Whitemarsh
Information Systems Corporation

A New Paradigm
For Successful Acquisition
Of Information Systems

Whitemarsh Information Systems Corporation
2008 Althea Lane
Bowie, Maryland 20716
Tele: 301-249-1142
Email: mmgorman@wiscorp.com
Web: www.wiscorp.com

Table of Contents

Abstract: Most information systems projects are over budget, under specified, delivered late, and fail to meet organizational expectations. A key reason for these characterizations is that during the information systems development, requirements both change and are being introduced. This paper presents a strategy whereby full development is started only after a series of realistic code-generator produced prototypes are created through mission-based database design techniques. The quickly accomplished prototypes are demonstrated to key user groups, and are iterated until they truly reflect a quiesced set of valid and accurate requirements. Production class systems created through CASE, metadata repositories and code generators will then be on-budget, correctly specified, timely, and will meet organization expectations.

1. Overview

Many, if not most, information technology (IT) projects exhibit these characteristics: over budget, under specified, delivered late, and fail to meet organizational expectations. [1], [2], [3]

The United States Government's General Accounting Office (GAO) has been studying IT projects for a number of years, and a review of 10 of the GAO studies clearly shows that the main reason s why IT systems fail has nothing to do with IT. [4], [5].

This paper presents a nine-step process that addresses many of the GAO IT findings. These nine steps presume, however, the existence of the following IT supports:

- An installed knowledge worker framework,
- Enterprise data architectures and effective data standardization,
- A data-driven systems development methodology,
- The availability of high-quality code-generation tools,
- Metadata repository environment, and
- CASE tools

All these supports exist today, and when employed, their cost is negative because of the time and money savings in the first several projects in which they are employed. The nine steps are:

- Mission Development,
- Database Design,
- Prototype Generation,
- Specification Evolution through Prototyping,
- RFP Creation,

- Vendor Responses Evaluation,
- Contract Award,
- Contractor Management, and
- Conformance Testing

This paper presumes that many public-sector agencies do not contain significant IT development organizations and have no interest in creating them. Rather, this paper presumes that public-sector agencies prefer to specify IT needs through some sort of central design authority, and then through the for-profit IT vendors, procure, install, employ IT solutions that are both functionally acceptable and conformance tested to ensure common functionality across public sector agencies. [6]

Whenever public sector agencies take on the full development, operation, evolution and maintenance of IT systems they commonly fail to achieve optimum results. While there can be many reasons for this, the GAO studies show the most common bo be:

- Failure to meet initial end-user expectations,
- Inability to continuously infuse advances in technology in the deployed IT environments, and
- Inability to successfully accommodate the long-standing tradition of individual autonomy.

This last reason, autonomy, prevents the effective deployment of IT systems based on a unified system's design and implementation strategy across a group of public sector agencies because:

- One size does not fit all,
- Public sector agencies often require slightly different functionality, and
- The cost of evolving IT systems through old technologies and approaches is neither cost effective nor viable.

A solution that does work is one that capitalizes on the strengths of both the public- and private-sectors, while avoiding their weaknesses. The proposed approach, modeled on the one employed for the internationally recognized and accepted data management language SQL, consists of a three-part paradigm:

- A vendor community with the profit motive to develop, sell, install, evolve, and maintain products,
- A central design authority to define, validate through prototyping, and maintain functional requirements, and
- Conformance tests and testing by the central design authority to ensure that all systems sold by the for-profit vendors conform to the minimal essential functionality.

The remainder of this paper describes the nine-step approach. Even though this nine-step approach is optimum for large, heterogenous hardware environments across multi-site public sector agencies, this approach can be simplified to accommodate homogenous and/or single-site public sector agency environments. The first four steps are the responsibility of the public sector, the next four steps are the responsibility of the for-profit private sector, and the last step is the responsibility of the public sector.

2. Mission Development

A common lament from IT professionals is that whenever new or changed requirements surface during the IT systems implementation phases, slippages, cost overruns and significant rework almost always result. Users counter that new or changed requirements arise because they didn't fully understand the "problem" that was being solved.

The impact of requirements' rework is based on the following axiom: If the costs associated with requirements and design costs $1 then the activities associated with detailed design through initial system implementation costs another $5. Figure 1 illustrates the main phases associated with the traditional first implementation life cycle of an enterprise-wide IT system such as human resources, or accounting and finance. The scale on the bottom represents the quantity of months spent in each phase. The curve clearly shows that the bulk of the effort (commonly about 70%) is expended before any demonstration is possible.

