Developing Collaborative Environments
A Holistic Software Development Methodology

Marge Petersen and John Mitchiner
Sandia National Laboratories
mbpeter@sandia.gov jlmitch@sandia.gov

Abstract

Sandia National Laboratories has been developing technologies to support person-to-person collaboration and the efforts of teams in the business and research communities. The technologies developed include knowledge-based design advisors, knowledge management systems, and streamlined manufacturing supply chains. These collaborative environments in which people can work together sharing information and knowledge have required a new approach to software development. The approach includes an emphasis on the requisite change in business practice that often inhibits user acceptance of collaborative technology. Leveraging our experience from this work, we have established a multidisciplinary approach for developing collaborative software environments. We call this approach "A Holistic Software Development Methodology".

Keywords

Collaborative Environments, Software Design, Methodology, Culture, Software Development

1. Introduction

A Holistic Software Development Methodology emphasizes the interdependence of business culture, users and technology. This methodology is the result of design success in several collaborative software environments at Sandia National Laboratories. The differentiating factor in this development methodology is the application of multi-disciplinary viewpoints, and the degree of influence each viewpoint has on the design. The design for the technologies will be business driven. Developing a business driven solution requires the application of many disciplines that we combine to form a methodology team. The establishment of an information architecture, a methodology team, and a process for creating collaborative environments has contributed to this holistic methodology.

2. Information Architecture

Architecture is an overloaded term, and can have many interpretations. An architecture, which is a style and method of design and construction, can be applied to any system, or group of interrelated, interacting, interdependent constituents that are functionally related. These functionally related constituents can best be analyzed and understood by evaluating the process, and letting that process drive the development of technical solutions to support the process.

We have defined an business driven information architecture that includes five elements:

- Activity: The activity(s) or process involved in the accomplishment of a goal.
- Information: The derived knowledge from the application of data that supports the process.
- Application: The transformation of data into information.
- Data: The detailed facts required providing information to support the activity.
- Infrastructure: The underlying business, social and technical foundations that support the process.

In developing a successful collaborative environment for a manufacturing supply chain, we addressed these elements from the top-down. Lessons learned from many software development efforts have proven time and again that technology driven solutions in collaborative environments are difficult to implement due to a lack of user buy-in. This often results in software that does not get used. Our experience indicates that technical solutions must be business driven.

3. Methodology Team

The idea of establishing a methodology team with many viewpoints originated from the concept of developing a "business driven solution" that would support the Information Architecture. A business driven solution lets the process drive the technology being developed.

Developing a business driven solution requires the application of many disciplines, not just computer science or business systems analysis. Leveraging our experience from work on collaborative supply chains and collaborative environments, we have instituted a multidisciplinary approach for designing software. Those disciplines we feel are important to software design include cultural anthropology, business systems analysis, industrial psychology, industrial engineering, systems engineering, data base analysis and computer science.

It is not necessary to employ experts in each of the disciplines identified. It is essential that in approaching a

DISCLAIMER

problem the team considers the "point of view" of each of these disciplines as they seek a business driven solution. The viewpoints applied to our methodology have been summarized as culture, modeling, users, use cases, human computer interaction (HCI) and computer science. The application of these viewpoints to the business driven architecture is shown below in Figure 1.

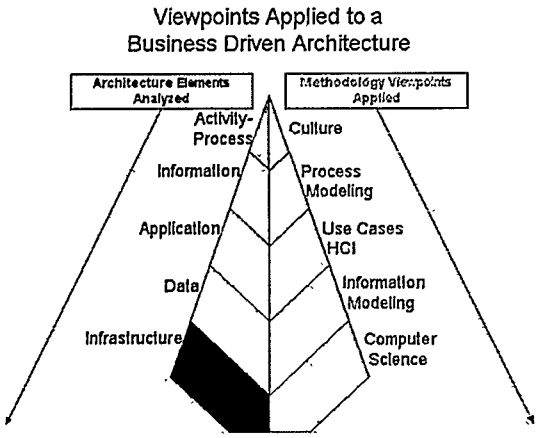

Figure 1. Architecture with Viewpoints

The methodology team may consist of six individuals, each filling a specific role on the team, or the team may only consist of two individuals, but each individual considers all of the viewpoints while proceeding through each step of the methodology. Throughout the development cycle, the team members (viewpoints) interact. The deliverables of each viewpoint are generally required by the other viewpoints to continue development.

3.1 Culture Viewpoint

The culture viewpoint analyzes the activities of the information architecture. It has to uncover the unstated contextual issues so that any proposed improvements to the supply chain or work environment are not in conflict with the real corporate culture.

