Application Delivery in the ‘90’s: A Framework for Change

Brenda Bangert
Technical Manager
Stanford University,
Stanford, California

Abstract

The approval of a multi-million dollar, 5-year strategic plan for replacing most administrative information systems causes a university’s information technology department to rapidly learn new technologies, methods, and business practices. In order to most effectively deliver new systems, the I.S. department re-engineers to provide project-based matrix management, a framework for “build vs. buy” decisions, application assembly and integration, and project support and improvement.
Background

During the 80’s and early 90’s, Stanford University implemented a set of integrated administrative systems that were written in SPIRES, a proprietary fourth generation language developed and supported by Stanford staff. All forms were put on line, electronic signature and routing were implemented, and on-line approval from electronic mail was integrated into the work flow.

By the end of the 80’s as we were completing the last of our major administrative systems, we were already beginning to see the need for change. The very factors that contributed to the success of the administrative systems at Stanford were now creating a new set of issues. Since paper forms had been eliminated, the mainframe capacity needed to support the University’s administrative needs was growing rapidly. Our reliance on proprietary systems made it difficult to find the skills we needed in the marketplace. In the highly competitive Bay Area, it is difficult to find technical staff who are willing to learn non-transferable languages and proprietary systems. Existing SPIRES expertise was diminishing as staff retired or pursued other career opportunities.

Technology was also changing very rapidly. What were viewed as “state of the art” systems when we started creating systems in the 80’s were already being viewed as out of date by the time we were delivering the later systems in our applications portfolio. Distributed, client/server systems utilizing graphical user interfaces were starting to be available in the marketplace. And, according to the wisdom of the times, these systems could provide the same functionality as mainframe systems much more cheaply and in a much more user-friendly manner.

The “business” climate was also starting to change. What had appeared to be an era of unlimited growth was rapidly coming to an end. There was increasing downward pressure on all sources of unrestricted revenue: tuition costs, indirect cost recovery, and unrestricted gifts.

The changing financial environment led to decreased base funding for computing and systems support. At the same time, there was an increased need for flexible systems to provide timely information and to respond to an increasing rate of change in business requirements.

Added to these pressures for change, for Stanford University, probably the most significant push for new, more flexible systems came as a result of the government’s indirect cost investigation of the early ‘90’s. One of the findings of the investigation related to the inadequacies of the existing financial systems. Although the on-line systems created in the 80’s provided capability to view information in the general ledger, and to perform some financial transactions, the central general ledger system was created in the 60’s and was difficult to extend, had outgrown the original schema for the chart of accounts, made it difficult to close the books at fiscal year end, and was updated only monthly, making timely financial reporting extremely difficult.

In 1991, it was determined that our core financial systems should be replaced for the functional reasons stated above, and increasingly, also for technology reasons.
The Change Process Begins

The first step (needless to say) was to re-organize the information systems staff. To provide a critical mass of resources to implement new systems, and to leverage economies of scale, a new centralized applications group was formed. This unit was called Business Information Systems Applications (BISA), and reported to the Chief Financial Officer. During the summer of 1993, an Administrative Systems Strategic Planning effort was launched under the sponsorship of the Chief Financial Officer, and led by the University Auditor. An AIS Planning team was formed with membership from the schools, central administration, BISA, and other technology support units on campus.

At this time the scope of the planning effort was extended to all administrative systems on campus. Concurrent with the AIS strategic planning effort, several business process improvement and/or re-engineering efforts were launched. The primary goal of these efforts was to eliminate non-value added activities and thereby reduce both central administrative costs, and administrative costs in the schools and departments.

The AIS planning effort concluded in the summer of 1994. Twenty-two major systems initiatives were identified, and scheduled for completion over a five year period. The Board of Trustees approved the plan, and projects were launched in September of 1994.

From Chaos Comes Knowledge

Knowledge can be divided into four categories: 1) what you know you know, 2) what you know you don’t know, 3) what you don’t know that you know, and 4) what you don’t know that you don’t know. In September of 1994, the number of items in category 2 seemed to far outnumber those in category 1. Over the past year, several items that were in category 4 have now moved into category 2. Category 3 represents the ‘nice surprise’: when you discover you know something that you weren’t aware that you knew or didn’t recognize the value of.

