RE-ENGINEERING THE MISSION LIFE CYCLE WITH ABC & IDEF

Daniel Mandl, Michael Rackley
NASA/GSFC Code 511
Greenbelt, MD 20771

Jay Karlin
Viable Systems Inc.
12236 Stoney Bottom Rd. Germantown, MD 20874

ABSTRACT

The theory behind re-engineering a business process is to remove the non-value added activities thereby lowering the process cost. In order to achieve this, one must be able to identify where the non-value added elements are located which is not a trivial task. This is because the non-value added elements are often hidden in the form of overhead and/or pooled resources. In order to be able to isolate these non-value added processes from among the other processes, one must first decompose the overall top level process into lower layers of sub-processes. In addition, costing data must be assigned to each sub-process along with the value the sub-process adds towards the final product.

IDEF0 is a Federal Information Processing Standard (FIPS) process-modeling tool that allows for this functional decomposition through structured analysis. In addition, it illustrates the relationship of the process and the value added to the product or service. The value added portion is further defined in IDEF1X which is an entity relationship diagramming tool. The entity relationship model is the blueprint of the product as it moves along the "assembly line" and therefore relates all of the parts to each other and the final product. It also relates the parts to the tools that produce the product and all of the paper work that is used in their acquisition.

The use of IDEF therefore facilitates the use of Activity Based Costing (ABC). ABC is an essential method in a high variety, product-customizing environment, to facilitate rapid response to externally caused change. This paper describes the work being done in the Mission Operations Division to re-engineer the development and operation life cycle of Mission Operations Centers using these tools.

1. Introduction

With NASA budgets becoming tighter each year, the Mission Operations Division (MOD), which is part of the Mission Operations and Data Systems Directorate at Goddard Space Flight Center (GSFC), has been forced to reevaluate and change how it has traditionally built Ground Data Systems (GDS). The MOD, as an enterprise, could very simply not afford to continue doing "business as usual".

The traditional GDS approach was to implement large facilities that supported multiple, simultaneous missions, with each facility providing a specific type of operational support function. The systems were also typically developed using the traditional development life cycle model, with formal reviews for requirements and design, and large amounts of formal documentation. This GDS architecture and development approach may have been appropriate given the technology and budgets

available at that time, but the MOD could no longer afford this approach. The development cycle was proving to be too long and expensive, and the operations costs associated with the architecture were accounting for too much of the overall budget. The MOD enterprise thus set out to improve itself in these two areas.

A new GDS approach has been adopted that takes advantage of the relatively recent advances in technology and industry standards. The new concept is to build a GDS that is tailored to a mission or family of missions. This required the development of an underlying architecture approach that was flexible, scalable and evolvable.

As mentioned, the MOD also set out to reduce the development life cycle cost. Analysis showed that while operations costs could be reduced with the new architecture, development costs associated with that architecture were not experiencing comparable cost savings. This was surprising to many since the new architecture employs high levels of software reusability across missions. The MOD came to the conclusion that though reusability was an important factor in reducing costs, any further substantial savings could only be achieved by improving the development process itself.

The remainder of this paper focuses in particular on the MOD's efforts thus far to improve the process for requirements analysis. Sections 2 through 4 introduce a costing method referred to as Activity Based Costing (ABC), and the Integration Definition for Modelling (IDEF) modelling used to support this method. The remaining sections describe how this method and tool were applied to the MOD's process improvement experiment.

2. Process Engineering with ABC; Pricing a Requirement

In order to manage processes effectively and to make appropriate decisions about changing them, a detailed and accurate set of metrics is essential. Pooled resources tend to distort the actual cost of a process. When a resource is shared, the traditional method for assigning cost to a project is to use the average cost of past missions, instead of assigning a cost that is tailored to the true needs of the project. With the new GDS approach of tailoring the system to each mission's needs, a comparable costing method was needed so that actual mission requirements could be individually costed, thus yielding a more accurate overall project cost estimate.

