Product Development Process Capture and Display Using Web-Based Technologies

Nader Sabbagian (nads@mit.edu)
Center for Innovation in Product Development
Massachusetts Institute of Technology
Cambridge, MA 02139

Prof. Steven Eppinger (eppinger@mit.edu)
Prof. Earl Murman (murman@mit.edu)
Massachusetts Institute of Technology
Cambridge, MA 02139

ABSTRACT

This paper presents a distributed knowledge capture method used for modeling the product development process. A Web-based solution is proposed to enable rapid collection, continuous update and structured display of organizational and task interactions in large projects.

Modeling the product development process in large projects is a complex exercise requiring numerous participants and the coordination and clarification of a considerable amount of collected information. Currently, this is performed through group meetings, where participants attempt to integrate their fragmented knowledge of the overall development process. Web technology is used in the proposed approach to address the limitations of present process modeling practices.

This paper presents the design and functionality of a web-based prototype system. The system is equipped with 'push' data capture and on-line multi-user issues resolution capabilities. It has been developed on the Windows NT platform using Java, Active Server Pages (ASP), MS SQL-Server RDBMS and JDBC middleware. The prototype system utilizes a multi-tiered Design Structure Matrix (DSM) configuration to present collected data. This paper will also provide sufficient background on DSM-based modeling and its current applications.

1. INTRODUCTION

There is mounting pressure in all industries to reduce the cost and time required to develop increasingly sophisticated products. Meanwhile, fast changing marketplaces, intense competition and rapid technological evolution have magnified the dynamic nature of product development, creating further need for flexibility and responsiveness in project management. Defining and coordinating development teams and activities under such circumstances is a real management challenge, especially in large engagements [2, 4].

Today's large product development programs can be characterized by the participation of thousands of designers, divided into hundreds of cross-functional teams, working on hundreds of thousands of tasks over a period of several years. Aircraft, satellite systems and automobiles are typical examples of products requiring development projects of such magnitude. The challenge in these programs is to overcome the tremendous complexity involved in planning and executing large numbers of interconnected and dynamic design [2, 4] and development tasks. Success usually depends on management's ability to collect and process a considerable amount of continuously changing information for efficient decision making.

Identifying instances of task iteration (planned or unplanned) is critical in reducing complexity and increasing program efficiency [3, 7, 8]. This important aspect of planning is most often neglected because of insufficient data collection and poor means of representation. Traditional project management tools provide a simplified view through the use of precedence network models and are unable to capture the iterative nature of the development process [1, 3].

This research attempts to address implementation issues related to the Design Structure Matrix modeling methodology. This matrix-based technique has proven to be an effective tool for planning and managing product development programs through information flow analysis [1, 3-8]. It is capable of intuitively representing complicated dependencies among numerous project entities and raising visibility on potential iterations in the development process.

2. THE DESIGN STRUCTURE MATRIX (DSM)

Donald Steward introduced the Design Structure Matrix in 1981 as a generic matrix-based framework for information flow analysis [7]. It consists of an N-square diagram showing the interaction of each element with every other element in the model. By reading across a row, one can observe these interactions through the cell contents corresponding to each cross-referenced column. Various conventions are used to define the content of the DSM cells. These conventions usually depend on the model type and the nature of the problem being tackled. The three most common uses of DSM are [1]:

- Parameter-based modeling - used to analyze system architecture based on parameter interrelationships
- Task-based modeling - used for project scheduling based on intertask information flow
- Team-based modeling - used to organizational design based on information flow among individuals and groups

2.1 Task-Based DSM

This paper will focus on the task-based use of the Design Structure Matrix. Figure 1 shows a sample task-based DSM. Tasks appear identically labeled in rows and columns of the matrix and are arranged top-down according to their sequence of execution. Each marked cell represents a task dependency. The convention adopted in this paper regards row elements as information "providers" and column elements as information "dependents" or "receivers". For example, in figure 1, the
marked cell found at row 2, column 4 represents an information provided by Task 2 to Task 4.

