1 of 1

COORDINATION OF SOFTWARE-DEVELOPMENT ACTIVITIES
AMONG SITES THAT ARE GEOGRAPHICALLY SEPARATED*

L. F. Truett J. P. Loftis P. C. Shipe
Oak Ridge National Laboratory
Oak Ridge, TN 37831

E. Z. Faby J. W. Grubb
University of Tennessee
Knoxville, TN 37996

June 29, 1993

Paper Submitted to
Seventeenth Annual International
Computer Software and Applications Conference (COMPSAC93)
November 3-5, 1993
Phoenix, Arizona

"The submitted manuscript has been authored by a contractor of the U.S. Government under Contract DE-AC05-84OR21400. Accordingly, the U.S. Government retains a nonexclusive, royalty-free license to publish or reproduce the published form of this contribution, or allow others to do so, for U.S. Government purposes."

*Research sponsored by the Military Traffic Management Command under Interagency Agreement DOE No. 1405-1351-A1 and the U.S. Department of Energy, under contract DE-AC05-84OR21400 with Martin Marietta Energy Systems, Inc.

MASTER

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

Coordination of Software-Development Activities Among Sites That Are Geographically Separated

L. F. Truett, E. Z. Faby, J. W. Grubb, J. P. Loftis, P. C. Shipe

Oak Ridge National Laboratory
Oak Ridge, Tennessee

Abstract

The Department of Defense is sponsoring development of a centralized, integrated database, which will be a repository of international surface cargo movement data. Scheduled for implementation in early 1994, the system will include a relational database management system, processing modules, and complex communication components. Four geographically separated sites are involved with development. Design and development of a major computer system is never simple, but when design and development occur at multiple sites, the problems are compounded, especially when the timeframe for project completion is extremely tight. Issues such as identical developmental platforms and communications strategies must be addressed. A design plan must be strictly followed to ensure consistency and to coordinate integration of modules developed at different sites.

1: Introduction

This paper describes problems faced by developers working on complex software development projects with tight deadlines at geographically separated sites. A project that faced these problems is presented as a case study. Solutions are discussed.

2: Problems

Design and development of a major computer system is never simple, but the problems are compounded when design and development occur at multiple sites which are widely separated geographically and the timeframe for project completion is extremely tight. Identical developmental platforms must be acquired and consistent design strategies must be used to ensure that the same program running at different sites will achieve consistent and identical test results.

Clear communication among sites is essential to coordinate tasks and establish system requirements and design. Finally, responsibilities must be clearly defined. To enhance maintainability, the code should appear to be written by a single individual. We are all aware that programmers sitting side by side can probably write more consistent computer code than programmers working at sites located hundreds of miles apart. The problem is to define responsibilities unambiguously and to maintain high standards for quality and for consistent workmanship.

3: Case Study

The purpose of the system being described in this paper is to accept data from transportation systems and organizations handling military cargo passing through common-user ocean ports; to process, redistribute, and maintain this data; and to provide a central tracking mechanism for inquiring about or obtaining status information on a piece of cargo at any point in its transportation history. The system will be used worldwide in both peace and war. It will consist of a centralized data repository and regional processing hubs. Data exchanges will occur between these regional hubs and another cargo management system, which will be located at the ocean ports. Both systems are under development by the Department of Defense (DOD). Designed around open-systems, client-server technology, processing can occur at the ports, the hubs, or the central data repository [1].

The database and processing modules of the system described in this paper and the processes of the companion port-level system must be integrated by early 1994 when both systems are fielded. The four sites involved in the development are located on the East Coast, the West Coast, the Washington, D.C., area, and Oak Ridge, Tennessee. The West Coast Site (WCS) is not directly involved in the development of the

integrated database; however, this site is developing the port-level companion system, which is the database's primary interface. Thus, changes in this port-level system's design affect development of the centralized database. The primary responsibility of the Washington, D.C., site is management and coordination of the project, and this site is referred to throughout the remainder of this paper as Headquarters (HQ). The East Coast Site (ECS) and Oak Ridge National Laboratory (ORNL) have responsibility for design and development of the integrated database, related processing modules, and all communication components.

4: Discussion

4.1: Establishment and maintenance of an identical development platform at every site

Establishment and maintenance of identical development platforms at all sites was difficult for two major reasons: (1) software and hardware arrived at different sites at different times, and (2) procedures for customizing installation and integration of the system components were complex and not standardized.