ABC is a method devised to model the cost of any process which has first been decomposed through modeling into primitive activities that serve as its building blocks. Once the primitive activities have been identified, costs can be assigned to those primitives. Then optimization of the general process can be performed in forums such as process improvement committees.

IDEF is a Federal Information Processing Standard (FIPS) that can be used as a tool to perform ABC. IDEF actually consists of an integrated pair of tools: the activity modeler (IDEF0) and the data modeler (IDEF1X). IDEF0 is used to model the activities that occur to produce a product or service and therefore shows the interrelationships of work being done in different groups. IDEF1X shows what is being passed between processes by defining a template (i.e. a data structure) for each item. This provides for more accurate, rapid and meaningful insight into interactions among groups. An example

of this might be a form sent to request a service from another group. IDEF therefore acts as an intergroup coordination tool by providing the overall blueprint for the entire process. The best way to use this tool to coordinate different groups is to put IDEF on a distributed network that is accessible on-line to all participants in the process.

3. The Power of IDEF & ABC

IDEF with ABC allows one to continuously assess the implementation of an overall process and thereby determine the point at which the implementation needs to be changed in order to reduce costs.

For example in figure 1, a conceptual process is depicted. The goal is to get from X to Z, however there is a constraint that regardless of what path is taken, it must cross Y as an intermediary point or constraint. For example, X might be the start of a project and Z might represent having a design. Before one can have a design, one must have the system requirements which is represented by Y. There are a variety of paths to get from X to Y, each costing a different amount. The cost to get from X to Y is the sum of the cost of each of the activities traversed to get from X to Y. In this case, path 2 happens to be the least expensive.

But what happens when a technology comes along that causes the cost of activity 3 to decrease from $3.00 to $0.50? This causes the least expensive path from X to Y to become path 1 instead of path 2. But what happens if there are thousands of activities performed by loosely coupled groups, with each injecting technology to perhaps automate an activity in their area? Thus what

