EFFECTIVE ORGANIZATIONAL SOLUTIONS FOR IMPLEMENTATION OF DBMS SOFTWARE PACKAGES

By: Doug Jones

Introduction/Strategic Plan

This paper will use the Space Telescope Management Information System development effort as a guideline for discussing effective organizational solutions used in implementing DBMS Software. The paper will not attempt to be all inclusive but rather will focus on the importance of strategic planning. In addition, methodologies will be examined that offer viable alternatives for successfully implementing DBMS software.

The paradigm for effective implementation of a Management Information System (MIS) is a strategic plan. The idea of the strategic plan is straightforward in concept: where do you want to be and how do you get there. More importantly, it is the background against which the necessary managerial and technical tradeoffs are made in order to successfully implement a Database Management System (DBMS) as part of the MIS effort.

It has been found to be useful by system development teams to express this strategic plan visually as an Information System Architecture. The general recommendation is to a avoid a rigid detailed hardware/software architecture which is incapable of responding to evolving managerial needs. Instead, the guiding principle is to construct the architecture to conform to the organization's management structure. This management structure or organizational setting includes the consideration as to whether the operation is centralized or decentralized, and what is the prevalent management philosophy.

This information system architecture does not have to be complex. In fact, simplicity is an asset. A few diagrammatic representations can prove to be very effective. As decision alternatives are developed, such as shifts in requirements, vendor evaluations, or software selection, they are all examined with respect to this Information System Architecture. It is several of these major decision alternatives which will be discussed, and implementation alternatives offered.

In harmony with this strategic plan, there must be a senior decision maker, functioning as the architect of the Information System Architecture. This authority should be one individual, or certainly no more than a few individuals, who can accurately determine and enforce organizational requirements.

For the NASA Space Telescope Program, this Information System Architecture includes the general requirement to integrate contractor financial data, reported via a diverse management tracking structure, with relevant program technical feedback. This data must be combined and reported in a timely fashion. More specific requirement goals were included, outlining the specific types and level of program financial data and the specific technical data needed. The organizational setting was complicated by the fact that the Space Telescope Program is a major NASA program, which at the time of the MIS start-up was well underway in terms of time and total dollar expenditures. The strategic plan included a need for offering a benefit to the on-going program in the near term, and the development of a system to meet future NASA program requirements. For the NASA Space Telescope Program Management Information System, the crucial chief architect role belongs to Mr. James Welch, NASA Space Telescope Program Manager.

Shifting User Requirements

The most immediate difficulty for implementation of a DBMS package, or any software system, is dealing effectively with shifting user requirements. All projects, whether they are software or engineering, involve a series of calculated trade-offs made during the difficult course to completion. Taken in this context, the problem of shifting requirements should not be surprising, nor unexpected.

Even more important than this realization, is the crucial combination of establishing a baseline requirement and top management commitment. It is this combination that permits effective solutions for handling shifting requirements. The importance of a strategic plan expressed as an Information System Architecture for establishing top management commitment has been discussed. One effective tool for establishing the baseline requirement is prototyping. Prototyping is the process whereby an initial working software system is constructed for presentation to management and the user community. In fact, the presentation process is the key. The goal of the presentation is to make the users an integral part of the development process. By demonstrating an electronic strawman early in the development cycle, users are presented with less of an abstraction than a simple verbal or written explanation would lend. This electronic strawman allows the system developers to elicit meaningful responses from selected users early on in the course of system development. This electronic strawman can be as simple as a series of example screen formats that may be paged through in the manner that the proposed system will function.

As development continues, the prototype continues as the vehicle for acceptance of changing requirements. It is important to note that this prototyping approach accurately reflects the iterative nature of software development. Changes will continue to be received and accepted as they are approved. Management's expectations should not reflect a static system, but rather should be for a controlled audit trail of changes, all of which relate back to the strategic plan as expressed in the Information System Architecture.

For the Space Telescope Program the prototype Program Management Information System (PMIS) has been completed. This PMIS 1.0, representing the baseline requirements, was accomplished using the Digital Equipment Corporation (DEC) VAX 11/780 executing the INFO Database Management System. In addition to continued user training, production control procedures are being established to identify, implement, and monitor requested changes to the PMIS 1.0 System. As data loading continues, it is expected that further enhancements will continue to be made.

DBMS Vendor

After establishment of a strategic plan in the form of the Information System Architecture, another early and crucial issue in the chronology of MIS development is the establishment of a reliable working relationship with the DBMS vendor. This is an important step, whether or not the DBMS package has already been selected. If the DBMS evaluation is being done, vendor support becomes one of the decision criteria. If an existing DBMS is already in place and is satisfactory, it still benefits the project to review the nature and quality of vendor support. Vendor support is always important because the vendor can be the major link to viable alternatives for use of the software product, both during and after the software development effort.

