Web-based collaborative feature modeling

Rafael Bidarra, Eelco van den Berg and Willem F. Bronsvoort

Faculty of Information Technology and Systems, Delft University of Technology

Mekelweg 4, NL-2628 CD Delft, The Netherlands

(R.Bidarra/E.vdBerg/W.F.Bronsvoort)@its.tudelft.nl

http://www.cg.its.tudelft.nl

1. Introduction

Collaborative modeling systems are distributed multiple-user systems that are both concurrent and synchronized, aimed at supporting engineering teams in coordinating their modeling activities. Instead of an iterative process, asynchronously sending product data back and forth among several team members, they turn design into an interactive process, in which several engineers are simultaneously involved to agree on design issues.

So far, only a small number of tools have been developed that somehow support collaborative design activities. For example, tools for collaborative model annotation and visualization via Internet are now becoming available, providing concepts such as shared cameras and telepointers. However, such tools are primarily focused on *inspection*, e.g. using simple polygon mesh models, and do not support real *modeling activities*. In other words, they are valuable assistants for teamwork, but no real CAD systems.

Some more recent research is focusing on the possibility of enhancing existing CAD systems with collaborative facilities [3, 6], with the goal of providing real *modeling facilities* to participants in a collaborative session. To the best of our knowledge, the only commercial system currently offering some collaborative modeling facilities is OneSpace [4]. However, its modeling capabilities are severely constrained by the modeler at the server, SolidDesigner, and by the format into which it converts shared models.

An interesting research challenge consists of developing a collaborative modeling system that offers all facilities of advanced modeling systems to its users, while at the same time providing them with the necessary coordination mechanisms that guarantee an effective collaboration. Among these mechanisms, solutions have to be provided to the critical problems of concurrency and synchronization that characterize collaborative design environments. *Concurrency* involves management of different processes trying to simultaneously access and manipulate the same data. *Synchronization* involves propagating evolving data among users of a distributed application, in order to keep their data consistent. This paper introduces webSPIFF, a new web-based collaborative feature modeling system that is a major step in this direction. In particular, this system demonstrates that a good and consequent exploitation of the *feature* concept can be very helpful in solving these problems.

2. System architecture

webSPIFF has a client-server architecture, consisting of several components; see Figure 1. On the server side, two main components can be identified: the SPIFF modeling system, providing all feature modeling functionality; and the Session Manager, providing functionality to start, join, leave and close a modeling session, and to manage all communication between SPIFF and the clients. The webSPIFF portal component provides the initial access to a webSPIFF session for new clients, and includes a web server where model data is made available for download by the clients.

The clients perform operations locally as much as possible, e.g. regarding visualization of, and interaction with, the feature model, and only high-level semantic messages, e.g. specifying modeling operations, as well as a limited amount of model information necessary for updating the client data, are sent over the network.

The server coordinates the collaborative session, maintains a central product model, and provides all functionality that cannot, or should not, be implemented on the client. In particular, as soon as real feature modeling computations are required, such as modeling operations, conversion between feature views and feature validity maintenance, they are executed at the webSPIFF server, on the central product model, and their results are eventually exported back to the clients.

An important advantage of this architecture is that there is only one central product model in the system, thus avoiding inconsistency between multiple versions of the same model.

3. The server

As a basis for the server, the SPIFF system developed at Delft University of Technology was chosen, which offers several advanced modeling facilities. First, it offers *multiple views* on a product model, each view consisting of a feature model with features specific for the application corresponding to the view. The current version of webSPIFF provides two such views: one for design and another for manufacturing planning of parts. In the design view, the feature model consists of both additive (e.g. protrusions) and subtractive (e.g. slots and holes) features. In the manufacturing planning view, the feature model consists of only subtractive features. All views on a product model are kept consistent by feature conversion [5]. Second, it offers *feature validity maintenance* functionality. This can guarantee that only valid feature models,

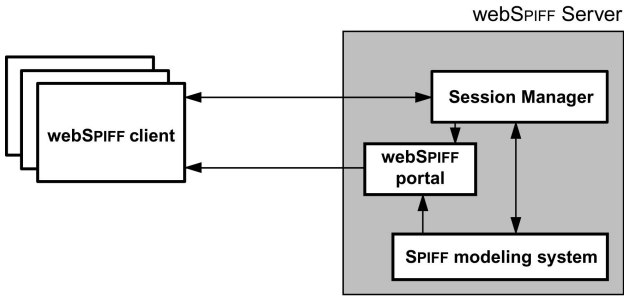

Figure 1 – Architecture of webSPIFF

i.e. models that satisfy all specified requirements, are created by a user [2]. Third, it offers *sophisticated feature model visualization* techniques, which visualize much more specific feature information than most other systems do. For example, feature faces that are not on the boundary of the resulting object, such as closure faces of a through slot, can be visualized too. All these facilities are computationally expensive, and require an advanced product model, including a cellular model with information on all features in all views.

