Improving the System Development Process

HEATHER A. SMITH
JAMES D. MCKEEN

QUEENS UNIVERSITY

ABSTRACT

A forum of senior I/S managers was convened to solicit ideas regarding how to improve the system development process -- a process which frequently falls short of its promise of delivering value to organizations. The forum was designed to challenge the thinking of I/S managers rather than to be a blueprint for change. As such, participants were encouraged to question the basic assumptions upon which system development practice currently rests.

The forum identified 10 commonly held assumptions which, when challenged, appeared to limit the way in which the system development process is conducted. Rethinking these resulted in recommendations to I/S management for the improvement of the process. These recommendations should be viewed as opportunities to develop an I/S organization that is much more closely aligned with and supportive of business than it is today and has been in the past.

INTRODUCTION

In the "old days" of system development, systems analysts hand-crafted systems to meet specific user needs. As a result, it was not unusual for a large portion of a project's total development time and budget to be spent **after** implementation, working bugs out of the system and refining system requirements. The imperfections and uncertainties of this approach led to more rational system development techniques and methodologies. These were designed to control the time and effort involved in system development by ensuring that user needs were properly identified **before** a system was created. Methodologies changed the system development process into a much more predictable and organized activity. Systems could be effectively planned and budgeted and better controls implemented. For the first time, users and I/S managers knew clearly what a system was going to do before major time and dollars were spent.

While the system development process became more manageable, methodologies also made it less flexible and considerably more bureaucratic by imposing reporting and documentation requirements and a rigid sequence of steps that all projects -- large or small -- had to go through. And they did little to decrease the overall time involved. Today, business conditions will no longer accommodate long development cycles and requirements that are "cast in concrete". In the 1990s, two key business conditions shape how systems are developed:

Time is a critical factor of differentiation among today's businesses. I/S cannot therefore afford the luxury of taking months or years to deliver a completed system.

Change is a constant. Whether companies are redefining themselves, resulting in re-engineering, merging, downsizing, or taking advantage of global markets, I/S is expected to provide the necessary support and services and do so in a cost-effective manner.

This paper investigates how I/S organizations are addressing these challenges in the context of the systems development process.

METHODOLOGY

It was our feeling after reviewing the literature and examining current practice, that none of the tools currently available for system development has had the dramatic effect on development productivity that was hoped. Therefore, in order to explore and assess what *other things* I/S organizations are doing to modify and improve the system development process, the authors convened a day-long focus group of senior I/S managers from ten leading Canadian firms. Five industry sectors were represented (see Table 1).

Industry	Number
Retail	2
Manufacturing	2
Insurance	3
Banking	2
Telecommunications	1
Total	10

Table 1. Industries Participating in the Measurement Focus Group

Before the session, each group member was asked to prepare a twenty-minute presentation outlining how his/her organization was working to meet these new organizational challenges in system development. The following questions were given to them as a guide:

1. *How are you managing system development within the new organizational imperatives of time and change?*

2. *What you are doing to make the system development process more time and change sensitive?*

3. *Can your current methodologies be adapted to these new conditions?*

   or,

   Do you need a fundamentally different method of system development?

4. *Are there assumptions about system development in your organization that limit the changes that can be made to it?*

5. *How can schedules and budgets be managed if user requirements can't be pinned down?*

6. *How should "maintenance" be handled?*

7. *How can scarce resources be allocated to projects?*

8. *What skills do I/S staff need to develop?*

Participants were asked to focus on the management issues associated with improving the system development process, rather than the tools. They were especially cautioned not to focus on a) the need for more tools that don't exist, or, b) hopes for new tools to solve all their problems.

During the day, each participant answered questions about their presentations. Extensive discussions followed. The researchers orchestrated the discussion in order to 1) clarify definitions making sure that everyone was using the same concepts, 2) explore areas of difference, 3) seek areas of consensual thinking regarding the system development process in order to extract management strategy, and 4) examine common assumptions. In addition, written copies of each presentation and supporting documentation were provided to the authors.

From the presentations, it was clear that commonly-held assumptions about system development were constraining efforts to improve it. Key concepts of system development, such as, What is a system? and, When does system development begin?, are not as cut-and-dried as has been previously assumed. Many managers are attempting to reorient their thinking about systems in one or more ways. This was difficult to do because the prevailing wisdom of systems development dominates many of the tools, people, and approaches to system development. Therefore, we decided that organizing the day's results around these assumptions would not only provide practical information for other managers about how organizations are seeking to improve systems development, but would also provide a framework for challenging thinking about systems development. This, in turn, could yield further insights for both practitioners and academics into how improvements to this process might be achieved.

In this paper, each assumption is discussed and challenged, using both arguments from the literature and from the focus group presentations. Following this discussion, concrete recommendations for management are made based on the collective wisdom and insight of the forum participants. All participants took part in determining the group's final recommendations for other managers. After the paper was drafted, it was reviewed by each participant for accuracy and completeness. Each participant was also polled to determine if the paper as a whole represented a fair assessment of their experiences pertaining to systems development. Some wording changes were suggested, but all members agreed that this paper summarizes the discussion and conclusions of the focus group session.

