OZCHI'96 Workshop on the Next Generation of CSCW Systems, Hamilton, New Zealand,
November 25, 1996, pp. 40-44.

Approaches for Utilising 'Work Contexts' in CSCW Systems

John C. Grundy

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

John G. Hosking

Department of Computer Science
University of Auckland
Private Bag, Auckland
New Zealand
john@cs.auckland.ac.nz

Abstract

When working together using CSCW tools, people have a "work context" within which they perform their work. A work context includes the tools they are using to perform the work, the work artefacts being modified, the tools used to communicate with collaborators, and the other people they are collaborating with.We describe our recent work with trying to represent work context information in a workflow system. We then describe our current work which tries to determine work context information, rather than requiring it to be explicitly specified, by monitoring user activities. We describe how both approaches allow work context information to be communicated between collaborating users, allowing users to better coordinate their work and understand why others have done certain things or are doing certain things. We claim such facilities are necessary in order to enable the next generation of workflow-based CSCW systems to better support work coordination amoung multiple users.

1. Introduction

When people work collaboratively together, they either formally or informally have a notion of the context of their work, which we refer to as a "work context". This includes the artefacts they use or modify to perform work, the tools used to view or modify these artefacts, the tools and artefacts used to articulate work to others (i.e. describe what a user is doing and why), and the other people they are working with. Work contexts may be fixed i.e. only certain artefacts (or parts of artefacts) and tools are used or modified and only certain communication techniques are used to discuss work with particular collaborators. Examples include specific phases of a software process (e.g. modify this particular code fragment using this particular tool/methodology). At other times a person's work context may be very volatile, with a variety of artefacts, tools and other users being utilised. Examples include aspects of office automation systems, where a person or persons may (or may not) use a word processor, spreadsheet, Information System or pen and paper, and their associated artefacts, to carry out some business process (e.g. processing invoices).

The notion of a "work context" is implicit in many CSCW systems, while made more explicit in others. Many CSCW tools have implicit work contexts depending on what problem domain they are used in. For example, email assumes data to transmit and receivers of this data, collaborative Web browsers assume people use any WWW pages and discuss these [5], and text chats, collaborative editors and whiteboards may be used to discuss or edit specific artefacts, thus their work context depends on what is being discussed or modified at a particular times and the participants of the discussion. Many such tools allow participants to come and go, and allow the focus of discussion or work to change while the tool is in use.

One example of explicit definition or representation of work contexts is the work of Schmidt and Bannon on the articulation of work [11], and their idea of dividing cooperative work into doing the work itself and articulating (or describing) the work done. Most workflow systems provide some manner of codifying the context of work for each process in the workflow. This is either just the roles involved, as in Regatta [12] and TeamFLOW [13], or more explicit artefact, tool and role definitions as in Action Workflow [10] and in process-centred environments, such as SPADE [1] and Oz [2].

Some systems, such as wOrlds [3] and Orbit [9], allow work contexts to be more flexibly specified and represented, with these contexts implicitly defined by the artefacts, tools, actions and collaborators involved, rather than explicitly codified. wOrlds and Orbit make use of the notion of a "locale" [4], where people perform both informal and formal aspects of work, with locales being very dynamic in their artefact, tool and participant make-up. Key problems with the use of work contexts in CSCW systems identified in the wOrlds/Orbit work

include lack of flexibility in defining a context, forcing people to be in only one context at a time, and lack of providing context through history and trajectory (what was done/is to be done next) [9]. Most existing workflow and process-centred environments suffer from such problems.

This paper describes the use of the work context notion in Serendipity, a flexible process modelling/workflow/CSCW system we have been developing. In Serendipity's initial design and implementation [6, 7, 8], work contexts were explicitly codified using a workflow-style notation. This has the disadvantages of inflexibility and focus on formal context specification as outlined above. Here we describe "agents" which can more informally "determine" work contexts as people use artefacts, tools and communicate with others. Mechanisms to help support multiple work context awareness for users, and to allow users to better visualise interrelationships between the work contexts associated with different aspects of their cooperative work are also discussed. We claim that adding such capabilities to workflow and process-centred environments will better support work coordination in such systems.

