Available at
www.ElsevierComputerScience.com
POWERED BY SCIENCE DIRECT®
Journal of Systems Architecture 49 (2003) 457–473
JOURNAL OF SYSTEMS ARCHITECTURE
www.elsevier.com/locate/sysarc

Architectural concerns in distributed and mobile collaborative systems

Schahram Dustdar *, Harald Gall

Distributed Systems Group, Vienna University of Technology, Argentinierstrasse 8/184-1, 1040 Wien, Austria

Abstract

Organizations increasingly coordinate their product and service development processes to deliver their products and services as fast as possible, and to involve employees, customers, suppliers, and business partners seamlessly in different stages of the processes. These processes have to consider that their participants are increasingly on the move or distributed while they are working. Expertise needs to be shared across locations and different mobile devices. This paper describes a framework for distributed and mobile collaboration, defines a set of requirements for virtual communities, and discusses a mobile teamwork support software architecture that has been developed in the EU-project MOTION. The framework together with the architecture enables to enhance current collaboration approaches to include the dimension of mobile participants and virtual communities for distributed product development. This is achieved by integrating process and workspace management requirements with Peer-to-Peer Middleware, Publish-Subscribe, and Community and User Management components.

Keywords: Process awareness; Software architecture; Mobile collaborative systems

1. Introduction

Software systems such as workflow management, Groupware, process modeling, and project management have been used to automate or to augment business processes in organizations [4,7,9,14,15,32]. In recent years there have been considerable attempts to merge or to integrate project management systems, workflow management systems (WfMS) [5,33], Groupware Systems e.g. [27], and business process modeling systems [12]. Corporate research labs [6,7] and product teams [2,22,26] have made significant steps forward. Future distributed and mobile collaborative systems focus on covering inter-organizational processes (e.g. product value-chains) and their activities on the Internet [4,10,25,28,30,31,34] regardless of location (mobility) and regardless of devices used. A simple sketch of a conceptual architecture for distributed and mobile collaborative systems was presented in [13].

In today's business environments participants in virtual project communities (VPC) demand process awareness to a relatively high degree of the software they use for collaborative work. In addition organizational awareness (e.g. roles) and

* Corresponding author. Tel.: +43-1-58801-18414; fax: +43-1-58801-18491.

E-mail addresses: dustdar@infosys.tuwien.ac.at (S. Dustdar), gall@infosys.tuwien.ac.at (H. Gall).

1383-7621/$ - see front matter
doi:10.1016/S1383-7621(03)00092-4

mobility aspects become increasingly relevant. Current WfMS and Groupware systems do not combine those features virtual project communities need: *information sharing*, *process sharing*, *process composition*, and *process configuration.* Future systems for virtual project communities need to facilitate not just mobility of content to group members, but also *mobility of context* of activities in business processes, i.e. providing information about process instances, the team configuration (i.e. participants and their roles or skills), their associated artifacts, and connectivity modes of group members (such as connected, disconnected, or ad-hoc).

Business processes in general and associated workflows in particular exist as logical models. Business Process Management Systems and WfMS complement each other. Workflow systems generally aim at helping organizations' team members to communicate, coordinate and collaborate effectively and efficiently. Therefore WfMS possess temporal aspects such as activity sequencing, deadlines, routing conditions, and schedules. WfMS are typically "organizationally aware" because they contain an explicit representation of organizational processes (process model) [8]. However traditional WfMS present a rigid work environment consisting of *roles* and their associated *activities and applications*.

In this context they do not sufficiently support virtual project communities, which require tools for frequent changes regarding process participants, ad-hoc formation of groups collaborating on a business process, and device-independent support of group activities. Unfortunately today's WfMS assume that each *work item* is executed by a *single* worker. Hence, distributed collaborative work in virtual project communities finds only limited or no support by WfMS. Most WfMS focus on automating structured (modeled) intra-organizational business processes. Groupware [16], on the other hand, typically does not contain any knowledge or representation of the *goals* or underlying business *processes* of the group [15,18,20,21,29].

Cooperative tasks in teams are increasing, and as a consequence the use of collaborative systems is becoming more pervasive. To understand current collaborative technologies we present a "distributed and mobile collaboration grid", which distinguishes two dimensions: *Process* and *Connectivity* as shown in Fig. 1.

A business process can be ad-hoc, semi-structured, or structured (modeled). For example a business process such as "*customer order entry*" can be modeled using a traditional WfMS. However, a *structured* process can only be enacted (instantiated) as it was designed. If an exception occurs, a workflow administrator needs to remodel the process before the execution can continue. This limits the usability of workflow systems in a world where constant adaptation to new situations is necessary and where teams are increasingly mobile and distributed. An example of an *ad-hoc* process is discussion of a project's design review using Groupware. A *semi-structured* process consists of groups of activities, which are modeled; however in contrast to a structured (modeled) process it also consists of activities, which are not pre-defined. Fig. 2 shows an example of a generic semi-structured process template for projects. For example, there might be one or more activities between activities "Project Research" and "Project Presentation", which are not known beforehand and, therefore, cannot be modeled in advance.

The second dimension presented in the DMC grid is *Connectivity*. Here we distinguish between three modes: fixed, mobile, or ad-hoc. We speak of *fixed* connectivity when users work on computers permanently connected to a network. For example in an office where each employee has a personal computer connected to the company-wide network or a wide area network (WAN). *Mobile* connectivity essentially describes a mode where people are "on the move" but access data and applications located on their remote network. The *ad-hoc* mode allows users to establish a "virtual" group of users on the fly. Participants in ad-hoc groups may have network connectivity either permanently or sporadically.

