In Proceedings of Vl'96, Boulder, Collarado, USA, IEEE CS Press.

Visual Language Support for Planning and Coordination in Cooperative Work Systems

John C. Grundy[†] and John G. Hosking[††]

[†]Department of Computer Science
University of Waikato
Private Bag 3105, Hamilton, New Zealand
email: jgrundy@cs.waikato.ac.nz

[††]Department of Computer Science
University of Auckland
Private Bag 92019, Auckland, New Zealand
email: john@cs.auckland.ac.nz

Abstract

This paper describes three new visual languages for use in the coordination of collaborative work in large CSCW environments. EVPL describes work plans and is used to define the context of work, VQL is a query language, and VEPL is an event processing language.

1. Introduction

Large CSCW systems, such as Software Engineering Environments (SEEs) and cooperative work via the Internet, require abstract, visual languages to support collaborative planning of work processes and activities [3,5]. The inadequacies of most workflow and process modelling approaches motivated Swenson to develop VPL (Visual Planning Language) [5] for collaborative work planning. VPL defines plans and subplans for work tasks, and useful plans can be abstracted into reusable policies.

In our current work, we have extended VPL and the SEE environment of [3] to allow the description of work artefacts, tools and developers. Extended Visual Planning Language (EVPL) elements include: process stages, denoted by elipses with a role and stage description; split stages, duplicated for each person involved; start and stop stages; and named enactment event flows between stages. Rectangles with "text" describe work artefacts; "faces" represent individuals and groups (roles); and a tool icon represents artefact manipulation tools. Figure 1 shows three views of a software process model: "m1:model - process' is the basic model; "m1.1:design changes-subprocess an exploded view of stage m1.1; and "m1:model1 - roles" shows additional artefact, tool and role usage by stages. Users enact models, resulting in highlighting of enacted stages. The stage with the bold border is the "current enacted stage", which specifies the part of the process the user is currently working on, thus giving a "work context" for tools. Work coordination is facilitated by capturing the context in which changes are made to work artefacts (from active stages), and presenting this to collaborating users.

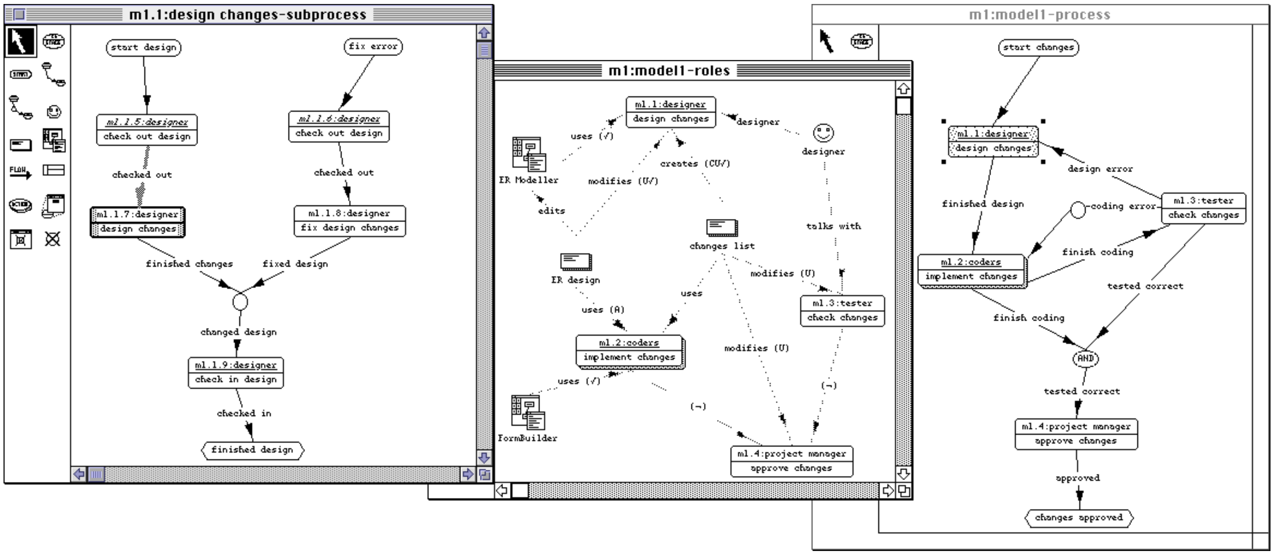

Figure 1. EVPL views for a simple software process model.

