Association for Information Systems
AIS Electronic Library (AISeL)

AMCIS 1997 Proceedings

Americas Conference on Information Systems (AMCIS)

8-15-1997

A Temporal Expert System for Engineering Design Change Workflow

Susan J. Chinn
Penn State Erie, The Behrend College

Gregory R. Madey
Kent State University

A Temporal Expert System for Engineering Design Change Workflow

Susan J. Chinn
Penn State Erie, The Behrend College
and
Gregory R. Madey
Kent State University

Abstract

Workflow management, which is concerned with the coordination and control of business processes using information technology, has grown from its origins in document routing to include the automation of process logic in business process engineering. Workflow also has a strong temporal aspect. Activity sequencing, deadlines, routing conditions, and scheduling all involve the element of time. Temporal expert systems, which use knowledge-based constructs to represent and reason about time, can be used to enhance the capabilities of workflow software. This paper presents a temporal expert system workflow component for tracking engineering design changes. We use Allen's theory of temporal intervals in our model to enhance the decision-making, timing, and routing activities in a workflow application. We test the model using information from a "real-world" engineering design situation and suggest further research opportunities.

Introduction

Workflow management, which is concerned with the coordination and control of business processes using information technology, has grown from its origins in document routing to include the automation of process logic in business process engineering. Workflow also has a strong temporal aspect. Activity sequencing, deadlines, routing conditions, and scheduling all involve the element of time. Expert systems, acting as intelligent agents, play a role in automating workflow by providing reasoning capabilities for knowledge-based work processes (Kalakota & Whinston, 1996). Temporal expert systems, which use knowledge-based constructs to represent and reason about time, can be used to enhance the capabilities of workflow software. By applying temporal representation and reasoning to workflow management systems, we can introduce a way to specify temporal constructs and to standardize the representation of temporal constraints. A key benefit to incorporating workflow in business process engineering is the reduction of cycle time. By reducing cycle time, a company can reduce its losses from scrap and rework, and can increase its ability to respond to new sales opportunities.

Complex coordination projects in engineering management involve volatile requirements, task uncertainty, and many interdependencies. Engineering change processes are representative of the complexity of workflow in engineering. Proposals for design changes in particular may be reviewed by many design teams, manufacturers, and financial planners, as well as external suppliers. An area suggested for research has been how to model these complex scenarios which require synchronization (Johns, 1996). The research presented in this paper demonstrates how a temporal expert system can be used as an intelligent agent in workflow systems for planning work sequences and for monitoring work status.

Workflow Problem

Our workflow problem focuses on the temporal constraints that occur in the design change review process. A design change review starts with a request from a customer or from a manufacturing plant. For example, a plant might request that a tool be changed. The request is reviewed by a work group of design engineers. The review process can last for weeks from the time the request is received until the design work group approves it and releases the changes to manufacturing for production. The design review coordinator may have to plan the review schedule to meet a client's timetable, or to ensure that the changes can be made in the plant during a preferred time (e.g., during a scheduled shutdown). Some factors that can slow the review process include idle time while a design folder is in transit; re-routing work if an engineer is not available; the time the coordinator spends receiving the request, manually assigning it a review number and releasing it for review; off-site reviewing; and meetings (Curtis, 1994).

The Coordinator in Design Engineering receives a Request for Engineering Action from the plant for the change to the turning tool. The request includes a description of the tool, reasons for the change, and suggested recommendations, such as coating the tool with a specific substance. The Coordinator evaluates the feasibility of the request and may respond by 1) routing the request to a team of experts as a formal assignment for further design review; 2) rejecting the request as not feasible; or 3) bypassing the review team and sending the change request to an Engineering Specialist in charge of making the changes. If the request is routed, the Coordinator will select a team with the appropriate skills in such areas as raw materials, heat treatment, grinding, and assembly, and which can meet the plant's timetable. Once a team is chosen, the Team Leader delegates the request to experts on the team. Team Members review the design changes and make their recommendations to the Coordinator. Based on their review, the Coordinator makes a recommendation either to implement the change or to reject it. If the Coordinator approves the change, the work, including revised drawings and the tool order request, is routed to the Engineering Specialist.

In order to plan the workflow, the Coordinator must know the priority of the request, deadlines for the request, and any projected time "windows" in which the changes should be evaluated and subsequently implemented. The Coordinator must also know or prescribe times that each step of the process should take, and then should select design teams on that basis. The Coordinator must be alert to signs that the process is falling behind schedule. The Team Leader, who actually assigns specific engineers to the review, must ensure that those engineers will be able to complete the review within the deadline for the team as a whole. Team Members may interact with each other during the process. Finally, the Coordinator must route the changes to the Engineering Specialist in time for them to be implemented.

