International Journal of Business, Human and Social Sciences
ISSN: 2517-9411
Vol:1, No:9, 2007

Iterative Way to Acquire Information Technology for Defense and Aerospace

Ahmet Denker, and Hakan Gürkan

***Abstract*—**Defense and Aerospace environment is continuously striving to keep up with increasingly sophisticated Information Technology (IT) in order to remain effective in today's dynamic and unpredictable threat environment. This makes IT one of the largest and fastest growing expenses of Defense. Hundreds of millions of dollars spent a year on IT projects. But, too many of those millions are wasted on costly mistakes. Systems that do not work properly, new components that are not compatible with old ones, trendy new applications that do not really satisfy defense needs or lost through poorly managed contracts.

This paper investigates and compiles the effective strategies that aim to end exasperation with low returns and high cost of Information Technology acquisition for defense; it tries to show how to maximize value while reducing time and expenditure.

***Keywords*—**Iterative process, acquisition management, project management, software economics, requirement analysis.

I. Introduction

DEFENSE and Aerospace sector had adopted waterfall standard, but they experience significant failure(estimates of about 70% of IT projects are long overdue or unusable), unfortunately the legacy of waterfall still confuses IT projects in Defense.

The largest contribution to this failure is to attempt full requirements definition at early stage. In defense and aerospace projects, there is a long gap before these requirements are delivered.

Our survey results agree with literature that on average 25% of the requirements change on IT projects. Thus, spending significant portion of time, budget and effort trying to define requirements to the maximum level is inappropriate and wasteful.

The results of our survey showed that 45% of the features defined at the requirements analysis stage were useless. Figure 1 shows the graphical presentation of the survey results.

Manuscript received October 14, 2005.

Ahmet Denker is with the Electronics Engineering Department, Ankara University, Besevler, Ankara, 06100 Turkey (e-mail: denker@eng.ankara.edu.tr).

Hakan Gürkan is with the Electronic Warfare Department, Turkish Airforce, Etimesgut, Ankara, 06100 Turkey (corresponding author to provide phone: +90-312-212 67 20 ext.1466; fax: +90-312-212 54 80; e-mail: hgurkan@science.ankara.edu.tr).

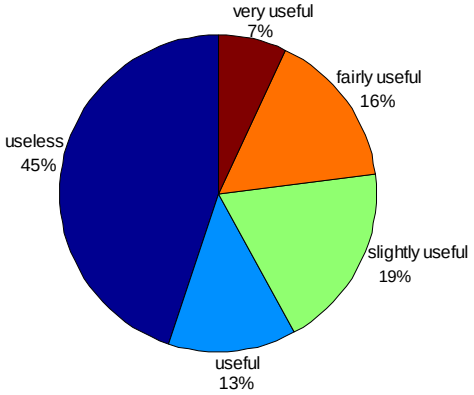

Fig. 1 Utility of the requirements defined at the analysis stage

II. Why IT Projects are Different

IT projects are different from other projects because they are domain dependent(Fig. 2). That's, it is not sufficient to manage the project itself, but they also require the management domain level. When the domain is defense environment, the task is even more difficult to achieve. Why?

- There is only one customer.
- There is a different corporate culture.

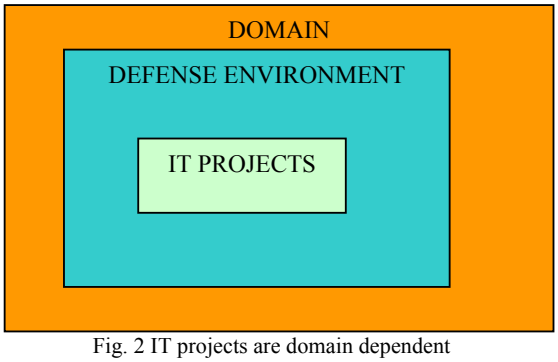

Fig. 2 IT projects are domain dependent

There are many different types of approaches at domain, methodologies, and models utilized in application development, but all have a common basis in IT development that involve defined steps to include: project evaluation and planning, requirements development (analysis and specification) and definitions, system design, program design, program implementation (coding), unit testing, integration testing, system testing (verification and validation), system delivery (implementation) and system maintenance, similar to