Beyond the first-implementation cost, the total life cycle expenditure for system revision cycles (not shown in the diagram) commonly costs 4 times more. The total cost is thus, 30 times the design cost. The problem, however is not that requirements change, it's that the effects of changes are too costly to reflect in the implemented system. Because $1 in requirements' changes cause $29 in additional life cycle costs, the exhortation is simple: Get the requirements right the first time because the cost of change is prohibitive.

A review of the GAO studies of IT systems' failure show that new requirements during systems development are such a common occurrence that they must be considered intrinsic. The problems that arise from new requirements fall into two categories: Database design changes and software changes.

Significant database design changes are often a result from an inadequate data driven methodology through which the database is initially designed. Experience has shown that very high quality database designs commonly return many times their design cost in reduced software development and evolution costs.

Software changes result from database design caused changes and also from intrinsic process logic changes. Database design changes can be largely eliminated through the use of a quality database design methodology. The onerous effect of intrinsic process logic changes, can be dramatically affected through the use of object-oriented analysis, design and programming

techniques employed within the environment of code-generators. The axiom of 1:30 may be reduced to 1:10. [7]

Given that requirements "naturally" change and are also significantly affected by accelerating technologies, the ability to posit an accurate and long-term set of requirements is close to impossible, and any IT system developed on the basis of unstable requirements is doomed from the outset.

An alternative to attempting to build an IT system on an unstable platform of requirements is to build from a platform of stable enterprise missions. Mission descriptions are characterizations of the end result independent of: technology, "who" and "how." Well done, mission description documents are timeless, technology free, and apolitical.

3.   Database Design

Database design is the natural next step as *Data is executed policy*. A database's design is thus the overall schematic of the policy domain of the problem space. A quality database design properly reflects the enterprise policy domain rather than ad hoc reporting needs.

The GAO studies show that most IT projects fail not because organizations chose the wrong technology or are not fast or flashy enough, but because they do not engage in the activities that result in proper mission descriptions and policy determinations. These are essential to form the foundations for quality database design.

Database designs, the schematics of policy domains, must be accomplished by public sector agency policy experts, not data processing experts. Abrogating database design responsibilities to IT professionals is a major cause of project failures. There are quality database design methodologies that are fast, efficient and above all engineered to properly reflect the policy domains of the enterprises.

4.   Prototype Generation

Code generators are a class of computer software that take in database design specifications and then produce working computer software systems. Generators have been growing in sophistication and capability over the past 15 years. The code generator environments, such as Clarion for Windows (www.topspeed.com) produce first-cut working systems from database designs in one-hour or less. In contrast, hand-coded systems take upwards to 2 staff weeks per module. Thus, for a 50-table database that requires 150 modules, a hand-coded first-cut working system requires about 300 staff weeks, or 6 staff years. As a prototype, that's unacceptable. But, the equivalent system in Clarion for Windows takes less than 2 staff weeks. As a prototype that's acceptable.

5.   Specification Evolution Through Prototyping

The value of the first-cut working version of a system is that it provides a first "look" at what was inferred from the "requirements." If each subsequent "look" is only a few days away, then, prior to committing to the first real version of a system, a prototype can proceed through 5 to 15 iterations. The value is dramatic. First, because there has been a real attack on the cost of initial system production, and second, because there has been an elimination of many causes of the major system revision cycles. Figure 2 depicts the process of specification evolution through prototyping. The scale, also in time, but weeks, is greatly reduced because of code generation. If the resultant prototype is considered acceptable, it could be turned over to production.
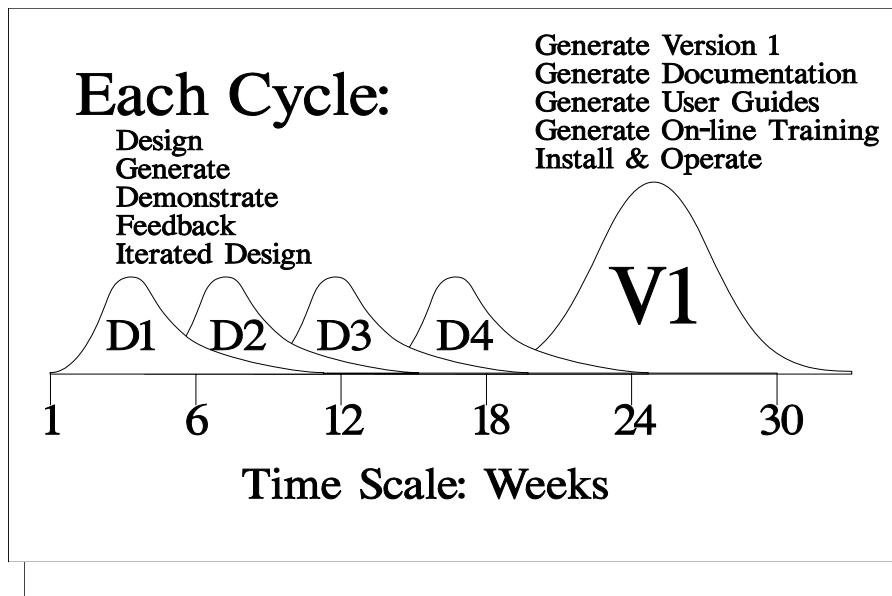
Figure 2. Evolution through prototyping.
Figure 1. Traditional information systems development life cycle and time line.