Fig. 1. Sample Task-based Design Structure Matrix

Three types of task interactions can be observed from the matrix. In figure 1, Tasks 1 and 2 are “independent” since no information is exchanged between them. These tasks can be executed simultaneously (in parallel). Tasks 3, 4, and 5 are engaged in a sequential information transfer and are considered “dependent”. These tasks would typically be performed in series. Tasks 7 and 8, however, are mutually dependent on information. These are “interdependent” or “coupled” tasks often requiring multiple iterations for completion.

Marked cells below the diagonal represent iterations in the process. This occurs when an activity is dependent on information from a task scheduled for a later execution. Such scenarios often lead to rework and are undesirable. A number of algorithms have been developed [3-8] to minimize such instances of iteration (below diagonal marked cells) by re-arranging the sequence of tasks in the process. Methods are also available on how to handle iterations in the process that cannot be eliminated through re-sequencing.

DSM models using simple binary representations strictly display the existence of a dependency between two tasks without providing additional information on the nature of the interaction. Further studies have extended the basic DSM configuration by capturing additional facts on the development process. For example, the so-called numerical DSM adds task duration in the diagonal elements, and replaces marks with numbers in the off-diagonal cells each representing the degree of dependency between two tasks [3].

2.2 The Data-Driven Approach

This research utilizes a DSM modeling technique pioneered by Dr. David Grose at Boeing called the Data-Driven approach. The method consists of creating process models through explicit capture of information exchange between project tasks. Deliverables/data produced and used by each activity are obtained in order to create a task-based DSM. A dependency (marked cell) is created once a task’s output is defined as input to another task in the process. Figure 2 presents a sample DSM constructed using explicit information flow. As seen, the dependency between tasks 5 and 7 in the DSM results from task 5 producing deliverable β required by task 7.

The practice of explicitly defining information interfaces among tasks presents several benefits. The model enables management to identify inconsistencies and inefficiencies in the defined process prior to the task sequencing analysis. This approach has demonstrated a remarkable ability to highlight redundant and obsolete activities in projects. From the modeling exercise certain tasks emerge as producing deliverables that are not required anywhere else in the process. These tasks are candidates for elimination since the model can clearly prove their lack of contribution to the project outcome. Redundancy is also easily identified anywhere a deliverable appears as the output of multiple tasks.

2.3 Multi-tiered DSM Configuration

Presenting very large models in a single matrix is challenging. When constructing models comprised of hundreds of tasks, the intuitiveness provided by the DSM representation diminishes. It becomes increasingly difficult to identify interfaces by observing the off-diagonal elements of a very large matrix. The method therefore loses its advantage of simplicity and becomes increasingly difficult to grasp.

A very large DSM can be effectively structured into a hierarchy of smaller DSMs. This configuration avoids problems related to presenting extremely large matrices by shifting the focus to smaller ones, obtained through hierarchical decomposition. It also provides the flexibility to analyze the process at different levels of detail. The multi-tiered approach was developed by Dr. Grose at Boeing and has been adopted by this research as an effective strategy for both data capture and presentation. Figure 3 provides a sample view of this multi-tiered approach.

The modeling effort begins from the highest level activities and deliverables in a project (called Level 1). Next, each high-level task is further decomposed into a set of sub-tasks forming a series of level 2 DSMs. In the example presented in Figure 3, the first two tasks in each matrix are decomposed to provide additional detail through the construction of lower level DSMs.

In this structure there are three possible forms of information exchange:
- **Internal Interaction**
  Referred to information being exchanged within a single matrix. Represented by marked cells within each matrix.
- **External Interaction**
  Consisting of information being exchanged between two or more matrices. Figure 3 depicts such interactions with
3. APPROACH

This paper presents a distributed and asynchronous modeling approach to address the outlined limitations of DSM use in large projects. The method is implemented through a Web-based prototype system with the following overall objectives:

- To reduce data collection effort by efficiently engaging a large number of participants in the modeling exercise.
- To promote DSM adoption in project planning and management by providing a distributed and user-friendly access to very large models.

Internet technology was chosen as the most suitable infrastructure for the proposed system due to its ease of deployment, cross platform capabilities and flexibility in data capture.

