful Knowledge| cos θ, where

$\theta = \arccos\left(\frac{a \bullet b}{|a||b|}\right)$ the angle between the two Task's Knowledge and Agent's Useful Knowledge vectors.

The agent's useful knowledge is limited by the amount of knowledge required by the task on each dimension; otherwise the excess of knowledge/skills in one dimension would compensate the lack of knowledge/skills in another dimension. The effectiveness of the task execution is proportional to the projections of the agent's useful knowledge into the task's knowledge/skills required.

APPLICATION CASE STUDY

This illustrative example is a case study based on the specification of the Super Seasprite SH-2G(A) Helicopter being acquired by the Commonwealth of Australia for the Royal Australian Navy (RAN). The first two levels of decomposition, comprising the roles and capabilities of the aircraft, are based on public information (DMO, 2006; NOC, 2006; Scott et al., 2004). The relative level of importance of roles and capabilities are hypothetical.

The Need

The Royal Australian Navy identified the need for a helicopter to support operations from the ANZAC class frigates. Two primary roles for the aircraft were identified:

a. Increase the ship's effectiveness by expanding surveillance capability;
b. Contribute to the ship's combat capabilities providing anti-surface and anti-submarine warfare.

The secondary roles for the aircraft comprise of supporting missions such as search-and-rescue and medical evacuation and crew training. The need also determines that the aircraft is to be operated by a crew of two, day and night and under adverse weather. The AHP was used to estimate the hypothetical relative contribution of the aircraft roles, shown on column P_1 in Table 3.

Table 3 – Aircraft Roles

Aircraft Roles	P_1	SV	ASH	ASB	SR	SM	TS
Surveillance (SV)	0.41	1	3	3	7	9	9
Anti-Ship Warfare ASH	0.28	1/3	1	3	7	9	9
Anti-Submarine Warfare (ASH)	0.18	1/3	1/3	1	5	7	9
Search and Rescue (SR)	0.07	1/7	1/7	1/5	1	3	7
Support Missions (SM)	0.04	1/9	1/9	1/7	1/3	1	5
Training Support (TS)	0.02	1/9	1/9	1/9	1/7	1/5	1

The need defines six roles (Table 3) and twelve capabilities: HMI for crew of two (HMI); Aircraft Monitor & Supervision (MS); Mission Preparation Support (MPS); Mission Debrief Support (MDS); Electronic Navigation (NAV); Radio Communications (COM); Surveillance Sensors (SVS); Electronic Warfare (EW); Tactical Data Management (TDM); Tactical Navigation and AFC (TNM); Anti-Ship Weapons (WAS); and Anti-Sub Weapons (WASb). Table 4 shows the need in the form of the dependency between roles and capabilities.

Table 4 – The Need

Roles			Capabilities											
			HMI	AMS	MPS	MDS	NAV	COM	SVS	EW	TDM	TNM	WAS	WASb
		P_2	0.10	0.08	0.08	0.02	0.10	0.12	0.14	0.09	0.09	0.08	0.06	0.04
	P_1													
SV	0.41		0.10	0.08	0.08	0.02	0.10	0.12	0.20	0.10	0.10	0.10	0.00	0.00
ASH	0.28		0.10	0.08	0.08	0.02	0.10	0.12	0.10	0.10	0.10	0.00	0.20	0.00
ASB	0.18		0.10	0.08	0.08	0.02	0.10	0.12	0.10	0.10	0.10	0.00	0.00	0.20
SR	0.07		0.10	0.08	0.08	0.02	0.10	0.12	0.10	0.00	0.00	0.40	0.00	0.00
SM	0.04		0.10	0.08	0.08	0.02	0.10	0.12	0.10	0.00	0.00	0.40	0.00	0.00
TS	0.02		0.10	0.08	0.08	0.02	0.10	0.12	0.10	0.08	0.08	0.08	0.08	0.08

The shaded column P_1 represents the hypothetical contribution of the aircraft's roles, as shown on Table 3. The set of non-shaded numbers defines a 6x12 transformation matrix T_1 and represents the contribution of each capability to the aircraft's roles. Although the numbers in T_1 were estimated without AHP, AHP could have been used to better accuracy (each row requires a 12x12 Priority Matrix). The shaded row P_2 represents the hypothetical contribution of each capability to the need as a whole, calculated by equation (1) below, where P_1 and P_2 are respectively the input and output product vectors and T_1 is the transformation matrix:

$$(1)\ P_2 = P_1 \,.\, T_1$$

The elements of the transformation matrix are tasks whose execution is affected by the ratio of the agent's useful knowledge and the knowledge required to execute it. It is interesting to observe that there are multiple dependencies between roles and capabilities. If Surveillance Sensors (SVS) is partially or not available, for example, all roles defined for the aircraft will be affected. Also, to express the need requires knowledge in multiple areas. To specify the need for Surveillance Sensors (SVS) requires not only knowledge of surveillance sensors but also on all of the aircrafts' roles. Lack of knowledge would cause deficiencies in expressing the need.