2. Visual Event Processing Language

To allow developers to specify automatic processing of EVPL events and how they are informed of collaborators' work, we have developed a Visual Event Processing Language (VEPL). This provides filters (boxes) and actions (shaddowed ovals) which specify what should happen when events, such as stage completion or artefact modification, relating to EVPL tools, artefacts, plan stages, etc occur. An example is shown in Figure 2. This specifies that if the "m1.3:check changes" stage is made the user's current enacted stage or testing completes, the people filling the coders role for m1.2 will be notified (by sending a message, iconic shading or writing a message on their screen). Filters and actions can also process artefact events, process multiple events and process events hierarchically for all substages of a stage.

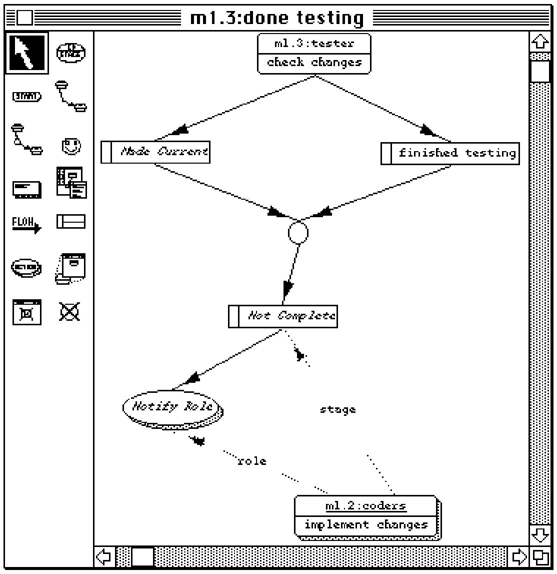

Figure 2. Processing enactment events with VEPL.

3. Visual Query Language

While EVPL is useful for defining work processes and coordinating work, and VEPL for context awareness and arbitrary event handling, extra facilities are also required. It is useful to be able to query SEE repositories to enable developers to locate required information. Common CASE tools and SEEs lack sufficiently flexible, understandable, and accessible query languages [4]. Liu proposed a two-view visual query language for MetaEdit [4], but this uses a limited and cumbersome notation to specify query constraints. We have developed VQL which provides a more concise, one-view visual query language. An example VQL query is shown in Figure 3. Shaded boxes are entities selected in the query output, round boxes are relationships, list box annotations are projected attributes, and optional constraint expressions are beneath entities and relationships. Queries may also include EVPL elements. We are currently extending VEPL to make use of VQL results for complex rule specification.

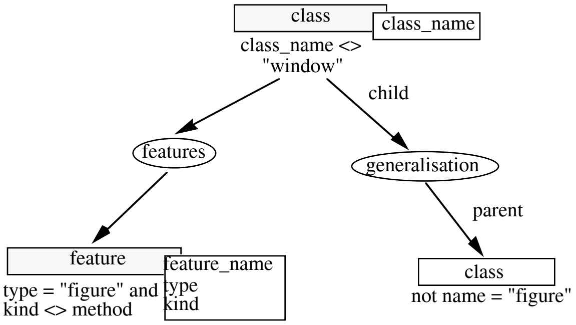

Figure 3. A VQL query example.

4. Architecture and Implementation

We have implemented a tool which supports EVPL and VEPL using MViews [1]. This has been integrated with SPE [2] to aid collaborative software development. MViews environments generate *change description* objects which describe events occuring on work artefacts and in tools. We intercept these in a non-intrusive fashion and route them to the "current stage" process artefact which attaches work context information. VQL queries will use a graphical representation of MViews data and thus be applied directly to SEE data to obtain query results. VEPL receives MViews change descriptions (events) from artefacts and stages, which are sent to VEPL filters, with actions invoked by events sent from filters.

References

[1] Grundy, J.C. and Hosking, J.G., "A framework for building visusal programming environments," in *Proceedings of the 1993 IEEE Symposium on Visual Languages,* pp. 220-224.

[2] Grundy, J.C., Hosking, J.G., Fenwick, S., and Mugridge, W.B., Connecting the pieces, Chapter 11 in *Visual Object-Oriented Programming.* Manning/Prentice-Hall, 1995.

[3] Grundy, J.C., Mugridge, W.B., Hosking, J.G., and Apperley, M.D., "Coordinating, capturing and presenting work contexts in CSCW systems," in *Proceedings of OZCHI'95,* Wollongong, Australia, Nov 28-30 1995, pp. 146-151.

[4] Liu, H., "A Visual Interface for Querying a CASE Repository," in *Proceedings of the 1995 IEEE Symposium on Visual Languages,* pp. 21-28.

[5] Swenson, K.D., "A Visual Language to Describe Collaborative Work," in *Proceedings of the 1993 IEEE Symposium on Visual Languages,* pp. 298-303.