22

Mini-workshop: revisiting the Denver model for groupware design

James A. Larson
Intel Corporation
Mail Station JF2-52
5200 N.E. Elam Young Parkway
Hillsboro OR 97124-6497, U.S.A.
Telephone (503) 264-8463
E-mail jim_a_larson@ccm.jf.intel.com

Participants

Participants were Hong-Mei Garcia, Noboru Koshizuka, Josef Voss.

1 REVIEW OF THE DENVER MODEL

Participants in an earlier workshop at the CHI'95 conference in Denver Colorado created the Denver model for groupware design [Salvador 95]. Illustrated in Figure 1, this high level model identified a hierarchy of submodels for describing various aspects of groupware designs. The first level of decomposition consisted of three submodels, (1) system requirements, (2) design model, and (3) technology model. The technology model describes the technology used to enable multiple participants to be aware of and control various aspects of the groupware system. The design model was further partitioned into five submodels, (1) the people model which describes users and the various roles they may assume, (2) the task model which describes the scenarios, tasks, and activities which users may perform, (3) the artifact model which describes the objects and their attributes and relationships which users may manipulate, (4) the interactive situation model which describes the participant locations (same, different), the number of participants (small, large), synchrony (asynchronous, synchronous), dependency (loose, tight), and degree of planning (planned, spontaneous), and (5) the interactive protocol model which describes the floor control mechanism, contention resolution, formality of address, and direction of conversation (unidirectional, multidirectional). Software practitioners have devised several models for people, tasks, and artifacts. Groupware systems are relatively new, and models for interactive situation and interactive protocols are just beginning to emerge.

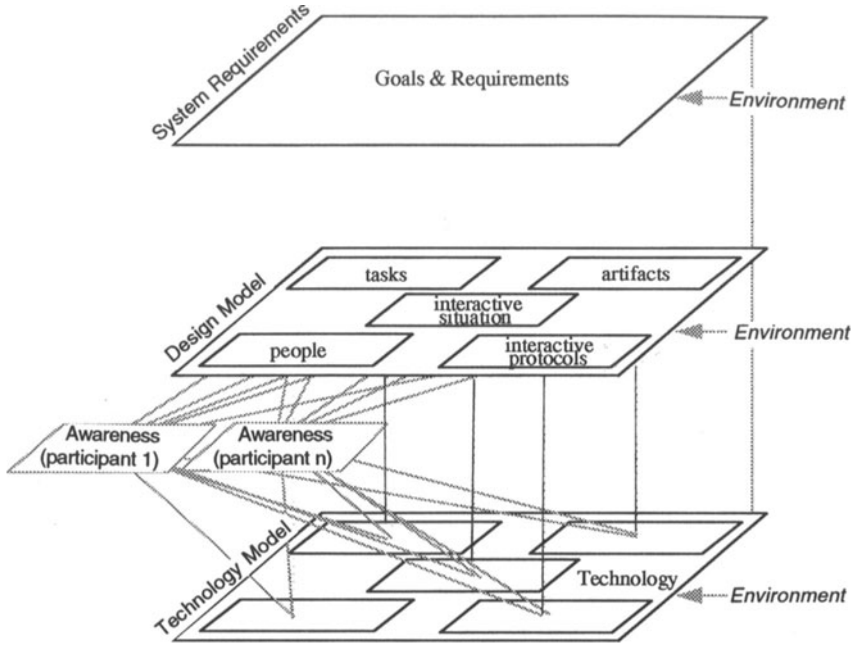


Figure 1 The Denver model for groupware design [Salvador 95]

2 REFINEMENTS TO THE DENVER GROUPWARE DESIGN MODEL

2.1 What is groupware

We defined a groupware application to be a computer application in which multiple users collaboratively strive to achieve a common goal involving multiple tasks. One example of a groupware system would be a conferencing system such as Intel's ProShare™ family of products which include tools for video conferencing, application sharing, and a shared notebook. Another conferencing system is Ventana, which includes tools for brainstorming, organizing, voting, etc. Lotus notes is an example of a non-real time groupware system with tools for e-mail, bulletin boards, and database management.

2.2 Uses of the Denver model

The Denver model could be used for several purposes, including (1) characterizing and comparing groupware systems, (2) defining groupware capabilities and measuring to what degree an existing groupware system supports those capabilities, and (3) guiding implementors in designing groupware applications. We decided that we would concentrate on using the

Denver model for understanding the design space of parameters, issues, and solutions for groupware UI design.