The expressed need is presented in the form of an Operational Concept or similar document, represented by equation (1).

The Problem

The expressed need is the input to the next level of decomposition. The problem is expressed in the form of what is required to resolve the expressed need. Capabilities endure further analysis to reveal products required to resolve the problem. The analysis continues until all products and their relationship are completely defined and specified. There are 26 primary products identified for the SH2G(A) helicopter, comprising of HMI devices, Main Data Processor, Mission Data Loader/Recorder, Navigation Devices, Communication Equipment, Aircraft Sensors, Radar, Forward Looking Infra Red (FLIR), Electronic Warfare (EW), Threat Warning, Counter Measure Dispenser (CMDS), Link-11 and various Weapons.

Capabilities are expanded into primary products and their contribution to the whole is estimated. Being P_2 and P_3 respectively the Capabilities vector input and the Primary Products vector output, the transformation matrix T_2 is a 12x26 matrix such that:

$$(2)\ P_3 = C_2 \,.\, T_2$$

The problem analysis is completed when the solution is defined and proposed in the form of a Function Performance Specification or similar document, represented by equation (2).

The Solution

The analysis of the problem and the proposed solution are the input to the next transformations. The solution comprises a series of transformation that produce enabling and operational products. Enabling products are the specification and design for the system, hardware and software, while operational products are hardware, software and procedures that will be delivered with the final solution.

Each of the 26 primary products identified in the proposed solution requires a system specification and design, as well as specification and design for human and hardware interfaces. The output of specification and design is then 156 enabling products. The specification and design process is represented by equation (3), where P_3 is the Product input vector, P_4 is the Specification and Design output vector and T_4 is the 26x156 transformation matrix.

$$(3)\ P_4 = P_3 \,.\, T_4$$

The system under development calls for the integration of products that require the development of dedicated software. As the focus of this paper is on software intensive systems, the transformations that follow address only software products.

Each of the 26 primary products depends on software and requires software specification and design. The output of software specification and design is then 52 enabling products. The specification and design process is represented by equation (4), where P_4 is the System Specification and Design input vector, P_5 is the Software Specification and Design output vector and T_5 is the 156x52 transformation matrix.

$$(4)\ P_5 = P_4 \,.\, T_5$$

The specification and design of software products should contain all information required to enable the production of software components. Each of the 26 primary products requires the development of dedicated software products. The output of the software productions is then 26 operational products. The production of software products is represented by equation (5), where P_5 is the Software Specification and Design input vector, P_6 is the Software products output vector and T_6 is the 52x26 transformation matrix.

$$(5)\ P_6 = P_5 \,.\, T_6$$

System specification and design documents intend to bring knowledge and information from previous domains into the software domain. When this intent is not achieved, lack of information and knowledge will propagate into the software domain and will carry distortions and omissions from specifications created on previous domains. To detect and correct these anomalies requires knowledge beyond the software domain, which usually is no longer or easily available.

Effectiveness and Efficiency

The effectiveness of the solution is given by the sum of the contribution of the end products, which in the ideal case is 1.0. Equation (6) represents the effectiveness of the solution, where in this example p_i are the 26 components of the vector P_6:

$$(6)\ Effectiveness = \sum_{i=1}^{n=26} p_i$$

Unless distortions are reduced, whether at the end of the development cycle in the form testing and rework, or as an integrant part of the engineering process, 100% effectiveness cannot be attained.

Equation (7) represents the efficiency of the system as the ratio between the actual and ideal effort, where actual effort includes the effort spent to produce enabling and end products plus the effort spent on additional enabling tasks and rework:

$$(7)\ Efficiency = \frac{\sum_{n=1}^{n=6} ActualEffort(Pn)}{\sum_{n=1}^{n=6} IdealEffort(Pn)}$$

The 'Chinese Whispers' Effect

The example shows that to produce 26 software operational products requires development of 252 enabling products in the form of operational concept, functional performance and other specification and design documents. Test products do not impact the solution directly and were not included in the example.