THE GOAL OF AN IMPROVED DEVELOPMENT PROCESS

According to Goldratt and Cox [1984], the goal of a business is to make money. It does this by increasing sales, by reducing operating costs, and by reducing inventories. All other "objectives" (e.g., increasing market share, producing more products, improving quality, improving customer service, being more cost-effective, keeping pace with technology, and employing good people) are important only to the extent that they further the overall goal. It makes no sense to produce more products if they can't be sold, or improve quality if operating costs increase and new sales are not generated.

The purpose of any subunit of an organization is to further the organization's goal. While this appears to be self-evident, Goldratt and Cox demonstrate that many subunits make assumptions about how they contribute to the organization's goal. These assumptions, which may not be substantiable and may even be counter-productive, unfortunately often go unchallenged. The result is the institution of objectives and measures that serve the subunit's goal but not the organization's.

As a subunit, I/S must question its built-in assumptions to ensure that every aspect of its work relates to the organization's goal. Clearly, the goal of an improved development process in business is to help the organization through creating systems that increase sales, reduce operating costs, and reduce inventory. The implications of these objectives are both internal and external to I/S:

> **Externally**, I/S and its users must ensure that the systems being developed further the goal of the *business* and deliver the expected benefits.
>
> **Internally**, I/S must ensure its throughput (i.e., the systems coming out of I/S) is optimized, its operational expense is minimized, and that there are no unnecessary delays in creating systems (i.e., inventory the company has invested in but has not seen a return on). Some managers have stated this goal as: reducing the "time to market" on a system request without increasing I/S budgets.

I/S organizations naturally have other concerns as part of running a satisfactory I/S organization. Quality, testing, training, long-term architecture, security and auditability, to name just a few, are important components of system development. A system developed without testing, for example, would not further the business's goal. However, these cannot be allowed to become goals in and of themselves. The primary goal of the system development process is still to help the company make money. Actions which further this goal are productive; actions which detract from this goal are counter-productive. This single goal should be the over-riding principle guiding I/S managers in their assessment of how the system development process can be improved.

The need to challenge assumptions about what systems are being developed is well-documented in the re-engineering literature [Hammer, 1990; Davenport & Short, 1990]. However, very little has been documented about challenging the assumptions of the systems development process itself. I/S managers all know that improving the systems development process is neither easy nor straightforward. If it were, they would have done it by now. However, as with any business, I/S managers may be holding onto assumptions about systems development which, if examined, may lead to new and more effective methods of developing systems. The rest of this paper challenges ten of these assumptions and suggests how I/S productivity might be improved by addressing them.

Assumption 1
The development process begins at requirements definition

Different methodologies may call it by different names, but the beginning of the development process is almost always a phase where the users' requirements are outlined, and costs and benefits are refined. Following requirements definition come phases concerned with system design, development and testing, and implementation. Figure 1 illustrates a generic model of this process. People involved in I/S will no doubt use some variant of this model in their own organizations and will have little difficulty relating their own methodologies to it. This is the process referred to when "improvements" to system development are considered.

However, a user sees the development process very differently (Figure 2). From the time a request for system development is made until the time the user sees something concrete is what he or she means by the development process. This usually involves some form of evaluation of the request and prioritization (as well as reprioritization if the project is delayed at this step) at the beginning of the process. It also involves modifications and enhancements at the end of the process to accommodate changes that have occurred since the request went into development.

When looking for ways to improve the system development process, most of the I/S managers in the focus group tended to explore ways to speed up their view of the process. But, as one manager pointed out, if I/S' goal is to truly speed up the delivery of systems, surely it must consider the *total time* involved in creating an effective system, including time spent in the front end getting the project started, and time spent at the back end of the traditional development cycle as well. *At the front end of the process,* lack of user understanding about how I/S evaluates and prioritizes projects, can result in poorly researched and documented system requests. I/S should therefore consider assisting users in preparing their requests to improve both the quality of requests and the speed with which they can be