The cultural viewpoint is one not frequently included in software development. However, as one takes time to understand the business culture, the subsequent pilots and/or software implementation are more prone to be business driven and accepted by customers.

3.2 Modeling Viewpoints

The process and information modeling viewpoints helps us understand the information and data of the architecture. It is central to understanding the "As-Is" process, "To-Be" processes, and the information supporting those processes. Modeling presents a "systems" view of the problem: understanding of people, processes, information, applications, and technology.

3.3 User Viewpoint

Users references all of the participants from the "content" community as opposed to the "design, analyze, and implement" software community. The user community includes corporate champions, managers whose organizations will be affected by modifications of the collaborative environment, and real users who would have to use the new processes or software to make the project successful.

3.4 Use Case Viewpoint

The Use Case viewpoint defines the application of the information architecture. It integrates culture and modeling viewpoint results with an understanding of the customers, software capability and development difficulty into a functional description of the proposed software product. The use case describes how each of the target users would use the system to do their jobs and how each user interacts with other users.

3.5 Human Computer Interaction (HCI) Viewpoint

The human computer interaction viewpoint is concerned with user interaction with the application. During user interaction design, HCI conducts a series of targeted interviews with users, understanding the users' tasks in the context of their work, identifying existing interface capabilities and constraints, assessing usability, and designing a functional user interface (Mayhew, 1999).

The main point is that the users are the users. We may provide them with a little training, but fundamentally they are what they are. The software has to be designed to fit them, their needs, thought patterns, and goals or they won't use it.

3.6 Computer Science (CS) Viewpoint

The computer scientists make the use-case and the HCI design a reality. They are responsible for the infrastructure of the information architecture. They design and build the software. This is extremely important and probably requires more effort than any of the other steps. Without careful consideration of the other viewpoints, what the CS viewpoint creates may be architecturally well designed, fast, pretty, and use the latest whiz-bang technology, but it will end up as shelf-ware without all of the other steps. Too often in our design of new systems we only build software, and are technology driven, not business driven.

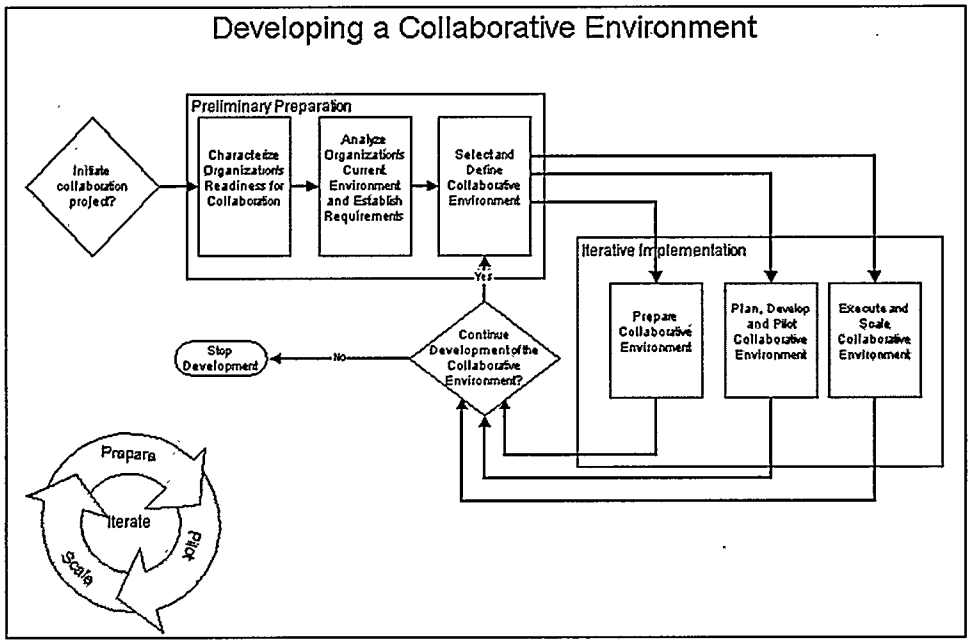

Figure 2. Process for Developing a Collaborative Environment

4. Process For Developing A Collaborative Environment

The Webster dictionary defines collaboration as the act of working together, esp. in a joint intellectual effort. The concept of working together implies a certain level of trust between the parties collaborating. The culture of the environment must support the premise that it is in everyone's best interest to collaborate if the collaborative environment is going to be successful (Bumbo, 1997).