2. Serendipity Overview

Serendipity is a process modelling, enactment and work planning environment, which also supports event handling, group communication, and group awareness facilities [6]. Serendipity's notations are designed to be high-level and graphical in nature, and its coordination and rule mechanisms are easily extended by users. The notation is based on Swenson's Visual Planning Language [12] but extends it to support artefact, tool and role modelling, and arbitrary event handling.

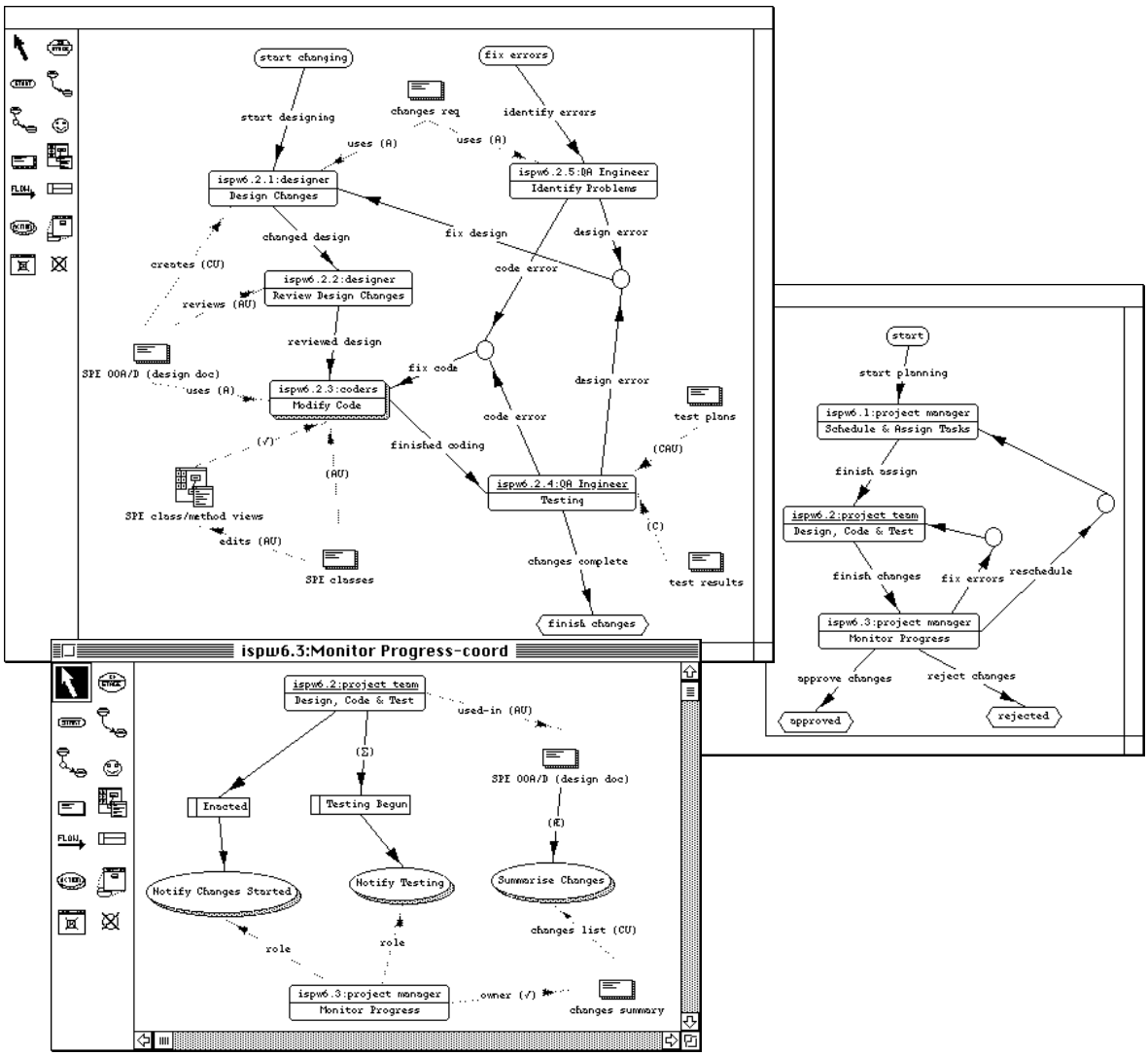

Figure 1. Part of the ISPW6 software process example modelled in Serendipity.