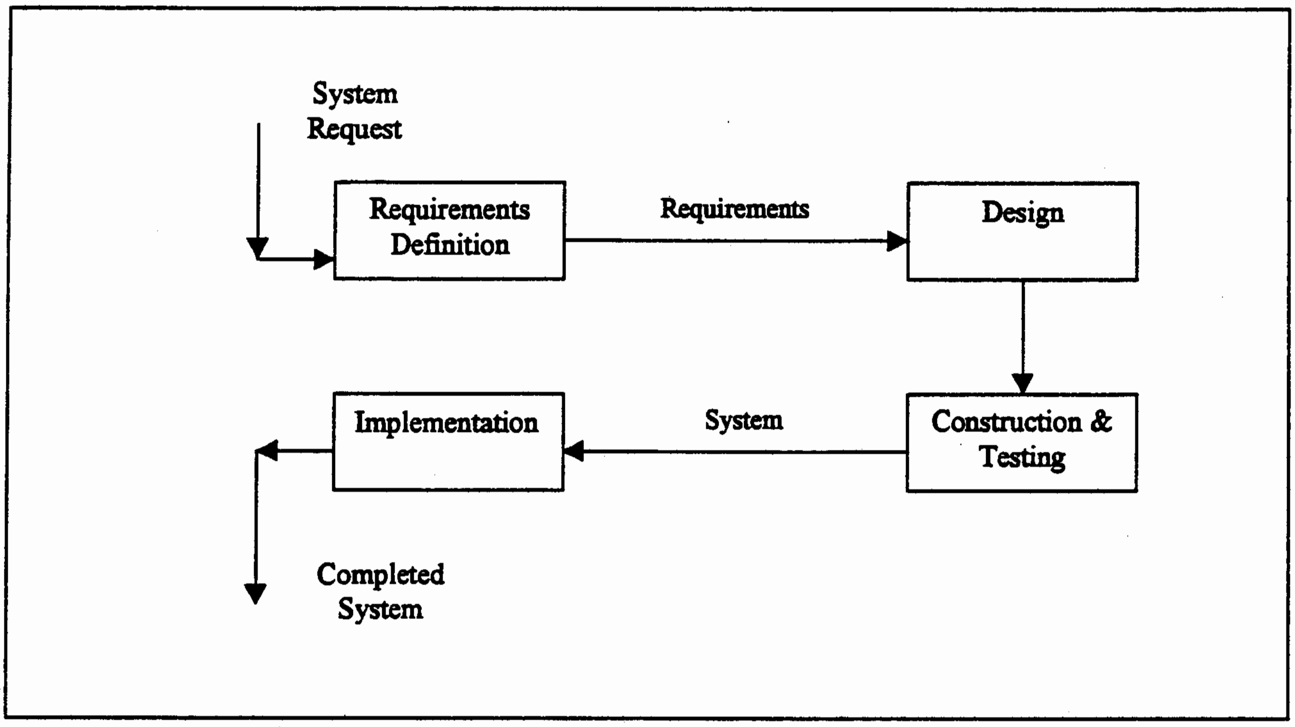


Figure 1 The I/S View of the Development Process

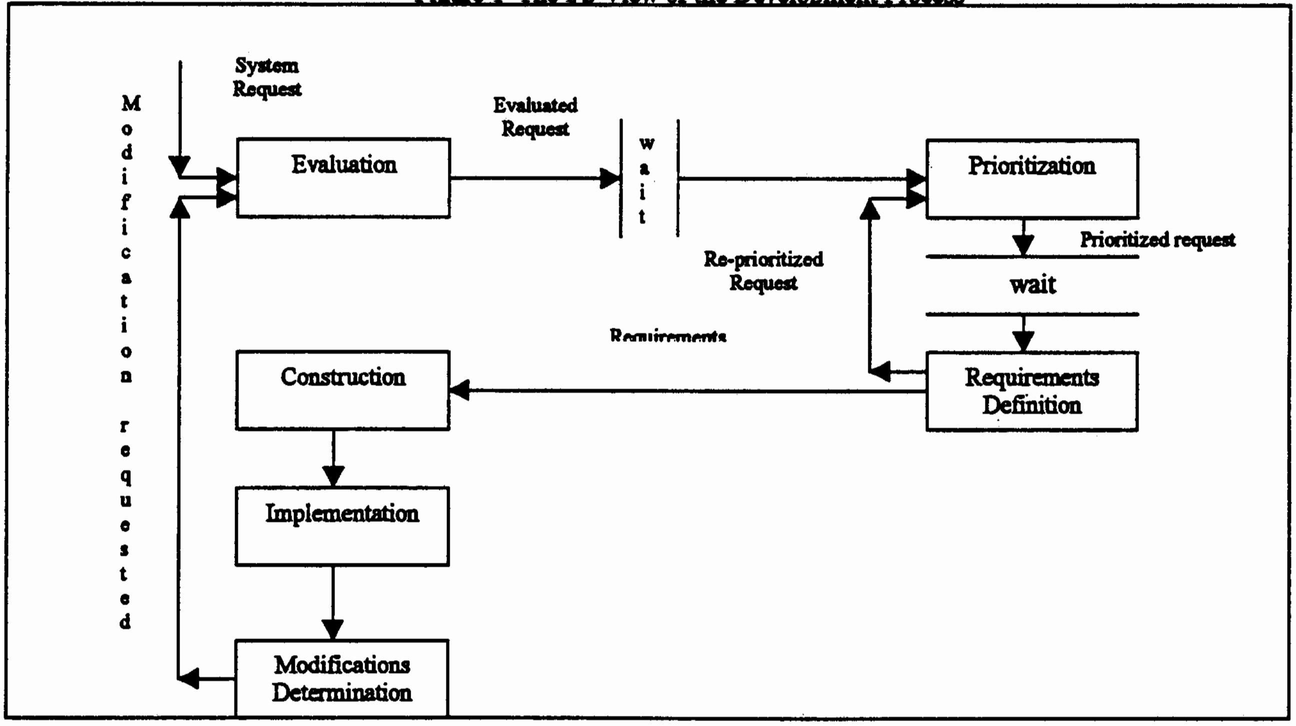


Figure 2. The User View of the Development Process

evaluated. *At the back end of the process,* valuable function can sometimes be cut out of system development in order to meet internal I/S deadlines. All too often, this results in ineffective systems and a spate of maintenance requests.

Recommendations for Managers

Focus group members made the following recommendations:

- ☐ To improve the system development process, examine the whole process, not just what I/S traditionally calls system development.
- ☐ Work with users to clarify and facilitate the evaluation and prioritization processes. By speeding these up, users may feel more satisfied with the overall time to market.
- ☐ Identify work that could be done during prioritization and evaluation that would ensure a request's speedy progress through subsequent phases, e.g., better clarification of user needs and organizational impacts so that there are fewer surprises to I/S and the organization.
- ☐ Evaluate the impact of cutting functions (or moving them to "Phase 2") of the overall process. This could delay significant system benefits and cause a large number of changes.