Through our work in designing collaborative software environments and forming collaborative supply chains, we have learned that collaboration is generally instituted gradually, and there are three distinct phases required to fully realize a collaborative environment. They are the preparation phase, piloting phase, and scaling phase. All of these phases must occur if collaboration is to succeed. These three phases are not necessarily monotonic, or linear. But a focused effort up front to prepare for collaboration will enable a successful pilot, and final effort to scale the software system to support a fully collaborative environment.

5. Holistic Development Methodology

The differentiating factor in our software development methodology from traditional methodologies is the application of the viewpoints, and the degree of influence each viewpoint has on the design. The design for the system will be business driven, and the culture viewpoint plays an important role at the beginning of the process. The concept of an iterative implementation for the process of establishing a collaborative environment is also applied, relying heavily on prototyping and piloting the collaborative environment.

Several standard process steps for developing software are maintained in this methodology:

- Characterize the environment
- Establish Requirements
- Develop Functional Design
- Develop Conceptual Design
- Develop Detail Design and
- Prototype and Scale the Software Environment

Figure 3 illustrates the application of the multi-disciplines (viewpoints) to the methodology. It also shows the relative position of the standard methodology process as it relates to the phases of development, and the activities being conducted by each viewpoint. The deliverables for each activity are shown as outputs to the activity.

5.1 Characterize Collaborative Environment

This step is required for the initial preparation of each organization to participate in a collaborative environment. Characterizing the environment includes making an assessment to determine the organization's readiness to engage in collaboration. It involves the organization members (users), and requires that during the process, the users also engage the Culture, Modeling, HCI and Computer Science viewpoint.

The culture viewpoint characterizes the organizations, and how individuals operate within an organization. The modeling viewpoint should determine if cross-functional