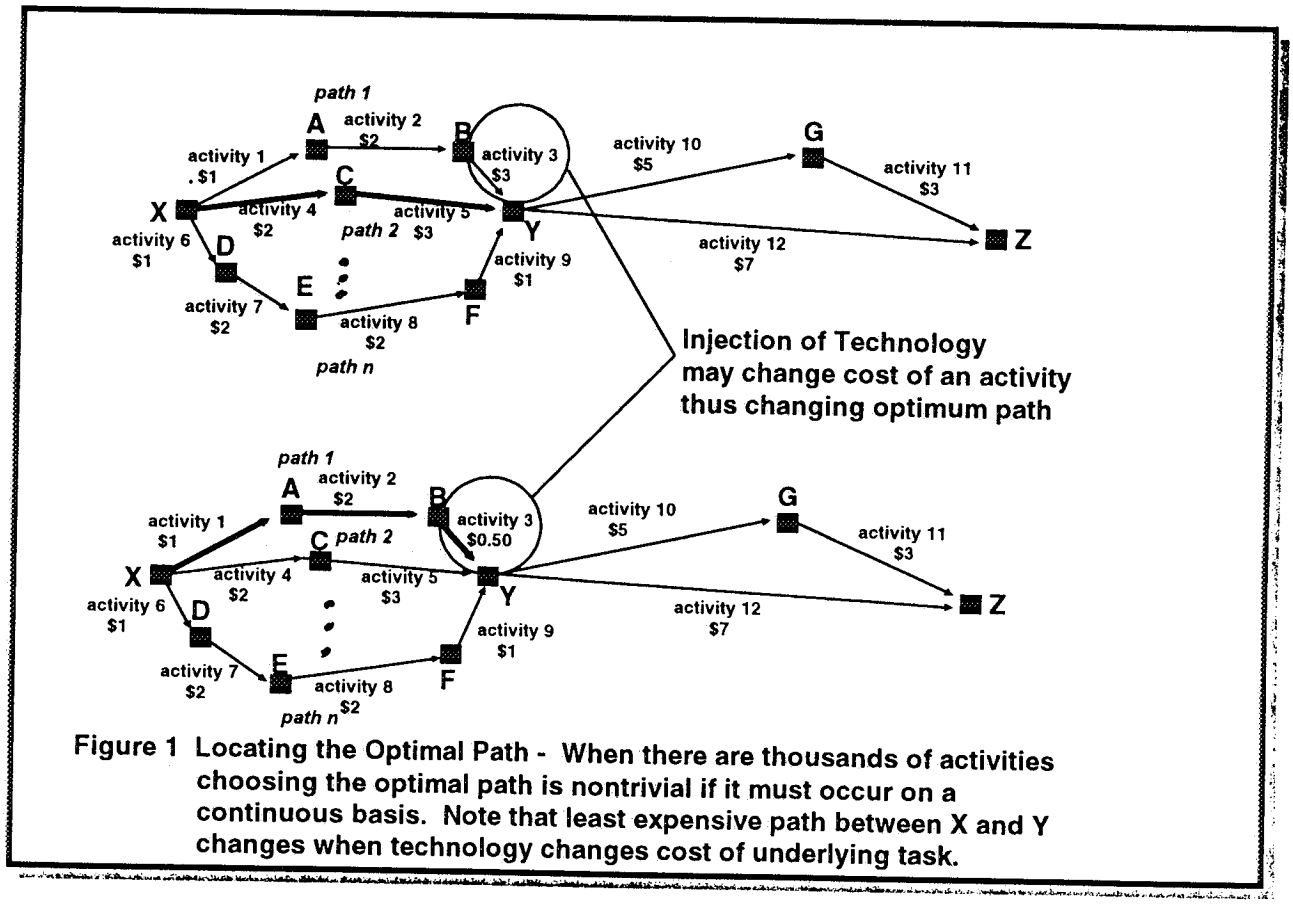

Figure 1 Locating the Optimal Path - When there are thousands of activities choosing the optimal path is nontrivial if it must occur on a continuous basis. Note that least expensive path between X and Y changes when technology changes cost of underlying task.

looked relatively simple in this example is in reality very complex!

What happens when the organization chooses to solve problems with the same processes without considering cost impacts of increasing complexity? Without examining the activities within processes and removing non-value added old activities , unneeded constraints are carried along like deadwood at extra expense. For example, it may be necessary to derive the system requirements, but it may not always be necessary to have a formal System Requirements Review (SRR) if there is a high degree of reusability.