The left and bottom windows shown in Figure 1 are Serendipity views modelling part of the ISPW6 software process example. Stages describe steps in the process of modifying an arbitrary software system, with each stage containing a *unique id*, the *role* which will carry out the stage, and the *name* of the stage. Enactment *event flows* link stages. If labelled, the label is the *finishing state* of the stage the flow is from (e.g. "changed design"). There are a number of specialised types of stage including *start*, *finish*, *AND*, and *OR* stages (empty round circle).

Modularity is provided in the form of hierarchical subprocess models. The window at the left of Figure. 1 is a subprocess model refining the "ispw6.2:Design, Code & Test" stage of the process in the bottom window. Underlined stage IDs/roles mark the presence of a *subprocess* model. The shadowing of the "ispw6.2.3:Modify Code" stage indicates that multiple implementers can work on this stage (i.e. the stage has multiple subprocess enactments).

Serendipity supports artefact, tool and role modelling for processes. *Usage connections* show how stages, artefacts (eg the change requirements document "changes req"), tools (such as SPE's OOA/D class/method views) and roles (such as the project manager) are used. Optional annotations indicate: data is created (C), accessed (A), updated (U), or deleted (D); whether a stage must use only the tools, artefacts or roles defined (√); and whether a stage cannot use a particular tool, artefact or role (¬).

In addition to specifying the static usages and enactment event flows between process model stages, Serendipity supports *filters* (rectangular icons) and *actions* (ovals), which process arbitrary enactment and work artefact modification events.

For example, the coordination of the process model via the "ispw6.3:Monitor Progress" stage is defined by the top-right window of Figure. 1. This uses two filters and three actions to carry out the coordination. The Enacted filter selects only stage enactment events, in this case when the ispw6.2 stage is enacted.

This triggers the "Notify Changes Started action", which notifies its associated role (in this case, the project manager) that changes have commenced. The other filter acts similarly to notify commencement of testing. The other action takes artefact modification events from the OOA/D design document and accumulates them into a changes summary. Serendipity models may be used to guide work or to enforce particular work processes (by defining rules with filters and actions).

3. Defining and Determining Work Contexts

As can be seen from the left-hand side process model in Figure 1, Serendipity allows users to specify and hence codify work context information (artefacts, tools and collaborators) for processes. Users have the ability to restrict the access of a process to particular artefacts, tools and other users. They can also specify more loose work contexts ("this process may make use of so and so..."). This approach works well when codifying work contexts for use with formal process models. For example, we have found this useful for software process modelling and enactment and the coordination of work in such projects.

The approach doesn't work so well for the more volatile "processes" (work contexts) in such systems, nor in domains where such volatile work contexts are the norm (higher-level system design, office automation, etc). In these situatiuons, people cooperating do not usually wish to formally specify exactly what artefacts, tools, collaborators and communication mechanisms they want to use [11, 9]. Most other workflow and PCEs suffer from the same deficiency in this regard as Serendipity - in fact most do not even support the level of work context and work coordinatioon capabilities as outlined above [8].

Serendipity's filter/action visual language allows users to specify a variety of mechanisms for handling notification of changes and "automatic" processes. Currently this has been used in a similar style to the example in Figure 1: to help coordinate work or automate some parts of work processes. However, this language has the potential to support informal determination of work context information.

For example, Figure 2 shows a filter/action to determine the artefacts, tools and collaborators someone is working with for a particular process stage. Whenever the user accesses or modifies an artefact, a tool they are using generates an event, or they communicate with another user with CSCW tools, this information is cached in a simple database. Other filter/actions can be then be defined to extract this information and make use of it.

This may be to formalise the work context/process model the person has been using, or to visualise the dynamic aspects of their work context. This filter/action can be applied to all possible process stages by applying the filter/action to a metaprocess model or to the top-level process stage for a system. More complex filter/actions can be defined and utilised which recognise specific artefacts/tools/people of interest, collate events before storing them, etc.