those described in the IEEE/EIA 12207 or ISO software life cycles and the Software Engineering Institute Capabilities Maturity Model (CMM) [1,2]. In discussing government and industry standards we feel it relevant to touch on the evolution of Defense software standards, specifically MIL-STD-498 "Software Development and Documentation," which endorsed Defense standards being converted to non-government standards and maximize the use of commercial practices in government software projects. MIL-STD-498 required industry to participate during the proposal phase of IT projects and recommend commercial solutions. Based on the processes, methods, and software engineering environments; MIL-STD-498 incorporated industry's best practices to include new developmental methodologies. MIL-STD-498 was later replaced by IEEE/EIA Standard 12207 in May, 1998. The "commercial" IEEE/EIA 12207 Standard expanded the scope of MIL-STD-498 to include: Standard for Information Technology: Software life cycle processes, Life cycle data, Implementation considerations, and it also specify the acquisition process from pre-contractual initiation of a project to acceptance and completion. The standard details a sequence of steps that the user and developer must undertake to assure a quality IT product [3]. This transition from Defense specific software requirements to commercial standards illustrated the basic need to combat the ever increasing cost associated with government IT projects (and the critical drivers specific to IT and methodologies in general) while taking advantage of the common software improvement activities in industry. With the advent of new technologies and the need for systems that focus on information intensive applications, there is a driving need for new iterative developmental approach that can rapidly adapt to the changing environment.

III. IT Acquisition Process

Acquisition includes design, engineering, test and evaluation, production, operations and support of defense systems. As used herein, the term "Defense acquisition" generally applies only to weapons and information technology systems, processes, procedures, and end products. Our goal is to make Project IT Acquisition process more efficient by decreasing the necessary time and expense for development.

Defense software-intensive systems are those for which software is the largest segment of the system development cost, development risk, functionality, or development time. Such systems are complex and must satisfy a wide spectrum of user requirements gleaned from diverse user communities. Defense software-intensive systems can be broken into the following three broadly generic categories: 1) Automated Information Systems; which include classic Information Technology and Management Information Systems for which privacy is typically a critical requirement; 2) Command, Control, Communications, Computers, and Intelligence Systems; those systems that assist mission planners and combat commanders in mission planning, control, deployment, and employment of defense for which security is typically a critical requirement; and 3) Weapons Computing Systems those embedded computer systems that are typically high performance, real-time systems designed as an integral part of a larger weapons system, and used by the Army for combat missions for which safety is typically a critical requirement.

Typically, a Software Development Plan (SDP) or an equivalent management plan has been used in Defense acquisition programs by developers to formally document their plans for the software development. Prepared by the developer, SDPs typically address:

The development process to be used, standards for products, reusable software, The handling of critical requirements (safety, security, and privacy assurance), computer hardware resource utilization and allocation, provisions for acquirer access during development, program planning and oversight, software testing, joint technical and management reviews, schedules, activity networks, program organization and resources.

Other plans, depending on the life cycle management standard being used (e.g., JSTD-016 or IEEE/EIQ 12207) may be placed upon the developer by contract in the form of such items as a software test plan, a software quality assurance plan, and/or a software safety plan, among other developer-prepared plans. A key acquirer-prepared plan that, while not required by defense policy, is encouraged at the service level, is a Computer Resources Life Cycle Management Plan (CRLCMP). Format of the CRLCMP varies widely, but it typically includes identification of major computer resource acquisition management and support risks, identification of critical issues, metrics and measures [4].

IV. Acquisition by Iterative Method

Building prototypes is an essential part of the Iterative Method. Prototype enables users and developers to examine some aspects of the proposed system and decide if it is suitable or appropriate for the finished product [5]. It is a technique for reducing risk by buying information. Knowledge is gained through creating a physical model without adding the effective means for communicating with the user community or the implementation details [6]. Prototyping ensures that the desired standards and requirements will be met by the final product. It can also aid developers in evaluating which model approach is most advantageous for a specific task and identifying the main requirements of a system [7].

We mostly visualize prototyping as constructing a scaled-down version of the system under development, which usually has limited functionality. Sometimes this is done to help stakeholders identify requirements and to aid the developers in determining if they are on the right track with the design or taking the correct approach. Generally, the current thinking on prototype development follows this model. A prototype is built, and then it is appraised for its functionality. It usually receives feedback from the stakeholders, who evaluate the functionality and determine from there any improvements that can be made. Subsequently, they either incorporate these

changes into a second prototype or incorporate the knowledge gained to the actual production model. In order to get to this point, developers use one or more of the process models listed below.