4. IDEF Nomenclature

Of course, models created through the use of IDEF are more sophisticated than the conceptual drawing in figure 1. Figure 2

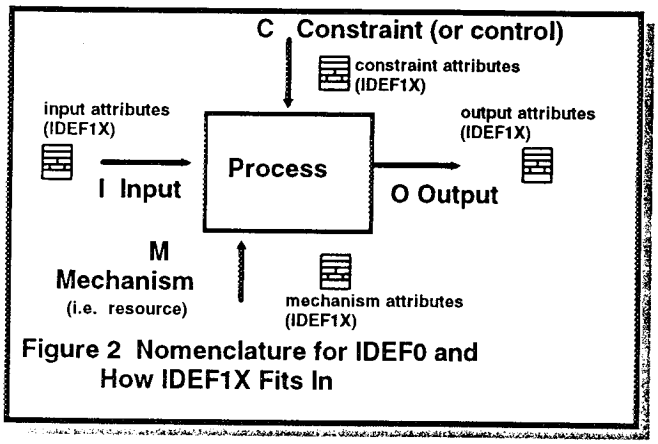

Figure 2 Nomenclature for IDEF0 and How IDEF1X Fits In

depicts the key to reading an IDEF0 drawing. Note the acronym ICOM helps to identify the key elements of an IDEF drawing where I represents Input, C represents Constraint, O represents Output and M represents Mechanism. One of the key features of IDEF is its ability to link the drawings to an underlying database. Also, the drawings are hierarchical to allow one to reveal more and more detail as needed.