The mobility of participants also offers new ways of distributed collaboration: processes are no longer bound to locations of resources (such as participants or artifacts) but can consider several availability modes.

Technologies used in organizations today basically can be associated to one cell of the grid in

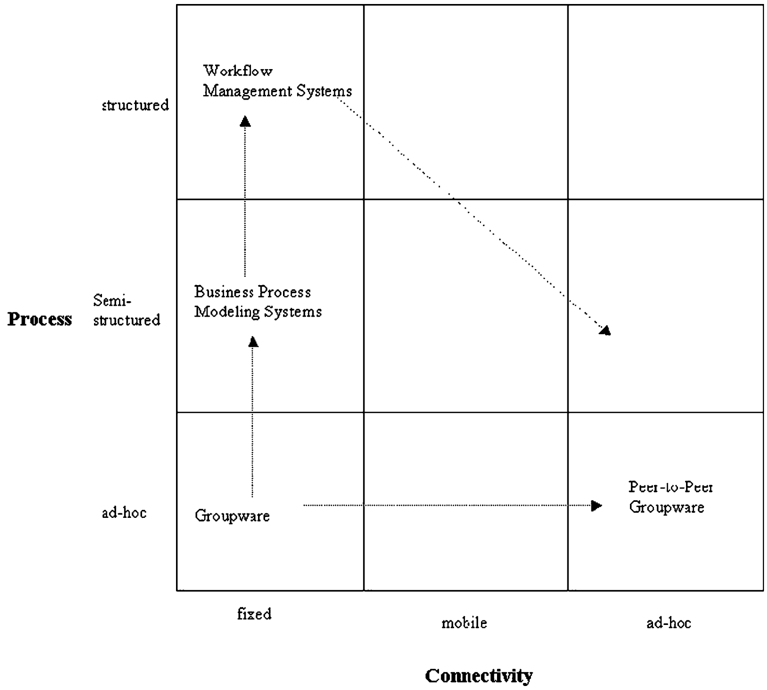

Fig. 1. Distributed and mobile collaborative systems grid.

Fig. 1. The arrows in Fig. 1 depict the evolution of recent technology developments. Workflow management systems traditionally provide support for structured processes (y-axis) and fixed connectivity (x-axis). WfMS are being enhanced to also provide support for mobile and ad-hoc modes of connectivity. This trend increases the flexibility, adaptability and traceability of process activities in WfMS to support mobility of context for business processes. Business Process modeling systems allow modeling of both semi-structured and structured processes (y-axis) but require fixed connectivity of users to a network. Groupware systems provide support for ad-hoc processes (synchronous or asynchronous). In most cases groupware requires fixed connectivity. Technologies such as Peer-to-Peer or mobile device support for Java facilitate the trend towards Peer-to-Peer Groupware, which per definition enables users to establish ad-hoc structures (y-axis) and in the same time ad-hoc connectivity (x-axis).

In Fig. 3 we summarize our evaluation of technologies supporting teamwork in a classification matrix using a simple scale with three types of support for the requirements we outlined above. Basically we differentiate between synchronous and asynchronous technologies for teamwork support. For each category we provide a well-known example system. During our case study requirements analysis we concluded that distributed product development in virtual communities requires a blend of synchronous and asynchronous systems support for communications as well a basic support for asynchronous coordination of team members and their activities.

The requirements for distributed product development teams in virtual communities cannot simply be met by using a combination of traditional synchronous and asynchronous systems since the criteria for successful systems in this area differ substantially compared to traditional "enterprise information systems." We identified and implemented four fundamental feature sets for our case study scenario (meta-data retrieval, expert search, information sharing, notifications) and refer to a system having successfully implemented

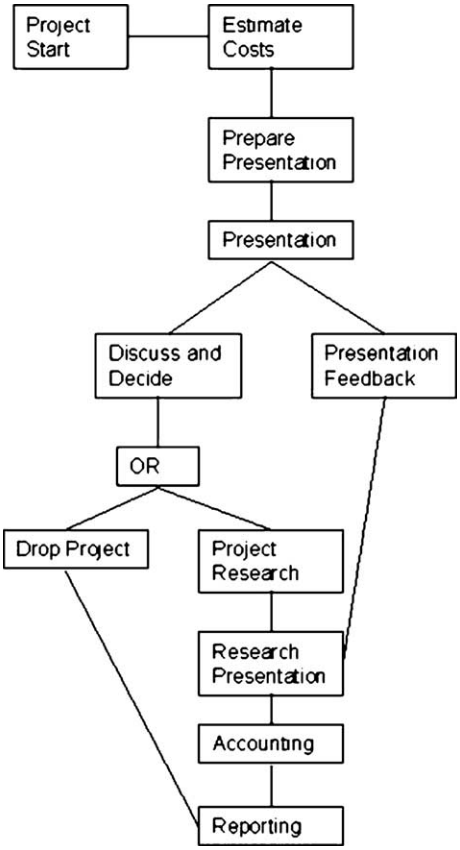

Fig. 2. Semi-structured process.

those as a DMC-system (Distributed and Mobile Collaboration).