Assumption 2
I/S has a huge backlog of work

It is common knowledge in many I/S shops that there is a huge backlog of requests for systems. They have queues of many person-years of work. Yet if you asked senior I/S managers which of the requests in the queue are ready to be worked on if resources were suddenly made available, you would get quite a different answer. In fact, while it is true that many more requests come into I/S than staff are able to work on, group members pointed out that a significant proportion of these requests will never be worked on because a) they are not clearly thought out b) they are too expensive or would not result in benefits or c) other systems already in development will address the problem. Thus, the "real backlog" in I/S is considerably smaller than a simple count of system requests would indicate.

Some group members noted that their organizations had an ineffective or obscure prioritization process. Although priorities are set annually, business changes and unplanned projects occur during the year which affect what is worked on. This creates an environment where the prioritization process has little impact on what actually gets done and where considerable informal lobbying can get projects started even if they are not on the priority list.

As well, almost all participants had experienced situations where users simply *must have* a system but who are unprepared to commit the people and make the decisions necessary to develop it properly. Many questioned whether these systems were really ready for development since business expertise is an essential component of effective system development. They felt that such systems should not be considered part of I/S' backlog.

Recommendations for Managers

The group made the following recommendations:

- ☐ Clarify how the I/S "backlog" is measured and managed to truly evaluate demand for I/S services.
- ☐ Distinguish between system requests, and evaluated, approved, prioritized system requests. Only the latter should count as backlog.
- ☐ Assess the prioritization process and make it truly reflective of the business' priorities. Ideally, the prioritization committee should meet at least quarterly and more frequently if emergencies occur.
- ☐ Ensure user commitment to a project is in place before prioritization and that users are aware it as a precondition to I/S participation.
- ☐ Measure I/S productivity and throughput based on the queue of evaluated, prioritized projects (with user commitment) that are completed.

Assumption 3
Users are the best judges of what systems are most valuable to the business

Goldratt and Cox [1984] describe a manufacturing plant where the "cost per part" measurement increased although the plant's operating costs stayed the same, its inventory reduced, and its throughput increased. The company's productivity manager gave the plant a bad report. This story illustrates how many "productivity improvements" are actually the result of local optimization which can have a negative impact on a business' bottom line. Such measures do not truly assess a subunit's contribution to the organization. Instead they assess whether people did their jobs and were fully occupied, and whether they met certain standards of output.

Operational expense cannot be reduced in isolation. For example, trimming a subunit's capacity can have an adverse effect on sales and inventory that is mathematically demonstrable. The objective of productivity is not to optimize an individual subunit, but the organization as a whole.

With systems, many of the same arguments are applicable. A system may reduce costs for a business unit, but may not achieve benefits for the organization as a whole. Systems that perpetuate traditional productivity measures or methods of work may do little to make money for the company. These are the arguments behind much re-engineering work in the organization. Therefore, an assessment of how a system will truly contribute to the *overall* goals of the organization, not merely those of the subunit is essential for I/S.

Some of the companies in the focus group allocate I/S resources on the basis of how much money a business unit is willing to spend. While this may be an easy way to allocate resources, some managers noted that it is questionable how this strategy will help less profitable business units become profitable. Other organizations do little to encourage joint sponsorship of projects across organizational boundaries. One focus group manager complained that this promotes short-term thinking and business unit decision-making in isolation and questioned whether these processes truly result in the most effective systems being developed.

Recommendations for Managers

The group made the following recommendations:

- ☐ Ensure that system requests with the maximum value for the *organization as a whole* get highest priority.
- ☐ Challenge system objectives and measures to ensure that true productivity will be improved.
- ☐ Promote different methods of resource allocation that reward additional resources to projects with joint sponsors, or discourage resource assignment to divisions with the largest budgets.
- ☐ Systems which will reduce operating expenses only should be challenged to prove that they will not affect sales (throughput) or inventory.
- ☐ Assess internal I/S systems on the same basis as other company systems requests e.g., what is the *business* benefit of reusable components?

Assumption 4
I/S throughput cannot be improved once adequate resources have been assigned

I/S managers have been fond of pointing out that "nine women cannot make a baby in one month", by way of highlighting the fact that the process of systems development is a series of dependent activities that cannot be speeded up by adding more people [Brooks, 1975]. Thus, in the past, I/S throughput has been increased by adding resources to enable more projects to be worked on simultaneously [McKeen et. al., 1990]. But has this assumption obscured some important facets of systems development throughput? By viewing time as the critical factor in improving throughput, rather than cost, significant opportunities can be identified for improving the system development process.

System throughput is dictated by elapsed time, not total effort, as I/S managers know. However, what is not always clear is that a significant portion of throughput consists of time waiting, not time working. Time spent waiting -- for decisions, for resources, for specialists -- could be a significant portion of throughput time on any individual system. Furthermore, if more than one system is waiting for the same valuable resource, that resource could be having a significant impact on I/S' total throughput. The mathematical principle of covariance demonstrates that delays caused by such a bottleneck accumulate and result in throughput delays disproportionately greater than the actual delay itself. Identifying and speeding up such bottlenecks could be the most important activity a manager could do to improve I/S throughput [Goldratt and Cox, 1984].