3.1 Distributed Data Collection

The responsibility for model construction and update is delegated to various individuals in the organization. A model coordinator handles the overall administration and coordination of the modeling activities. This individual kick-starts the modeling exercise by constructing the first level matrix through information collected from a small group, typically comprised of high-level management possessing a broad view of the development process and its goals. From this point onward the modeling activity can be delegated to a larger group of professionals. This is accomplished by assigning the decomposition of each task to suitable individuals in the organization.

The Web interface allows a user to select a task for decomposition and assign another individual to model this lower level DSM. The system then automatically generates an e-mail to the target individual requesting his/her cooperation in the modeling activity. The e-mail contains information such as the user ID and password needed to logon the system as well as a hyperlink to the web-application’s home page. Upon accessing the system, the user is greeted with information aimed at ensuring a full understanding of process modelling and its benefits.

Each data collection participant is provided with a web-page, used to communicate specific requests, issues and announcements. Upon login, the user is presented with details of the modeling request in his/her web-page. When choosing to create the requested DSM model, the participant navigates through a series of screens that prompt him/her to enter all tasks and corresponding responsible individuals/teams in the process. Upon completion of task entry for the model, the user proceeds to the definition of existing interfaces by specifying input and output information elements for each outlined activity.

All areas of the model are accessible to the user community in a read-only mode. The model coordinator has editing rights to the entire model, while each participant is capable of modifications to matrices he/she originally created. This ensures needed discipline and control over potential changes to the model.
3.2 Usability

Figures 4 and 5 present a sample view of the matrix representation and some of the supporting screens used by the Web-based tool. To successfully engage users in the modeling exercise and promote the use of DSM methodology, the tool is equipped with an easy to operate user interface. A number of key features have been introduced to achieve this goal. These are summarized as follows:

- Users are able to quickly pinpoint the dependency represented by each cell in the matrix. As seen in Figure 4, when positioning the mouse over a particular cell, the two corresponding tasks are highlighted. In the illustrated example, the two tasks “Define High-Level Specs” and “Perform High-Level Design” are highlighted when pointing to the cell depicting their interaction.
- The individual or group responsible for the execution of each activity is presented in its corresponding row. For example, according to Figure 4 the activity “Gather Customer Requirements” is performed by the “Marketing Group”.

<table>
<thead>
<tr>
<th>Responsible Group</th>
<th>Task Name</th>
<th>Level 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Marketing Group</td>
<td>Gather Customer Requirements</td>
<td>X</td>
</tr>
<tr>
<td>Configuration Team</td>
<td>Define High-Level Design</td>
<td>X</td>
</tr>
<tr>
<td>Certification</td>
<td>Develop Test Plan</td>
<td>X</td>
</tr>
<tr>
<td>Research &amp; Development</td>
<td>Perform Feasibility Studies</td>
<td>X</td>
</tr>
<tr>
<td>Manufacturing &amp; Operations</td>
<td>Define Manufacturing Plan</td>
<td>X</td>
</tr>
<tr>
<td>Engineering</td>
<td>Design High-Level Design</td>
<td>X</td>
</tr>
<tr>
<td>Certification</td>
<td>Create Model</td>
<td>X</td>
</tr>
<tr>
<td>Engineering</td>
<td>Perform Detailed Design</td>
<td>X</td>
</tr>
<tr>
<td>Verification</td>
<td>Execute Product Test</td>
<td>X</td>
</tr>
<tr>
<td>Operations</td>
<td>Execute Assembly Test</td>
<td>X</td>
</tr>
<tr>
<td>Support Team</td>
<td>Test Development Plan</td>
<td>X</td>
</tr>
</tbody>
</table>

Fig. 4. Sample DSM in Web-based tool

- Activities are reported in the matrix from top to bottom according to the user-specified order of execution. Clustered black and white stripes in the second (narrow) column indicate which tasks the user has planned for parallel execution. In Figure 4, for example, the user has indicated his/her intention of simultaneously executing tasks “Execute Product Test” and “Execute Assembly Test”.
- Marked elements in the diagonal indicate the decomposition of the corresponding task (existence of a lower level DSM). For instance, in Figure 4, “Develop Test Plan” has been further decomposed into detailed subtasks. Simply clicking on the marked diagonal element displays its lower level matrix representation.