To summarize the challenges: To build software systems supporting fully distributed and mobile collaboration requires functionalities currently found in *different* software application domains such as WfMS, Groupware, or Business Process modeling tools. A challenge for future DMC systems is to develop a sustainable software architecture, which provides support for both dimensions presented in Fig. 1 and is flexible, adaptable, and traceable regarding processes of collaborative work. It is of paramount importance for a DMC system to enable geographically dispersed users with different modes of connectivity (fixed, mobile or ad-hoc) to share information in various kinds such as using middleware and/or peer-to-peer technologies. Users need to register themselves and receive notifications on events, regardless of their location or device they use. Collaboration partners need to be empowered to locate each other, find experts in required domains and link all coordination information with artifacts such as documents. Therefore the mobility of context (who, what, why, when and using which resources) is essential for DMC systems.

Example *use cases* of distributed and mobile collaboration include: information updating and notification of availability (of resources); searching and inviting people for diverse synchronous communication (e.g. chat, video/telephone conference); expert search; information retrieval about resources (e.g. users, artifacts, processes); synchronous and asynchronous communication in a community; synchronous collaboration on artifacts (e.g. Groupware); community establishment and updating. Those use cases are presented in

	Requirements for Virtual Project Communities			
	Meta-data retrieval	**Expert Search**	**Information Sharing**	**Notifications**
Synchronous Communications (Audio/Videoconferencing)	1	1	3	1
Asynchronous Communications (e-mail)	1	1	3	5
Synchronous Collaboration (Shared Editing Groupware)	1	3	3	3
Asynchronous Collaboration (Workspace Groupware, Intranet)	3	1	3	1
Asynchronous Coordination (Workflow Systems)	3	3	1	3
DMC Systems (Virtual Community Software)	5	5	5	5

Fig. 3. Technologies and features. Legend: full support: 5; limited support: 3; no support: 1.

more depth in the next section discussing our case study.

The contribution of this paper therefore is to elaborate on architectural concerns for distributed and mobile collaborative systems. We achieve this by decomposing process- and workspace management issues and presenting a three-layered architecture, which integrates process awareness with the easy to use groupware (workspace) metaphor. The remainder of this paper is organized as follows: Section 2 presents a case study for distributed and mobile teamwork. Based on the experiences and lessons learned in Section 2, Section 3 discusses architectural concerns such as mobility, publish/subscribe, peer-to-peer middleware, and web services. Section 4 decomposes the proposed system architecture and elaborates on the layers and components. Section 5 concludes the paper.

2. Case study: mobile phone design

We have investigated the process requirements for the distributed product development of mobile phone software for a large European telecommunications company whose development sites are located in several countries all over the world. Each development site has main responsibilities of some products but the development sites co-operate quite extensively in many phases of the development of the products. The complexity of the product family, the structure of the development organization, and the market pressure in terms of price, performance and rapid need to introduce new features as they become available in the networks, make mobile phone software development a challenging task. The company applies two different techniques to cut down product development costs and reduce time to market:

- *Product family architecture*: Common parts of hardware and software of mobile phone generations are developed for the entire family and only product-specific variations or parts are developed in individual product development projects.
- *Concurrent engineering*: Common parts and the product specific parts are developed concurrently in different development sites. Therefore, new products can be introduced to the market at the same time world-wide.

The development of mobile phones software involves many different steps and processes among which are, for example, platform definition, platform management, component development, as well as platform and component integration for product development. In the MOTION project we have performed a detailed analysis of collaboration activities and technologies used for processes such as software development, configuration management, or design review in particular. Project managers of single-site and multi-site projects were interviewed on how they achieve their project goals in terms of collaboration, technologies and processes used. We summarize our findings below, but details of this study are beyond the scope of this paper.

One of the critical aspects in the company's product development is effective, focused, and timely information sharing. In the studied projects it turned out that information sharing is done in a rather traditional way: The most widely used tool for information sharing is e-mail; further, tools such as Lotus Notes, Intranet, and phones were used to communicate within and across project teams. Some of the interviewees remarked that an effective way to gather and share information was to have casual conversations with people, for example, in communication areas or hallways. Personal networking was also ranked high as a medium to acquire filtered and focused pieces of information. Some interviewees noted that personal networking is more effective than Lotus Notes and Intranet to get information because of limited searching capabilities of these tools. In some projects specific information plans were developed to support effective information sharing. These plans were established at the beginning of a project and they described how and what information was to be shared and with whom.

The interviews also showed several problems of information sharing and distribution: especially product managers indicated that there are clear information sharing boundaries between projects. Often projects do not want to publish all of the requested information, especially if persons who enquire information are not members of a particular group or project. In addition, locating relevant information is difficult and asking via e-mail was considered to be too slow. Product managers see this as a problem because they need overall information of different components from different groups to build up the final product. Another difficulty is locating and finding relevant information effectively.

The business process analysis further investigated the collaboration tools and technologies used so far. It showed that different synchronous and asynchronous means are used: (1) phone and e-mail; (2) Intranet and Lotus Notes; (3) shared network directories; (4) shared work spaces; and (5) videoconferencing.

E-mail is the most frequently used means for information sharing and communication, but it was experienced to have several drawbacks: large mailing lists transform e-mails to chain letters; roles of process participants fade over time; group conversations are quite powerless and usually no decision can be made; the length of e-mails grows fast and lengthy e-mails are hard to read; many messages are received per day, but important messages easily get lost; many messages are saved but usually they get lost in various folders; many attachments are saved as messages instead of storing them into a database such as Lotus Notes; discussions and information sharing is possible only for a small group of participants; effective conversation can hardly be achieved, if at all; and cultural differences are likely to affect the e-mail discussions.

