Software Design Issues: A Very Large Information Systems Perspective*

Gerald B. Williams, Chunka Mui, Vairam Alagappan, Bruce B. Johnson

Andersen Consulting
Center for Strategic Technology Research
100 South Wacker Dr.
Chicago, Illinois 60606

Abstract

Research in software engineering is concerned with the enhancement and automation of the processes of building computer application systems. While there is a broad consensus on the problems associated with software development, a specific understanding of the software engineering problem and the appropriate solutions are inevitably driven by the target application domain. Much of the current software engineering research is driven by the development of large scale embedded software systems. Our understanding of the problems is based on a different domain: that of very large information systems (VLIS). In this paper, we identify some significant software engineering problems from the context of developing very large information systems.

1. Introduction

The ultimate goal of research in software engineering (SE) is to improve the productivity and quality associated with both the processes and products of software development. The foreseen benefits are improvements in the management of the development process, increases in user satisfaction, and the delivery of greater functionality to the marketplace in the form of software and related products. As a means to this end, software engineering research is aimed at reducing costs and improving time in production by automating portions of the development process.

Some of the major problems associated with the automation of software development occur with respect to requirements specification, design, maintenance, reusability, validation, verification, and testing. While there is a broad consensus on the existence of these problems in general, a specific understanding of the software engineering problems and their appropriate solutions is inevitably driven by and is sensitive to the target application domain. *Software engineering solution methods are dependent on the application domain.*

Much of the current software engineering research is being driven by the development of large scale embedded software systems. The nature of an embedded system can be characterized by asynchronous parallelism and urgent performance requirements [1]. The concentration of effort in this domain has resulted in attempts to develop formal, high level specification languages, methods of specification validation, and methods of transforming specifications into executable code.

Our approach to research on the problems of software engineering is founded on a different class of application domains; namely that of very large information systems (VLIS). In the following sections we briefly define and characterize VLIS and the problems associated with VLIS development. The intent is to contrast the nature of the software development problems related to large information systems with the development problems of the commonly investigated domain of embedded systems. As a result we hope to illuminate new perspectives on the software engineering problem and construct a foundation from which an integrated, productive research program can be launched for software engineering issues in VLIS.

2. VLIS Viewpoint

An information system is not a single monolithic structure. VLIS are federations of subsystems developed according to a system wide design plan to provide information to support the operational, managerial, analytical and decision-making functions of an organization. These subsystems are integrated and evolved over time for the purpose of supporting these organizational functions.

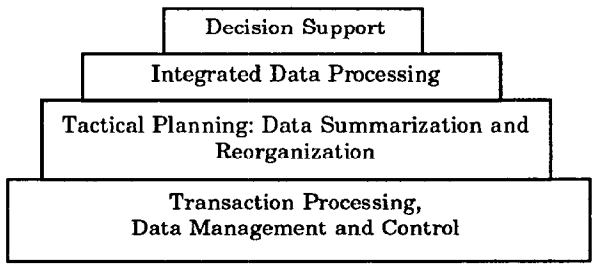

Figure 1: Information Systems Pyramid

* The full version of this short paper is available from the authors.

Permission to copy without fee all or part of this material is granted provided that copies are not made or distributed for direct commercial advantage, the ACM copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Association for Computing Machinery. To copy otherwise, or to republish, requires a fee and/or specific permission.

©1989 ACM 0-89791-305-1/89/0500/0238$00.75

The operational scope of VLIS across different levels of business activities can characterized as a pyramid structure similar to that shown in Figure 1. Each layer of the pyramid represents a class of information processing activities and provides a processing foundation for the levels above it. As one ascends through the layers of the pyramid the complexity of the business problem being addressed increases, additional use of business specific knowledge is required, the degree of integration across the activities increases, the accumulation of information increases and the form of the information being processed becomes more complex.

As is the case with embedded systems, VLIS development is certainly concerned with the functional complexity and the performance of software. However, in contrast to embedded systems, VLIS are more strongly characterized by relatively simple functional demands, massive size, enormous volumes of data, continuous use and constant evolution. In essence, the complexity of a VLIS is rooted in the size of the system and the demographic complexity of the application environment.

Base upon our experiences in developing very large information systems, we believe a significant amount of progress can be made to enhance and automate software development methods for VLIS by exploiting the unique properties of those systems. These properties place a set of real-world constraints on the systems development process and will allow problems which, in the general case, seem intractable to be addressed.

SE research therefore needs to contain a strong empirical component and be grounded by data generated from industrial experiences. The formulation of research problems and the strengths and weaknesses of the alternative approaches need to be subjected to the test of industrial practices, time constraints, resource limitations and economic pressures.

3. Software Development Issues

This section describes some of the issues associated with the software development process in the VLIS context. In each case we define the issue, identify the sources of its existence, and discuss the implications of what needs to be done. We see these issues as highly interdependent and regard the approach to an overall solution as necessarily dependent on concurrent development in each area.

3.1 System Evolution

In current practice, the process of maintenance is too often thought of as the patching of software bugs. Software maintainability is wrongly assumed to be a natural by-product of good initial systems development. These perceptions are faulty and contribute to the maintenance crisis being experienced today. In order to support the full scope of maintenance, it must be recognized that software systems need to *evolve* and that this evolution cannot be treated in the same manner as initial development.

The ability to continually evolve is crucial for VLIS. Information systems are typically a critical arm of the business and must adapt continually to the changing business environment. *Change is not the exception for these systems, it is the norm.* Unless they can effectively adapt to the needs of the company, the stability and competitive position of the company will be jeopardized.