2.3 Refining the Denver Model for groupware UI design

In order to design user interfaces, it is necessary to understand what objects the user may manipulate. We thus refined the objects in the design submodels of the Denver model to describe the major objects of a groupware system. The result is shown in the Entity Relationship diagrams illustrated in Figures 3-6. Figure 2 summarizes the notation used in these figures.

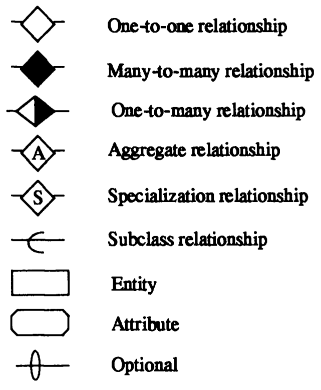


Figure 2 Entity-relationship notation

2.3.1 Task submodel

Figure 3 illustrates the task submodel. A scenario consists of several tasks, which in turn consists of several activities, each of which may be accomplished by performing zero or more computer operations. Operations may be initiated by the user or the groupware system, and each operation is performed by either a user or the groupware system.

2.3.2 Person submodel

Figure 4 illustrates the person submodel. The two principal objects are person and role. Each person may assume one or more roles, and each role may be assumed by one or more persons. A special role called Facilitator will be useful in many groupware applications. Some roles may also be assumed by a synthetic person such as a software agent or process.

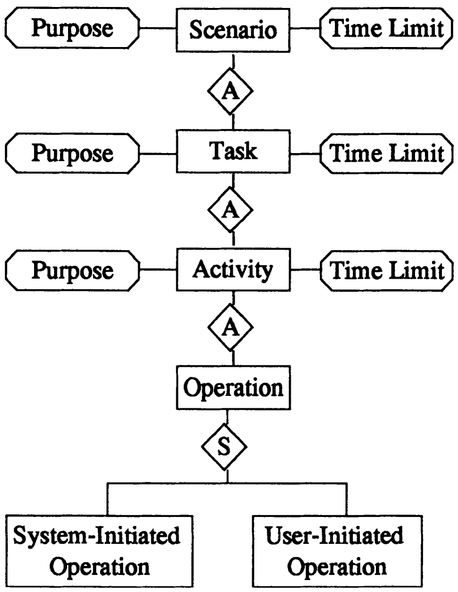


Figure 3 Task submodel

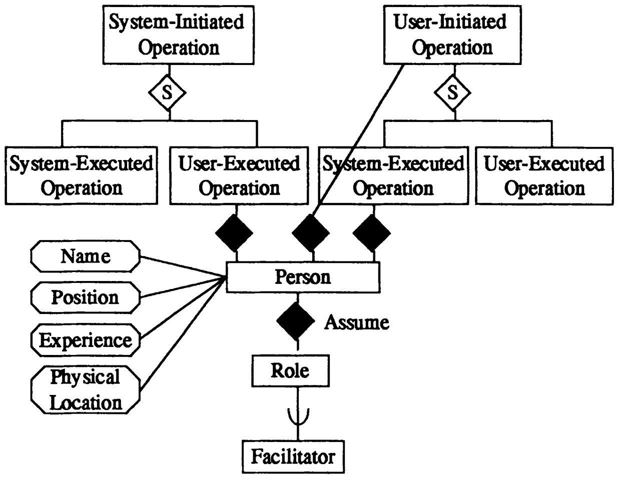


Figure 4 Person submodel

2.3.3 Artifact submodel

Figure 5 illustrates the artifact submodel. Each artifact has a type, for example, text, illustration, graphic, bitmap, audio, video, or computer program. Each artifact has one or more purposes. Artifacts may have attributes and be related to other artifacts. At this level of abstraction, we didn't want to list all possible attributes and relationships, so we represented attributes and relationships themselves as entities. Each artifact is implemented using one or more technologies. Example technologies include storage devices, display devices, input/output devices, and networking mechanisms. From a design point of view, each of these technologies can be characterized by cost, performance, reliability, capacity, etc.