5. Experiment Approach

The following steps were taken in conducting the MOD process improvement experiment:

a. Identify target process for improvement.
b. Gather baseline cost data.
c. Define activities that comprise the process using IDEF0 (referred to as AS-IS process).
d. Identify potential problem areas.
e. Identify the underlying business rules using IDEF1X
f. Develop improved process (referred to as TO-BE process).
g. Quantify potential cost improvement using ABC.
   (1) Show main cost driver activities.
   (2) Identify resource cost drivers, e.g., needing specialized skill only half-time but required to hire a full-time person.
h. Measure the new process to verify improvement.

6. Target Process Selection

In order to define the target process for improvement, a typical mission life cycle was first defined as follows:

- Develop System Requirements
- Design System
- Build and Test System
- Operate System
- Maintain System

This top level process is illustrated in figure 3. Although this figure applies specifically to the development and use of a Mission Operations Center (MOC), it was actually derived from a higher level diagram of the

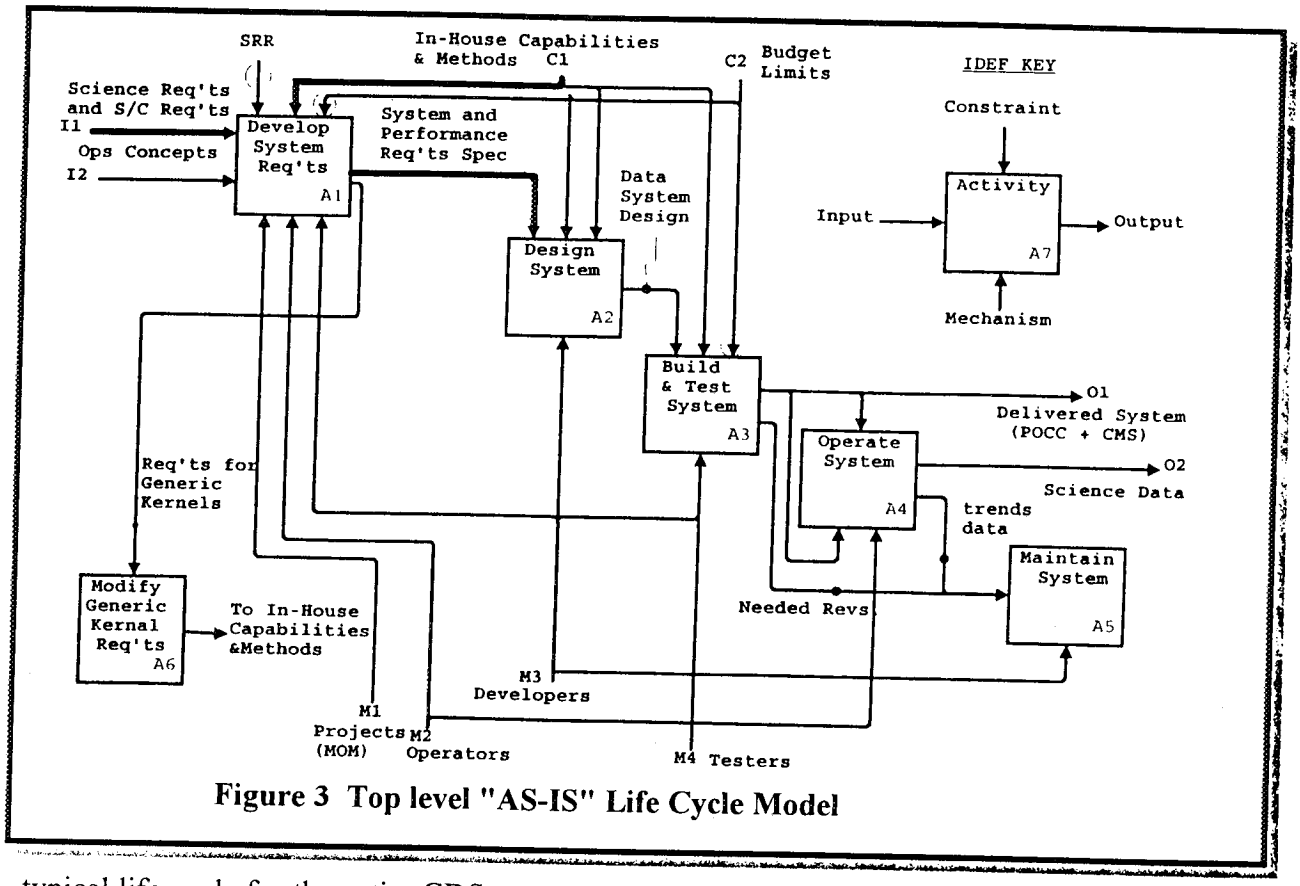

Figure 3 Top level "AS-IS" Life Cycle Model

typical life cycle for the entire GDS.

The next step involved taking some gross measurements of the different life cycle phases to see where the largest portion of the resources was spent. The build phase of the life cycle was found to be an increasingly smaller portion of the total cost. This was due to the employment of reusable building blocks. This meant that the major cost drivers no longer resided in the generation of software. They instead resided in the other life cycle phases, primarily developing system requirements and testing. Therefore the greatest remaining potential for cost savings was in these other phases. The requirements analysis phase was thus chosen as the target process for improvement.

7. Process Analysis

The "AS-IS" and "TO-BE" requirements processes are illustrated in figures 4 and 5 respectively. Figure 6 shows the underlying cost spreadsheets and figure 7 is an example of the underlying entity relationship model.

A cursory examination of figure 4 , suggested the following problems:

a. Function A13 "Negotiate Exceptions" is a trigger on Functions A11 and A12. That is, A13 is required to feed certain previously identified exceptions back to their source for reconsideration. This reiteration (loop-back) multiplies the effort.

b. Additionally, there is conflict between inputs into the "Analyze Mission Req." from two different sources. This indicates that the changes resulting from the unresolved constraints and inputs (exceptions) perturb the on-going preparation of other requirements, necessitating

