Methodology for Rapid Development of C2 Planning Systems

Sheena Kelsey and Simon Snell
QinetiQ
Darenth House, Sundridge Business Park, 84 Main Road, Sundridge,
Nr. Sevenoaks, Kent TN14 6ER, United Kingdom
Tel +44 (0)1959 54 7419, Fax +44 (0)1959 54 7571
smkelsey@qinetiq.com
shsnell@qinetiq.com

For presentation at the 8th ICCRTS Command and Control Research and Technology Symposium in the stream entitled C2 Decision Making and Cognitive Analysis

QinetiQ

Methodology for Rapid Development of C2 Planning Systems

Sheena Kelsey and Simon Snell

QinetiQ
Darenth House, Sundridge Business Park, 84 Main Road, Sundridge,
Nr. Sevenoaks, Kent TN14 6ER, United Kingdom
Tel +44 (0)1959 54 7419, Fax +44 (0)1959 54 7571
smkelsey@qinetiq.com
shsnell@qinetiq.com

Abstract

This paper proposes a methodology for rapid development of computer systems that offers benefits for the rapid and accurate elicitation and definition of user and system requirements. It also aids user acceptance and can reduce procurement time scales.

The development of a successful planning system depends on a user-developer link where final user requirements can be developed through an adaptive process of learning and evolution. The methodology QinetiQ have developed is based on iterative and interactive 2 way links between User and System, User and Designer and between System and Designer. This 'middle-out' approach relies on the quick delivery of an initial system to which users can respond and thus clarify what they really need. This is known as rapid prototyping or Rapid Application Development (RAD).

The refined method called Parallel Rapid Application Development has been developed in response to Urgent Operational Requirements (UOR). This involves designing and building an initial system using the methods detailed above but in a very short and concentrated time scale. This system is then fielded and further requirements are developed in response to information from the field. The system is updated and a new version sent to fielded system via secure email.

Introduction

Problem

Procurement processes certainly for the UK MoD can take up to 5 years or longer. Projects tend to be very large, and the fixed time scales and costs can potentially result in functionality being sacrificed in the implemented system. In addition, long procurement time scales result in more significant technology gaps, and ultimately result in shorter effective in-service lifetimes. This is particularly pertinent for areas of rapid technology development, such as software systems.

Method of development is also an issue. Requirements can be difficult to elicit and also be based on current technology, restricting potential design solutions. Current

sequential step-by-step development though acceptable for most military hardware is not, in the authors' view the best method for the procurement and development of computer based planning systems. Older methodologies have proved to be inflexible, over ambitious and 'opaque to the user'. C2 system development requires formal methodologies which aim not to produce 'optimal' solutions but to facilitate an enriched decision making process. It is also of limited use when a system is required under an Urgent Operational Requirement (UOR). Time scales for these are usually months or even weeks.

***C2 Relevance**:*

The ability to conduct pre-planning is a vital factor in assisting C2 decision making. Creating, amending and running through various courses of action for both blue and red forces based on Intelligence Preparation of the Battlefield (IPB), Decision Support Overlay (DSO) etc is of great benefit if not vital preparation. Current planning methods however are manually based and relatively slow, being largely based on map table overlays.

Computer based planning systems have the potential to be a great asset in the planning process, with their ability to store and manipulate large volumes of information, very rapidly. They can also rapidly perform complex calculations to assist with, for example line of sight and cresting analysis.

However to be effective these systems must be based on well defined and documented requirements with extensive user input and be available in a timely manner.
For a system required under an UOR to deal with a specific problem the time constraint is paramount and development methods need to be available to cope with this.

Approach to Topic:

The development cycle has been researched and amended to reflect the problems specific to computer systems. Through experimentation and practical application of various techniques to projects over the last five years the team have developed methods that they believe greatly assist in the rapid and accurate elicitation and definition of user and system requirements. These methods have been enhanced to assist with the provision of systems to meet UOR' s and to facilitate the further refinement of these systems when they are in the field.

UK MOD Methodology

The main procurement principles followed by the UK MoD are SMART (Symbolic Method Aiding Representation of Tactics) procurement and the CADMID (Concept, Assessment, Development, Manufacture, In-service, Disposal) lifecycle.

