View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by CiteSeerX

ADJUSTING A TOOL FOR COLLABORATIVE PLANNING TO REQUIREMENTS IN PRACTICE - REALISATION OF A CLIENT-SERVER ARCHITECTURE

Felix Hofmann, Volker Berkhahn

Institute for Computer Science in Civil Engineering,Leibniz University Hannover,Germany

ABSTRACT: The planning of projects in building engineering is a complex process which is characterized by a dynamic composition and many modifications. For a computer-aided and network-based cooperation a formal description of the planning process is necessary. In the research project "Relational Process Modelling in Cooperative Building Planning" a hierarchical process model was defined and divided into three parts: an organisation structure, a building structure and a process structure. Furthermore, we implemented a prototype graph modelling tool in Java to build up the process model dynamically. Our tool includes functions to instantaneously check the structural correctness of the graphs. The usage of critical path and Petri net methods is possible.

In our transfer project "Verification of a Tool for Co-operative Planning in Practice", we currently use a practice building project to test our process model and the prototype implementation. With many engineers working on the process model in collaboration, our implementation needs a client-server architecture to allow distributed work. This architecture comes along with different types of problems: simultaneous work demands a real-time status and thus Client-Callback, for instance through firewalls. The separation of model and view is difficult, and finally concurrent modifications have to be prevented. In this context, problems and solutions are discussed.

KEYWORDS: project management, process modelling, network based collaboration, client-server architecture.

1 INTRODUCTION

In building engineering every state of design, planning, construction and usage is characterized by specific processes. These processes can be organized very efficiently with the support of modern information and communication technology.

Within the research project "Relation Based Process Modelling of Co-operative Building Planning" we have defined a consistent mathematical process model for planning processes and have developed a prototype implementation of a tool to model these processes.

This research project was embedded in the priority program 1103 "Network-based Co-operative Planning Processes in Structural Engineering" supported by the German Research Foundation (DFG). The research work is now continued within the transfer project "Verification of a Tool for Co-operative Planning in Practice" promoted by the DFG for the next 18 months.

Besides the process model, this paper will discuss problems adjusting our prototype implementation to collaboration in practice regarding client-server aspects.

2 MATHEMATICAL PROCESS MODEL

2.1 *Overview*

Our process model is divided into three sub models: a process structure with states and activities for time scheduling, a building structure with structural and spatial building components and an organisation structure with participants and roles for resource planning (see figure 1). Between the sub models various relations exist. For the description we use the algebra of sets and relations as well as the graph theory. Each sub model is represented by a hierarchical bipartite graph.

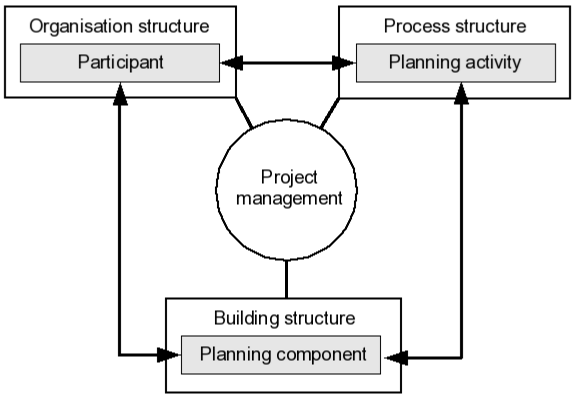

Figure 1. Process model.

2.2 *Process structure*

The process structure covers all planning activities. Activities represent work packages carried out by planning participants within a prescribed time period. They are specified on the basis of tasks for components of the building structure. The directed relationships from one activity to a successive activity are specified by states. Activities and states form the bipartite graph PS which is acyclic and called workflow graph.