Each element of the transformation matrices, if not zero, is a task that requires knowledge of input and output products, spread over 300 areas of knowledge. Equations (1) to (5) represent five transformations required to produce software products. In summary, to produce 26 outputs needed some 250 intermediate outputs in a five stage process. The distortion level of each transformation will determine how correct the end product will be if compared with the ideal solution. This reflects the 'Chinese Whisperers' effect caused by the propagation of distortions and omissions due to lack of useful knowledge. Distortions will also occur if less than the nominal effort is applied. Wrong estimation can be associated to lack of knowledge and skills and contribute to reduce the effectiveness of the system.

Distortions and rework caused by lack of useful knowledge, whether in engineering or management, could partially explain schedule delays and cost overrun in software intensive projects.

CONCLUSION

The proposed task model represents the ideal products and transformations found in software intensive acquisitions. Products are represented numerically by their respective contribution to the whole estimated using the AHP. Tasks are matrices that transform input products into output products, and require a nominal level of knowledge, skills and effort. If less than the required knowledge, skills and effort are applied, output products will contain distortions and omissions. Output products are input products of subsequent transformation and distortions propagate in the form of 'Chinese Whisperers' effect.

Lack of knowledge and skills, can be associated with failure of software intensive acquisitions. Knowledge is acquired and shared through learning and cooperation, and these are driven by motivation. Software intensive acquisitions are complex adaptive systems, and the factors that influence knowledge, are affected by the system and vary during the course of the acquisition.

A case study based on the acquisition of the Super Seasprite helicopter was presented to demonstrate how the proposed task model can be applied to software intensive acquisitions.

The proposed task model is a component of a more comprehensive socio-organisational model founded on the ACTS theory, and will be applied to agent-based simulations to explore non-linearities and factors of success and failure of complex software intensive acquisitions. Simulation can be used to investigate how the effectiveness of the system is affected by variations in input parameters and by the engineering process models (e.g. waterfall versus incremental). Simulations can also help to understand how software components are likely to be affected by the propagation of lack of useful knowledge and inadequate estimation of effort; how troubled projects are likely to get worse; cause and effect of tension between engineering and project management; and how individual and corporate motivation, staff morale, learning and cooperation affect the effectiveness of the system in meeting the need.

REFERENCES

Aslaksen, E. W. 1996, *The Changing Nature of Engineering*, McGraw-Hill Australia, Sydney.

Carley, K. M. and M. J. Prietula 1994, *ACTS Theory: Extending the model of bounded rationality*, In Carley, K. M. and M. J. Prietula (Eds.), Computational Organization Theory. Hillsdale, NJ: Lawrence Erlbaum Associates.

DMO, 2006, *Project SEA 1411, ANZAC Ship Helicopter*, Defence Materiel Organisation, http://www.defence.gov.au/dmo/asd/sea1411/sea1411.cfm viewed on 12 June 2006.

EIA, 1999, *EIA-632 Standard: Processes for Engineering a System*, Electronic Industries Alliance.

Hitchins, D. 1992, *Putting Systems to Work*, John Willey & Sons, London, England.

Jones, C. 1995, *Patterns of Software System Failure and Success*, International Thomson Computer Press, Boston, MA, December 1995.

Jones, C. 2006, *Social and Technical Reasons for Software Project Failures*, Cross Talk, The Journal of Defense Software Engineering, June 2006, p.4-8.

NOC, 2006, The SH-2G(A) *Kaman Super Seasprite*, Naval Officers Club, http://www.navalofficer.com.au/seasprite.htm, viewed on 12 June 2006.

Saaty T. L. 2000, *Fundamental of Decision Making and Priority Theory with The Analytic Hierarchy Process*, Vol. VI of the AHP Series, RWS Publications, Pittsburgh, PA.

Scott, R, Holdanowicz , G, Bostock, I, 2004, *Second coming for the Super Seasprite*, The Aviation Forum, http://forum.keypublishing.co.uk/showthread.php?t=24443, viewed on 13 June 2006.

Standish Group 1998, *CHAOS: A receipt for success*, Standish Group, http://www.standishgroup.com/sample_research/PDFpages/chaos1998.pdf, viewed 13 Nov. 2003