Foremost in the evaluation of vendor support is their perceived ability to meet the organizational goals, as outlined in the Information System Architecture. This however, is the composite of many indicators. One recommended first step is a phone call to other user sites, followed by visits and evaluations of these operations. This should yield a realistic impression of the sites relative success and quality of on-site support. Users who are performing a similar operation to the proposed plan are the most valuable contacts. As part of this initial evaluation, the financial position of the vendor should be examined to ensure at least the possibility for a long-term relationship.

Another valuable indicator for evaluation is vendor supplied training, and the existence of a user group. The emphasis placed on training is a strong indication of the vendors intent to support the user sites for the long-term. In particular, multiple class offerings both on and off-site with flexible scheduling are all positive signs for vendor support. In addition, a strong user group can be a considerable asset. The user group can be another source for training and creative ideas for use of the DBMS product. No one single vendor will put their product to the variety of uses that a user community will in just a short period of time. During this second phase of evaluation, plans should be formulated to ensure staff training and making contacts for later direct support.

Another consideration is the vendors' philosophy towards processing maintenance requests and changes to their DBMS product. Maintenance requests should be handled in a disciplined fashion. The presence of a strong user group can yield additional leverage for insistence that a selected change be implemented. Another leverage factor is the size of your account and the type of work you are proposing. A project for a large organization, that is making full use of a vendor's product will command the vendors attention. Such a relationship can be advantageous to both sides. The user gets prompt attention while the vendor gains potential marketing advantage. With more hardware vendors supporting DBMS products, a user with large dollar expenditures tied up in hardware may consider using the same vendors DBMS software. Bargaining leverage is gained and technical risks may be minimized because the software was developed for the hardware in question. Finally, the contract itself offers the possibility for bargaining leverage. A lease agreement may make better financial sense and offers greater bargaining flexibility than an outright purchase. In particular, the prototype process allows real time evaluation before commiting to a product.

Whether the DBMS package has been procured or not, a technical evaluation should be made to determine compatability with design requirements. This subject is thoroughly discussed in current literature and will not be greatly expanded on here. One consideration does ,however, deserve emphasis. Any DBMS package selected will be lacking with respect to some capability or capabilities desired by the user community. This is why the types and likely strengths of your leverage should be evaluated beforehand. The reality of work-arounds will soon become apparent. Work-arounds is the concept of seeking alternative software solutions for a problem that is unresolvable in the short-term. This inevitability should be reflected in the evaluation process. A more complex DBMS is more difficult to manipulate. Make sure the application(s) planned require the degree of complexity you are buying. Again, the prototype process can be useful in making this final determination of DBMS suitabliity.

For the Space Telescope Management Information System, the alliance with the software DBMS vendor was complicated by an intermediate agreement with another third-party vendor. This lead to some negotiating difficulties when faced with technical deficiencies with the DBMS. This situation was resolved within the vendor's organization by using the leverage NASA had due to the lease agreement, and the implied pressure of negative publicity because of the importance of the NASA account.

Admittedly, it is more difficult to break-out intermediate deliverables for tasks such as development of the DBMS system calculation logic. For such tasks, the recommendation is to break them down to correspond to the functional capability of the system. For instance, the calculation logic may be broken out into three sections. The input of the data to be calculated, performing the calculations, and reporting the calculation results. To the development staff the process is of course, more complicated. But, this approach is more understandable to general management. It is the responsibility of the system developers to represent the difference in relative difficulty by demonstrating time differences in the schedule. For the calculation logic example, the necessary design, writing, and testing of the calculation logic is more difficult than either inputting the data to be calculated , or moving the results to be reported.

For the Space Telescope Management Information System this general approach was followed. A time schedule with associated deliverables was established and reported to. Of the three schedule delays identified, only one was related to a software problem with the INFO DBMS. In all cases the underlying problem for the delay was worked until a solution was reached.

Effective implementation of a DBMS requires the continuing support of management considerations. The strategic plan is the starting point for establishing management commitment. The prototype is an excellent tool for handling shifting requirements. Vendor support is a requirement for identifying viable software solutions. Everything is set except for the time line of development. Continued management support requires an accompanying schedule.

The system development staff should be able to track and report to a time line at all times. The idea of a strategic plan and corresponding schedule is not a revolutionary concept. However, since software projects are frequently late, it is equally obvious that merely developing a schedule is not adequate. Several suggestions are in order.

The reality of shifting requirements has been discussed. Such shifts need to be documented and approved. The prototyping approach allows for inclusion of these changes, which is important to prevent uninterrupted development. In parallel these changes must be noted with respect to the changes in schedule. The realities of schedule slippages should be discussed immediately and not appear as a surprise on the original anticipated project complete date. Tracking the inevitable deviations from the original plan in concert with the prototype development is an on-going process.