Fig. 5. Sample auxiliary screen depicting task information flow

- Two pop-up windows have been designed to provide a graphical representation of the information flow and assist users in editing the matrix content. The screen in Figure 5 is used to display information requirements and output for a given task. The task is presented in a box, with arrows indicating the information flow. Below each arrow a list box includes all input and output data elements involved in the flow. Authorized users (see section 3.1) are able to add and remove information elements from this list. The window is presented each time an activity in the “task” column is selected.

Fig. 6. Sample auxiliary screens depicting task interaction

- The screen in Figure 6, displays the task interaction pop-up window. This window is accessible by selecting any off-diagonal element in the matrix and is used to present a potential interaction between two tasks. An arrow indicates the existence and direction of an information transfer between the two activities. A list box contains the names of specific data elements being exchanged. Information entities can be added or removed from this list (by authorized users only). To create a dependency a user can simply click on its corresponding off-diagonal element in the matrix and use this graphical interface to specify the data exchange between the two activities in question.
- A pop-up window is provided to add, edit or delete data elements (input or output) in the model. This screen also allows users to search for a particular data component and view its description and usage. Currently such a list is presented alphabetically. Plans are on the way to provide users with a categorization of such information to further facilitate the search process.
- A web page is created for each participant in the modeling process. This page is utilized for communication purposes relating requests and displaying general notifications to users in the modeling community.

3.3 Model Integration

Integrating information received at different times from a large group of dispersed individuals is a major challenge in this approach. One of the biggest issues is the general lack of a commonly understood terminology for teams, deliverables and tasks, the basic ingredients of the Data-Driven model. This often leads to inconsistencies requiring further interaction among participants for a negotiated resolution. The system's goal is identify such scenarios and devise an effective communication mechanism for such negotiations to take place through exclusive involvement of necessary participants.

The proposed prototype addresses two major integration issues labeled Data Disconnect and Inter-level Disparity. Several batch processes perform periodical analysis of the collected data in order to detect such inconsistencies. A set of procedures and interfaces are then responsible for facilitating
issue resolution through direct user involvement. Each scenario is explained as follows:

- **Data Disconnect**
  This occurs when a task's output is not an input to any other task in the process. In essence, there are no customers for the information being generated. Similarly, a Data Disconnect occurs when a task's input is not generated by any other activity in the model. In this case information is requested that is not being produced anywhere in the defined process. This clearly does not apply to boundary interactions (see section 2.3) where the input and output information exchange takes place with entities outside the model.

The above situations can be attributed to one of the following:

a) **Nomenclature**: Interaction has been defined but not detected because of the difference in terminology used for the same information element. For example, Task A defines "Center of Gravity" as an information requirement while Task B indicates "C.G." as it's deliverable. Clearly, this nomenclature difference needs to be resolved for this interaction to appear in the matrix (from Task B to Task A).

b) **Timing**: Disconnect is present because of other participants' delay in providing data for the modeling exercise.

c) **Information Obsolescence**: For output data disconnects the scenario marks the existence of a process inefficiency. The output information element is deemed obsolete since there is no evidence of its use by another task in the process.

d) **Incomplete Model**: When modeling a process for the first time or during task decomposition it is difficult to fully define all activities and their scope. Therefore tasks and deliverables are sometimes left out of the model by mistake.

Upon detecting a Data Disconnect, the prototype system creates an "issue" record for the affected participant. Users with outstanding issues are periodically notified via e-mail to request their participation in its resolution. The customized user page (accessed upon login) presents each participant with his/her data disconnect issues. The user is asked to select one of the following courses of action:

1. Identify the participant likely to be the recipient or provider of the information element in question. This option triggers the creation of an "issue" for the specified participant who is then drawn into the online discussion through the same Web interface.

2. Fix the problem by editing existing model. Upon selecting this option, the participant is transferred to the affected matrix where he/she can proceed with modifications necessary to resolve the issue.

3. Provide feedback to the model coordinator. When the first two options are not applicable, the user can provide comments to the model coordinator. The resolution of the issue in question will then be delegated to the model coordinator.