Solution and Implementation Using CLIPS

To handle the workflow process and its temporal aspects, we built a temporal expert system. Our system represents time as slots in a frame-like structure, and uses rules to generate Allen's temporal relations (Allen, 1983). These relations model the temporal constraints imposed by the schedules of the different engineers and the interactions among members of the review team. Other rules use specific time points as criteria in monitoring and planning the workflow. We chose CLIPS (C Language Integrated Production System) Version 6.0 as our expert system development tool. CLIPS is a shell that was first developed in 1984 by the Artificial Intelligence Section of NASA's Johnson Space Center and is derived from the OPS5 family of languages (*CLIPS Version 6.0 Basic Programming Guide*, 1993).

We use deftemplate slots to hold attributes about the entities *engineer* and *folder* as shown in Figure 1. To represent temporal attributes, we use slots for the planned starting and ending times that each engineer is expected to work. A duration slot holds a derived value of the starting time subtracted from the ending time, which is used to calculate the total duration of the review. An engineer "starts" work when the design folder is passed to that individual. In our implementation, all starting and ending times are known to simplify the model.

Figure 1. Deftemplates used to hold information about design folder and engineers.

```
(deftemplate folder          ; design change object      (deftemplate eng        ; engineer
      (slot name)            ; folder name                     (slot folder_name) ; name of folder
      (slot due date)        ; date changes due                (slot beg_time)    ; start time
      (slot changes_done)    ; when team finishes              (slot end_time)    ; end time
      (slot which_eng))      ; who has folder                  (slot duration)    ; planned duration
                                                               (slot dec1)        ; initial feasibility
                                                               (slot dec2)        ; make changes
```

Figure 2. Sample deffunction and defrule that establishes *during* relation and defrule using the *during* relation in the workflow application.

```
(deffunction during                          (defrule test-during
        (?t1 ?t2 ?t3 ?t4)                            (eng (name ?name1)(beg_time ?time1)
        (and (> ?t1 ?t2)                             (end_time ?time2))
         (< ?t3 ?t4)))                               (eng (name ?name2&~?name1)(beg_time ?time3)
                                                     (end_time ?time4))
                                                     (test (during ?time1 ?time3 ?time2 ?time4))
                                             =>
                                             (assert (during-rel (rel1 ?name1)(rel2 ?name2))))

(defrule alert-during
        (phase Team)
        (eng (name ?name1&MA|MB)(folder_name ?fname&~nil))
        (eng (name ?name2&~?name1&MA|MB)(folder_name ?fname&~nil))
        (during-rel (rel1 ?name1)(rel2 ?name2))
=>
(printout t "Member " ?name1 " alerts " ?name2 crlf))
```

Figure 2 shows how one of the interval relations is created and used by the application. The *test-during* rule passes the beginning and ending time points for two facts to the *during* deffunction. If the time points meet the requirements for a *during* relation, then a fact is asserted in working memory. For example, if Team Member MA has a beginning time of 3 and an ending time of 3.5, and Team Member MB has a beginning time of 2.5 and an ending time of 3.7, then MA would occur *during* MB and the relation *(during-rel (rel1 MA) (rel2 MB))* would be asserted. The same pair of facts is also evaluated by the other seven rules that establish Allen's interval relations: *before*, *after*, *overlaps*, *starts*, *finishes*, *meets*, and *equals*. In our example, *during* is the only relation out of the seven that describes the temporal connection between MA and MB. The rule *alert-during* then uses the interval relation created as part of a stage in the workflow process where one Member alerts another, perhaps signalling an imminent bottleneck.

Conclusion

We have shown that there is an opportunity for temporal representation and reasoning to play in the development of workflow. We have also developed an expert system using CLIPS to model the temporal aspects of a specific engineering design change review process. Our expert system improves the review process by controlling the advancement of the review folder through the stages of the process, and by alerting the engineers to delays or temporal conflicts that may occur. The objective was not to build a full-blown system for design change review, but rather to demonstrate how an expert system acting as an intelligent agent could be used for the temporal component of the process Future research includes extending the expert system to handle more complex projects.

References

Allen, J. F. "Maintaining Knowledge About Temporal Intervals," *Communications of the ACM*, (26:11), November 1983, pp. 832-843.

CLIPS Version 6.0 Basic Programming Guide, Houston, Johnson Space Center, Software Technology Branch, 1993.

Curtis, D. *What About Product Manager? IBM's Solution for Engineering Design Change Control*, Florida, Maximum Press, 1994.

Johns, B. *Experiences Implementing Workflow Based Systems in Manufacturing Enterprise Product Development and Change Processes*, http://optimus.cs.uga.edu:5080/activities/NSF-Workflow/bjohns.html, 1996.

Kalakota, R., and Whinston, A. B. *Frontiers of Electronic Commerce*, New York, Addison_Wesley, 1996.