Category One: What We Knew

- The Board of Trustees had approved a high level 5-year systems plan, which outlined the need for 22 new or replacement administrative systems.
- Since the financial systems form the foundation needed for many of the enterprise-wide systems used at Stanford, they were first in the place for replacement.
- Stanford wanted to re-engineer its business processes in conjunction with the delivery of new systems whenever possible to help maximize potential savings.
- Stanford had decided to look to the marketplace for new systems solutions. The ongoing and exit costs of proprietary software solutions were proving to be quite high. Our goal was and is to use vendor packages, industry standards and expertise.
• A new Information Technology Systems and Services organization was being formed to be led by a vice-presidential level Chief Information Officer.

• Our new systems would be open, standards-based, state of the art, distributed client/server (insert your favorite buzzword here)....

Category Two: What We Knew We Didn’t Know

We knew we wanted to be client/server and distributed, but as an organization, we possessed little knowledge and experience in these newer technologies. We knew we needed to quickly gain and/or buy (contract for) expertise in:

• distributed computing
• relational data bases
• application development tools
• object technology
• new languages, including visual programming languages
• frameworks
• middle ware
• analysis and design methodologies
• project management processes, standards, and techniques

Since we knew we wanted to implement vendor provided and supported products whenever possible, including application packages, we knew we needed to find out more about vendors of client/server packages.

The category of what we know we don’t know has grown significantly as we began the process of finding answers to our questions. That’s how we know that items are moving from Category 4 (what we don’t know we don’t know) to Category 2(what we know we don’t know). For example, we didn’t know that we needed to understand maturity of client/server vendor solutions until we began to explore the package system marketplace.

Issues that we now understand need to be addressed and/or questions that need to be answered include:

• What is our target technical architecture? What is our strategy for achieving it? Current marketplace solutions often do not conform to the target environment.
• What criteria will we use to evaluate vendor packages? What is the decision making process?

• What industry standards do we want to adopt? There are many to choose from. Microsoft sets the de facto standards, just as IBM did for mainframes.

• What ‘flavor’ of client/server architecture is desirable: two-tier, three-tier, multi-tier?

• Just how open do we want/need our architecture to be?

Plug and play sounds great. Are we ready for the multi-vendor, multi-site license, version control headaches this approach brings, or can we make progress more quickly by purchasing application suites that provide proprietary solutions.

• What exactly is meant by integration in a client/server world anyway?

• What relational database vendor do we want to use? Can we afford to support more than one?

• How is security provided in a client/server world

• How object-oriented do we want to be?

• How do we measure system success? Is the goal efficiency and cost reduction? Or is the goal effectiveness and better service and support?

• How is a project-based organization different from a traditional, hierarchical organization?

In what is probably a common approach when there is so much work to be done, so many questions to be answered, and so many decisions to be made, we started everything at once.

Good News and Bad News

The good news about starting many efforts at once is that knowledge is quickly added to the “what we know we don’t know” category. The bad news is that forward progress is quickly impeded by the large number of unknowns that must be dealt with. This can quickly lead to demoralization and “analysis paralysis” when it appears that so many decisions in so many areas need to be made at a time when there are so many unknowns in our industry.

In order to provide focus on the new systems work needing to be done, a separate organization was formed, Information Systems. Staff in the newly formed IS are not responsible for support of existing systems. Working with clients, they formed project teams to perform the detailed requirements gathering process needed to begin package evaluation and selection. As part of the AIS plan, it had been decided to look to the
marketplace for system solutions and to limit the amount of custom development; however, we needed a framework for when to buy and when to build.

Although work could progress on functional requirements in the absence a defined technical architecture or a uniform process for vendor and package evaluation, we quickly determined that a defined systems delivery process for the new I.S. organization would streamline our work. In the autumn of 1994, an I.S. focus team was formed to examine concepts of component re-use and assembly, and to recommend an approach for delivery of new systems. The team identified several inhibitors that could lead to less functional, less integrated systems than our current legacy systems. The team made the following recommendations:

- A new process for system delivery that would include:
  - portfolio and functional domain analysis
  - enterprise data and process views
  - alignment of the delivery solution with the value of the business process
  - componentizing of applications

- A new method for identifying reusable components:
  - functional and process domain analysis to meet the needs of the central administrative offices (functional view) and the schools and departments (processes such as ‘establishing a new course’ that go across many functions)
  - intersections identify potential areas for reusable components
  - frequently used functions, for example: cut a check, within a functional domain are also candidates for re-use