The aim behind SMART Procurement is to enhance defence capability by obtaining and supporting equipment more effectively in terms of time, cost and performance thus delivering the projects within the agreed performance time and cost and to cut the

time for the introduction of new technology. It has been designed with the military but the principles apply to any other industrial sector.

The CADMID processes are the lifecycle of SMART procurement and follow the classic waterfall method, moving from one phase to the next in a sequential manner.

Concept starts with the stimulus for a new project, where the Mission Statement is defined with the customer's problem, not the solution.

Assessments is the investigation of solution options with discrimination of key design drivers, evaluation and trade-off of systems concepts, prototyping/demonstration of high risk issues, determination of program feasibility and recommendation of final system solution.

Demonstration is detailed development of the proposed solution with manufacture of the first-off systems, integration, verification and validation against the system requirements and demonstration to the Customer of the fulfilment of his needs.

Manufacture is the manufacture of follow-on systems against a proven design and may include concurrent development of the training and infrastructure systems, to support the deployment field deployment of the prime system.

In-service is field deployment including maintenance, upgrades etc.

Disposal is the de-commissioning which may include re-sale and re-cycling and may be complex, hazardous or has legal implications.

During the concept and assessment phases the User Requirement Document (URD) is prepared and the Directorate Equipment Capability team owns this. During the assessment and demonstration phases a System Requirement Document (SRD) is prepared and the Integrated Project Team owns this.

Current Software System Development Methods

Waterfall Model

The waterfall model was developed in response to previous ad hoc approaches that had little regard for business needs or usability. The waterfall approach is an effort to provide a more coherent and structured way to develop an information system.

It is a highly structured, sequential and predictable set of stages and techniques but it is now commonly associated with systems failure, long time scales and expensive overheads. There are still, however, hundreds of development methodologies based on the simple sequence of the waterfall model:

Feasibility study

System investigation

System analysis

System design

Implementation

This sequential system development method can also be shown as a sequence of generic processes, from user requirements to the delivery of a complete operational capability. At each boundary, reviews allow progress to be monitored and a commitment made to the next stage. Changes and feedback from the next stage affect the previous stage.

The V-diagram shown in figure 1 illustrates the simple system-lifecycle with verification between the definition stages and across the horizontal links. The left-hand side defines what must be built and the right-hand side builds it from the components and verifies the end products against the left-hand specifications.

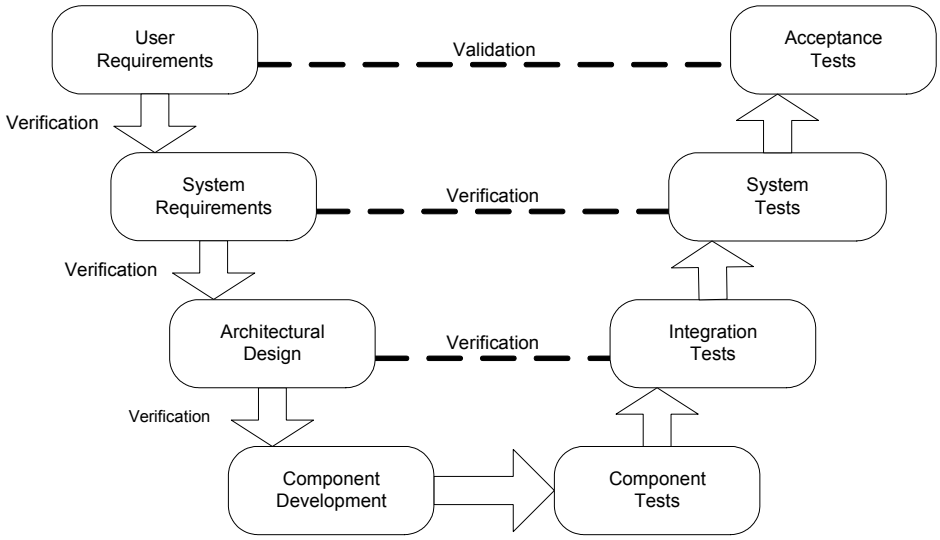

Figure 1: The V-Model

Structured System Analysis and Design Methodology (SSADM)