Phone, on the other hand, is rather used when answers are needed quickly. These situations usually occur at the end of projects when time to delivery is short.

Asynchronous groupware such as Lotus Notes is considered to be an important information management tool and most of project data is stored in such systems. Nevertheless, the study pointed out that users of such systems have perceived several shortcomings: finding information or even the right databases is difficult; access rights management for database causes a lot of additional effort; people do often forget to place their documents into the database; the information representation is not formal enough, which makes searches even more difficult for users; the data placement strategies are manifold and usually there is not enough time to update all concerned databases; and some people found it difficult to use mostly due to lack of training.

Intranet is used frequently, because it provides mostly general information and guidelines. But information is hard to find without adequate effective links in the Intranet, old information often is confusing when searching and it Intranet provides almost no support for locating people or expertise in the company.

As a consequence, *shared directories* are still often used. Project members place their documents and other artefacts into commonly shared disk spaces. Since such directories usually are shared within project teams, only little time is spent on searches and information location. But this concept only works effectively for smaller project groups.

Shared editing and whiteboards as provided by tools such as NetMeeting are needed to decrease the distance between distributed project members. These tools are used for reviews, document editing, drawing, concurrent engineering, and application sharing. However, problems with time zones and different devices used remain open.

Videoconferencing, as an additional synchronous communication means, restricts communication to be more formal than in face-to-face meetings. Meeting materials are difficult to supply especially for technical meetings as they are needed for telecommunications product development. Besides technical problems with connections and multicast support, gestures or moods cannot be transferred effectively such that face-to-face meetings cannot be fully substituted by this means. Further, ad-hoc meetings are ineffective due to the high effort for setup and connection establishment. The participation is limited and not every group member can participate in the discussion so that valuable comments or remarks are not expressed.

Multi-site project analysis—for project sizes greater than 100 participants distributed over more than seven sites in different countries—further revealed project collaboration habits in working with other people, teamwork, knowledge work and personal work. This multi-site dimension introduced additional challenges related to sharing of project status and news, getting feedback on people's work and finding information. User requirements are grounded in limited technological capabilities: information, for example, that has been stored in an asynchronous groupware database needs to be accessible for all group members. Since change notifications are weakly developed, change awareness cannot be achieved so that information pull is the most frequently used pattern.

Distributed knowledge access and distribution via personal contacts (i.e. expertise of people in different sites) is an open issue in multi-site collaboration. Roles of project participants and their expertise are invisible and the relevance and context of specific documents is difficult to gather. The need for keeping documents up-to-date across location boundaries raised the importance of shared spaces to check in/out and control work products (versioning, easily accessible data repository).

We have investigated several processes within that telecommunications company: software configuration management, software release management, and conducting peer-review meetings for software designs. In the following, we focus on the *design review process* and show its requirements, special needs, and system support through the teamwork services platform that we have developed in the MOTION project. The findings are based on discussions with quality engineers, software release managers, configuration managers and software tool support managers. Among the different instances of design reviews across product development, we distilled common best practices in performing the process in a multi-site dimension.

A design review process in the telecommunications company is defined in a separate handbook for mobile phone software development. It follows the SEI Software Process Definition Guide and uses the SPICE process model defined by ISO. The particular instance of a peer (design) review in that company is conducted when a work product has been created and checked to be ready for review. The design review team consists of three to six participants (usually from the same development team) each having one or more roles in the design review.

The work product may be distributed to the reviewers in advance for their individual checking prior the actual meeting. During the meeting the author(s) of the document present(s) the work product, walk(s) through it in detail and reviewers give their comments on defects, suggested changes and improvements. These findings are recorded and the work product is improved (by revision or refinement) by the responsible author(s) after the meeting. The reworked artefact is verified again. Measures and statistics are collected and stored for analyzing the review process.

The design meeting has the following goals:

- Evaluate and improve the work product.
- Find as many defects as possible.
- Consider alternative implementations/solutions.
- Educate and exchange knowledge between the review participants.
- Collect software engineering data.

2.1. Activities, processes, and roles

At the beginning, the particular peer review plan is created by the review leader and then the review meeting is conducted starting with a preparation phase and finishing with a follow-up phase after the review. For that, information concerning the ongoing project is retrieved from project archives including reference information such as checklists. Experts and all participants are selected and then the review plan is distributed to all review participants. The work product under review is also distributed to the review team optionally including known defect items. Required changes are done according to identified defects and time for re-working the artifact is recorded. The review (project) leader is responsible for creating the peer review plan and conducting the review. Software developers and other participating software reviewers act as experts for the process.

In creating the review plan, the project leader and process participants select the work products to be reviewed, identify checklists, define the standards to be used, and establish completion and (re-)review criteria. In conducting the actual peer review those reviewers that are not involved in that specific software development project act as experts for that review process. Further, due to the different locations of the enterprise and people on the move, it is often difficult to find expert reviewers allocating time and readiness for such a review process.

2.2. Requirements for a distributed product development platform

Based on the process description and the rules for carrying out the design review, we distilled particular requirements for a product development platform to effectively support distributed (software) product engineering.

