Working Group Report on Computer Support in Project Coordination

edited by

Frank Maurer
University of Kaiserslautern, P.O Box 3049, 67653 Kaiserslautern Germany
e-mail: maurer@informatik.uni-kl.de

Abstract

This paper summarizes the presentations, discussions, and results of the Project Coordination Workshop of the IEEE Fifth Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WET ICE '96).

1 Introduction

To compete globally, enterprises must reduce the time and money needed for project execution. For example, reducing product development times and costs is increasingly important for engineering processes because it enables the enterprise to sell its products earlier and cheaper than its competitors.

In pursuit of this goal, more and more organizations are forming virtual corporations. Virtual corporations are temporary associations among heterogeneous groups and organizations based on the key competencies of its members. The purpose of these temporary alliances is to quickly respond to market opportunities, improve product/service quality, and achieve more flexibility in work and work organization, or to work on projects larger than can be managed by any single company, for example the International Space Station.

A major requirement for the creation and management of virtual corporations is distributed project coordination. This includes the planning and scheduling of projects, execution of projects, coordination of tasks, resolution of competing objectives, achievement of global coherence, change propagation, communication across heterogeneous groups, and maintenance of access to valid information. Without these and other similar functions, the work becomes disorganized and the twin goals of time compression and cost reduction cannot be achieved.

Yet, most existing computer support for distributed work is mostly "groupware," which addresses only the sharing of (informal) information among people. Computational methods that use process models and structured data in order to link software and people at the task level are still not widely employed, and the lack of coordination of distributed processes is the most critical bottleneck limiting the size and complexity of projects and products today.

As an alternative to groupware, workflow management systems are used to support routine tasks that can be planned and implemented before enactment. However, for many projects this is not possible because the tasks are inherently non-repetitive: in software engineering, for example, the planning of later stages of design and implementation has to be based on the results of the earlier stages.

Thus, groupware systems provide too little structure and current workflow techniques are too restricting to support virtual corporations, except a very limited way.

This is particularly true in distributed engineering projects, where many of the tasks cannot be analyzed prior to execution and where the duration of cooperation is typically very long. Engineering projects are in a sense the most challenging case of virtual companies that, in general, are very dynamic. Distributed engineering projects are also particularly interesting because of the obvious need to reduce cost and increase speed by allowing engineers and designers in different organizations to work directly on a peer-to-peer relationship rather than to be mediated by several layers of management.

This workshop focused on techniques and systems that allow flexible planning and control of processes in distributed projects in all kinds of virtual enterprises, but especially in engineering ones. Nearly all of the applications that were presented at the workshop dealt with software engineering or mechanical engineering. There is an urgent need in these domains to improve project coordination by using automated computer support.

At the workshop, the use of the World Wide Web (Web) for agent interfaces, data structuring, project planning, and team coordination was emphasized. Clearly, the

participants see the Web as a kind of infrastructure which can be used to develop more sophisticated and intelligent project coordination tools.

But before we can discuss project coordination, we have to characterize from the participants' viewpoints what a project is. A project can be characterized by the following attributes:

- **Goals:** A project tries to reach a given goal, e.g., to develop an aeroplane which carries 400 persons and uses 20% less fuel. Goals may be decomposed into design subgoals; the satisfaction of the latter leads to satisfaction of the former.
- **Planning:** An inherent feature of projects is the occurrence of one or more planning phases. By project planning, the goal can be decomposed into a set of activities that will achieve the project's objectives. Additionally, the plan may contain scheduling information, such as who should do what, and when.
- **Alternative decisions:** Within a project, many decisions have to be made. For every problem, a set of alternative solutions may be found. A decision selects one of the alternatives. Because one decision may interact with other decisions, one goal of project coordination is to reach an overall agreement.
- **Several agents:** In a project, typically several persons have to cooperate to reach the common goal. Sometimes an activity may be executed by a computer. Therefore, we prefer the term "agent" instead of "person" to characterize an active entity which works on activities.
- **Constraints:** Projects and their artefact designs have global constraints over the achievement of their goals. Examples are resource limitations, such as time, money, and personnel; and design constraints, such as the total weight of an aeroplane.
- **Distributed conflicts:** Even if there is a single top-level goal, its decomposition into subgoals to be achieved by different agents may lead to conflicts among them. For example, a car engine designer may want to increase power while at the same time another engineer is trying to use a lighter frame that may not support a larger engine. Together with the transmission engineer, they all are trying to keep the car under a certain weight. To solve this conflict, negotiations have to take place.

All of these aspects have to be taken into account if a project coordination tool is to support its users adequately.

