

Net-distributed applications in civil engineering: Approach and transition concept for CAD systems

Most software applications in civil engineering that are commercially successful were originally conceived as single-user systems. Fundamentally, the software was never designed to be used in a highly cooperative engineering environment. This is specifically true for most CAD applications in civil engineering. CAD systems have been advanced from the use of “simple”, attributed geometry to engineering objects that are semantically rich. The requirements of engineering cooperation, however, are quite often still based upon the traditional paper-based process definitions of engineering projects.

Functionality that is available for a net-distributed cooperation of engineering teams was quite often built into existing systems as an enhanced feature on top of existing solutions that were never designed for such a purpose. Such solutions can at best be regarded as network-enabled solutions. Enabling existing solutions will, of course, constitute an improvement for the work of the engineers but will be restricted within severe conceptual limitations. Starting from scratch, however, with a fundamentally new approach will be very hard with regard to a successful acceptance by the users because the required breadth of functionality will be difficult to achieve.

Research on process definitions for civil engineering projects lead to the approach of a versioning system for engineering objects as a basis for distributed cooperation. Cooperation in civil engineering projects is typically characterized by engineering teams working independently in parallel and by long transaction times in these teams. Independent solutions are then coordinated as a joint solution at predefined milestones. This fact resulted in a distinction between a *workspace* and a *project* concept with engineering objects that are never discarded but rather enhanced with new versions. Therefore, relations to existing objects will never become inconsistent in the context of the *project* but rather will only refer to “older” versions of objects.

Existing engineering applications may be enhanced to be usable in a *project* / *workspace* environment. Necessary functionality of the *workspace* can be divided into application dependent and independent functionality. This leads to different application specific classes and one generic implementation. The generic part is predominant. It has the following tasks: Communication with the common project data at load and store time (long transaction) via the internet, performing operations on the transient part of the common planning material (*workspace*) using a set algebra and transforming the transient object model of the application into a generic schema that is accessible by the set algebra. The interaction with the user is application dependent. A concept to enhance the graphical user interface providing the new functionality is required. The common *project* implements a generic access to different data stores for the persistent model, the required operations on the persistent model, a flexible and problem specific query language for selection of subsets of the planning material and the communication with several clients (server functionality).

Many existing engineering applications are based upon object-oriented models and provide a programming interface. Furthermore, they support event handling mechanisms and object access required for recording modifications to the model when implementing long transactions. Thus, they are basically prepared to be enhanced with *workspace* functionality and user interaction. The *project* system is a new implementation with the functionality outlined above. The basic concepts are demonstrated with a prototypical CAD system.

There are still some open questions for further research. A mapping mechanism for internal data structures of applications is needed to allow for different applications using objects of the same persistent class and to support changes to interfaces of class components in different application releases. Furthermore, the concept has to ensure access to persistent objects of obsolete classes. So far, the set algebra implementation used is not based upon transactions. Therefore, every operation operates directly on the datastore. This may be changed to improve performance. The theoretical concept and its pilot implementation will be tested in cooperation with practical partners on real-world scenarios in order to ensure practical usability of the approach chosen.