Once a prototype is created, it can be taken on a "road-show" wherein critical audiences can see what the system actually does. If the "road show" appearances are a day or so apart then minor revisions to the system can be put in place and tested with the next audience.

The goal of the prototyping activities is the finalization of IT system specifications. The result of these activities is not only a valid system speci-fication, it is also a complete set of metadata that can be employed during implementation and then subsequently evolved during the system life cycle. Prototyping also enables the creation of test data that be used for training, documentation, implemented system test cases, and the all important conformance tests.

There is a great temptation to turn the final iteration of a prototype into a production class system and be done with it. There is, however, much more to a production class system than just a working set of computer programs. For example, there are likely functional differences among the different levels and types of public sector agencies. There is the need for multiple types and sizes of hardware, operating systems, database management systems, and so on from different vendors. There are also different approaches to system implementation depending on whether the systems are to be implemented on a network of micros and servers, or mainframes with multiple LANS, servers and clients. Finally there is also the need for formal and informal training, technical support, documentation, and user group meetings.

6.   RFP Creation

The role of an RFP (request for proposal) under this approach of central specification development and for-profit vendor community implementation, delivery, support and evolution is non-traditional.

Traditionally, a procuring agency creates a requirement's specification, issues the specification to a number of bidders, receives proposals from vendors, selects a winner and then provides funding to the winning contractor as it creates the information system. Subsequent to the delivery, the contractor performs warranty maintenance for a period of time and thereafter makes functional changes on a fee for service basis. Under the traditional approach there are three main types of development contract alternatives:

- Firm fixed price
- Time and materials
- Cost plus fixed fee

Under the firm-fixed-price approach, whenever there is a requirement's change the contract's funding is increased and the original delivery date is slipped. While, under the time-and-materials approach, whenever there is a requirement's change the contract's funding is increased and the original delivery date is slipped. And, finally, under the cost-plus-fixed-fee approach, whenever there is a requirement's change the contract's funding is increased and the original delivery date is slipped. Different? Not much.

While the costing and fee structure may be different, the ultimate result is the same. That is, under any of the three RFP procurement alternatives all the risk for proper specification is with the contracting agency. Whenever there are new or changed requirements the contractor immediately requests a contract modification which almost always means a schedule slip coupled with an increase in funding. All the risk is with the public-service agency, and the relationship with the contractor is almost always adversarial.

Under the new approach proposed in this paper the procurement paradigm is fundamentally changed. Rather than adversarial, the role of the public sector agency is to promote maximum involvement by for-profit implementation vendors by providing:

- Technical support for the proven technical specification,
- A completely working prototype of the system,
- Access to the public service agency's metadata repository,
- Immediate release of any specification updates,
- A complete set of initial and iterated functional conformance tests, and
- Test and certification environments of vendor developed systems.

The implementing vendor-community has the following freedoms under this approach:

- They can choose the implementation platforms, hardware, operating system, telecommunications and interoperability environments, and database management system,
- They are free to enhance and evolve the functional specification, and
- They are free to sell to whomever they want providing of course that they have passed state mandated conformance testing

The benefits to the public sector agency under this approach are:

- It's financial obligation is limited to the development of a completely functional prototype,
- It does not have to deploy system development organizations
- It does not incur technical support and maintenance organization costs.