The next section summarizes the paper presentations. In Section 3, a generic architecture of a project coordination support tool is described. Techniques that are needed to build its components are discussed in Section 4. Section 5 boils down the issues that were discussed most heavily by the workshop participants.

2 Presentations

The nine long papers accepted for the workshop were selected from 20 submissions and span a wide spectrum in the area of project coordination approaches.

M. Barbuceanu proposed a conceptualization of the coordination task based on "structured conversations" [1]. His language uses KQML messages for communication among agents. It allows a rule-based definition of flexible conversation protocols. His major application is a multi-agent system which supports the supply chain for manufacturing enterprises.

C. Boldyreff's presentation dealt with techniques that will support the software development process in virtual software corporations [2]. It identified a number of open problems, including tools for virtual software configuration, distributed metrics collection in heterogeneous tool environments, and the application of process quality standards in virtual enterprises.

F. Brazier and J. Treur discussed the DESIRE framework, which can be used to model multi-agent systems [3]. The DESIRE language has a formal basis and can be used to support the analysis, modeling, and implementation of design coordination systems. The authors analyzed the cooperation and communication of the participants of a real-life design project: the design of a part of the interior of a specific aeroplane.

K. J. Cleetus described the software tool PACT, which supports project management and the coordination of people [5]. PACT uses Microsoft Project for planning purposes and stores the resulting information in a database. This database is then used for generating worklists for the team members. The up-to-date information in the database supports project coordination because all participants are able to access the current state of a project.

B. Dellen and F. Maurer presented the integrated project planning and execution environment CoMo-Kit [6][9]. This systems uses project plans to generate causal dependencies between information entities and thereby improves the traceability of decisions. Because of the improved traceability, CoMo-Kit is able to support reaction to changes. Further, the workflow engine of the system is able to alternate between project planning and execution steps, providing more flexibility than in conventional workflow approaches.

S. Goldmann's Procura system supports the planning and scheduling of agent based design projects in a hierarchical, top-down approach [7]. The tool uses Charles Petrie's REDUX' system [10] to enable the re-planning and revision of planning/scheduling decisions. This paper received the WET ICE '96 Best Paper Award.

L. Gupta uses a constraint approach to improve the communication and coordination in concurrent design

projects [8]. His agent-based system supports multiple perspectives, contains a notification mechanism, and a design knowledge management component.

S. Ramakrishnan's approach uses the World Wide Web as a basis for managing software projects and collecting software metrics [4]. His WISE system tracks open issues in the project, handles to-do lists for its users, and provides informal communication between its users.

3 Approaching project coordination

Based on the presentations and the subsequent discussions, Miro Benda proposed a top-level architecture of a project coordination tool. A slightly modified version is described in this section. First, we define the objectives of a coordination system.

3.1 Objectives

To be considered useful, a project coordination support tool has to satisfy several business and technical objectives. The overall business goals for this kind of software are to reduce the costs, time, and risks of a project and to improve the product and process quality. Basically, the task of a coordination support tool is to *give the right information to the right people at the right time.*

On a more technical level, one can distinguish between objectives concerning the process, the information flow, the change management, and long-term improvements.

Process objectives

The workshop participants stated the need for more flexible tools for project coordination. This requirement arises from the fact that often the team members have to cope with incomplete specifications and knowledge. Hence, a supporting tool has to allow for dynamic changes to the project plan and the workflow.

In reality, changes will often occur: new requirements arrive from the customer, new techniques have to be introduced, etc. The system therefore needs to be adaptable to cope with these changes. This adaptability is strongly related to the flexibility of the system.

In addition, when changes occur, the system should be able to predict their costs and risks.

Objectives concerning the information flow

Coordination is improved by easy access to all information about the project, such as information about which tasks can be executed, who is working on each task, which information is currently available and valid, and what background knowledge has to be used in an activity. Making the entire information store accessible, however, may result in an information overload. Therefore, the coordination tool has to find a balance between information overflow and information underflow.

Because many decisions made in a project are based on incomplete information, mistakes can happen. A tool has to support its users in the early detection of mistakes and in the avoidance of conflicts between decisions.

A desirable feature of a project coordination tool would be to show improvement opportunities to its users. These opportunities might concern the project plan as well as the decisions made.

Tools for coordinating engineering projects should allow for more design options to be considered by reducing the time needed for generating one option. Thus, a larger part of the design space can be explored.

Change management

Change management is needed to achieve process adaptability. A central feature is to automatically inform the appropriate users about a change and its consequences for their work.