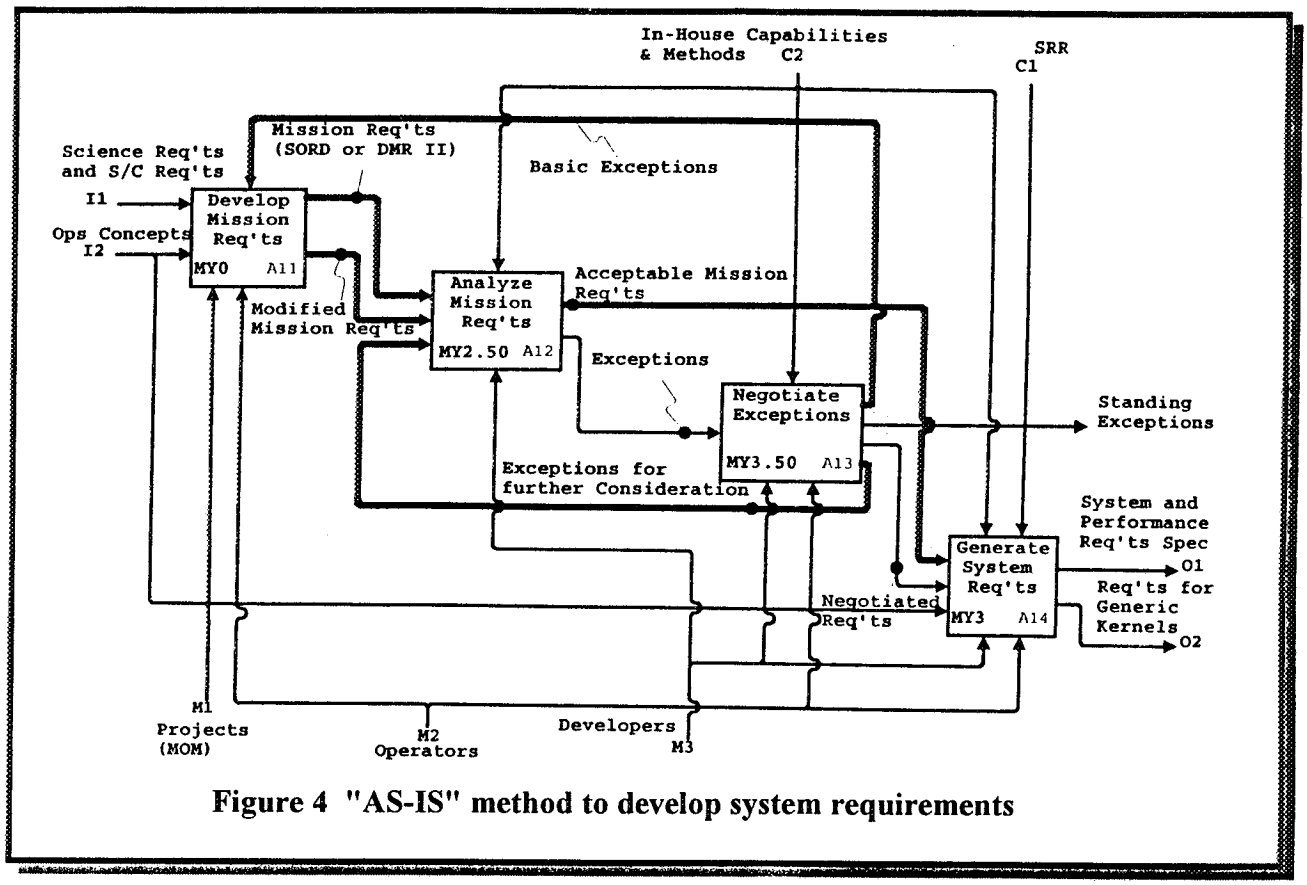

Figure 4 "AS-IS" method to develop system requirements

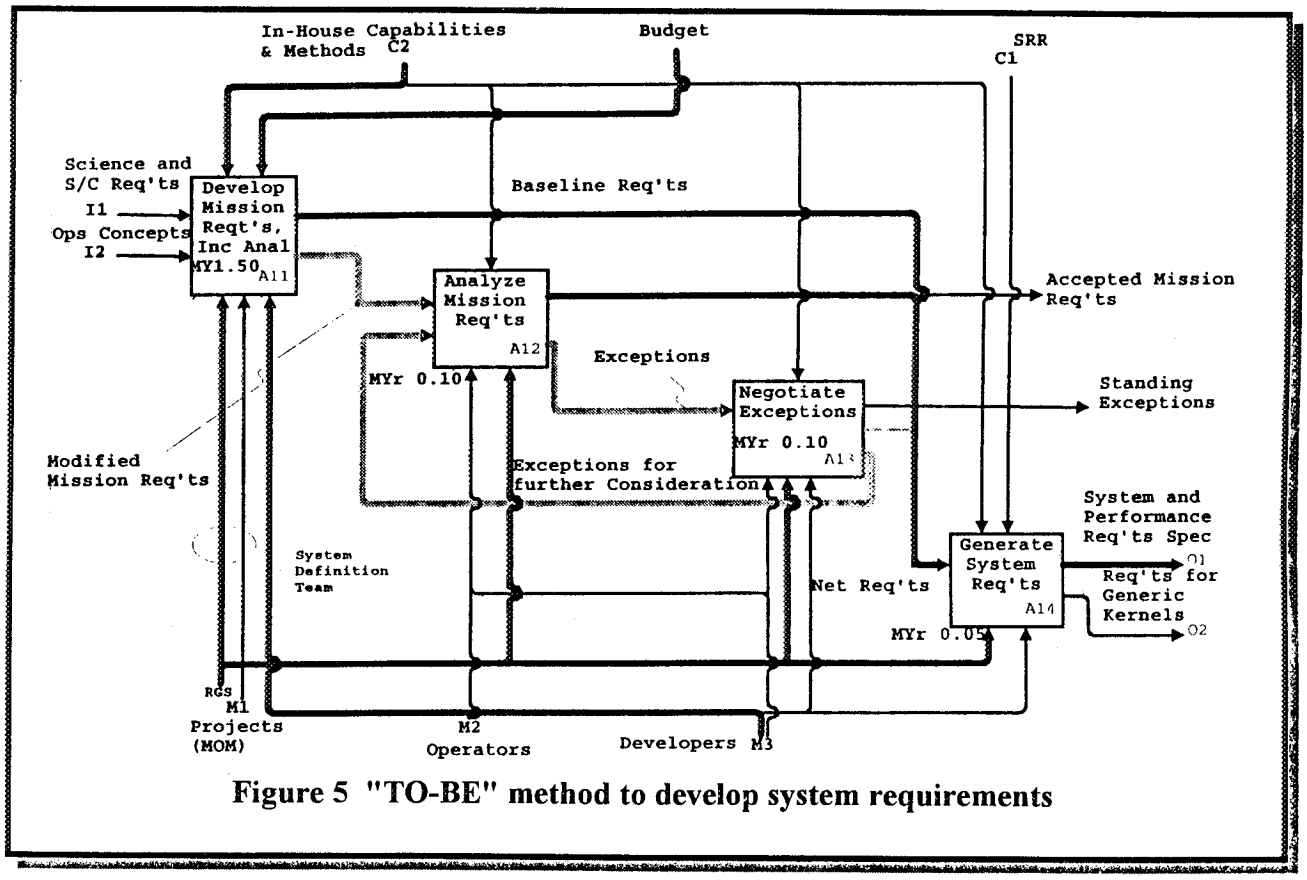

Figure 5 "TO-BE" method to develop system requirements

coordination. Thus, management and developer resources, as well as operator and projects' personnel time, is consumed unnecessarily.

8. Process Improvement

The AS-IS process has been redesigned to largely eliminate feedback as shown in fig. 5, TO-BE, while retaining its basic functionality. This was accomplished by making the Developers, mechanism M3, and their personal knowledge of the constraint C2, "In-House Capabilities", available to A11, which is the initiating point in the requirements process. This early developer involvement has removed many of the information interfaces that used to require translation and documentation "at-a-distance" between A11, A12 and A13, and that had been burning up a significant amount of manpower. To facilitate dynamic person-to-person interaction, the process designers specified there be a System