In order to fine-tune this process, system developers have found it useful to establish intermediate product deliverables throughout the lifecycle development process. The purpose of this effort is to prevent serious misunderstandings from being identified late in the development process. Although the complete DBMS-based software system is the final deliverable, there are identifiable intermediate products which demonstrate progress to that goal. This methodology forces a prior planning approach to be initiated. For instance, a working electronic prototype is the logical product deliverable of a functional specification. The prototype might include one dozen screens and the ability to view the screens in the same relationship as in the final system. The deliverable schedule might have included the completion of two electronic screens per week. Difficulty in agreement with a screen format would appear as a delay in completion of the particular electronic screen in question. By identifying the specific delay point the system development staff avoids appearing unaware of the schedule and can more effectively identify delay points. These delays are not of themselves a software problem.

Post Development Control

The majority of the alternatives discussed so far have pertained to solutions applied during the development phase. To complete the discussion of effective implementation of DBMS software, viable mechanisms must be discussed for continued efective use. The strategic plan should continue as the cornerstone of the maintenance effort. As the development effort transitions to a close, it is imperative to establish specific control structures. These controls fall into three areas. Production software control, production data control, and scheduled software enhancements. As the system is released into the user community, changes will be required to satisfy the user community. In all three cases it is important to track adjustments in a regulated fashion.

A simple but proven technique is control forms. These forms should detail the requested change or difficulty, a description of the change or difficulty, requested date, and description of the solution. In addition, the system development staff should arrange for assignment of responsibility and provide for management signoffs. The desired result is an audit trail of controlled changes. Such control forms are appropriate for any of these three areas. For legitimate scheduled enhancements, where the scheduled work is more than a few days, prior review approval is necessary before starting.

The Space Telescope Management Information System is now entering this post development phase of implementation. Control forms for data changes, software changes, and enhancements have been created. Initial system changes have been requested and implemented.

Summary

The purpose of this paper has been to discuss effective organizational solutions for implementing DBMS software. Paramount to this successful implementation effort is development of a strategic plan. For Management Information Systems (MIS) this is often expressed visually in an Information System Architecture. The purpose of this Architecture is to gain top mangement commitment. Most importantly, this strategic plan should reflect the organizations' structure. If the orgainization is decentralized then the information gathering and dissemination plan must reflect this reality. In the context of the Space Telescope Program this organization structure included the demands of an on-going program with the requirement to integrate diversely reported financial data with selected relevant technical feedback.

The most immediate difficulty in implementating DBMS software is dealing with shifting user requirements. This problem was set in the context of calculated trade-offs, a reality for any project. Prototyping, the creation of an initial working system, is a powerful tool to establish the baseline requirement. Users are made an integral part of the development process in reviewing this so called electronic strawman. With continued development, the prototype continues as the vehicle for accepting change, reflecting the iterative nature of software.

Another traditional problem with implementation has been the establishment of reliable DBMS vendor support. For this, techniques are less of an issue, rather the evaluation should center on a series of factors in which to make a weighted estimate. Emphasis should be concerned with the likely leverage the user may have in case of difficulty. The financial strength of the vendor should be examined as well as evaluation of other user sites. Other important considerations include, a strong user group, and a vendor who supplies the hardware and the DBMS. Signing a lease agreement may be a more powerful lever than a purchase agreement. The technical evaluation should ask the question, does the application require the degree of complexity that the DBMS software offers. More complexity translates to a longer learning time and possibly more maintenance difficulties.

As the develoment lifecycle continues, management considerations must continue to be supported. Although the prototype continues to demonstrate the development effort, a schedule is esssential to demonstrate progress against a defined timeline. Because software development is a recursive activity and shifting requirements are a reality, it has been demonstrated to be useful to identify intermediate deliverables. These deliverables should be expressed in functional terms that a general user could understand. This will be important because as delays are identified management will need to be informed. Many delays in implementation of DBMS software are not directly software related. An accurate means of identifying these sticking points will permit an accurate determination of the underlying difficulty and permit a managment decision to be made. Intermediate product deliverables are a form of development insurance.

As development winds down, the importance of continued monitoring does not diminish. Continued support falls into three areas: prodution software control; production data control; and software enhancements. For all these areas a regulated means of tracking is essential. Production control forms are one such way of providing an audit trail of changes. These forms should include requestor, change requested, description of change or problem identified, accompanying description, and provision for approval signoff.

The dynamics of a DBMS implementation are many. No technique discussed here is dificult to understand, many were borrowed from other fields of study. If there is one key it is the discipline to maintain the effort necessary to see the Management Information System successfully implemented.