Change management also includes the reduction of undesired changes (e.g., after an erroneous decision) as well as an increase in change tolerance of the process by supporting the users in their reaction to changes.

Another important feature of a change management system is the ability to predict the costs and risks of a revised plan.

Long-term improvements

Although the workshop concentrated on coordination aspects within a single project, the participants noted the need for long-term process improvement in the sense of Total Quality Management. This means that a system should support its users in learning what was good and what went wrong in the project so that they might use this knowledge in the coordination of the next similar project. Clearly, this requires a kind of "Experience Factory" which stores reusable parts of the process.

3.2 A generic architecture of a project coordination support tool

Figure 1 illustrates a generic architecture of a project coordination support system.

The input of the system is the budget, usable resources (teams, machines etc.), and the "philosophy" of the mission.

The results of the project are products, processes, and metrics (which allow evaluation of the project's performance for long-term improvements).

The tool has to be built on top of the enterprise infrastructure platform; CORBA or the Internet/Web are possible standards on which to build.

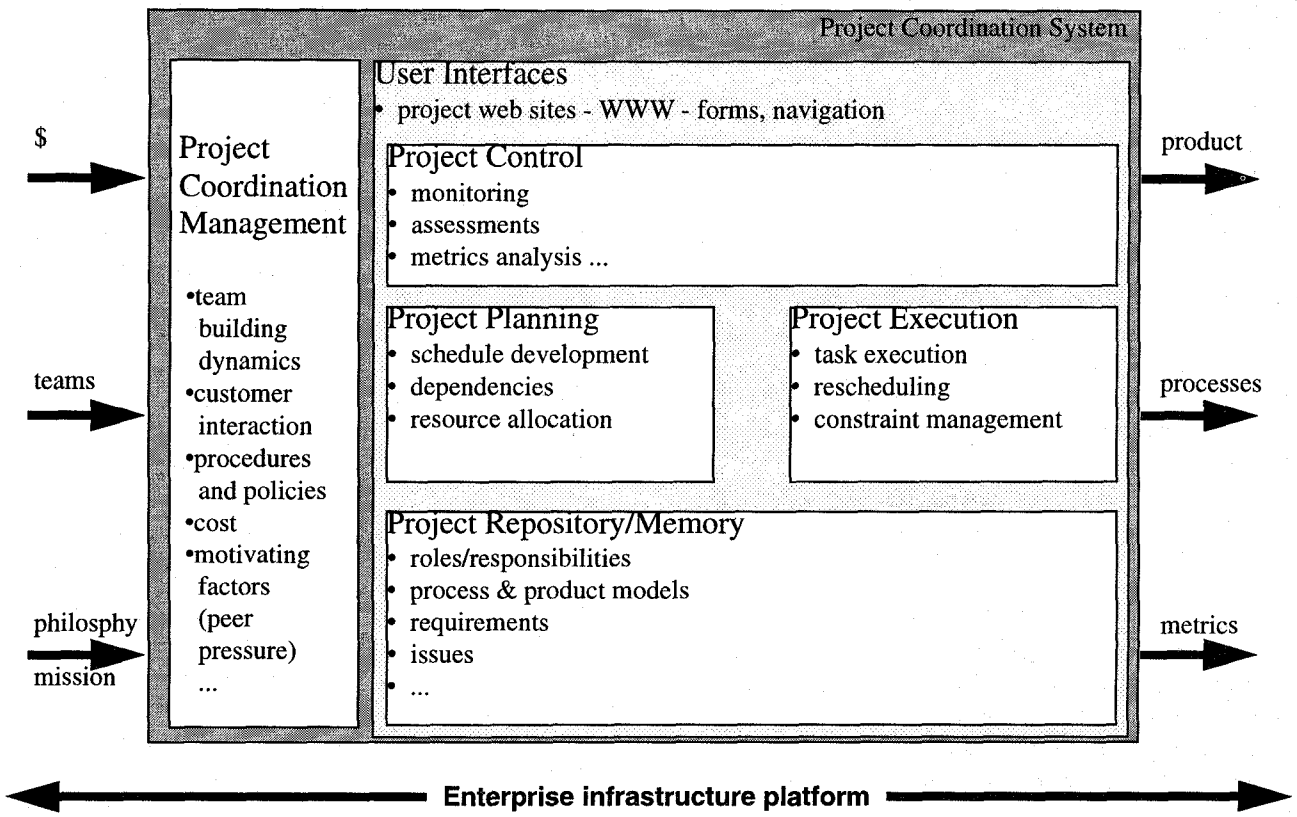

Figure 1 Project coordination architecture

Before the project begins, many activities concerning the softer organizational and management issues, such as team building, have to be addressed. Computer support for this is difficult at this stage because it mainly tackles personal interactions.