$PS:=(A,T;AT)$ with $AT \subseteq A \times T$

A	Set of activities
T	Set of states
AT	Relation between activities and states

2.3 *Building structure*

The building structure covers the planning states of all building elements. Relations between components are defined by connections. Within the context of planning processes, the building structure only contains topological information about the building. All information on dimensions, material and documents are part of the product model. For each component a planning schedule with a set of planning tasks has to be defined. Every task has a certain planning state with references to the corresponding objects or documents of the product model. The planning of a component is finished, if all tasks are carried out. Components and connections form the bipartite graph BS.

$BS:=(C,F;CF)$ with $CF \subseteq C \times F$

C	Set of components
F	Set of connections
CF	Relation between components and connections

2.4 *Organisation structure*

The cooperative planning process requires an organisation structure for planning participants and their different planning roles. Planning participants represent planning actors, planning groups, offices or subcontractors. For every planning participant one or more planning roles can be defined. To carry out certain planning tasks planning appropriate roles are required. Planning participants and planning roles are managed in two sets. These disjunctive sets of roles and participants in combination with the corresponding relation build up the bipartite graph OS.

$OS:=(R,P;PR)$ with $PR \subseteq P \times R$

R	Set of roles
P	Set of participants
PR	Relation between participants and roles

2.5 *Hierarchy*

During the term of planning, most structures are specified in more detail. Therefore, nodes can be refined: they can contain or represent subnodes. Thus, all three structures of our process model are hierarchical.

The recursive refinement of nodes to a level with more detail is called decomposition. The reverse operation is the composition. Figure 2 shows an example of a hierarchical building structure.

The consistent composition of components and connections with their relationships is important for further analysis. The hierarchical bipartite undirected graph is consistent, if an undirected relationship on a higher level is associated with an undirected relationship on a lower level and vice versa.

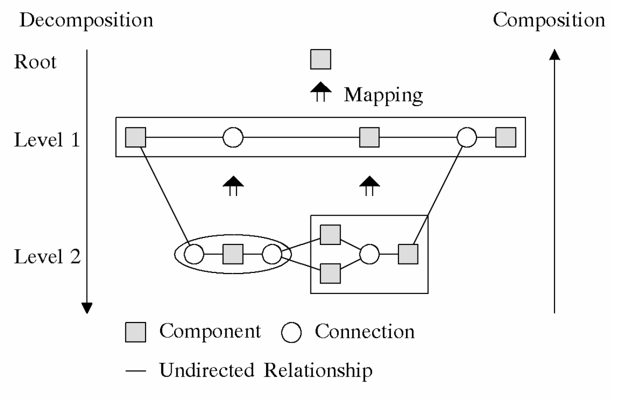

Figure 1. Hierarchy in the building structure.

3 PROTOTYPE PROCESS MODEL EDITOR

To build up the process model dynamically, we developed a graph modelling tool, implemented in Java. The graph structure is represented by bipartite graph classes and trees for the hierarchy. A geometry model provides the layout of the graph.

There are several software products on the market for modelling graph structures but most of them lack of features for testing structural properties. Our tool includes functions to instantaneously check the structural consistency and the structural correctness of the graphs. Structural attributes for instance like deadlocks, synchronisations or the lack of synchronisations are displayed in different colours and described by an index (see figure 3).

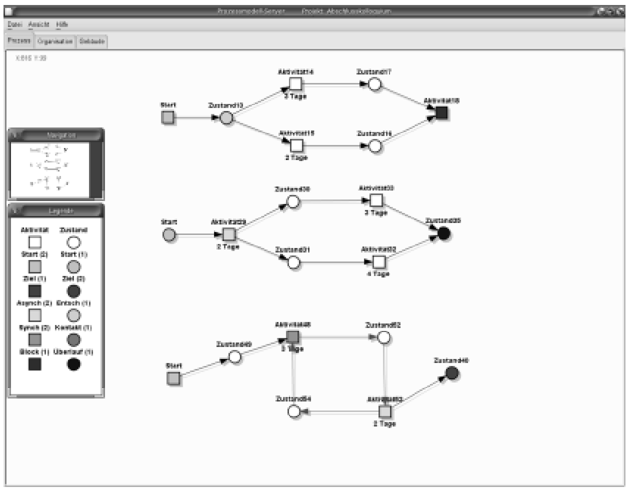

Figure 2. Screenshot of the process model editor.