Group members pointed out several common backlogs in I/S which could be areas of improvement:

- ☐ Approvals for what to do or what has been done
- ☐ Access to database and other technical specialists
- ☐ Test data production
- ☐ Information about business procedures or technical capabilities.
- ☐ Approvals from audit, security, or quality control groups.

They suggested that managers should focus on speeding up such bottleneck processes. Perhaps not all projects need to follow the same approval procedures. Perhaps procedures can be streamlined. Are technical specialists bogged down in paperwork? Again, the solutions to these issues are not straightforward or even obvious, but attention to areas where bottlenecks occur can yield some important results in increased throughput – without additional cost.

Recommendations for Managers

The group made the following recommendations for other I/S Managers:

- ☐ Identify bottlenecks in the development process – anywhere work must wait for resources.
- ☐ Speed up bottleneck processes e.g., by streamlining, adding resources.
- ☐ Maximize throughput, not individual productivity.

Assumption 5
The traditional SDLC methodology is the best way to develop most systems

Many I/S managers in the group felt that systems development life cycle methodologies (SDLCs) are necessary tools. They argue that as long as an SDLC methodology is applied with common sense, then it is the best way to ensure a quality product. Without it, they contend, business would lose control of the process. But some questioned whether SDLCs really contribute to the goal of I/S. Some managers in the focus group advocated two alternate methods of systems development:

> **Rapid Application Development (RAD or timeboxing)**. Applications are broken down into small, manageable chunks (of about four months) and developers are given complete control over a project. No formal phases are undertaken, and very little time is spent on management reports or schedules.
>
> **Evolutionary Development.** An iterative approach to systems development, designed to address the critical problems of clarifying user requirements and throughput. It attempts to deliver usable, albeit incomplete, pieces of systems in a series of short, rapid iterations (of about 2-3 months).

Both approaches represent a change from the traditional SDLCs, breaking systems up into smaller pieces, gaining additional flexibility and responsiveness to change. Smaller projects are more productive and easily managed and require fewer management controls or formal processes. Shorter cycles are also more satisfying for users because they feel more involved and in control of the process [Alavi, 1984; Smith, 1986].

Managers from organizations using these approaches cautioned that they require a change of mindset and new approaches to the user-I/S relationship. Developers used to thinking "big is better and more important" or who worry less than optimal solutions will be developed, and users who fear they may never get I/S resources if they don't cram all their needs into one request, can sabotage the process. While all the group members agreed that traditional development methodologies work best in some situations, those using RAD and evolutionary prototyping have found that these approaches solve many problems with throughput and user satisfaction.

Recommendations for Managers

The group made the following recommendations:

- ☐ Learn about RAD and evolutionary development from organizations using them. Many companies are happy to share their expertise in this area with others with whom they are not in direct competition. Although many industry groups (e.g., LOMA) have more formal means of exchanging information, informal networking often works just as well.
- ☐ Establish the architecture as a first step for any projects using these methods.
- ☐ Ensure users and developers understand how these processes will affect their work.
- ☐ Appoint people who can focus on getting the job done, rather than getting enmeshed in the process.
- ☐ Use only those tools which help the process.
- ☐ Manage for results, not the process itself.

Assumption 6
I/S productivity measures assess I/S productivity

Typically, I/S productivity has been assessed by such measures as the number of function points delivered, the number of lines of code produced, or the number of projects delivered on time or on budget. Usually, a considerable amount of time and effort is spent preparing and collecting these statistics. While the group participants recognized that these measures aren't perfect, they contended that better measures simply aren't available and senior management must have a way to control I/S expenditures.

The group was asked to consider whether these measures address I/S' goal in any way. For example, a common measure used is the number of projects delivered on time and on budget. Research has shown that in organizations where these measures are tracked, project leaders will slash function in order to deliver a project within these parameters, regardless of whether it reduces the benefits of the finished project [Smith & McKeen, 1990]. Function points too, according to many users, bear little relationship to the benefits of a system request.

The group agreed that I/S productivity measures need to reflect two things: how well I/T is producing systems; and how beneficial systems are to the organization. There *are* measures available that do this. Measures of average throughput time for each prioritized request, for example, are straightforward and capture how quickly systems are produced. One I/S manager measures benefits achieved, as estimated by the user, and measured against the original project request, to determine how a system contributes to the organizational bottom line. This manager also computes these benefits by I/S employee. Other measures, such as revenues per headcount, can be used to compare I/S productivity over time [see McKeen & Smith, 1992]. These types of measures relate closely to what I/S is being asked to do for the company and will result in closer attention being paid to facets of development that have direct bearing on company performance.

Recommendations for Managers

The group made the following recommendations for other I/S managers:

- ☐ Evaluate all performance measures collected by I/S. Determine how they contribute to the overall goal of I/S. Consider how they might detract from this goal -- either in terms of time spent collecting the data, or in terms of focusing management and staff on the wrong objectives.
- ☐ Consult users regarding how I/S performance could be better measured from their perspective.
- ☐ Investigate alternative, non-traditional measures that more closely tie to I/S' overall goals.