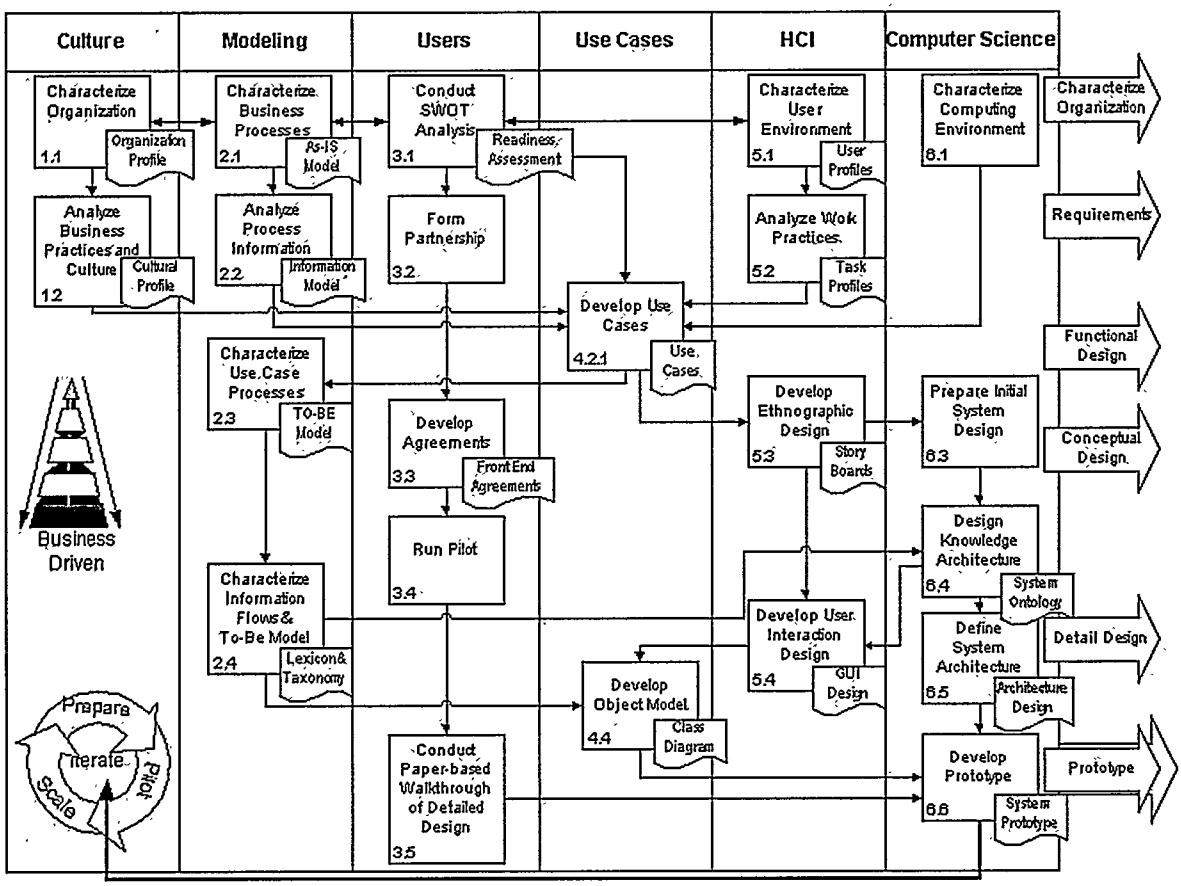

Figure 3. Software Development Methodology

teams exist within the organization, and determine how they share information, through what activities. The Users characterize the organization through a Strength, Weakness, Opportunities Threats (SWOT) analysis, enabling each organization and the supply chain as a whole to carefully determine their readiness to participate in a collaborative environment. The initial assessment conducted by the HCI viewpoint is to develop the User Profiles, while the Computer Science viewpoint characterizes the computing environment at a relatively high level, focusing on the main capabilities and constraints of the environment.

5.2 Establish Requirements

In order to establish the requirements of the technology to support a collaborative environment it is essential for the development team to have a full understanding of the current business practices and culture. Establishing the requirements involves the Culture, Modeling, User and HCI viewpoints.

Business practices and individual behavior take place within a given business environment or corporate culture, and are conditioned by it. Behaviors and texts provide clues to corporate culture and relevant aspects of individual behavior. The culture viewpoint should pay particular attention to the *business environment, business practices and individual behavior.*

The modeling viewpoint looks to understand the sources of information, and how it is used. This step will result in an information model that describes the organization.

The HCI viewpoint studies the work being automated and the socio-culture work environment to derive a task structure.

The users form the partnerships for the collaborative environment, and define the objectives of collaborations. These objectives become a key component in defining the requirements for the collaborative environment.

5.3 Develop Functional Design

The information that has been gathered by the various viewpoints comes together in the functional design. The Use Case viewpoint uses input from the Culture and

Modeling viewpoints, and works interactively with the Users and HCI viewpoints to continuously refine the Use Cases. Characterizations of the user work practices and culture strengths and weaknesses are incorporated.

5.4 Develop Conceptual Design

The key deliverables for the conceptual design are the To-Be Business Model of the proposed system and User Interaction Storyboards developed by the HCI viewpoint.

The modeling viewpoint documents the To-Be Business Model, incorporating the proposed business processes and information flows into one context, as specified in the use cases.

Also using the use cases information along with the real world context from the cultural viewpoint, the HCI team designs User Interaction Story Boards are used to describe to the user exactly how the system will be used, and how they should expect the system to work. By applying anthropological methods to this design, we ensure that the system's features are central to real world use. (McCleverty, 1997).

The computer science viewpoint prepares the initial system design, considering system constraints at a high level. What is the legacy code? Can it be used? What networks & databases need to be in place to support the Use Cases that have been designed? What system security and data security should be incorporated to support the collaborative environment?

Our project experience with collaboration has proven that collaborative agreements are critical to the success of a collaborative environment. The deliverable of this process is a Front-End Agreement and a Pilot Plan. These documents should include measurements for success so that the team can evaluate if they should continue in an unbiased manner. The plan should include organizational strategies, partnership strategies, and metrics that will be used to measure the success of the pilot.

5.5 Develop Detail Design

The detail design is documented by formalizing the proposed To-Be Business Model of the system and the User Interaction Storyboards. Developers can begin to develop code for prototypes or scale the system to support a collaborative environment. The detail design can be developed as the users pilot incremental prototypes of the system, including the business processes that support the new collaborative environment.

The Modeling viewpoint characterizes information flows and the To-Be Model to document both the lexicon and the taxonomy for the environment. This provides a common understanding for both the users and the development team of the participants, products, processes, and data elements (classes and attributes) to be incorporated in the proposed collaborative environment. It also provides a classification of objects in an ordered system that indicates natural distinctions. (Ontology.Org, 1999).

The class diagram is the most common object model that is developed in the detail design. It will describe the types of objects in the system, the attributes and operation of a class, and the constraints that apply to the way objects are connected (Rosenberg, 1999).

The HCI viewpoint starts with the Storyboards that were developed for the conceptual design and begins to develop prototype screens. The interaction and GUI prototype form a foundation for the final walkthrough of the detailed design with the Users.