In order to support system evolution, processes which directly support maintenance must be developed and information required during maintenance (design rationale) must be available. Taken together, design rationale and the maintenance processes must help the designer (1) understand the functionality, structure, and behavior of a system (2) assess the impact of alternative changes, and (3) control the impact of necessary changes. Of course, the integrity and maintainability of the system must be preserved.

Supporting system evolution must be viewed as a major driving factor in shaping an approach to formalizing the design process. Maintenance requires more information because an existing system represents a larger set of constraints that does not exist during initial design and the design tradeoffs made during maintenance differ from those in the initial design process. For example, data structure decisions may be driven by efficiency during initial development but by minimization of impact during maintenance.

3.2 Leveraging Expertise and Experience

The necessarily large size of VLIS development teams coupled with the relative scarcity of highly experienced systems builders mandate the judicious use of less experienced personnel in most phases of the development effort. Leveraging expertise and experience is concerned with the ability to manage and successfully complete large software development projects with a relatively small amount of highly experienced, proven resources. This is a critical issue given that the success of a development project is directly correlated with the levels of experience and talent across members of the project team [2].

The focus of SE research should be on supporting the high level design activities where the decomposition of high level problems into relatively independent components take place. The simple functional demands of individual programs allows for their construction by less experienced personnel.

3.3 The Design Process

The design process is the task of creating or generating design artifacts and subsequently evaluating, refining, integrating, and modifying these artifacts until the result satisfies the requirements of the problem definition. In essence, the design process is the task of mapping problem requirements into design solutions. The design process should be guided by a productive, economic, and controllable methodology that will ensure a high quality product.

The reasons why the design problem is important are fairly obvious. Good design decisions made early have a positive

effect on the quality of the ultimate product as well as the efficiency of the development process. Poor design decisions play havoc with the quality, efficiency and cost of the development process and the design products.

Design is characterized by a necessity to deal simultaneously with a large number of diverse constraints that are highly intertwined. In general the design problem is intractable. However, we believe that the combination of the restricted domain of VLIS development, the simplicity of its algorithmic requirements, and the knowledge accumulated from the experience of building a large number of VLIS provides good insights into the available expertise, known solutions and alternatives. Our intention is to exploit the natural structure of the domain in ways that allow us to reduce the complexity of the problem to manageable levels. We cite several issues that help focus some of the design process concerns.

The Paradigm Problem: The paradigm problem refers to the failure to recognize and develop a manageable, productive, economically feasible process model for SE. Much attention has been focused on the development of new software engineering paradigms [3, 4] but no results have proven completely satisfactory for VLIS development. Any successful model must deal with the interdependent facets of making design decisions while recognizing the need for adequate leverage and project management.

Bridging the Functional to Technical Gap: A significant portion of the design process occurs during the creative process of translating the business problem description to a high level systems design. Presently techniques at this level of the development process are not adequately understood. The result is an inability to adequately map problem requirements to technical solutions. No good "language" has been developed in which the requirements of the problem can be expressed and ultimately transformed to technical solutions.

Design Evaluation: In order to make good design decisions, one must have the ability to assess the validity of a particular design decision or weigh the relative merits of competing design alternatives. The lack of evaluation ability leads to inadequacies in assessing the impacts of a design decision on all levels of the design process. Under the umbrella title of evaluation we include testing, validation, verification and, as a specific instance, prototyping.

The Representation Problem: The representation problem is a fundamental requirement for advancement in each of the areas mentioned above. The issue is the ability to express, manipulate and make inferences about design objects and processes. In current practice, major development takes place at very low levels of design for at least two reasons. First, current methods of software engineering encourage designers to think in terms of low level issues such as databases, data structures, performance measures, screens, and interfaces because low level representations are the only mechanisms that provide feasibility measures and evaluation feedback. As a result designers move quickly to lower levels without adequately investigating alternative early design decisions. Second, the nature of the business is that cost pressures often do not allow for an adequate investigation of high level design alternatives. As a result projects designs are often inadequate and brittle.

3.3 Large Scale Integration

The problem of large scale integration is concerned with the understanding, use and exploitation of the functionalities, standards, protocols, and communication interfaces of a heterogeneous set of technologies in developing large systems. Integration of component systems differing both in functionality and platform is important because systems within the commercial environment can no longer be treated as isolated entities. In fact there is great disincentive to do so.

The large scale integration problem points out the need to understand the interrelationships of all systems within a company. Efforts need to be concentrated on large scale design at the *enterprise*, or company wide, level before detail design and implementation of a particular component is undertaken. Understanding the enterprise level connectivity and integration issues is extremely important and will have tremendous impact upon the design process.

4. Conclusions

The intent of this paper is to discuss the implications that the process of developing very large information systems (VLIS) has on the approach to the software engineering problem. Although the same SE problems are found in many domains, they take on a unique set of constraints when considered in the context of developing VLIS. It is our position that possibilities for enhancing the software development process are functions of the domain in which one participates.

Acknowledgements

We wish to thank Mehdi Harandi, Wojtek Kozaczynski, and Bill Sasso for their contributions to the ideas presented in this paper.

References

[1] Pamela Zave, "An Operational Approach to Requirements Specification for Embedded Systems," in New Paradigms for Software Development, (ed. William W. Agresti), IEEE Society, Los Angeles, Ca., 1986, 159 -178.

[2] Bill Curtis, Herb Krasner, Vincent Shen, & Neil Iscoe, "On Building Software Process Models Under the Lamppost," in the Proceedings of the 9th International Conference on Software Engineering, Monterrey, CA., 1987, 96-103.

[3] Barry W. Boehm, "A Spiral Model of Software Development and Enhancement," *IEEE Computer*, May 1988, 61- 72.

[4] Cordell Green, David Luckham, Robert Balzer, Thomas Cheatham, and Charles Rich. "Report on a Knowledge-Based Software Assistant," Kestrel Institute, 1983.