For the preparation of the design review, the project leader and the software author(s) select the reviewers, designate their roles, set timetable, and invite them to a synchronous walkthrough session. These preparation activities take place a week before the actual session. Participants will get notified and are asked to respond with their availability. The review leader then stores all this information and documentation into the distributed product development platform called MOTION system (a few days before the session). Reviewers will get notified and get access to the documentation (e.g. a URL, access information, and downloadable forms and documents). Reviewers can give their comments on defects, suggested changes and improvements any time before the session and enter them into the MOTION system. A reviewer thereby should be enable to follow comments of other reviewers. It should be noted that all these activities in the preparation phase should be done asynchronously just using an information space to put together all the required and generated documents.

The holding of the session itself needs synchronous communication among the review participants. At the proposed date the review leader invites (calls) all reviewers to a synchronous session and a session chairman is assigned including someone taking minutes. For that, the MOTION system should support the leader and the rapporteur to manage their work effectively. The software author presents the material that implies a voice connection and the reviewers present their comments (defect items) that may have been earlier attached to the documentation (asynchronously) in the preparation phase or are attached during the synchronous meeting. All defect items and their originating authors have to be managed in the MOTION system. The provisioning and handling of synchronous communication is outside the MOTION platform, but interfaces for a seamless integration are provided.

For the design review follow-up the software author(s) need(s) to rework the work product based on the list of accumulated defect items. The review chairman then checks that all defect items have been integrated and corrected. For the follow-up also the time spent is recorded. The chairman checks and decides about approval of the reworked artifact. In case the result is rejected, a new meeting will be held to clear up the issues.

2.3. Services for distributed product development

Given the above activities and roles for the design review process and the requirements for a supporting software system such as MOTION, we elicited and defined the following services, which have been implanted in a prototype DMC system:

- S1: information updating.
- S2: search for expert.
- S3: contacting and inviting people.
- S3: web research (for enquiries).
- S4: asynchronous information transfer.
- S5: synchronous information transfer.
- S6: notification of availability.
- S7: discussion in a (virtual) community.
- S8: virtual (review or expert) community establishment and updating.
- S9: archive updating (community information space).

3. Architectural aspects

An architecture that supports mobility of participants and computers has to be highly flexible and adaptable to new requirements and new collaboration scenarios. In contrast to traditional software architectures, architectures that support mobility are faced with several additional difficulties: because of bandwidth restrictions, unreliable connections and disconnected operations, mechanisms and components are necessary to locate participants, synchronize data and query available resources. Depending on the location of the participant, the number of offered services may vary in quantity and quality. Sophisticated subscription mechanisms and notification services are necessary to disseminate information to the mobile participant instead of forcing the participant to find it. Furthermore, the information in mobile environments needs to conform to different standards such as WML or WAP for data representation. The varying display sizes limit the amount of information that can be displayed on a small mobile device. Thus, components are required that render the information according to the display capabilities of a certain device. High-quality wireless multimedia communications such as UMTS will further improve the quality and quantity of services on the participant's mobile device and have to be considered for a distributed and mobile collaborative (DMC) systems' architecture as well. Much of the required functionality already exists as Web applications, so the key is to efficiently migrate the features to mobile applications. The most common approach is called Web-to-wireless: this relies on existing Web services, but adds a wireless channel to them, providing stakeholders such as customers, employees, and business partners with an access point.

Mobile architectures have to integrate both fixed and mobile components. Hence, we designed the DMC architecture with the following specific design goals in mind. The architecture has to be: *open* with respect to integration of existing technologies and tools; *generic* to be deployed in organizations with varying internal organizational structures, business processes and IT infrastructures; *scalable* for different number of participants, future extensions and new requirements; and *adaptable* to restrictions imposed by mobility both of the mobile participants and the mobile devices.

3.1. Mobility

Process participants are not confined to one location, but may be on the move while working. For this, we consider three connectivity modes in our DMC infrastructure: *connected mode*, *disconnected mode*, and *ad-hoc mode*. Connected mode is used whenever (fixed) network connectivity is available. There is a single global context determined by all hosts on the network. Information can be accessed and shared from any point in the network, at any time.

Disconnected mode is an operational mode, which is required for mobile working when most of the information available in the system cannot be obtained by the occasionally disconnected mobile user. In our architecture, the disconnected mode is a special case of the connected mode and the mobile user is able to continue working (with limited functionality) even if he is disconnected. Once the mobile user is connected again, the changes he has performed locally are synchronized with the rest of the system.

In the ad-hoc mode, mobile users, for example, run a face-to-face meeting and do not have any fixed network connectivity, so they create a temporary network (e.g. in a wireless LAN without Access Point). The context for information sharing and cooperation is limited to the one provided by the parties in communication and to the resources (PDAs, laptops, etc.) they bring along. Hence, participants can build *ad-hoc* networks for collaboration and information exchange.

3.2. Information sharing in virtual project communities

We use the notion of a virtual project community (VPC) in the context of work or research, but not as places for social exchange (e.g. persistent meeting rooms). A VPC is, therefore, defined as *a persistent technological environment that supports multiple styles of interaction and multi-user engagement*. Different groups—including participants on the move—may need to interact in a variety of ways among different levels of a virtual work space.