The computer science viewpoint is responsible for designing the knowledge architecture. The ontology specification provides a comprehensive explanation of the entities, their internal structure, and their rules of combination. From the ontology and the models documenting the information flows, the knowledge architecture will result which supports the capture of knowledge and learning shared in the collaborative environments, which in turn can be used to design and develop future knowledge based systems (Coleman, 1997).

The information generated from the detail design provides the users an opportunity to conduct a paper-based walkthrough of the detailed design. In this walkthrough, missing information and miscommunications are often revealed. As corrections are made, and the detail design is validated, the process of iterating on the design, and developing and extending the prototype until it is a fully scaled system continues.

5.6 Prototype and Scale the Software Environment

The prototype developed may be accepted as designed and expanded to scale a fully collaborative environment, or the process may return to any of the design stages to rework the prototype for a new iteration.

By taking the object models and knowledge architecture, the computer science viewpoint can now add system objects, factor in legacy concerns, look for reusable code and incorporate maintenance and test plans. The language and tools will be selected. At this stage the Capability Maturity Model (CMM) or other traditional methods become useful in preparing the final design for the system architecture.

Frequently, a "thin-line" prototype is implemented in the early stages of development. This is a working system that is designed to handle a subset of problem instances.

It will implement all aspects of system architecture (e.g. user interface, network communication, data storage, major algorithms), although some may be done in a partial manner. Its purpose is to provide a working test of all major areas of system functionality, which can incrementally be extended into a full working system.

Once the collaborative environment has been selected and the pilot plan is completed, piloting can begin. As the pilot is being executed, the development team can study the pilot to begin documenting some of the detail design. This characterization is in preparation for the final development and scaling phase

6. Summary

The Holistic Methodology that we have developed is iterative in nature. Iterative development is crucial to this process. It is important to get experience-based feedback from the participants early and continuously throughout the development cycle. We have found 2-6 months can provide time for an entire iteration. We have successfully developed collaborative software environments by employing the principles of this holistic methodology, in part or in total, on several projects. Specific projects that have contributed to the development of this methodology are Demand Activated Manufacturing Architecture (DAMA) Project, Manufacturing Process Selection System (MPSS), Collaboration Architecture Environment (CAE, formerly PWBDE) and Engineering Authorization Manager (EAM).

Demand Activated Manufacturing Architecture (DAMA)

The U.S. textile industry was steadily losing market share to foreign competitors. The DAMA project addressed this problem by developing processes and technologies to streamline the manufacturing supply chain and associated business processes to provide quick response to measured customer demand.

Manufacturing Process Selection System

A partnership with the nuclear weapons design community has brought together content experts and information technologists to create a suite of design advisor computer tools useful to the weapons design community. These knowledge-based design advisors address geometry, materials, and processes in an integrated fashion.

Collaboration Architecture Environment

The Collaboration Architecture Environment (CAE) is a prototype for the web-based integration of tools, information and people across the product life cycle. The CAE is based on a general object-oriented framework that is applicable to a wide range of engineering design environments. It is structured to provide a consistent approach to managing project data, sharing information, and accessing commonly used design tools.

Engineering Authorization Manager

The Engineering Authorization (EA) Manager is a web-based tool for creating and managing engineering authorizations. It is one of a set of related projects aimed at improving data management in the weapons community. The EA Manager will support creation, retrieval and tracking of EAs, maintain each user's workspace, including old and "in progress" engineering authorizations

7. Acknowledgments

The following individuals assisted in developing and documenting this methodology:

DAMA Project

Leon Chapman, Jessica Glicken, Ruby Lathon, John H. Linebarger, Elaine Raybourn

MPSS, CAE, EA Projects

Bill Stubblefield, Steve Kleban

8. References

1. Bumbo, Notty and Coleman, David (1997) Case Study: Knowledge Sharing the Old Fashioned Way. An Inquiry into the Culture and Technology at St. Luke's.
   Available at http://www.collaborate.com/publications
2. Coleman, David (1997) GroupWare: Collaborative Strategies for Corporate LANs and Intranets. Prentice Hall.
3. Mayhew, D. J. (1999). The Usability Engineering Lifecycle. San Francisco, CA: Morgan Kaufmann.
4. McCleverty, Amy (1997) Ethnography, *Partial Requirements for Computer Science 68I: Research Methodologies*; University of Calgary, Canada
5. ONTOLOGY.ORG Enabling Virtual Business. Available at http://www.ontonogy.org
6. Rosenberg, Doug with Scott, Kendall (1999) Use Case Driven Object Modeling with UML. Addison-Wesley Longman, Inc.

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy under contract DE-AC04-94AL85000.