Using such an approach means Serendipity can dynamically determine the contexts of peoples' work by monitoring their process stages and seeing which tools, artefacts and other people they use or interact with. By storing this information, it can be made available to users or to other filter/actions. The lack of such a capability in existing workflow/PCEs means they can not adapt to situations which require dynamic work contexts which can not be explicitly codified (or that users may not want to explicitly codify) in their workflow models.

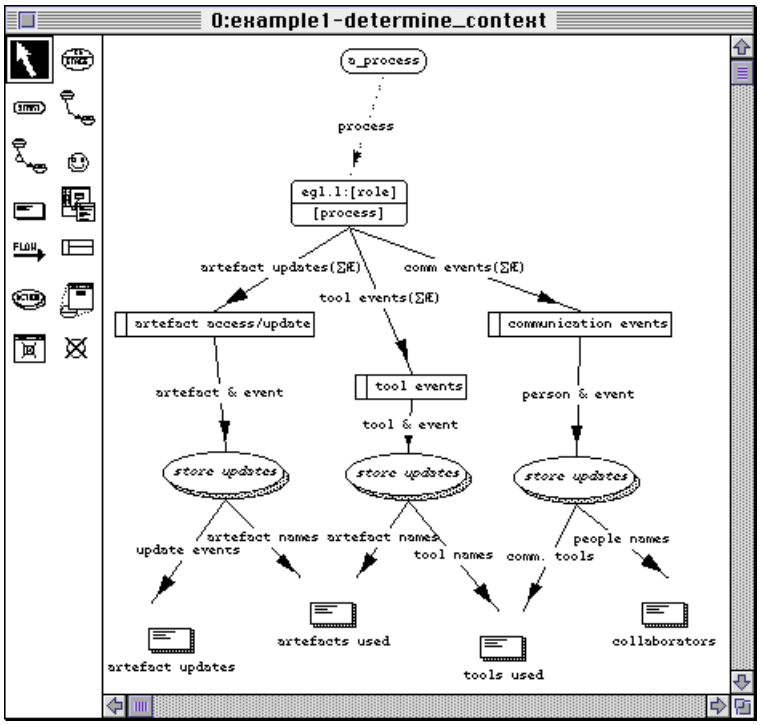

Figure 2. Determination of a work context.

4. Visualising Work Contexts

If we utilise work context determination techniques as outlined above, we also need mechanisms to access this information, visualise it, and use it to make others more aware of their collaborators' work and the reasons for it. Most workflow/PCEs which support multiple users rely on explicitly specified work context information to keep users aware of each other's work (if they even provide any work context awareness information at all).

Serendipity currently annotates events generated by tools integrated with Serendipity (i.e. tools for performing work) with the currently enacted process stage to indicate the work context [6, 7, 8]. This allows people to see not only the changes others have made to artefacts, but the "reason" these updates have been made (i.e. the work context they were made in). Serendipity also highlights the current enacted process stage of cooperating people, allowing a user to see just what aspect of a system other users are working on at the current time. Users can also define filter/actions which inform them of others' work (when they access/update artefacts, use tools etc.) in various ways (open a dialog box , highlight artefacts, send a message, etc.). Figure 3 (a) shows a filter/action to highlight the artefacts being being modified on a process model diagram when an artefact update event is received.

While we have found this approach to work context visualisation useful, it relies on collaborating users to explicitly enact and advance process model stages to keep others informed of the context of their work. Requiring people to explicitly enact or complete stages can interfere with the informal aspscts of cooperative work. In addition, the approach does not, currently, take into account that the artefacts, tools and collaborators for

someones work context may be dynamica, and thus need to be determined (as outlined in the previous section) rather than explicitly specified in process model views or in filter/action models.

We can make use of the "determined" work contexts discussed in the previous section by providing ways to visualise the stored event data, and to utilise this data to e.g. highlight any effected artefacts in tools being collaboratively used. For example, Figure 3 (b) shows a filter/action which highlights class icons in the SPE software development environment when another user modifies part of that class. We are working on other filter/actions which help keep collaborators aware of others' work using awareness widgets in tools which are updated by utilising stored, determined work context information.