Sharing data among members of particular VPCs is one essential requirement for a DMC architecture. Participants decide what information they want to share with the members of their community (or even across communities). An architecture for mobile users that supports such a loosely coupled information sharing would typically follow the fully distributed information sharing principles of Gnutella et al. Such a peer-to-peer communication infrastructure is especially advantageous to support different modes of connectivity. Because every computer (= peer) can work as a server as well as a client [19], it is possible to build ad-hoc networks rather easily.

3.3. Distributed searches and data delivery

To further enhance the effectiveness of virtual project communities, resources are described in meta-data that cover a description of each resource (such as documents, processes, users, communities, etc.) and enable a much more powerful search for information in an enterprise network. People can join communities based on their expertise or interest. Others can also search for information or other people's expertise and be notified whenever this is available in the system (on some peer or server peer). Meta-data is represented in XML and can be searched through, for example, XQL [26]. This allows flexibility in the definition of attributes for meta-data and provides additional extensibility for changing requirements. Artifacts themselves do not need to reside on server peers but can be located on the peer of the community member who shares his document with some community. If meta-data are additionally stored on server peers then such searches even provide results if some peers hosting the particular documents are offline. Notifications and messaging services can be used to ask community members for certain documents or process descriptions in a location independent way since members can be notified by means outside such a process aware framework (e.g. SMS).

Publish/Subscribe mechanisms allow distributing information to where it is wanted. Unlike point-to-point messaging, providers and consumers of information do not need to know about each other. Push and event-based systems are closely related. The purpose of push systems is a timely distribution of data and information to consumers whereas event-based systems focus on notification of events. Both push and event-based systems play an important role in the DMC architecture. We use a push system to notify participants based on a *profile* [23]. The participants define profiles for the kind of information they are interested in, and the push system delivers the information whenever it is available. Participants may subscribe to (and unsubscribe from) specific topics. Furthermore, participants can also subscribe to other participants and be notified whenever these participants are online (or available etc.), which is particularly interesting for mobile collaboration.

3.4. Business process support

The scalability and the distributed nature of the Web has made it a popular platform for building collaborative tools. Thus, many Web-based tools have been introduced and there are countless Web applications for improving communication, information exchange and process management. Being, for example, has been successfully using Web-based collaborative applications in the construction of its airplanes. To meet the requirements and to cover the scenarios mentioned earlier in this paper, the DMC architecture utilizes the existing Web infrastructure and exploits many of its advantages: Web access is widely available (e.g. on airports), Universal Resource Locators (URLs) are a simple and unique way of identifying resources on the Internet, and Web technologies also offer security against sniffing attacks by using secure HTTP connections through SSL.

4. Architectural components for distributed and mobile collaborative systems

Based on the recent results in software architecture research and practice [1–3,11,24] we adopt the quasi-standard terminology to describe a DMC architecture: A software architecture typically includes the description of *components*, *connectors*, and *configurations* [1]. In terms of DMC systems these terms have to be discussed from a mobile and distributed collaboration perspective.

Since such an architecture has to cope with three connectivity modes we decided to strive for a peer-to-peer (P2P) style rather than a classical client-server (CS) style. P2P facilitates ad-hoc meetings and distributed information sharing without the presence of some particular server; but it also offers ways to exploit CS structures in supporting distributed and mobile collaboration (e.g. persisting artifacts, distributing information using hierarchies of computers etc.).

Our DMC architecture has a P2P nature in cases where this is beneficial but also exploits classical CS structures where appropriate. The following descriptions will point out the respective architectural style used in a particular layer or component.

4.1. Architectural units

Before describing each of the components depicted in Fig. 4, we group them into logical units with clear responsibilities. A DMC system consists of the following three layers:

- The *Middleware Layer* provides communication means between peers and their software components; it is a communication layer that supports P2P protocols such as Gnutella or P2P architectures such as JXTA [19].
- The *Service Layer* provides the functionalities required for mobile and distributed collaboration: *Basic Services* such as Authentication and Access Control, Resource (i.e. artifact) Management, Process Composition and Configuration, Publish-Subscribe and Distributed Searches as well as *Collaboration Services* such as User and Community Management. The *Collaboration Layer* provides uniform access to all kinds of teamwork services that can be used in applications such as WfMS or Groupware in a DMC context.
- The *Application layer* offers service access and configuration facilities for business-specific services such as running a Design Review or a Production process. It includes process management to configure and instantiate particular business processes in terms of communities,