The classic waterfall model has developed in many forms over the past twenty-five years or more. SSADM has become a British Standard and brings together a wide range of technologies for analysis and design and provides a coherent and widely used framework for their use. SSADM provides a high level of control achieved through documentation and through structure. More recently SSADM has evolved and adapted to include Object Orientation ideas and techniques. Additional specifications have been added to SSADM that attempt to reflect some of the issues found in C2 environments, specifically mapping and geographical issues. These include:

- Geographic data places specific and significant demands on the design process
- Geographic data entities and procedures are inherently complex
- Users place high demands on the analysis of requirements
- Specific knowledge of geographic data and Geographical Information Systems (GIS) is essential for good design

The reality, however, is a limited adoption of SSADM by organisations undertaking C2 projects. The reason for this is unclear. Some theories suggest that user involvement is too controlled and input only at specific stages. The eventual users of the system are allowed to make an input to *what* the system should be but not *how* that system should be developed. This *how* bit is important since it defines the scope of requirements and this especially reflects on particular decision making processes which need to reflect the nuances, intuition and skills of an expert decision maker.

Other methods

Newer methodolgies have recently emerged against the background of rapid developments in technology. Developers have to get grips with changing technologies and the increased demand of users. Within the C2 domain, developers are faced with existing rigorous processes that have withstood the test of time, yet have not been re-structured to work in a digitized context. There is no single methodology which represents a simple response to these factors but recently there have been ideas about the role of individuals and the human issues in information system development, especially in complex environments.

Object Oriented (OO) design has provided a new approach to C2 system design and maintenance. The main benefit of OO design is the re-usability of components which means it is not necessary to design and build systems by writing all software modules from scratch. Competitive commercial industries have flooded the market with component software that provides a multitude of functionality suitable for exploitation within C2 systems. There has been a shift in C2 systems development away from algorithms to component customisation using Commercial-Off-The-Shelf (COTS) products. New projects are rarely involved in programming or in developing algorithms as most core functionality(i.e. databases) already exist. Even C2 specific functionality can be created by combining existing functionality.

Limitations/problems of the methods for C2 systems

As mentioned previously, the structure and formal approach of SSADM can disregard the important factors of C2 development. Without sufficient involvement in planning the system users are left feeling that they have had a system imposed on them. Traditional system analysis methodologies had a strong technical bias and there was a relative neglect of the important human issues which are critical to the success of any C2 system.

The waterfall model was based on a number of assumptions:

- that all its stages could be completed in sequence
- that the costs and benefits of an information system could be calculated in advance
- that users knew what they wanted
- that the work needed was known and could be measured
- that programs once written could be altered
- that the right answer could be produced first time

For the development of C2 systems, none of these have been shown to be true. C2 systems have proved difficult to manage because of at least two major problems which cannot be dealt with by traditional management methods. These are (Grindley, 1993):

- the difficulty of stating what is required
- the difficulty of measuring pioneering work

The traditional life cycle approach effectively ignores these issues and developers have been forced to consider other methods to overcome these problems. One solution is to integrate prototyping into the user and system requirement processes.

Requirements Capture Process

Requirements Engineering is the capture, documentation and maintenance of the users requirements and their transformation into system requirements. It is the vital initial stage in the Systems Engineering process. Investment of time and effort at the requirements capture stage has been proven to increase the success rate of a project and reduce error rate and changes at later stages. This is especially important for large-scale projects as errors/omissions at the early stages are exponentially more difficult to resolve the later in the project life cycle they are discovered.

Capture and maintenance of requirements, both initially and during the life of the project, is a difficult and crucial task. The use of defined, proven techniques and tools greatly improve the accuracy and speed of requirements capture.

User Requirements

The user requirements phase defines the requirements of the user and produces a User Requirements Document (URD). The customer defines the scoping, users are defined and initial requirements captured.

The main techniques for capturing user requirements are interviewing users, analysing existing systems, modeling current and proposed processes, use of available documentation on requirements, processes, doctrine and operating procedures, prototyping and use of scenarios to validate/confirm requirements.