In an attempt to ensure an identical platform, HQ provided the hardware, operating system, relational database management system (RDBMS), and software tools for three development sites (HQ, ECS, and ORNL). Unfortunately, because components did not arrive at every site simultaneously, several months passed prior to having a common working platform. In addition, the setup/load procedures were not identical at each site, and software version upgrades never arrived at a time convenient for uploading at all sites concurrently. Although changes to the architecture should have been coordinated, the more common approach was to solve the problem at one site and then attempt to transfer the solution. Eventually, the attempt to coordinate the platforms at three sites was abandoned, and only ORNL and the ECS continued to attempt to set up consistent architectural structures.

Government procurement practices caused most of the hardware delays and made the hardware platforms difficult to synchronize. Maintaining a similar set of third-generation and fourth-generation language software versions was nearly impossible also, since different people were managing system component integration at each site. Individuals interpreted and implemented software loading instructions differently based on past experience. Frequent communication

among the various sites and direction from management (HQ) did not prevent this problem.

The second problem involved customizing the hardware and software installation and integration. ORNL experienced this difficulty with the acquisition of the Santa Cruz Operations (SCO) Open Desktop (versions 1.1 and 2.0). The major problems we encountered were those of matching personal computer (PC) hardware such as graphics cards, ethernet cards, and video monitors with the SCO system. Installation was difficult and time consuming. Both ORNL and ECS experienced problems but they did not always appear to have the same cause or require the same solution.

The difficulties are partly caused by DOD's recent trend toward an "open system" computing environment. Proprietary systems, which have been the norm in the past, tend to trap the user into an environment of unsupported hardware and planned obsolescence. An open system environment allows the user to change computer systems, hardware, application software, and even personnel without losing all of the computing system investment. This is accomplished through standardized operating systems and standardized interfaces which may be implemented by a wide variety of vendors. This all sounds wonderful; however, the advantages of an open system environment do not come without a price. At the present time, the reality of most open systems is that the user may very well need to be an integration expert to purchase and install such systems.

An additional delay was created when it became apparent that a new version of the RDBMS offered significant enhancements. The new version would be on the market very early in the development process. After discussion of the advantages and disadvantages of beginning system development with a beta release, the decision was made to proceed with the new version. ORNL discovered several bugs in the software tools associated with the RDBMS and this had a slight impact on software development.

To solve concurrency and consistency problems, ORNL took the lead in documenting the system administration/system operation standards for the development platform after the system (i.e., hardware, operating system, RDBMS, off-the-shelf tools) was stabilized. Because of the difficulties in finalizing the development architecture, approximately six months passed after the approval of the project plan before the first draft of this documentation was produced. This report provided a list of procedures for ensuring that ORNL and ECS would be developing on equivalent platforms. It also documented the process of preparing the system hardware, network, operating system, and RDBMS and documented the sometimes daunting task

of properly tuning the whole system. Configuration of the local area network (LAN) was also documented.

The reasons for establishing identical platforms at each developmental site were valid. The plan for acquiring all platform components through a single source (in this case, HQ) was valid. Unfortunately, setting up multiple sites took a lot longer than we expected. We have no solution for this platform consistency problem and its implication on development time; primarily we wish to issue a warning about the potential hazards of attempting to acquire and set up identical computer laboratories at multiple sites. The problems caused by this architectural jockeying have been a lesson that will not be repeated during implementation of the operational system on the target hardware.

4.2: Communication issues

Because programmers were not located nearby, on the same LAN, or even in the same time zone, solving minor technical problems and sharing in technical or functional discussions, though critically important, were very difficult. Because of this inherent difficulty, all of the sites seldom had access to all needed information. Addressing the issues associated with communication among the widely separated sites involved solving technical issues and establishing communications procedures for people to follow.

Some of the technical issues were addressed by establishing network communications. A computer network is generally a collection of computers which are connected together via some sort of electronic media. The individual computers all have both software and hardware which facilitate message traffic between them.