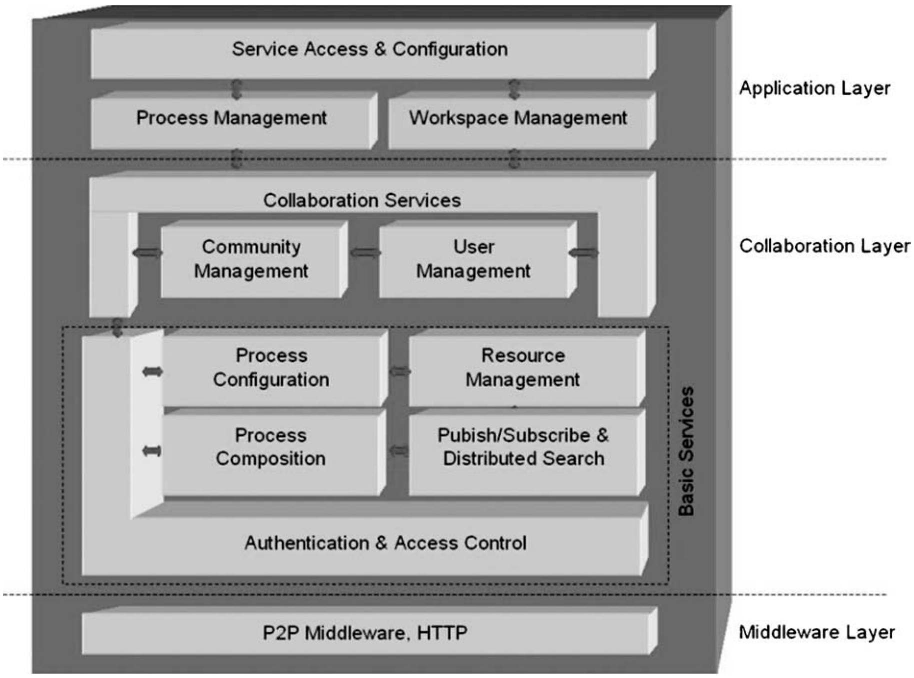

Fig. 4. DMC conceptual architecture.

processes, and workflows. Further this layer includes workspace management to assign artifacts and community spaces to project teams.

4.2. Peer-to-peer middleware

An event-based middleware is used as a *scalable* and *flexible* infrastructure to transfer messages to the participants and components [11] in the connected, disconnected and ad-hoc modes of operation. The middleware exhibits a peer-to-peer architecture (P2P) following the fully distributed information sharing principles of Gnutella et al. The peer-to-peer communication infrastructure is especially advantageous to support different modes of connectivity. Because every computer (= peer) can work as a server as well as a client, it is possible to build ad-hoc networks rather easily. The P2P middleware is the central underlying component of the system. It manages the subscriptions and provides an event-based system to notify participants on the subscribed events. It is also responsible for managing the virtual communities and for locating the actual physical location of a requested resource.

The middleware is also amendable for queuing participant actions and events that cannot be processed when working in the disconnected or adhoc networking mode. Notification messages that cannot be delivered because of an unreachable peer are queued as well. All these queued actions and events are processed as soon as the peer reconnects to the DMC platform (depending on the time-to-live for such events).

4.3. DMC architectural components

In the description of the key components of the DMC architecture we focus on the connectivity and process awareness as basis for WfMS and Groupware systems, which are denoted as *Basic Services* in Fig. 1. Users (or process participants) should be granted access via various types of devices ranging from PCs, notebooks to PDAs or mobile phones for connected, disconnected or adhoc mode.

Participants can be addressed and reached via the concept of a *community* that resembles a project group. This concept allows building communities for specific purposes and tasks as the basis for distributed and mobile collaboration of people. Both participants and artifacts are connected in communities and share their information in a peer-to-peer style.

User and Community Management includes setup and configuration of community leaders, community members and also community friends (as a more loosely coupled variant of a team member). Adding/removing participants to/from a community, giving participants specific access rights to resources etc. define the responsibilities of this component. It provides community as central abstraction to other components for addressing groups of people and sharing and exchanging information with them.

Resource Management: resources cover various kinds of artifacts required for a particular process (or process template) and can be of any MIME-type (text, audio, video, graphics etc.). Resource management also includes information about particular resources such as searches for artifacts, notification about the availability of some artifact(s) etc. In this context information about a resource includes both meta-information about artifacts and the artifact itself. So searches and subscriptions/notifications can be handled on a meta-data level more easily and efficiently for large sets of users.

Process Configuration is concerned with managing the relationships between process participants and artifacts and providing this information to other components. Process participants may be human users or software agents (i.e. components). Artifacts may be documents or other resources such as database records or applications. Such a process configuration, for example, can be that user (process participant) "Smith" requires the document artifact "invoice" in a process (or process instance) named "Sales cycle."

Process Composition is concerned with managing process models including coordination and synchronization of its sub-processes and tasks. Each process model consists of a set of tasks. The degree of granularity of process tasks can vary. On a generic level a process model (template) consists of a directed graph consisting of tasks and connection constructors such as OR and AND. On an instance level a process model consists of instantiated tasks (activities) performed by process participants (human agents or software agents).

Publish/Subscribe and Distributed Search is a component that provides loosely coupled communication among components. Its focus is on subscription to all kinds of resources (including artifacts, users, communities, processes, access rights etc.). A participant can use this functionality to declare interest; for example, in the state of a particular artifact (whenever it is changed or updated he should be notified). The same applies to users, communities, or processes. As a result this component allows notification of specific activities and can be used for process composition and configuration within or across communities.

Distributed searches are based on meta-data stored in so-called profiles. These profiles describe artifacts, users, processes, or communities in a concise way and represent it in XML. A distributed search, therefore, queries XML repositories (of different content) on each peer and—if successful—returns the requested piece(s) of information. Distributed Searches further allow querying for information that a user wants to be notified whenever it becomes available.

Distributed searches further can be used to search for experts in a particular problem domain and invite them upon availability and reachability to join a (virtual) community. This enables the exchange of expertise across communities and processes, which is especially important in mobile and distributed collaboration in large enterprises where people are on the move very often.

The *Authentication and Access Control* component consists of an access control system called Dynamic User Management System (DUMAS) [17] and a security component responsible for integrity, confidentiality and authentication. The access control system covers three responsibilities: user control, community control, and authorization.