The Session Manager stores information about an ongoing session and its participants. It manages all information streams between webSPIFF clients and the SPIFF modeling system. Since several session participants can send modeling operations and queries to the webSPIFF server at the same time, concurrency must be handled at the Session Manager. It is also the task of the Session Manager to synchronize session participants, by sending them the updated data structures, after a modeling or camera operation has been processed. The Session Manager has been implemented using the Java programming language.

4. The clients

The clients of webSPIFF make use of standard web browsers. When a new client connects to the webSPIFF portal, a Java applet is loaded, implementing a simple user interface, from which a connection with the Session Manager is set up. Different clients can connect from various locations, local through a network or remote via Internet, in order to start or join a modeling session.

Once connected to the server, the user can join an ongoing collaborative session, or start a new one, by specifying the product model he wants to work on. Also, the desired view on the model has to be specified. Information on the feature model of that view is retrieved from the server, and used to build the client's graphical user interface, through which the user can start active participation in the modeling session.

The bottom line is obviously that clients should be able to specify modeling operations in terms of features and their entities; for example, a feature, to be added to a model, should be attachable to entities of features already in the model (e.g. faces and datums). After a feature modeling operation, with all its operands, has been fully specified, the user can confirm the operation. The operation is then sent to the server, where it is checked for validity and scheduled for execution. Notice that this can result in an update of the product model on the server, and thus also of the feature model in the view of each session participant.

In addition to the above functionality, several visualization and interactive facilities of the SPIFF system have also been ported to the clients. webSPIFF clients provide two ways of visualizing the product model, both making use of so-called *camera* windows, i.e. separate windows in which a graphical representation of the product model is shown. First, a *sophisticated feature model image* can be displayed. Second, a *visualization model* can be rendered that supports interactive modification of camera viewing parameters, e.g. rotation and zoom operations. After the desired viewing parameters have been interactively set, they are sent to the server, where a sophisticated feature model image corresponding to the new parameter values is rendered and sent back to the client, where it is displayed. webSPIFF cameras also use a *selection model*, providing facilities for interactive specification of modeling operations, e.g. assisting the user in selecting features or feature entities by having them picked on a sophisticated image of the model. The visualization and interactive functionality of webSPIFF cameras is described in detail in [1].

To support all these facilities, the webSPIFF clients need to locally dispose of some model data. This data is derived by the server from its central model, but it does not make up a real feature model. webSPIFF clients need just enough model information in order to be able to autonomously interact with the feature model, i.e. without requesting feedback from the server. Client model data is never modified directly by the clients themselves. Instead, after a modeling operation has been sent to the server and executed, updated model data is sent back to the client. In addition, if the central feature model has been changed, appropriate updated model data is sent to all other session participants as well. This consists of, possibly several, new model images, a new visualization model, and an incremental update of the selection model (containing only new and/or modified feature canonical shapes).

5. Conclusions

A good distribution of the functionality between the server and the clients has resulted in a well-balanced system. On the one hand, the server provides participants in a collaborative modeling session with the advanced functionality of the original feature modeling system. On the other hand, all desirable interactive modeling functionality is offered by the clients, ranging from display of sophisticated feature model images to interactive model specification facilities.

All functionality described here has been implemented in the webSPIFF prototype system. The Java-based client application is quite simple. So far, webSPIFF clients running on Unix, Windows and Linux platforms have successfully participated in collaborative sessions. webSPIFF has a demo version available on Internet for users to experiment with, at www.webSPIFF.org.

References

[1] van den Berg, E., Bidarra, R. and Bronsvoort, W.F. (2000) Web-based interaction on feature models. Proceedings of the Seventh IFIP WG 5.2 International Workshop on Geometric Modelling: Fundamentals and Applications, Parma, Italy, pp. 113–123

[2] Bidarra, R. and Bronsvoort, W.F. (1999) Validity maintenance of semantic feature models. Proceedings of Solid Modeling '99 – Fifth Symposium on Solid Modeling and Applications, Bronsvoort, W.F. and Anderson, D.C. (Eds.), ACM Press, NY, pp. 85–96

[3] Chan, S., Wong, M. and Ng, V. (1999) Collaborative solid modelling on the WWW. Proceedings of the 1999 ACM Symposium on Applied Computing, San Antonio, CA, pp. 598–602

[4] CoCreate (2000) Shared engineering. *http://www.cocreate.com/onespace/documentation/whitepapers/shared_eng.pdf*

[5] de Kraker, K.J., Dohmen, M. and Bronsvoort, W.F. (1997) Maintaining multiple views in feature modeling. Proceedings of Solid Modeling '97 – Fourth Symposium on Solid Modeling and Applications, Hoffmann, C.M. and Bronsvoort, W.F. (Eds.), ACM Press, NY, pp. 123–130

[6] Lee, J.Y., Kim, H., Han, S.B. and Park, S.B. (1999) Network-centric feature-based modeling. Proceedings of Pacific Graphics '99, Kim, M.-S. and Seidel, H.-P. (Eds.), IEEE Computer Society, CA, pp. 280–289