- **Inter-level Disparity**
  Interactions reported in the multi-tiers of matrices must be consistent across different levels. Once a task is decomposed, the lower level matrix must inherit as external interactions (see section 2.3) the input and output information flow of the parent task. In essence, the lower level matrix provides additional details on the parent task and is therefore required to be consistent with its information interactions. For example, if Task I requires information elements A and B and produces information element C and D, these four information elements must also be present at the lower level matrix as inputs and outputs.

Upon detecting an inter-level inconsistency the system generates an "issue" for the participant responsible for the lower level matrix. The participant is then notified via e-mail and presented with the issue upon logon through his/her user page. The problem is clearly explained and the user is asked to either add the missing information element(s) to the lower level matrix, or resolve the inconsistency by contacting the owner of the parent task. In fact, discussions may lead to the resolution of the issue by simply removing the data element(s) in question from the parent task.

There are also cases where task decomposition leads to the discovery of new external interactions. These interactions are initially unknown to the process modeler for the higher level matrix and are exposed as a result of a more detailed analysis through decomposition. The system identifies such instances and proceeds with its resolution by simply adding the newly found information elements to the list of inputs and outputs of the parent task. The owner of the parent matrix is notified of the change in the model through his/her user page upon system logon.

To ensure that a number of basic modeling guidelines are followed, the data collection interface performs a first pass, online validation on entered information. Examples of such data entry guidelines include:

- Users are required to limit the number of tasks presented in a matrix to 20. This measure is critical to ensure that each layer in the decomposition represents a reasonably comparable level of abstraction thus preventing users from combining detailed and high-level tasks.

- Two tasks are prevented from generating the same deliverable. Users are notified when attempting to add as output a deliverable being already produced by another task in the model. In this case, additional information is provided to assist a participant in resolving the potential conflict. This includes facts on the source of output and the individual currently claiming ownership of the data/deliverable in question. It must be noted, however, that not all scenarios point to the existence of redundancy in the process. Certain information is sometimes deliberately created by multiple sources in order to capture different opinions or obtain results using distinct approaches. Such cases can easily be incorporated in the model by simply assigning different names or version numbers to the deliverable in question.
3.4 Software Development Environment
The prototype is developed on the Windows' NT platform using the following suite of tools:
- Microsoft SQL Server 6.0
- Symantec Visual Cafe® 2.5
- Symantec DB Anywhere
- Microsoft Visual Interdev
- Microsoft Active Server Pages (ASP)
- Microsoft Internet Information Server (IIS)
- Microsoft Front Page 98

4. SUMMARY
This research attempts to provide an effective strategy for product development process modeling in large projects. A distributed and asynchronous knowledge collection and presentation approach has been developed through the use of Web-based technologies and a multi-tiered Design Structure Matrix (DSM) configuration. The method relies on the direct involvement of a large group of geographically dispersed participants for the creation and continuous update of the matrix-based model. User interface features are have been developed to provide an improved process analysis framework to project managers, team leaders and other product development users engaged in the modeling exercise. To resolve complex integration issues during data collection, a web-enabled technique is presented to facilitate on-line interactions among participants.

5. FUTURE WORK
Experiments must be conducted to measure the effectiveness of this approach. The system is therefore planned for trial deployment at MIT industrial partner sites in the fall 1998. Time and effort required to produce a Data-Driven DSM model using the proposed approach will be measured and compared against the traditional centralized and synchronous data collection method. In addition, feedback will be collected from the trial user community on the tool’s usability and its effectiveness to promote the adoption of DSM-based process modeling in the organization. Such facts are documented to support future work in this area.

6. ACKNOWLEDGEMENT
The authors are affiliated with MIT’s Center for Innovation in Product Development, sponsored by the National Science Foundation and a consortium of MIT industrial partners. The research was conducted in close collaboration with the Boeing Commercial Airplane Group (BCAG). The study has benefited from the continuous guidance provided by Dr. David Grose and the support of Mr. Walt Gillette and Mr. Brian Jobes at BCAG. The authors also recognize the contribution of Mark Phillip and Robert Marsh as undergraduate research assistants on this project.

7. REFERENCES