A LAN is a small collection of computers which are generally positioned in very close proximity to one another, typically in the same building. The system discussed in this paper consists of workstations and PCs connected together via ethernet cable (other methods and technologies are on the rise) all in the same room. This LAN is also attached to the main ORNL backbone and, as such, is part of a much larger network in the Oak Ridge area, which is also a part of the global "Internet." The Internet consists of thousands of connected LANs from many government, commercial, and educational sources. During system development, ORNL, HQ, and the ECS communicated via the Internet using various programs [e.g., the Unix File Transfer Protocol (ftp)] to facilitate file transfers.

Communication procedures were developed for exchanging data, software, and information. Knowing that, as development progressed, all sites needed visibility of shared information, ORNL and the ECS produced a design/development change plan for managing data transfers for changes occurring to the database and programs. This plan documented database procedures for construction of tablespace structures, table ownership and privileges, and transfer components. It also established how updates would be accomplished, the frequency of data exchanges, and the methodologies for providing the appropriate types of exchanges. This document might be seen as an abbreviated configuration management plan for a system under rapid development. Although brief, the plan was crucial to prevent havoc.

For casual communications (electronic mail), a LAN-based commercially available package was chosen because the DOD community was already using it. This package, which includes a scheduler, office manager, and mail system, among other features, was standard within the DOD user group, who did not want to check two different systems for messages. It also had a gateway to Internet. HQ obtained a user-id for ORNL on this system and ORNL could dial-in via modem as a remote user. Because of security reasons, the e-mail system used for simple communications was not attached to the LAN containing the RDBMS and associated programs. However, ORNL developers could send and receive e-mail via the Internet gateway.

It was also necessary to share information orally transmitted during meetings not attended by every developer from each site. The participants at each site agreed to write and share "trip notes." Faxing and phone calls helped resolve any questionable issues that were identified via the meeting notes. This procedure assured that oral communications provided in meetings were documented and were not interpreted differently at individual sites. The greatest problem was production of the notes, which was not high on anyone's list of priorities. The notes took a lot of time to write, but participants agreed that they were very valuable references throughout the project development period.

Nonavailability of WCS development staff to answer questions and coordinate design issues was a problem. The initial prototype of the primary interface with our system (the port-level system built by WCS) was developed based on a European model (involving many countries with different cargo documentation requirements and a different user community) rather than on a Continental U.S. (single country) viewpoint. The prototype's applications and database had to be revised for use in the U.S. Although the RDBMS that we were developing could be designed based on the

European prototype, major changes were expected prior to a final U.S. design. Because the WCS development staff members were busy changing the European design for U.S. operation, they were not readily accessible to our development staff for answering questions. This communication problem was never completely resolved, although interaction with the developers of WCS was facilitated as much as possible by HQ. Questions were addressed through e-mail or in formal meetings; documents (e.g., the Functional Description and the Architectural Analysis) were used for review and concurrence.

4.3: Design Plan

To meet a very tight deadline, development of the database system required the participation and close coordination of programming activities at ORNL and ECS with cooperation and guidance from HQ and interaction with WCS developers. The geographic separation of the development sites forced early modularization. While ORNL and ECS worked on establishing a consistent platform and solving technical problems, a clear plan was established by HQ to determine who (i.e., which development site) was responsible for what process at which point in time and who "owned" the data or process. The design plan provided (1) a process flow model which divided tasks into modules and established responsibilities; (2) system standards and conventions which enforced design consistency and increased efficiency through use of templates for screen design; and (3) a schedule for early documentation, continual testing and integration, and accountability.

Because the potential for wasted energies and deflated egos increases in direct proportion to the lack of clarity in task definition, this clear delineation of design tasks was an attempt to keep politics and emotions out of the isolated enclaves of the project. A process flow model helped to establish which development site was responsible for each module and task and to coordinate developmental activities. The goals in defining the tasks and modules were (1) to facilitate independent system development at each site and (2) to best use the skills and knowledge possessed by developers at each site.

The two sites developing processing modules (ECS and ORNL) possessed very different skills and experiences. ECS personnel had far more functional knowledge of military transportation regulations and user needs than did ORNL; however, ORNL had a broader technical knowledge of state-of-the-art

software, computer, database, and communications technologies. A program of technology and information transfer was initiated in which HQ, ECS, and ORNL personnel participated in an attempt to provide a more even mix of skills. This program allowed ORNL to provide technological support to the ECS and HQ while those sites provided functional knowledge to ORNL.