Providing work context awareness facilities which utilise dynamically determined work context information allows Serendipity to be used in situations where such dynamic work contexts are utilised, while still providing similar context awareness capabilities to when work contexts are explicitly codified. Such an approach is not supported in most current workflow systems and PCEs.

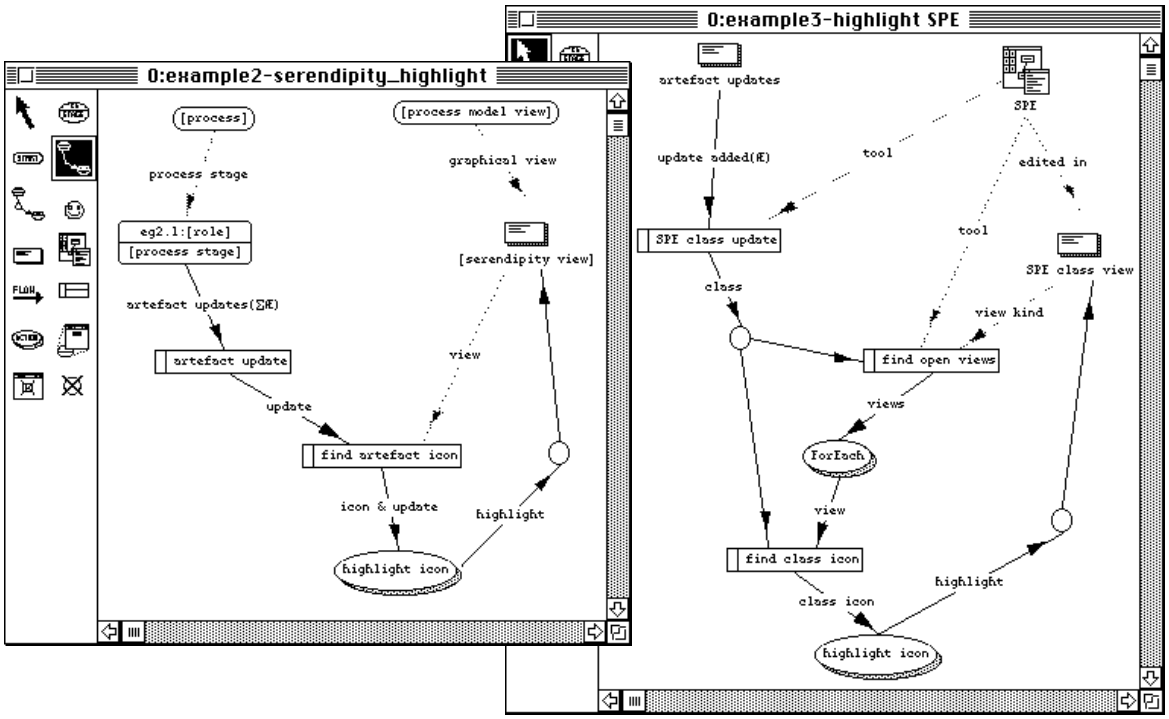

Figure 3. (a) Highlighting modified Serendipity artefacts. (b) Highlighting affected artefacts in tools.

5. Changing Work Contexts

Currently Serendipity assumes that either users explicitly enact/advance workflow/process model stages, stages are enacted by enactment events flowing into them from stages they are conencted to, or actions send process stages enactment events in response to some other filtered event(s) which have occured.

If work contexts are specified in process models, we can utilise determined work context information to "automatically" change the work context for a user. This can be achieved by a filter/action determining someone is now using a tool or artefact or communicating with another user, not specified for their current enacted process model stage. The filter/action can attempt to determine an appropriate alternative process stage and then enact that stage. Even if work contexts are not fully specified, if events caused by user actions utilise a tool, artefact or person not before seen when determining work context information, this could be recognised by filter/actions as some form of potential work context switch.

Alternatively, Serendipity could record all events that occur while the user is in a "non-determined" stage, and when the user subsequently enacts a stage or a suitable stage is determined, these events can be processed for the newly determined work context. We envisage that this "optimistic" work context determination could also be

used to predict future actions of the user, and hence make others aware of this, by using the history of actions and work contexts which have been stored.