Assumption 7
Productivity can best be improved by automating development activities

Most of the I/S organizations in the focus group are investing significant amounts of time and money in new system development tools, e.g., CASE tools, fourth generation languages, code generators, in the hopes of dramatically improving system development productivity. In spite of considerable evidence to the contrary, I/S managers continue to manage as if technology was the limiting factor in I/S productivity. DeMarco & Lister, [1987] point out that some typical ways I/S organizations attempt to improve development productivity include: mechanizing product development; creating standardized procedures; forcing a tradeoff with product quality; and having staff work significant amounts of overtime.

In contrast to this prevailing perspective, DeMarco and Lister believe that the major causes of lost productivity are sociological not technological. Their book Peopleware, outlines ways to achieve significant performance improvement through more effective ways of handling people, modifying the workplace, and changing the corporate culture. Comparing programmer productivity across numerous organizations, they found a ten to one differential between organizations, not due to language or years of experience or salary, but as the result of how much control a programmer had over his/her work pace. To promote effective working methods, many of the group's managers support such things as training, peer reviews, proven standards, and tools. I/S managers are beginning to believe that improved system development comes from looking for factors that allow people to do their best work, not by controlling and limiting how they should work.

Recommendations for Managers

While many of these management approaches are gradually seeping into the I/S workplace, the focus group noted that some I/S organizations find it more difficult to change than others. It therefore had the following recommendations for other I/S managers:

- ☐ Identify ways your organization is trying to improve productivity. How many relate to automation of methods? How many are non-technical?
- ☐ Examine how much control development staff has over the development process. Look at the paperwork required, whether a team can adjust its office space, if staff have adequate control over interruptions and, how much control management exerts.
- ☐ Identify ways that managers can *help* staff work, rather than *make* staff work.
- ☐ Evaluate your methodology. Is it truly flexible? Or are large portions of it compulsory? Consider replacing methodological control with minimum standards and peer reviews.

Assumption 8
Maintenance should be minimized to enable more new system development

Most I/S managers try to minimize maintenance, (i.e., *optional* changes to existing portfolios, *not* error correction or mandated changes) in order to maximize their resources for new systems development. As a result, maintenance is seen as a cost and many I/S organizations rigorously limit this work in the belief that maintenance detracts from the development of larger, more beneficial systems. This has been a major source of frustration to users who often cannot get small, but significant (to them) changes made, and some focus group managers questioned the assumptions on which this principle is based, including:

- Is maintenance always less beneficial than system development?
- Are current methods of doing maintenance the most effective?
- Can maintenance throughput (i.e., the turn around of maintenance requests) be improved?

They noted the benefits of maintenance are more immediate and easier to achieve than those of new systems. Not only is it harder to make unsubstantiated claims of benefits sometime in the future, maintenance benefits are more likely to be achieved, and achieved more quickly and at a lower cost than new system benefits.

These managers pointed out that like system requests, maintenance requests must be evaluated on the basis of their contribution to the productivity of the organization overall, and not to local optimums. However, for requests that pass this test there are good arguments to make maintenance a priority over new systems, notably the likelihood and speed with which the benefits can be achieved relative to the resources invested.

Finally, maintenance throughput needs to be assessed as carefully as other I/S throughput. Many of the steps outlined above can be applied to maintenance as well. Participants noted that the testing and implementation processes are particularly time- consuming for maintenance work. Some companies limit system releases to a few times a year to reduce these costs. Since benefits do not accrue until after implementation, some group members believed the assumptions underlying this practice should also be challenged. Is the practice of releases truly necessary? Would more releases be beneficial to the company? Can testing be improved? A careful look at the *process* of maintenance may well yield significant results in throughput at no additional cost.

Recommendations for Managers

The following recommendations were made by the group:

- Examine limitations on how much maintenance can be done. Consider whether beneficial work is held up because of limited resources or whether limitations are arbitrarily based on industry statistics.

- Evaluate maintenance and system requests together, based on the benefits to be achieved versus the cost to develop.

- Assess the maintenance process. Can throughput be improved? Are restrictions on implementation (i.e., releases) justifiable from a business perspective or for the ease of I/S?

- Assess the lifecycle of individual systems to determine an appropriate maintenance strategy. If the system is supporting a growth area, for example, more maintenance may be appropriate.

Assumption 9
Expensive rebuilding is the only way to upgrade a system technologically

Most I/S organizations must maintain technologically out-of-date systems because redeveloping them merely to upgrade the technology is an extremely expensive proposition. Often, it is only when modifications and the addition of new functions must be limited because a systems is technologically unstable, that I/S is able to justify redevelopment. While recreating a system's functions and data from scratch may be the ideal, group members pointed out that it is not the only way to upgrade a system technically.