HQ, which took the lead in developing the process flow model, established responsibilities for developing specific processing modules. HQ also led the effort in documenting official data-exchange contracts between the central database and other interfacing systems and organizations. ORNL produced written standards and conventions for the design of the user interface, which included the philosophy for using generic stored procedures when appropriate, guidelines for screen layout, the use of help messages, standard keys, and standards for commenting modules. The ECS produced code for three major processing modules. Data produced by these modules was fed to the database and to the appropriate water port via communication programs written by ORNL.

Figure 1 shows a simplified process flow for one module that ORNL, ECS, and WCS developed in collaboration. Major responsibilities for each site's activity are documented on the figure. In the formal design plan, detailed lists of tasks and responsibilities, dependencies, and schedules, including the testing and integration timetable, back up the process flow shown in Figure 1.

ECS developed programs to accept Transportation Control Document input and process the file for entry to a temporary transportation database. ECS also examined the file for "fatal" errors and determined if the file should be sent to recipients that were not yet part of the system or to a suspense file for administrative review and correction of an error of some type. ORNL developed communications processes for reading the data prepared by ECS and for supplying it to the appropriate water port and to the central RDBMS. ORNL was responsible for database design as well as for development of the communications processes. WCS was responsible for the system development activities at the port level.

Data, database, and system administration modules were also part of ORNL's responsibility. Other major ORNL responsibilities included providing query capability and user interface design. Table 1 shows a broad division of responsibilities and the skills required to complete the responsibility.

Figure 1. Process flow activity: transportation control documents

We anticipated a problem with competitive attitudes between developers at the different sites. Although it is common knowledge that many programmers have very possessive, personal attachment toward code they have produced, this attitude was suppressed during development of this project. For example, the site responsible for massaging raw data and converting it to a usable format (ECS) designed its output table considering the requirements of the port-level system and the requirements of the RDBMS with respect to input format. All three sites met and worked together to produce the most effective design for project requirements. The developers of each module looked at the "big picture" (i.e., all interface requirements) rather than just their limited universe. Similarly the database design was a compromise to incorporate the design of the interfacing prototype system which was much further along in development.

The system described in this paper is not a perfect design, but it fulfills the system requirements (including limitations imposed by DOD restrictions) and accommodates the WCS's prototype system design. We believe that the design plan provided necessary

guidelines to understand when compromise was appropriate and to coordinate developmental activities among sites.

5: Conclusions

Coordination of programming efforts among geographically separated locations is difficult but possible with precise planning and cooperation of all sites. Problems associated with establishment of a consistent hardware/software tools platform at each site have to be solved prior to actual programming. Technical problems associated with communications must be addressed, and procedures for information exchanges also have to be worked out. A detailed design plan, which emphasizes specific site responsibilities and controls is required.

Table 1. Development responsibilities by site

Responsibility	Skill Required	Site
Preliminary data preparation and processing	Knowledge of military transportation requirements, flat file processing, translation of COBOL programs	ECS
Document distribution	Knowledge of report distribution requirements; address formalities; formats required at various sites	ECS
Report preparation	Functional knowledge of reports	ECS
Processes at the water port level	Functional knowledge of cargo requirements at the port; programming of the port-level system	WCS
Database design	RDBMS technologies	ORNL
Queries and user interface design	Knowledge of database optimization theories; experience with user interface designs	ORNL
Communications strategies	Knowledge of and experiences with communication topologies and architectures	ORNL
System component integration	Knowledge of system integration procedures	ORNL

Early modularization, which is forced by this development mode, is actually an advantage. Modules are more balanced and better documented.

During development of the system discussed in this paper, we learned that communication among sites is a top priority; developers must be kept informed. We also learned that generating an atmosphere of consensus, moving toward agreements quickly, and documenting decisions is important. Finally we learned that sites and individual developers must be willing to compromise and to change course, if necessary. Because of the mix in knowledge, skills, and experiences and because of the rigorous discipline

required in this development mode, the final product was more efficiently designed and should be more easily maintained than a system developed under more conventional circumstances.

6: References

Krasner, Herb, Jim Terrel, Adam Linehan, Paul Arnold, and William A. Ett. "Lessons Learned from a Software Process Modeling System," Communications of the ACM. Sept 1992. 34(9): 91-100.

**DATE
FILMED**

10 / 15 / 93

END