Once again, the lack of such capabilities in current workflow and PCEs means users are constrained to using codified process models and work contexts. Thus the formal aspects of work are well-supported, but where processes and/or work contexts are more dynamic, these CSCW systems do not adequately support work in these domains. Providing filter/actions which allow people to work in "non-determined" stages but retain a history of work allows process models to be specified incrementally as users develop models while carrying out their work. Automatic context-switching allows users to do their work while Serendipity determines their work context from their historical work. The histories of work allow future events to be anticipated based on previous actions taken during a work context, or previous actions in relation to the artefacts, tools and people a person is currently working with.

6. Summary

We have discussed the current use of work contexts in the Serendipity workflow/process modelling environment. We are trying to utilise Serendipity's built-in event processing language to allow more flexible determination, visualisation and utilisation of work context information. This, we believe, will better support users and application domains where formal codification of work processes and work contexts is not appropriate, or where this codification unduly interferes with cooperative work. Adding such capabilities to the next generation of workflow and process-centred environments will, we believe, enhance the usefulness of these tools in such situations.

References

[1] Bandinelli, S., Fuggetta, A., Ghezzi, C., and Lavazza, L., *SPADE: an environment for software process analysis, design and enactment*. Finkelstein, A., Kramer, J. and Nuseibeh, B. Eds, J. Wiley, 1994.

[2] Ben-Shaul, I.Z. and Kaiser, G.E., "A Paradigm for Decentralized Process Modeling and its Realization in the Oz Environment," in *16th International Conference on Software Engineering,* May 1994, pp. 179-188.

[3] Bogia, D.P. and Kaplan, S.M., "Flexibility and Control for Dynamic Workflows in the wOrlds Environment," Procs *of the Conference on Organisational Computing Systems,* ACM Press, Milpitas, CA, November 1995.

[4] Fitzpatrick, G., Tolone, W.J., and Kaplan, S.M., "Work, Locales and Distributed Social Worlds," in *4th European Conference on Computer-Supported Cooperative Work,* Kluwer Academic Publishers, Stockholm, Sweden, 1995.

[5] Gianoutsos, S. and Grundy, J., "Collaborative work with the World Wide Web: adding CSCW support to a Web browser," Procs *of Oz-CSCW'96,* DSTC Technical Workshop Series, University of Queensland, Brisbane, Australia, August 1996, pp. 14-21.

[6] Grundy, J.C., Hosking, J.G., and Mugridge, W.B., "Low-level and high-level CSCW in the Serendipity process modelling environment," *Procs of OZCHI'96,* IEEE CS Press, Hamilton, New Zealand, Nov, 1996.

[7] Grundy, J.C., Venable, J.R., Hosking, J.G., and Mugridge, W.B., "Coordinating collaborative work in an integrated Information Systems engineering environment," *Procs of the 7th Workshop on the Next Generation of CASE tools,* Crete, 20-21 May 1996.

[8] Grundy, J.C. and Hosking, J.G. "Serendipity: integrated environment support for process modelling, enactment and improvement," Working Paper, Department of Computer Science, University of Waikato, 1996.

[9] Kaplan, S.M., Fitzpatrick, G., Mansfield, T., and Tolone, W.J., "Shooting into Orbit," Procs *of Oz-CSCW'96,* DSTC Technical Workshop Series, University of Queensland, Brisbane, Australia, August 1996, pp. 38-48.

[10] Medina-Mora, R., Winograd, T., Flores, R., and F., F., "The Action Workflow Approach to Workflow Management Technology," Procs *of CSCW'92,* ACM Press, 1992, pp. 281-288.

[11] Schmidt, K. and Bannon, L., "Taking CSCW seriously: Supporting Articulation Work," *Computer Supported Cooperative Work: An International Journal*, vol. 1, no. 1-2, Kluwer Academic Publishers, 7-40, 1992.

[12] Swenson, K.D., Maxwell, R.J., Matsumoto, T., Saghari, B., and Irwin, K., "A Business Process Environment Supporting Collaborative Planning," *Journal of Collaborative Computing*, vol. 1, no. 1.

[13] TeamWARE, Inc., *TeamWARE Flow*, 1996. (http://www.teamware.us.com/products/flow/)