Initial interviews and analysis of current systems lead to an initial user requirements document. Process models and prototypes are created and discussed at workshops to validate understanding of processes and of requirements and to further refine them. User roles and types are identified and linked to requirements for traceability.

System Requirements

User requirements define the capabilities that the user wants from an operational point of view, not in terms of functionality or equipment. System requirements show the functions required of a system, they do not specify how this functionality is achieved but what the system must do based on the user requirements. They are the intermediate stage between the user requirements and the design and are documented in System Requirement Documentation (SRD).

Prototypes created for the user requirements can be utilised and discussed at workshops to validate understanding of system requirements and to further refine them. This is a key component to QinetiQ's approach to gathering and validating URDs and SRDs.

Use of Prototyping for Requirement Capture

The development of a prototype system is one of the best ways to successfully extract what the user really wants and for the designers to demonstrate what can be done. Prototyping is employed to model, explore and sometimes pre-empt the requirement changes that will result from emerging doctrine and technology. Prototypes can also be used to handle requirement changes.

Prototyping is the mechanism by which participation is ensured and through which the user's view of the system can be expressed. Prototyping is a response to the problem of not knowing either what the user wants or what he will do with it when he's got it!

Bestebreurtje (1997) expresses the problems prototyping addresses succinctly:

> 'Perhaps the most difficult problem in a project is to define what has to be built. In order to do this the client has to tell what the system should look like. But in practice*: 'They do not know!'* because for them it is the first system of its kind they have experienced and they lack the knowledge which comes from experience.'
>
> 'Human beings almost never perform a complex task correctly the first time. However, people are extremely good at making a mediocre beginning and then making small refinements and improvements.'

Prototyping has long been applied as an approach to systems development in the defence industry for everything from aircraft to tanks but has not been considered as a major component in systems development until recent years. Changing technology has opened up prototyping as a cost effective approach to capturing the key

requirements, functions and design previously attempted by the traditional design methodologies.

The implications of prototyping for C2 systems are enormous. The special characteristics of C2 – complexity, new technology, complex data types, security, validation and the difficulty of the technology all make C2 system development ripe for a prototyping approach. If simple, more transparent, less ambitious anaytical tools are deployed as prototypes, user participation becomes more of a reality. Prototypes activate social interaction and analytical methods need not be so complex.

The QinetiQ C2 Approach for a Rapid Development Methodology

The best way to successfully extract what the user really wants and for the designers to demonstrate what they can actually do is through the development of a prototype system. It also has to be accepted that users can change their minds and prototyping recognises this and actually encourages this as part of the process.

The Geospatial Decision Support Group, part of QinetiQ, have used these principals over the past five years to extract user requirements for large C2 projects and to develop smaller, focused solutions for C2 planning problems. The group have also expanded the wider concept of prototyping to take in the following concepts:

- a method to extract and validate user requirements
- a means of designing a user interface
- a quick and dirty approach to system building
- to refine data structure for database queries and testing
- to demonstrate proof of principal concepts
- to evaluate decision support interaction
- to validate models

The development of a successful prototype depends on a user-developer link where final user requirements can be developed only through an adaptive process of learning and evolution. The successful approach adopted by QinetiQ is to develop decision support for C2 systems using a ‘middle-out’ approach. The ‘middle-out’ approach relies on the quick delivery of an initial system to which users can respond and thus clarify what they really need.

Figure 2 *(below)* indicates how the user can evolve the system by customisation once they have initially learnt from the system. The *Designer-System* link adapts the system after the user has pointed out shortcomings and researched possible improvements.

The *System – User* link indicates that learning is stimulated by the system. After a period of use, the user discovers the full potential of the system and becomes familiar with new analysis techniques. The *User – System* link refers to any customisation made by the user, possibly using it in a manner that was not intended or envisaged by the designer. This allows the system to evolve around any existing planning process and it allows the user to experiment with existing planning processes using the new technology.

The two adaptive processes should work together as an effective prototype encourages the user to explore new alternatives and approaches to a task (*System – User*). This in itself stimulates new uses of the system (*User – System*).