Definition Team (SDT) in order to ensure "eyeball-to-eyeball" operator and developer physical proximity, which eliminated the shuttling of documents back and forth. The Requirements

Node	Develop System Requirements "AS-IS"	Developers	Myrs
A0	Develop and Operate a Project Ops Control Center	9.00	9.00
A1	Develop System Req'ts	3.00	9.00
A11	Develop Mission Req'ts	0.00	0.00
A12	Analyze Mission Req'ts	2.50	2.50
A121	Decompose Mission Req'ts	0.60	0.60
A122	Document Exceptions	0.70	0.70
A123	Determine Acc. and Unacc. Mission Req'ts	1.20	1.20
A13	Negotiate Exceptions	3.50	3.50
A131	Evaluate by Proj/Ops	0.00	0.00
A132	Consider Possible Alternatives	1.50	1.50
A133	Interact on Issues with Affected Parties	2.00	2.00
A14	Generate System Req'ts	3.00	3.00
A141	Develop Initial Ops Scenarios	0.00	0.00
A142	Compare With In-House Methods	1.50	1.50
A143	Identify Performance Req'ts	1.50	1.50
A2	Design System	0.00	0.00
A3	Build & Test System	0.00	0.00
A4	Operate System	0.00	0.00
A5	Maintain System	0.00	0.00
A6	Modify Generic Kernal Req'ts	0.00	0.00