A coordination support system consists mainly of four components:

- The project repository serves as a kind of memory for the people involved. It stores the roles and responsibilities of the team members, open issues, all information on the product (e.g., requirements), and process and product models.
- A project planning component allows users to plan and schedule activities, determines dependencies between information items, and supports resource allocation.
- A project execution component uses the project plan for workflow management. It handles the worklists of the users, supports task execution, and is responsible for constraint and change management. This component has to support re-planning and rescheduling and, therefore, has to provide a highly flexible execution environment.
- The project control component supports the monitoring of the project, helps in the assessment of the current state, and allows metrics collection and analysis.

All tools need an easy-to-use interface, preferably with a common look and feel. The participants agreed that using Web browsers to access project Web sites would provide such a common interface and would support navigation through the project information space.

To build a project coordination support system, many different techniques are needed. Some of these are discussed in the next section.

4 Techniques for project coordination

Although all the participants agreed that the social aspects of project coordination are very important, these aspects were not discussed in the workshop because of the group's computer-oriented background. Instead, we concentrated on computer-based approaches which support coordination. Clearly, due to space constraints, this discussion here does not address every technique that might help coordination.

A tool for project coordination support has to be based on explicit representations (or ontologies) of the processes, products, resources, organizational structures, interactions, and negotiation & co-operation strategies. Explicit representations allow the system to have a "kind of understanding" about what is going on in the project.

There is, for example, the necessity for dependency tracking and mechanisms that inform the right people about changes.

To develop these explicit models, some effort has to be spent on task analysis. Existing techniques from requirements analysis or knowledge acquisition can be used for this purpose.

Explicit process models can be seen as an electronic process handbook. This is also very useful for total quality management.

Electronic access to project-related product and process information should be supported. Using the World Wide Web for this is a sensible choice because of its ease of use and the (more or less) transparent access to information.

In real-world projects, much time is wasted because of difficulties with information exchange. Sometimes information that is electronically available is printed on paper, sent to a collaborator, and re-entered into a computer; sometimes a lot of effort is wasted because of incompatible data formats. To reduce this waste, it is necessary to establish standards. The important issue is not *which* standard is preferred, but *that* a standard is used.

Constraint management techniques reduce the amount of conflicting decisions by restricting the possible alternatives.

Information filtering approaches reduce information overload and alert users to relevant information only.

A basic problem in project coordination is: Who has to be informed when a change occurs? To answer this question, one may appeal to developments in impact analysis and change propagation (using, e.g., truth maintenance systems). Tracing the rationales for decisions is a prerequisite.

The automatic generation of design alternatives using AI approaches supports engineering projects because it enlarges the design space that can be evaluated. Because of the enormous size of the design space, the system should support users in navigating through it.

To support communication and coordination, Internet agents can be used to work on behalf of human users. For example, many of the less important negotiations (e.g., concerning the date of the next project meeting) can be carried out automatically by using this technology.

To improve the process robustness, it is possible to run redundant, parallel processes. Additionally, what-if analysis and sensitivity analysis may give a better insight into the process and can be used to reduce the risk of running out of time and/or budget.

Long-term process improvements can be gained by storing "good" project plans in an experience base (or project memory) that can be accessed, manually or by a computerized agent, for future project planning.

Using reusable components — processes as well as product data — may prove to be a key factor in future project coordination.Therefore, the retrieval and instantiation of these components has to be supported. This is expected to reduce the amount of duplicate work performed.

Last, but not least, the participants discussed the instrumentation of the process: metrics collection, for example resources used or money spent, improves the monitoring of project performance and reduces the risk of poor management that results in missed deadlines.

5 Hot topics and open problems

This section summarize the main topics of the workshop discussions and some open problems.

5.1 Important issues

A major part of the discussion dealt with different agent models and methods.

On one hand, several approaches consider agents as black boxes. Here an agent can be accessed externally and has a given functionality which supports one of the tasks in project coordination. For example, there are agents that handle dependencies, perform scheduling, or manage constraints. The internal structure of a black box agent is not visible.

On the other hand, some approaches consider agents as transparent boxes. The internal structure is visible and can be defined/changed to get a special functionality. The participants agreed that the second kind of agent model is more flexible and can be used to implement the first. The trade-off is that some additional human work is needed to rebuild functionality which is already incorporated in the black box agents.