Furthermore, we implemented critical path and Petri net methods for the hierarchical process structure. With the critical path methods, it is possible to plan and observe the time schedule for the processes. A positive real time value is assigned to every activity, in order to enable the calculation of the critical path.

With the help of the Petri net methods, several tasks can be accomplished. Besides analysing the process structure, it allows an event oriented communication: when a state is reached, certain users are notified by electronic mail. Then, they can for instance withdraw results, make decisions or start their work.

4 PRACTICE BUILDING PROJECT

Currently, we are working on our transfer project to research the use of the developed process model and the implementation in practice. In collaboration with our practice partner Sellhorn Engineering Company we selected a completed building project in Hamburg, Germany.

The planning of this office building, in which 14 engineers were involved, is completely remodelled with our planning tool. The structures of the organisation, of the building itself, and of the planning process are set up dynamically by different actors in a collaborative environment. The technical problems of this collaboration are our main concerns right now: the process model editor has to support distributed and simultaneous working.

5 CLIENT-SERVER ASPECTS IN PROCESS MODELLING

5.1 *Architecture*

In a collaborating environment, where many involved persons work on the same project and thus on the same process model, a client-server architecture is needed. The only process model is kept on the server. Clients can connect to the server an submit their changes.

Our Process Model Editor software uses Java RMI (Remote Method Invocation) for the communication between server and clients. All the transferred objects must be serializable. The client side software is provided via Java Web Start. The server uses an object-oriented database, DB4O, to store the process model. The changes, which a client transmits to the server, are called updates. They are equipped with a GUID (Globally Unique Identifier) and for logging reasons (the succession of changes can be reproduced) not only executed on the process model but also stored in the database.

This configuration works fine as long as only one user changes the process model at a time. But the more people are involved in the project, the more likely it is, that they work on this project at the same time. Therefore, the software has to assure each client a "real-time" status. When one client submits an update, the server has to send this update to all connected clients. Thus, the server must contact the clients (not vice versa), which is called Client-Callback.

Client-Callback often causes problems, since many connections have to pass firewalls and PAT-Routers (Port Address Translation) (see figure 4). Thus most connections are initiated by the client, kept alive, and then used by the server for sending information.

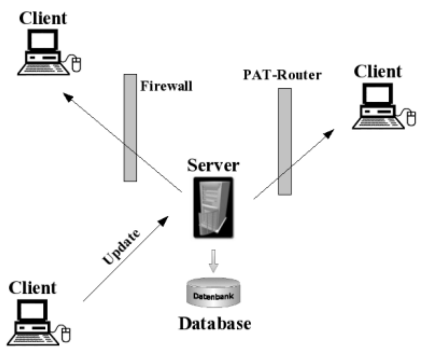

Figure 3. Client-server architecture.

5.2 *Model – view – control*

The separation of client and server causes a chain of problems that have to be solved. First of all, it is difficult to stick with the MVC-Concept (Model-View-Control), to clearly separate the model from the view. The "model" of our implementation is the process model, which is represented by hierarchical bipartite graphs. The view is represented by a geometry model, which mainly consists of the coordinates of the nodes.

In a first approach, every user should have his own view. But furthermore, every user should also be able to log in from any computer around the world. Therefore it is impossible to save the view on the user's computer (client). A solution could be to save one view per user on the server. In a multi-user-environment this can cause huge amounts of data on the server. Furthermore, this solution does not solve the following problem: if one user adds a node, where does it appear in the views of the other users ? Random ?