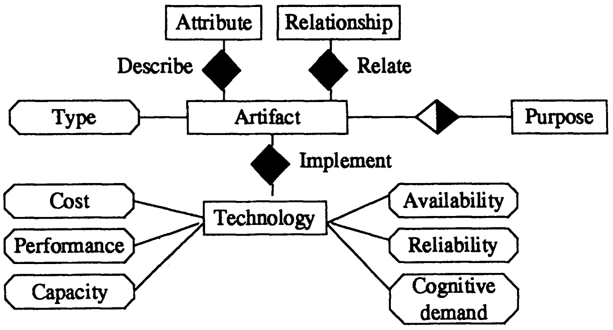


Figure 5 Artifact submodel

2.3.4 Interface situation

Figure 6 illustrates how the interface situation is a many-to-many relationship between person and task. This relationship characterizes the tasks in which each person is involved during a point in time, and includes information about the common culture and context of the involved persons, the synchrony of their interactions and (optionally) where the participants are in an agenda. The interactive protocol attributes as closely associated with the interface situation, but because they can be clustered together, we placed them in a separate entity called interactive protocol. There is a single interactive protocol for each interface situation.

While we do not claim that this entity-relationship is complete or entirely accurate, we felt it was useful in defining important entities in the design model and how they relate to each other. Given this insight, we turned our attention to designing user interfaces for groupware systems.

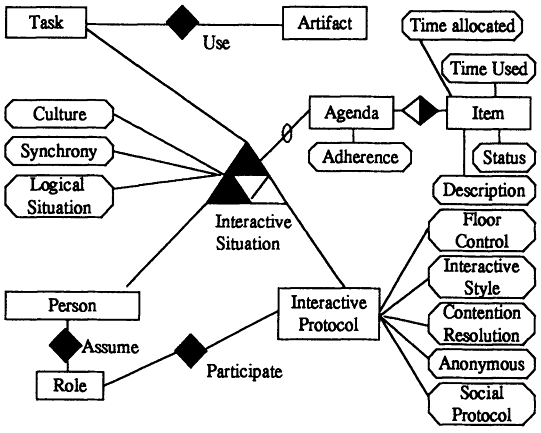


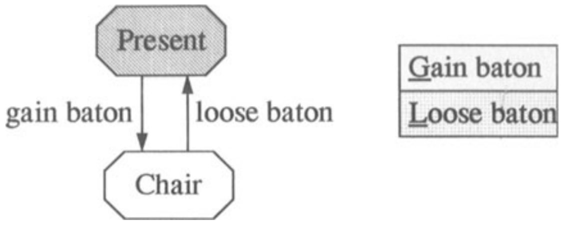
Figure 6 Interface situation

3 UI DESIGN

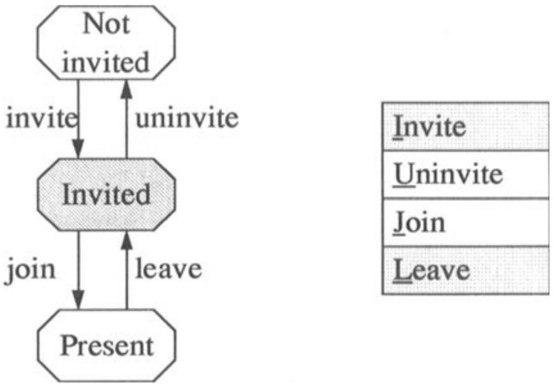
As a starting point, we suggested that each major entity might correspond to a UI interactor (widget, or control). For example, the agenda could be presented to the user as a list containing several items, with each item containing information about its description (the task to be performed), its status (completed, in progress, not started, ...), time allocated for this task, and the amount of time already spent on the task.

The interactive protocol interactor would contain information about interaction style (the relationship among each pair of participants, including which participants are anonymous to other participants), floor control options, and social protocol options. Figure 6 illustrates state transition diagrams for some floor control and social protocols. Each transition between states correspond to a user command. Users change states by selecting currently available commands from an interactive protocol interactor such as a menu. For example, Figure 6b illustrates a social protocol for joining a conference. The state transition diagram indicates that a participant is currently invited. A menu interactor illustrates that the operations of join and uninvite are currently available, and operations invite and leave are not currently possible.

Interactors can also be associated with relationships. For example, a many-to-many can be represented by a graph, by values, by icon position, and by charts [Larson, 92, Appendix C]. This leads to several alternative interactors for the interface situation relationship between task and person, the participants relationship between role and interactive protocol, and the use relationship between task and artifact. Choosing compatible interactors for multiple many-to-many relationships is a challenging UI design problem.



(a) Example floor control protocol and associated menu



(b) Example social protocols and associated menu

Figure 6 State transition diagrams

4 References

Tony Salvador (ed.), "The Denver Model for Groupware Design," SigCHI Bulletin, September, 1995.

James A. Larson, Interactive Software, Tools for Building Interactive User Interfaces, Prentice Hall, Englewood Cliffs, New Jersey (1992).