- A new process for determining value
  - priority processes support identity process such as teaching and research
    - drivers are increased market share
    - often differentiate a “business” from its competitors
    - customized system solutions are often required
  - background processes are business processes, such as purchasing, that are needed by any business
    - drivers are low cost and efficiency
    - package solutions usually meet all requirements
  - mandatory processes, such as those required by environmental health and safety, are those required by outside agencies
    - package solutions will meet common requirements, like compliance with tax laws via a payroll system

The team created a process model for I.S. that we called the Software Shop. Like any process, the Software Shop has inputs and outputs. The inputs are: architecture, standards, business process redesign, and knowledge from other systems. The output is a new application. The process is supported by an application delivery environment (ADE) infrastructure, and is managed via planning, quality assurance procedures and organizational direction and oversight.

The Software Shop process has three components: analytic framework, delivery strategy, and deployment. The analytic framework requires three kinds of analysis. Enterprise
integration analysis determines how the proposed system interfaces with others. The data and process models for the new system are integrated into existing enterprise data and process models and both are modified as needed. Areas of potential overlap or missing functionality are noted. Business process analysis maps a business process from beginning to end and notes the need for information or process logic that may be provided by other systems. Functional domain analysis identifies common procedures within a functional domain (such as financial systems) that can be re-used by many applications.

The delivery strategy is based upon whether or not the system is a priority system or a background system. Most administrative systems support background and mandatory processes. There are five delivery strategies. These strategies are not mutually exclusive, but represent mix and match approaches that will probably be used in varying proportions for the delivery of all new systems. The first approach is: buy the service. This suggests that no systems are installed at Stanford to support this business process. Both the business process (or function) and the systems that support that function are outsourced. An example of this might be a lab store. The store would be run by an independent business and that business would be responsible for keeping it stocked appropriately, and for providing any computer systems it needed to do so.

The second strategy is to buy a package and use it ‘as is’. This strategy implies that the business practices may need to change to fit the package. A third strategy is to leverage our investment in legacy systems. Stanford has created excellent authority, routing and work flow applications. Potentially these systems can be ‘componentized’ or ‘wrapped’ so that newer systems can make use of their functionality. The remaining two strategies would be used primarily for priority systems. These strategies call for building systems and components and assembling them to create new applications.

The deployment component of the Software Shop model begins with the integration of system components, and testing of the resulting application. This new system is then rolled out for general use and turned over to the application support organization.

The I.S. Software Shop recommendations were completed in early 1995, and the I.S. organization was subsequently re-organized around the concepts in the model. The I.S. organization was structured into six units: Application Project Integration (API) was formed for the analytic framework process; Application Assembly and Integration (AAI) was formed to deliver and deploy applications; the Projects and Planning Office was formed to introduce and support standard project management methods and quality assurance practices; the Application Support Center (ASC) was formed to provide and support the application delivery environment; and Application Resources (AR) was formed to support the acquisition of staff for the projects. The sixth unit, Data Administration, was already in existence.

Parallel Efforts

While the I.S. organization was being defined and formed, many projects had already begun. Each team was defining its own processes and approaches, and making its own
assumptions regarding technology, package usage, vendor selections, use of object technology, etc. Outside the I.S. organization, infrastructure projects were being prioritized, and a project management and system development methodology was being chosen.

As might be anticipated, the result of these uncoordinated and sometimes conflicting activities led to organizational confusion. In addition, the sheer magnitude of work to be done often resulted in communication breakdown. There was a general lack of understanding of the new I.S. organizational structure, both within I.S., and among our clients and other parts of our parent organization, ITSS. The lines of authority between and among line managers and project managers were not clear. Project managers were accustomed to having line manager responsibilities. Line managers found themselves largely in support roles, with little or no staff.

At the same time, the University senior management team (RASOG, restructuring and administrative systems oversight group) charged with overseeing both business process redesign initiatives and the new system initiatives began to realize that it needed to define a clearer decision making process, and to identify when it needed to be involved in review of the initiatives. As a result of the selection of the Navigator methodology from Ernst and Young which defines the roles and responsibilities of sponsors, executive sponsors, project managers, steering committees, and executive steering committees, RASOG began to realize that more and consistent structure and support was needed across all systems initiatives.