Node	Develop System Requirements "TO-BE"	Developers	M Yrs
A1	Develop System Req'ts	1.75	1.75
A11	Develop Mission Reqt's, Inc Anal	1.6	1.6
A111	Document Mission Req'ts	0.5	0.5
A112	Compare with In-House Capabilities	0.5	0.5
A113	Reconcile Differences	0.5	0.5
A12	Analyze Mission Req'ts	0.1	0.1
A121	Decompose Mission Req'ts	0.05	0.05
A122	Develop Initial Ops Scenarios	0	0
A123	Determine Acc. and NonAcc. Req'ts	0.05	0.05
A13	Negotiate Exceptions	0.1	0.1
A131	Evaluate Documentation	0.05	0.05
A132	Document Exceptions	0.05	0.05
A133	Interact on Issues with Affected Parties	0	0
A14	Generate System Req'ts	0.05	0.05
A141	Develop Initial Ops Scenarios	0	0
A142	Compare With In-House Methods	0	0
A143	Identify Performance Req'ts	0	0

Figure 6 Underlying cost spreadsheets

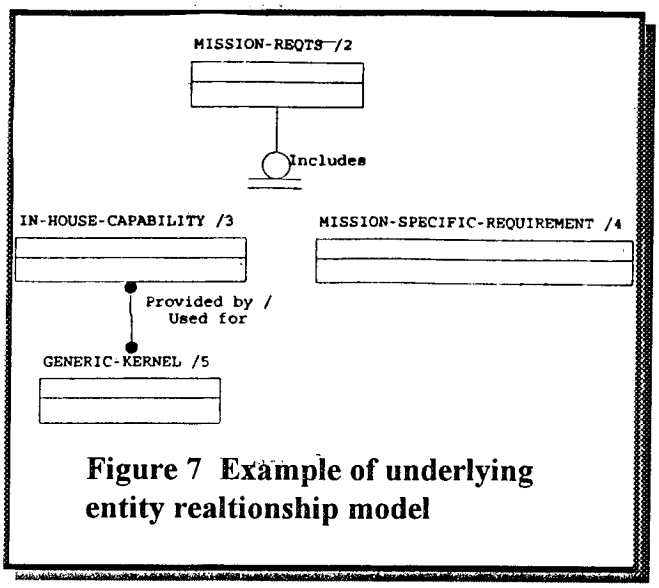

Figure 7 Example of underlying entity realtionship model

Generation System (RGS) replaced this physical function by recording the agreements in real time, thus furthering the cost reduction effect.

9. Results

Chart 1 shows the results that were achieved. Note that, in spite of the fact that they were favored by very high levels of reuse, previous missions still required significant developer effort to get through the requirements analysis process. The shaded area indicates the newly installed process .

10. Future Efforts

The remainder of the life cycle can also be treated in this manner. Since technology changes so quickly, this analysis, including an activity cost breakdown, has to be constantly monitored. The next step, once these models are in place, would be to feed the results into a simulator such as Work Flow Analyzer to do probabilistic analysis on the feedback loops.

	S/W KDSI	% Reuse	DevMyrs to Do Req
SAMPEX	180	40	11
FAST	220	72	9
SWAS	220	87	7
XTE(AS-IS)	250	50	9
ACE(TO-BE)	250	70	1.75

Chart 1 Results of experiment; Only developers' effort considered, no management or publications

11. Conclusion

Together, IDEF and ABC allow large organizations to coordinate their processes and to create a living blueprint for changing and improving business practices. These tools also provide a forum for each individual to identify his or her viewpoint and to comment on these processes from such a perspective. For the business entity and its organization to remain viable, and since the primary cost savings potential is in the process and not the product or product architecture, these types of management methods must be instituted along with such items as product innovation. They in fact provide a means to achieve process innovation. Finally, without these type of tools, large organizations find themselves in the situation of making decisions without the necessary metrics.

12. References

Kaplan, Robert S., *Management Accounting for Advanced Technological Environments*, Articles, August 25, 1989.

Moravec, Robert D. & Yoemans, Micheal S., *Using ABC to Support Business Re-Engineering in the Department of Defense*, Journal of Cost Management, Vol 6, No 2, Summer 1992.

NIST, *Integration Definition for Function Modeling (IDEF0)*, FIPS Publication, Dec 21, 1993.