A. Iterative Method

The Iterative model (IM) performs the waterfall in overlapping sections, thereby attempting to produce usable functionality earlier in the project life cycle. This allows the development team to demonstrate results earlier on in the process and obtain valuable feedback from system users. As some modules are completed before others, well-defined interfaces are required. In addition, there can be a tendency to push difficult problems to the future to demonstrate early success to management. The Iterative Model can be used when it is too risky to develop the whole system at once.

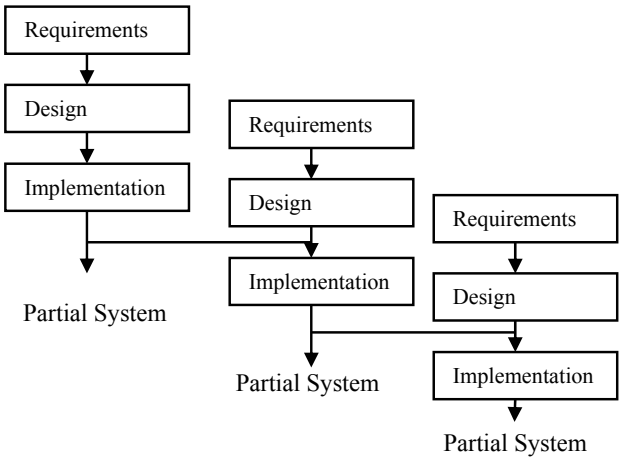

Fig. 3 The Iterative Model

The Iterative Model tackles many of the problems associated with the Waterfall Model; however, it does present new issues [2].

- Users need to be actively involved throughout the project. While this involvement is a positive for the project, it is demanding on time,and staff and can add project delay.
- Communication and coordination are a must during project management, i.e. requests for improvement after each phase may lead to confusion - a system for handling requests will have to be used.
- The Iterative Model can lead to "scope creep," since user feedback following each phase may lead to increased user demands. As users see the system develop, they may realize the potential of other system capabilities, which would enhance their work. This is one of the big slow-down areas when using this model, which of course, takes longer to get the finished product to the market.

Unlike the waterfall approach, the IM approach is dynamic and provides capability to the users in varying increments or stages. The IM approach provides an integrated process that allows users, developers and PMs to interface and validate the status of the program from fielding of the initial to the final capability. Because the user is continually involved, adjustments to the core requirements are fairly straightforward. The major drawback to this approach is uncertainty. Constantly improving and developing technology creates difficulty in ascertaining risk associated with cost or schedule. In short, technology is both the benefit and weakness of the IM approach.

TABLE I
COMPARISONS OF THE WATERFALL AND ITERATIVE METHOD

Issue	Waterfall model	Iterative Model
Requirements	Requirements are known at the start and remain stable throughout the program.	Requirements are broadly defined. They evolve and are refined as the program develops.
Design /Technology	Determined Early in the Program or when mature.	The basic architecture and initial functions are determined early, but the detailed designs and other functions evolve. Cutting edge but fully tested.
Acquisition Cost	Known based an awarded contract. Capped based on contract type.	Known based on the Phase-I contract. Follow-on phases are estimated and capped.
Risk	Performance: Higher because of the single step approach. One chance at success. Management: Lower because requirements are stable. Cost: Lower because technology is mature.	Performance: Lower because of iterative approach. Several opportunities for success. Management: Higher because of requirements uncertainty and dynamic nature of project. Cost: Higher because of uncertainty and high cost of cutting edge technology.

With all the changes in today's culture, nothing is ever fast enough. Current software and system engineering research has promoted rapid prototyping, which is a combination of both a throwaway prototype and an evolutionary prototype, where sections of the proposed system are built in order to determine the viability of requirements. This type of prototyping, which integrates requirements, design, completion, and testing in one step, aids in understanding the requirements and determining the ultimate design [5].