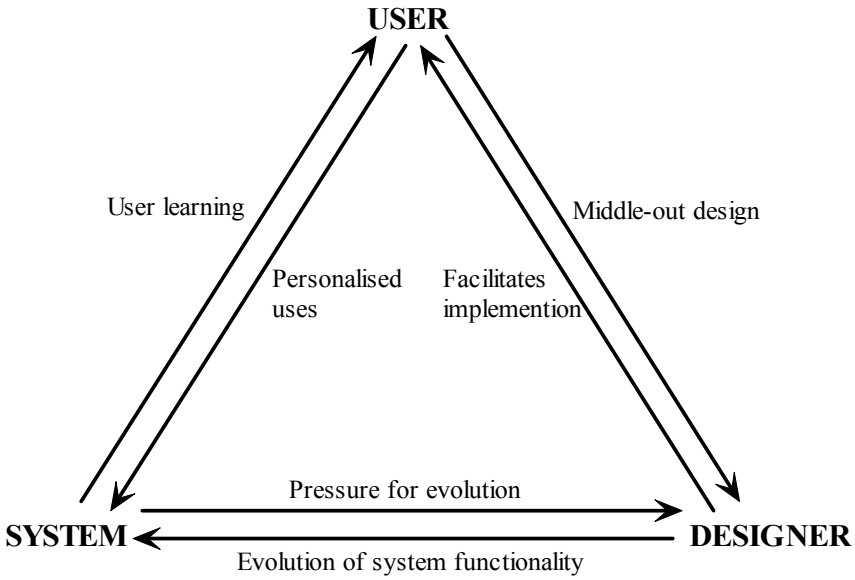

Figure 2: An adaptive framework for prototyping (Keen 1980)

The *User – Designer* link is a key aspect of adaptive design as users do not know what they need. Middle-out design allows the designer to learn from the user and ensures that the user drives the design process. The *Designer – User* link allows the designer to understand the users' perspective and processes. This is where the designers really get to grips with the nuances and details of the more intricate C2 decision making processes.

The *System - User* link (learning) and the *User – System* link (customisation) put a strain on the existing system. This builds pressure for evolution which is represented by the *System – Designer* link. The designer is forced to provide new functions through the *Designer – System* link and this drives the adaptive evolution of the prototype.

As mentioned previously, prototyping is an established part of engineering practice to which software developers have only recently woken up to. This practice is especially relevant to C2 systems as the introduction of new technology has a profound impact on the current doctrine and processes which means that some sort of process re-engineering has to take place. This involves not only the designers/analysts understanding the capabilities of the system in practice but the users going through some sort of education process. Prototyping is essential for this, as Hobson (1991) states:

> 'Some sort of prototyping, using the productive system or a simpler one, is the only way to really educate prospective users of new systems. Only by being shown what is possible can the users decide what is desirable, which itself must be tested for feasibility and understanding and shown once more.'

Generally prototypes can only adapt and evolve so much before they become 'bloated' and technically unfeasible to maintain and extend. Depending on the role of the prototype the usability of the system does not have to stop here. If a prototype is successfully capturing user requirements it can carry on as a demonstrator where the *Designer – System* link does not evolve the actual prototype but the feedback is captured in the User Requirements Documentation (URD). The whole prototype loop then becomes the main drive behind the URD process when the prototype demonstrator is distributed to everyone in the C2 community who wishes to establish themselves as future users of C2 systems.

Parallel Rapid Application Development (PRAD)

Over the past year QinetiQ have received Urgent Operation Requirements (UORs) from components of the British Army who have become engaged in the recent operations that have stemmed from the actions that took place on 11 September 2001. UOR's require immediate attention and in these circumstances developers are given extremely short timescales to provide a system solution to the problem in hand.

One such UOR required the modelling of ballistic trajectories for the complex terrain of Afghanistan. The emphasis was on providing a planning system that would allow the correct placement of an artillery gun in an environment that presented many cresting problems in a situation where it was subsequently very difficult to move the gun once it had been deployed by helicopter.