Some managers believe that "information systems renovation" can be equally as effective as rebuilding at a fraction of the cost in time and effort. Ricketts [1993] has defined system renovation as consisting of four steps:

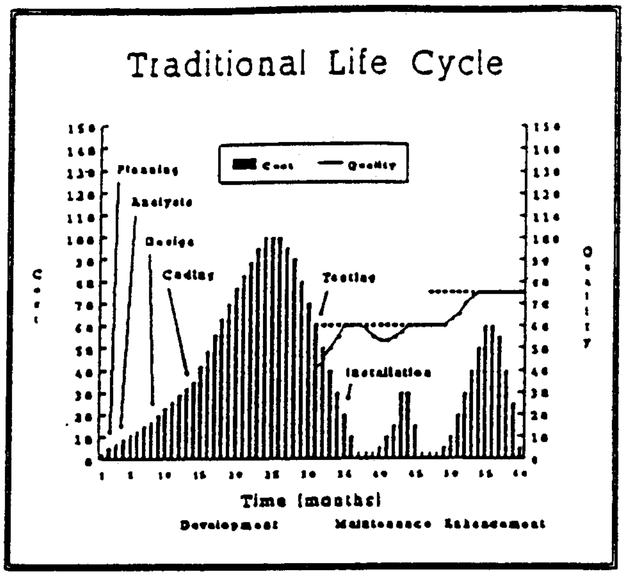
1. **Restructuring** - transforming old source code into new source code, and rationalizing data definitions, using the same technical platform and data access method.

2. **Reverse Engineering** - extracting specifications from existing systems via abstraction and modelling. A system design is extracted from source code and recast as entity-relationship

models, data-flow diagrams, structure charts and screen layouts, and stored in CASE repositories.

3. **Transverse Engineering** - transforming derived specifications into specifications designed for a new system via integration of models. This step moves the existing system to another generation of technology.
4. **Forward Engineering** - creating a working system from designed specifications via code generation. The new code can be generated to perform most original functions and new development can begin with a loaded CASE repository instead of an empty one.

In contrast to a traditional system life cycle project, renovation can be accomplished in less than half the time and effort (see Figure 3). Some of the group's managers noted that not all these activities have to be completed in order for benefits to be gained. Restructuring results in greatly improved technical quality and is also quick and well-supported and is most often used. Reverse engineering is longer and more costly, but results in modest technical and functional gains. Transverse engineering greatly improves both functional and technical quality. Taken together, these three steps form a ladder that enable systems to be developed using forward engineering principles.



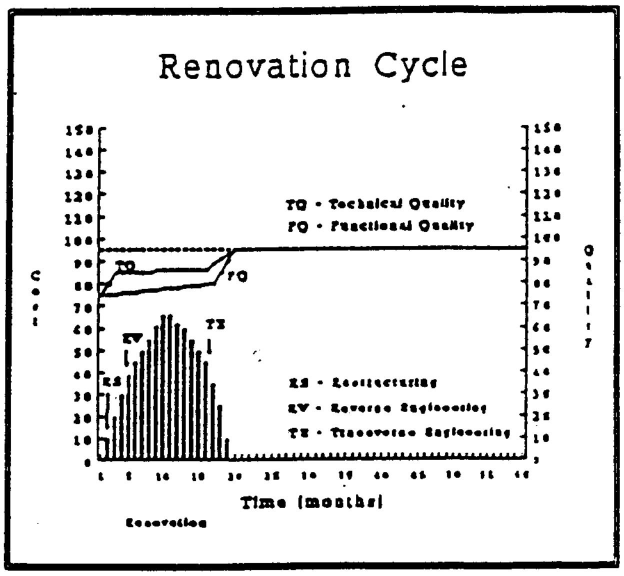


Figure 3. The Traditional and Renovation Life Cycles (after Ricketts, 1992)

Recommendations for Managers

The group made the following recommendations for other I/S managers:

- ☐ Investigate the system renovation process as a substitute for expensive redevelopment.
- ☐ Use inexpensive restructuring to gain technological improvements and enable new function to be added without putting an existing system in jeopardy.

Assumption 10
Systems consist of inputs, processes, and outputs

A final assumption that needs to be explored in order to improve the system development process is the nature of systems themselves. Traditional systems theory states that systems consist of inputs, processes, and outputs. But newer theories and some of the I/S managers in our group suggest that we may be better off to do away with the concept of a "system" altogether, because it encourages tying data and processes together. Why not conceive of "data management" as one form of I/S work and transaction processing as another?

Looked at from this perspective, I/S work becomes an exercise in assembling the appropriate components to fulfil a user need. I/S' responsibilities would include establishing certain components of the infrastructure (e.g., security) and ensuring consistency of presentation. I/S' technical objective would be to achieve interchangeability of access across platforms. Its business objective would be to empower users with tools to extract, process, and manipulate the data needed to do their job. The ultimate goal would be to simplify system assembly through such things as object-oriented techniques until many user needs can be addressed in a matter of minutes rather than months.

How can this perspective be used to improve the system development process today? Changing the way "systems" are viewed will encourage I/S staff to develop data and transactions for the organization, not for individual business units. From a technical point of view, adopting interchangeability of access and consistency of presentation as standards now will further this goal, with minimal impact on throughput and will maintain many of the disciplines implicit in traditional structured development methodologies. Finally, viewing process building as the least significant element of I/S work will discourage the hand-crafting of transaction processes wherever these can be assembled with pre-built pieces.

Recommendations for Managers

While only a few of the I/S managers in the group are actively using this approach, many more believe that it is likely the way 'systems' will be developed in the future. They had the following recommendations for others considering this approach:

- Separate data management from processes as much as possible.
- Encourage users to jointly sponsor data files.
- Adopt standards for accessing data from multiple platforms and to ensure consistency of presentation.
- Encourage I/S staff to look for less expensive ways to process data than programming transactions themselves.
- Provide consulting and technical support for users to enable them to manipulate data for individual business units.
- Encourage system renovation (see above) as first steps towards a "systemless" organization.