Our survey results show that the above models have become relatively obsolete and a revolutionary prototype for software can be effectively designed, after requirements are identified, that can essentially go straight into production after testing, saving organizations significant amounts of both time and money. The resulting iterative method is outlined as follows:

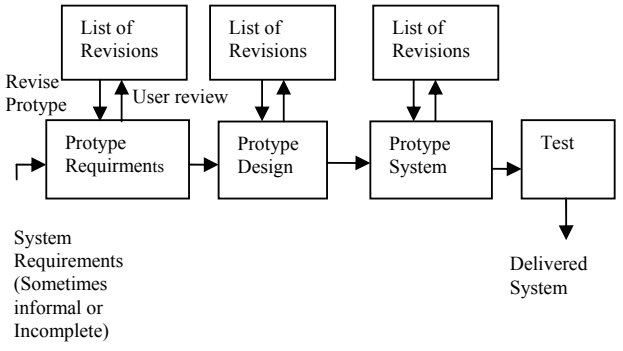

Fig. 4 Iterative Method Outline

This model can be the foundation for a successful process model where there is continual analysis so that the user, developer, and customer are aware of what is needed and anticipated. One or more of the loops for prototyping requirements, design, or the system may be eliminated depending on the goals of the prototyping [5]. In this model, the prototype is developed, refined, tested, and sent on to production. This model will save time for the developer. We found that even though this model looks different, it incorporates the benefits of the other models we researched without the disadvantages. While using this model, the prototype is iteratively being revised throughout each development phase in the least amount of time without sacrificing quality. This keeps the project moving, especially when the scope or requirements from defense and aerospace client would unexpectedly change.

V. Iterative Requirements Development

Gathering requirements plays a large role in prototyping. Most of the time in systems application development, requirements are not entirely known prior to the development efforts of system prototypes. In most cases this tends to slow the overall systems development cycle (as a direct result of the lengthened time between prototype and production). However, in systems where the requirements are more fully known, a system could be produced with the "working prototype" put into immediate production.

Requirements development consists of the following three related activities: gathering candidate requirements, specifying requirements, and analyzing requirements. Gathering candidate requirements are done by interviewing potential customers about the system they want, reviewing competitive products, building interactive prototypes, and so on [8]. This statement may appear to be a generalization to the inexperienced developer, but we feel it is perhaps the most important aspect of requirements analysis. Major reasons for the problematic nature (or resultant failures) of a software project can be directly related to a lack of detailed requirements or inadequate systems specifications; both of which can lead to project creep of both scope and time…and those two factors relate to increased cost.

In order to extract significant requirements, the dynamics of basic requirements principles must be identified and criteria for measuring those requirements be established [9, 10]. The basic principles of significant requirements are listed as follows:

- Requirements extraction follows a formal process
- All customers, users, stakeholders are identified – different viewpoints of the system are utilized
- Requirements are not simply taken as given but are re-validated using in-depth interviews
- Requirements statements avoid methods of implementation
- Requirements are testable – testers are involved in requirements definition
- Requirements are documented (hierarchical structure and shows traceability of requirements)
- Documentation has version numbers A formal change procedure is used
- Requirements are prioritized

VI. Conclusion

Defense technology is becoming increasingly complex and diverse, demanding more flexible and shorter acquisition procedures. Armed Forces are facing less predictable threats and a wider range of tasks, so new technology needs to be deployed more quickly. The Armed Forces can not keep pace with the rate of technological change which in many areas now commercially led. In the present procurement procedure and organization, major weapons system are still taking some twenty years to bring into service, cost continue to exceed planned levels and reliability and maintainability of new equipment frequently remains a problem.

The present acquisition process can not strike the right balance between cost, time, and performance in the very early stages of a project. Insufficient investment in the risk reduction at this stage has cost more to Armed Forces later on. The present procurement process and organization has deficiency to give project managers sufficient delegated authority. They have also failed to provide properly targeted incentives to both contractors and staff [11].

Many Defense analysts believe the conduct of warfare is entering a period of fundamental change, literally, a "revolution in military affairs," driven by advances in information technology and precision guided weapons. Past experience suggests that revolutions in military affairs are not produced solely by rapid technological advancements, but also require changes to prevailing operational concepts, doctrine and force structure to fully harness the technology in a manner to dominate the battlefield.

As corporations and government organizations continue to downsize and outsource in an attempt to restructure cost, they must rethink the way they exploit iterative acquisition method in order to save their organizations both time and money. The Ministry can certainly benefit by utilizing findings (from other shared innovations that have been mutually beneficial between the civil and government sectors) as it continues on a path of transformation intended to radically improve its enterprise-wide business processes.