A 'quick and dirty' approach to system design was adopted to enable a laptop system to be placed in the Theatre of Operation within five weeks. Prototypes were developed to test ballistic models and terrain analysis techniques before the system was deployed. Once deployed to Afghanistan a new approach to the prototyping loop illustrated in Figure 2 was established to gain user feedback while the system was in-theatre.

Secure email and telephone connections were established between the developers in the UK and the operational headquarters in Baghram. The system users provided regular feedback and requested enhancements to the system. The developers were able to respond to these requests and prepare new software solutions. The software was placed on a testbed that consisted of an identical laptop to one that had been deployed to Afghanistan. A new system could then be emailed to the user with the confidence that it had worked on an identical laptop setup. This technique was iterated several times during the conflict and gave birth to the Parallel Rapid Application Development approach using parallel systems that is illustrated in figure 3.

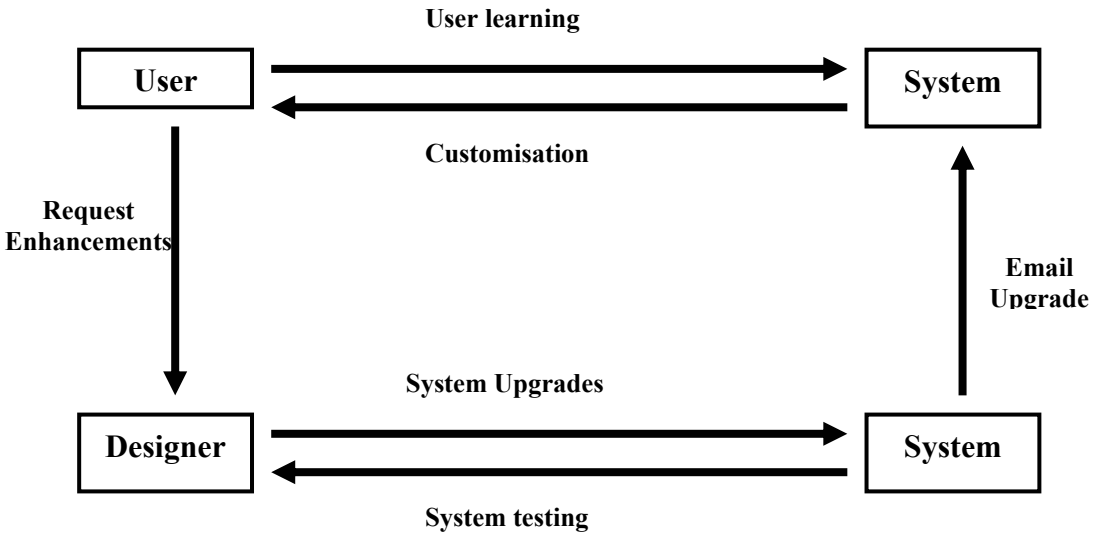

Figure 3: Parallel Rapid Application Development

The circumstances around the development of this artillery system have demonstrated that with rapid pressure and feedback along the paths in Figure 3, users gain access to new capabilities that enable them to explore the application of technology in new areas. This is unlikely to occur if a traditional life-cycle approach to system design is used.

References

Bestebreurtje, J. G. A., 1997, GIS Project Management, (Manchester Metropolitan University).

Densham, P.J., 1991, Spatial Decision Support Systems. In Maguire, D. J., Goodchild, M. F. and Rhind, D. W., (Eds.) Geographical Information Systems Principles and Applications, (London: Longman).

Gindley, K., 1993, Managing IT at Board Level: The Hidden Agenda Exposed, (London: Pitman).

Hobson, S. A., 1991, Methodology issues in GIS Introduction. AM/FM Conference Proceedings, pp 207-211.

Keen, P.G.W. 1980. "Adaptive Design for Decision Support Systems". Data Base vol. 12. Fall 1980. pp.15-25.

Reeve, D. and Petch, J., 1999, GIS Organisations and People, (London:Taylor & Francis).

Stevens, R., Brook, P., Jackson, K. and Arnold, S., 1998, Systems Engineering: Coping with complexity, (Prentice Hall Europe).