The benefits to the for-profit vendor-community are:

- Vendors are relieved of having to absorb 100% of the cost of requirement's determination, system design and prototype generation
- Vendors do not have to absorb the cost of design iterations and possible loss of reputation resulting from prototype iterations through beta site testers who are seldom happy.
- Vendors do not have to find the market to which to sell its new product.

- Vendors do not have to prove their products acceptable beyond mandated functional conformance tests

All in all this strategy is WIN-WIN as contrasted with the adversarial relationship between the public-sector agency and the for-profit vendors. The RFP must clearly set out this new strategy of "procurement" and invite the for-profit vendor community to participate.

## 7.	Vendor Responses Evaluation

Vendor responses, which would actually be requests for acceptance into the product development effort, should contain:

- Descriptions of the system's design and development strategies,
- Methodologies and quality controls that the vendor is willing to undertake,
- The management and controls that would be placed on systems development to ensure quality product development,
- Proposed schedules,
- The expected quantity and duration of meetings with the public sector agency staff, and
- References of past efforts of a similar nature that would ensure the public sector agency that the vendor would likely complete the effort.

These types of proposal materials are critical because unlike a traditional contract-for-delivery effort, the vendors are largely on their own to produce products. The arrangement is really a partnership that is governed by voluntary participation and expectation of the mutual rewards listed in the RFP Creation section.

## 8.	Contract Award

Contract is certainly the wrong word to express the result of this type of cooperative process between public sector agencies and for-profit development vendors. The proper phrase should either be "cooperative agreement" or "memorandum of understanding."

Once a number of vendors are selected, memorandums of understanding can be scripted that enable vendor access to materials, key members of public sector agencies and special interest groups who would air and then resolve ongoing detailed design and implementation problems, progress status, and the testing of early releases of products.

These group meetings would enable the intended public sector agency customers to be aware of progress and to factor the impending availability of products into their budget cycles.

9. Contractor Management

Contractor management within this type of cooperative arrangement is unique as participation is not really mandatory. The best type of arrangement would be a series of meetings over a period of months wherein vendors could work with public sector agency committees addressing in-progress work, demonstrations, public sector agency development of test data and functional test scenarios, vendor-surfaced anomaly resolutions, central public sector agency development and promulgation of process handbooks, fundamental-process user guides, common process training materials, and the detailed formats for data exchange through the export and import functions.

10. Conformance Testing

The final step is conformance testing. It simply answers the question: How will the public sector agencies know if they received a properly working system? Traditionally, the answer has always been, *caveat emptor*. To resolve that healthy scepticism, the procuring agencies must develop a set of functional acceptance tests. To be useful, the tests must be valid and comprehensive.

The benefits from a comprehensive and valid set of conformance tests is compelling, and include:

- All parties agree-to a minimum essential and testable set of functionality,
- The public sector agency knows that all sold products perform the minimum essential set of functionality,
- The procuring agency procurement costs are reduced because test development and testing is centralized, and
- The for-profit vendor selling costs are reduced because they only have to pass one set of centrally administered tests.

11. Conclusion

The solution proposed in this paper works because it capitalizes on the strengths of the government sector and the private sector, that is,

- A vendor community that has the profit motive to develop, sell, install, and maintain IT systems,
- A public sector central design group that defines, validates through prototyping, and maintains functional requirements, and
- A cooperative conformance testing activity that ensures that all systems sold by the for-profit vendor community conform to the minimal essential functionality.

9

Refernces

1.	The Standish Group. *CHAOS: 1998: A Summary Review:* 1999. The Standish Group International, Dennis, MA 02638.

2.	Matson, Eric. Speed Kills (the Competition). Fast Company (August 1996). Page 1. Web: www.fastcompany.com/online/04/speed.html.

3.	Strassmann, Paul A. *40 Years of IT History*: 1997. Page 6. Web: www.strassmann.com/pubs/datamation1097/index.html

4.	Gorman, Michael M. Knowledge Worker Framework: 1999. Web: www.wiscorp.com

5.	United States Government Accounting Office. Managing Technology: Best Practices Can Improve Performance and Produce Results,: 1997 (GAO/T-AIMD-97-38). Washington, D.C. (web: www.gao.gov)

6.	Strassmann, Paul A. *Outsourcing IT: Miracle cure or emetic*: 1998. Web: www.strassmann.com/pubs/outsourcing.shtml

7.	Matson, Eric. Speed Kills (the Competition). Fast Company (August 1996). Page 3. Web: www.fastcompany.com/online/04/speed.html.