We chose a different approach: basically the same view for all users, one geometry model. Needless to say, this solution is not problem-free. When a user moves a node, this node also has to be moved in the views of all other users. Every client has to be notified. This is not the case for node refinement. If a user "opens" a node (its view), so that subnodes are visible, this is not part of the geometry model and shall not be transferred to the other users, because other users may work on different parts of the model.

However, in the single-user mode of our editor, there exists a connection between node movement and node refinement: if a node is opened, it changes its size, and nodes in the direct neighbourhood are automatically moved away to avoid overlapping (see figure 5).

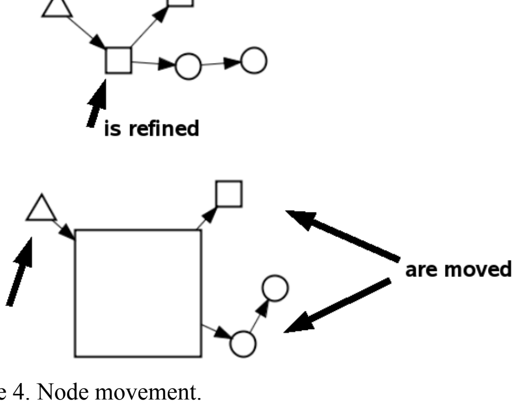

Figure 4. Node movement.

In multi-user mode though, this is impossible because of the eventuality of locked nodes, many refined nodes and their recursion. Thus we now allow the overlapping of nodes.

5.3 *Concurrent modification*

Last but not least we have to mention the problem of concurrent modification: several users want to change the attributes of the same node at the same time. This has to be prevented in order to avoid inconsistencies. For this purpose we use object locking. When one user edits a node, no other user can change the attributes of this node. This also applies to the location of the node. And since in a hierarchical system the attributes of nodes are often dependent on the attributes of sub- or supernodes, a user is not allowed to change or move a node when another user is editing a sub- or supernode.

6 CONCLUSION

In this paper we introduced a process model for planning processes in building engineering and our prototype implementation of a graph editor to build up the process model. In order to test the process model editor in our practice building project with many collaborating engineers, we had to deal with client-server aspects. In this context, the paper's focus was put on the separation of model and view. We chose an approach, which uses the same view for all users. Because of occurring problems though, we eventually will have to reconsider this decision. In further development of our tool, we will find out whether to prefer an individual zoom function or saving one view per user.

ACKNOWLEDGEMENT

After a successful completion of the research project “Relational Process Modelling in Co-operative Building Planning” embedded in the German Research Foundation (DFG) priority program 1103 “Network-based Co-operative Planning Processes in Structural Engineering”, the authors thank gratefully the DFG for supporting the transfer project “Verification of a Tool for Co-operative Planning in Practice”.

REFERENCES

Klinger, Berkhahn, König (2006). Formal Treatment of Additions in Planning Processes, ICCCBE, Montreal 2006.

König, M. (2004). Ein Prozessmodell für die kooperative Gebäudeplanung, Dissertation, University of Hannover, Germany.

König et al. (2004). Process Modelling in Building Engineering, 5th European Conference on Product and Process Modelling in the Building and Construction Industry (ECPPM), Istanbul, Turkey.

König, M. and Klinger, A. (2002). Modellierung von Planungsprozessen mit Hilfe von Hierarchischen Strukturen, 14. Forum Bauinformatik in Bochum, Fortschrittsberichte Reihe 4, Nr. 181, VDI-Verlag Düsseldorf, Germany.

Stilhammer, Klinger, Berkhahn (2006). Navigation within Hierarchical Graph Systems for Process Modelling, ECPPM, Valencia 2006.

Van der Aalst, W. M. P., Hirnschall, A & Verbeek, H. M. W. (2002). An Alternative Way to Analyze Workflow Graphs, Proceedings of the 14th International Conference on Advanced Information Systems Engineering (CAiSE´02), volume 2348 of Lecture Notes in Computer Science, pages 535-552, Springer-Verlag, Berlin, Germany.