Where We Are Today

Several resignations among the management of the I.S. organization created the opportunity to re-examine our structure. The basic organizational unit is still the project. There is a high level of acceptance of project structures recommended by Ernst and Young’s Navigator methodology which calls for involved and committed project sponsorship, full time project managers and project team members, and project steering committees. The I.S. organization has been reconfigured and now has only 3 line units: data administration, project support and improvement, and architecture and infrastructure services. In order to keep infrastructure projects aligned with application projects, we are forming steering committees for infrastructure projects that include project managers of the application system initiatives. We’ve come to realize that the sheer magnitude of the work to be done is far too great for existing staff to accomplish in the time required. We are now in the process of ‘pre-qualifying’ vendors as technical partners.

A change support team is being formed which will provide support to the multiple change initiatives, especially supporting the entire University community in the areas of training and job redesign and many new systems and procedures are introduced across campus. Reporting to executive steering committee. The change support team will also be responsible for high-level coordination of the initiatives in order to help manage the level and rate of change being introduced during any one period of time. The team will define and implement consistent communication to the campus.

Several related projects are being integrated into a core financials program. This will help to keep these projects aligned and coordinated. An integrated package suite will be
chosen to support the core financials, and will help provide, by default, an applications architecture, infrastructure and language that can be built upon as needed. This decision will move more items into category 1: the things we know.

**Lessons Learned**

1. The senior management of the information technology (in our case, the direct reports to the Chief Information Officer) need to agree on technical direction and approach.

   Differences of opinion in areas such as build vs. buy, whether or not to embrace object technology, and methodologies make it difficult for the project teams to proceed.

2. There needs to be general understanding, acceptance and buy-in of new organizational approaches.

   The project managers in the I.S. organization did not understand the functions and purpose of the line units. This led to resentment, lack of trust, and a disassociation with the organization itself. Projects became self-contained units with little or no management or oversight from I.S.

3. Line managers must have expertise in their areas of responsibility.

   This would seem to be self-evident, but when it is overlooked or ignored for whatever reason, the entire organization suffers.

4. Don’t use inexperienced contractors in key management roles.

   The position of Manager of Application Assembly and Integration was outsourced, since it was felt that no one at Stanford had experience in component creation and assembly. Having a contractor in a management position created many problems. These were compounded by the fact that the contractor selected for the position was an unseasoned manager.

5. The culture of the organization needs to be used to introduce change.

   Stanford, like many Universities, has a very consensus-based culture. This fact was largely ignored when establishing the new I.S. organizational structure. Without taking the time to get understanding and buy-in, the new approach was never accepted, and in some instances, was actually fought.

6. Line managers must understand their primary role is to support project managers and project teams.

7. Don’t turn process into structure

   To be successful, projects will use the analytic framework, delivery and deployment strategies outlined in the Software Shop. The attempt to turn system delivery
processes that teams should follow into an organizational structure was not a good idea.

8. First you plan

Stanford jumped immediately into starting projects, largely because the Strategic plan had a start date of September 1994. We now realize our job would be easier today if we had taken the time to plan to the next level of detail, identified an optimal sequencing of application and infrastructure projects, selected application delivery tools, production data bases and platforms.

Steps to include in a planning effort include the following:

- Define a long term technical direction and strategy
- Identify impacted systems and business processes
- Define an architectural approach
- Determine infrastructure, middle ware, and the common business services (modules) needed by enterprise-wide applications
- Map dependencies
- Create a tactical plan

9. Change takes time

Implementing systems and the associated analysis and planning effort takes time. Acceptance of new structures and approaches takes time, consistent messages, and patience.

10. There are no silver bullets

Enterprise-wide client/server systems have much to offer; however, there is some step back in functionality from highly integrated mainframe environment. Processes taken for granted on the mainframe, such as authentication, version control, workflow, routing and email integration are just starting to appear in client/server systems.

Required flexibility and ease of change require closer attention to architecting systems in a standard manner, building (or if possible componentizing packages into) smaller single or few function modules with standard published API’s

Be prepared to implement short team solutions that can be replaced; concepts are ahead of the marketplace.

The Nice Surprises

Remember the third category of knowledge: what you don’t know you know. During the past year, we’ve had some pleasant occasions when we realized that we knew more than we thought we did. Mainframe knowledge is transferable. Screens may be called the presentation layer and be GUI, but they are still used for the same purpose. Business
logic is still business logic and data is still data. The processes used for gathering requirements still work. Data models are still quite useful. Project Management skills are still very important. Every once in awhile the old adage is true: the more things change, the more they stay the same.