CONCLUSION

This paper has examined ten common assumptions about system development with an eye to improving how the system development process delivers to the organization. It was designed to challenge the thinking of I/S managers, rather than to be a blueprint for change. Not every idea outlined here will be useful in every organization, nor will every idea bear fruit if implemented. Most will not be easy to implement. (It is hoped by now that I/S managers will have learned that there are no "silver bullets" in system development.) Nevertheless, these ideas are worth serious consideration as opportunities to develop an I/S organization that is much more closely aligned with and supportive of business than it is today and has been in the past.

Some common themes have been developed throughout this paper. First, focus on activities that bring I/S closer to the basic goals of the organization. Never forget that businesses are in business to make money. Other goals can only be realized if this main goal is being achieved. I/S is responsible for helping business realize this goal. Second, I/S must continue to question assumptions about what it does and what is important. Often, I/S managers "go with the flow" in the industry rather than questioning if something is good for their organization. *Third*, I/S must work to develop and use meaningful measures to evaluate the work it is asked to do. These need not and should not be complex. Beware of measuring something that does not contribute (either directly or indirectly) to the goals of the organization. Third, I/S must use tools where appropriate but only where they make sense. New technology should not be adopted for its own sake but because it can improve on what I/S can deliver to the organization.

Improving the system development process is a challenge for all I/S managers. While there are alternatives to the traditional lifecycle approach, they are not straightforward and could very easily be equally as problematic for organizations. Today's I/S managers therefore must look amongst the available alternatives to find those that meet the specific needs of their organizations. Only one thing is certain: time and change will continue to create pressure to find new solutions. These solutions can only be found if I/S managers 'push the envelope' of current thinking about systems.

REFERENCES

Alavi, M. 1984. An assessment of the prototyping approach to IS development. *CACM* V. 27, No. 6, June.

Brooks, Fred. 1975. *The Mythical Manmonth: Essays on Software Engineering.* Reading, Mass., Addison-Wesley Publishing.

Davenport, T. and Short, J. 1990. The new industrial engineering: information technology and business process redesign. *Sloan Management Review*, Summer, V31, no.4.

DeMarco, T. and Lister, T. 1987. *Peopleware: Productive Projects and Teams*, Dorset House Publishing, New York.

Goldratt, E. and Cox, J. 1984. *The Goal.* North River Press, Croton-on-Hudson, New York.

Hammer, M. 1990. Re-engineering work: don't automate, obliterate. *Harvard Business Review*, July-August.

Jones, C. 1991. *Applied Software Measurement.* McGraw-Hill.

McKeen, J., Smith, H., Agrawal, P. and Smyth, D. 1990. The Investment in Information Technology, Proceedings of the *Administrative Sciences Association of Canada*, V.11, part 4.

McKeen, J. and Smith, H. 1993. The relationship between information technology use and organizational performance in *Strategic Information Technology Management: Perspec tives on Organizational Growth and Competitive Advantage*, R. Banker, R. Kauffman, and M. Mahmood (eds). Idea Group Publishing.

Ricketts, John. 1992. How information systems renovation affects systems analysis and design. *Informatica* V.1 No.1, December.

Smith, H. 1986. Get your requirements right – prototype!, *Canadian Data Systems*, October.

Smith, H. and McKeen, J. 1991. Do system development tools deliver? An assessment of their impact on the information systems bottom line. Queen's University Working Paper 91-21.

Smith, H. and McKeen, J. 1992. Computerization and management: a study of conflict and change, *Information and Management* 22.

AUTHOR'S BIOGRAPHIES

***James D. McKeen** is a Professor of Management Information Systems and Chair of the Research Program in the School of Business at Queen's University. He received his Ph.D in Business Administration from the University of Minnesota. He is the MIS area editor for the* Canadian Journal of Administrative Sciences. *His research interests include methods of user participation, the design and implementation of application systems, and strategies for selecting application systems in organizations. Most recently, he is involved in a large program of research to determine the value of information technology. Jim is a frequent speaker at business and academic conferences throughout North America and is a founder and co-convenor of two industry forums – the **I/T Management Forum** and the **CIO Brief**. His research has been published in the* MIS Quarterly, *the* Journal of Systems and Software, Information & Management, Communications of the ACM, Computers and Education, OMEGA, Canadian Journal of Administrative Sciences *and* Database. *Jim has just published a book entitled* Management Challenges in IS: Successful Strategies and Appropriate Actions *which will be available in August 1996 (Wiley Publishers).*

***Heather A. Smith** is a Research Associate at the Queen's University School of Business and with the Society for Information Management's Advanced Pracitices Council. She has an M.A. from Queen's and has worked and researched in the field of I/T Management for over twenty years. Her research interests include effective I/T Management practices, the value of information technology, the strategic application of I/T and integrating academic understanding with practitioner experience. She is a founder and co-convenor of two industry forums - the **I/T Management Forum** and the **CIO Brief**. Heather has published articles in* Information & Management, Database, Canadian Business Systems, Canadian Journal of Administrative Sciences *and several books. She is the author of* The SIM Practioners' Guide to I.S. Measurement *and has just published a book entitled* Management Challenges in IS: Successful Strategies and Appropriate Actions *which will be available in August, 1996 (Wiley Publishers).*