Defense IT Acquisition Management has a great need for iterative approaches, because it has significant investment in current systems and a limited budget for innovation. It's important to point out the need for the development of iterative approaches that are paramount to not only the future of Defense operations, but to industry as a whole. The requirement for increased innovation on limited budgets is a reoccurring theme among many organizations regardless of whether they are a Defense or commercial entity. Although our initial focus discusses the requirements analysis phase, we are in no way diminishing the relevance or importance of the project evaluation and planning phase. The requirements development process is where that critical link or relationship between user and developer is consummated in order to produce a clearly defined specification.

We identified what we believe is a pertinent development strategy recommendation for requirements analysis for future IT project applications. In gaining a greater understanding of the requirements analysis phase of application or use of IT, the first requirement is to understand the basics of the overall use of IT cycle.

The relative importance of a structure requirements analysis approach is that it greatly enhances the strategy of system development. Structured requirements analysis in combination with the most promising and desired approaches and methodologies of Prototyping, we believe will result in a strategy that will allow any developmental initiative to effectively take a concept from prototype to production the most efficient and effective means.

The iterative acquisition method was iteratively being revised throughout each development phase in the least amount of time without sacrificing quality. This kept the project moving, especially when the scope or requirements from our client would unexpectedly change.

REFERENCES

[1] IEEE/EIA 12207 Standard for Information Technology – Software Life Cycle Processes or relevant International Standardization Organization (ISO) standards. They define a set of recommended development activities and documentation alternatives for software intensive systems.
[2] The Software Engineering Institute (SEI) Capability Maturity Model (CMM) for software development - Feb., 1993.
[3] Barrow, Patrick D. M. and Mayhew, Pam J. "Investigating principles of stakeholder evaluation in a modern IS development approach." Journal of Systems and Software 52, Iss. 2,3 (June 1, 2000): 95-103.
[4] PMI, A Guide to the Project Management Body of Knowledge (PMBOK® Guide) First Edition Version 1.0 June 2003.
[5] Pfleeger, Shari Lawrence, Software Engineering: Theory and Practice, Upper Saddle River: Prentice Hall, 2001.
[6] Hall, Elaine, M. Managing Risk: Methods for Software System Development, Boston: Addison-Wesley, 1998.
[7] Housel, Thomas J. and Bell, Arthur H. Measuring and Managing Knowledge. McGraw-Hill Irwin, 2001.
[8] Software Project Survival Guide, Requirements Development, Steven C. McConnell, http://www.stevemcconnell.com/sgreq.htm
[9] Ralph R. Young, Effective Requirements Practices, pp82-83, Addison-Wesley, 2001.
[10] Malhotra, Yogesh. "Knowledge Management for e-Business Performance." Information Strategy: The Executives Journal (2000).
[11] Strategic Defense Review, SDR, 1998.

Ahmet DENKER. B.S in EE, Bogazici University (1977); M.S. and Ph.D Sussex University, England (1978-1981). Dr. Denker worked as a full time professor at Bogazici University between years 1982 and 1999, gave lectures on Control and Robotics and supervised several theses. He was entitled "Chartered Engineer" (C.Eng.) by the Council of Engineering of U.K.. His particular areas of interest are applications of industrial control and Information Technology. He was a project manager at the Robotic Science Division of TUBITAK (Turkish Scientific and Engineering Research Council) Marmara Research Center between 1992 and 1996 whereby he lead a team working on an industrial robot with visional capabilities. He held visiting positions in Sussex University (U.K.), Open University (U.K.), Eastern Mediterranean University (Cyprus) and Keio University (Japan). In 1996 he was awarded a medal by Matsumae International Organization of Japan for his contributions to science and peace. He acted as the General Manager of HAVELSAN Inc. between years 1996 and 2003. Through his leadership, Havelsan had undertaken critical responsibilities in initiating various e-government, IT and defense industry projects which elevated HAVELSAN to the top 100 defense companies world-wide as well as to number one IT company in Turkey. Dr. Denker is a full time professor at Ankara University since 2003 and giving lectures in Information Systems Engineering and Management.

Captain Hakan GÜRKAN currently serves as the chief of the Electronic Warfare and Missile Systems Branch for Turkish Air force of the Air Logistic headquarter in Ankara. He earned his BS and MS in Electronic Engineering from Ankara University. He carries 8 years of high technology experience where he worked as a project engineer, project manager and trainer of fighter pilots. His research interests are project management as a system engineering and Requirement Engineering.