Along with the flexibility of the agent model comes flexibility in the cooperation protocol of the presented approaches. Typically, the black box approaches use a predefined, purpose-specific protocol for the co-operation between several agent, whereas transparent box approaches allow system designers to dynamically define the co-operation strategies between the agents.

Many coordination problems result from a lack in notification and change propagation support. As mentioned before, changes occur frequently in projects. The question then is: Who must be informed about the change? A coordination support system can make different assumptions in its change handling mechanism:

- **Active notification versus passive management:** The tool may be proactive and could automatically send messages to its users to inform them about a change. On the other hand, it may passively store the current state of

the project, allowing its users to access up-to-date information. The first approach produces more communication overhead, the second may lead to some users missing a change.

- **Notify all versus notify only the particular affected:**
  A proactive tool may send messages about a change to all team members or to only the users who are affected by a change. The first approach may produce an information overload because many changes are not relevant for the recipient. The second requires a dependency theory to determine who is affected by a given change. If the theory is not complete and correct, some users may not be notified even though they are affected.
- **Dependency theory:**
  A generic theory can be used to automatically generate dependencies between decisions. For example, all users of a specific information are notified when it becomes invalid, or, if a task becomes invalid, all people working on its subtasks are notified. Domain-specific theories stored in a knowledge base can be used to generate additional dependencies. For example, if the power of an engine is changed, the engineer working on its connection to the car is informed. Additionally, the user may explicitly request to be informed about a particular kind of change.
  There is a trade-off between the generality of the theory and the user's work-load: A generic theory can be built into a coordination system and can be reused in every application domain. A domain specific knowledge base, on the other hand, can be used in one domain only. Switching to another domain requires the development of a new knowledge base (thereby increasing the amount of human work). Explicit requests have to be entered by the users, increasing their work-load.
- **Time of change propagation:**
  A change may be propagated before or after it is actually carried out. Pre-propagation can be seen as a request to find out whether a proposed change is agreeable, whereas post-propagation basically informs that a necessary change has taken place already.

5.2 Outstanding issues

Two open problems for which no direct solutions were found were addressed in the workshop. The first dealt with feedback loops; the second tackled the problem of early conceptual design.

- **Feedback loops:**
  This problem results from (at least) two interdependent tasks where the output of one is the input of the other, and vice versa. There is a cyclic dependency between the tasks, and conflict resolution is difficult.
- **Early conceptual design:**
  Most of the presented approaches rely on a decomposition of tasks into subtasks. In the early design stages of a project, this kind of decomposition may not be clear or possible at all.

Acknowledgments

This paper summarizes the results of the WET ICE '96 Project Coordination Workshop. I want to thank Mihai Barbuceanu, Miro Benda, Cornelia Boldyreff, Frances M. T. Brazier, K. Joseph Cleetus, Mark Cutkosky, Rob Davis, Barbara Dellen, Sigrid Goldmann, Lokesh Gupta, Charles Petrie, Shinya Sekimoto, Jan Treur, and Paal Ytterstad for their participation in the workshop and their contribution to the discussions. The good ideas can be credited to them; the mis-reporting is mine alone.

References

[1] M. Barbuceanu and Mark S. Fox, "Coordinating Multiple Agents in the Supply Chain," in [11].

[2] C. Boldyreff, J. Newman, and J. Taramaa, "Managing Process Improvement in Virtual Software Corporations," in [11].

[3] F. M. T. Brazier, C. M. Jonker, and J. Treur, "Modelling Project Coordination in a Multi-Agent Framework," in [11].

[4] J. Callahan, S. Ramakrishnan and W. Sun, "Software Project Management and Measurement on the World-Wide-Web," in [11].

[5] K.J. Cleetus, C. Cascaval, and K. Matsuaki, "PACT - A Software Package to Manage Projects and Coordinate People," in [11].

[6] B. Dellen and F. Maurer, "Integrating Process Planning and Execution," in [11].

[7] S. Goldmann, "Procura: A Project Management Model of Concurrent Planning and Design," in [11].

[8] L. Gupta, John Chionglo, and Mark Fox, "A Constraint-Based Model of Communication and Coordination in Concurrent Design Projects," in [11].

[9] F. Maurer, "Project Coordination in Design Processes," in [11].

[10] Charles Petrie, "The Redux' Server," *Proceedings of the Intl. Conf. on Intelligent and Cooperative Information Systems* (ICICIS), May 1993, Rotterdam, The Netherlands.

[11] *Proceedings of the Fifth Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises*, June 19-21, 1996, Stanford, California, USA. Los Alamitos, CA: IEEE Computer Society Press, 1996.