The above Basic Services components are shielded by the Collaboration Services to provide uniform access for DMC applications. Based on this DMC Service layer any specific DMC application such as WfMS or Groupware can configure the Collaboration Services according to their specific requirements and also build new business-specific services on top of the DMC Services layer. Such a service configuration, therefore, includes the instantiation of processes (templates) and communities (including artifacts, users, and access rights) for specific tasks (e.g. holding a Design Review while process participants are on the move, in different branches of the enterprise, and/or work on various devices). For more detailed component descriptions we refer to [26].

4.4. Connectors

Connectors typically define the kind of communication that occurs between software components. The description of connectors is often enriched by information/data that is *required* and/or *provided* by a component to perform its functionality to the environment. In our case we distinguish connectors depending on the connectivity mode. When *disconnected*, participants can work in their local workspace and follow pre-defined initiated processes. Artifacts and certain process information reside on their local device enabling them to continue working while not connected to a network. Components such as user (community) management or resource management communicate via common service requests (e.g. method invocations in a JVM). Once a participant connects to the system and is in *connected* mode, he can share his work products with others in his community and can fully exploit the functionality of DMC. For this different communication protocols (i.e. connectors) between components come into operation: middleware protocols, HTTP, or RMI. The architecture utilizes existing Web technologies such as Universal Resource Locators (URLs) or secure HTTP connections through SSL. This allows widely available access to the DMC platform from various devices (ranging from Web-terminals at airports to full-fledged computers). Especially interesting in the context of distributed and mobile collaboration is the mixture of connected and disconnected working. In this case the different communication scenarios alternate depending on the network availability. The *ad-hoc mode* empowers users to quickly set up communities in situations where there is no network connectivity available or necessary. Processes can be instantiated from templates and information can be shared on a peer-to-peer basis allowing quick coordination and synchronization of tasks and easy information sharing within a community.

4.5. Configurations

Configurations of the DMC architecture depend on the specific business requirements and range from workflow and process to workspace settings. Design Reviews or Production Process Support, for example, can be configured and instantiated company-wise. This includes all relationship information regarding process participants and artifacts they use during those processes. By utilizing this information, it is possible for DMC based systems to combine features regarding flexibility, adaptability, and traceability of processes. For example in a DMC based system it is possible to support collaborative work in a flexible way since the system is "aware" of relationships between artifacts used by a participant. The system is adaptable because it supports a set of connectivity modes (from connected to ad-hoc). Virtual project communities can be instrumented in many different ways considering the requirements of the actual organizational unit, the process, and the location: some instrumentations consider the location-aware dimension, i.e. it is of particular interest where the resource actually is residing; others focus on a location-transparency in which it is important that some task is carried out but independently of where the actual resources are. Traceability is an important aspect for mobility of context; one of the design goals of DMC based systems. For distributed and mobile collaborative work it is essential to provide process state and artifact information in a location independent manner.

In terms of systems support, we distinguish between two phases: the setup phase and the operational phase. Fig. 5 depicts a sequence diagram for the setup phase by providing an in-depth analysis of the activities, actors, and artifacts during this phase. The *setup phase* consists of process composition and subsequent configuration.

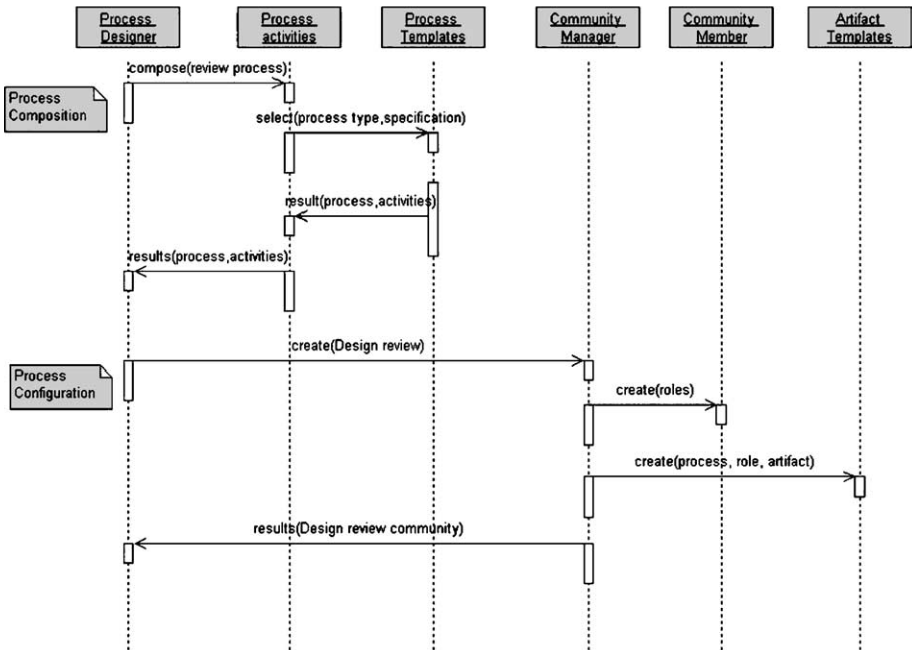

Fig. 5. Process composition and process configuration—setup phase.

During *Process Composition*, a *Process Designer* composes a review business process (Design Review Process) consisting of (pre-modeled) *Process activities*. Process type and specifications are selected and results of the selected templates are chosen from a *Process Templates* repository.

In the *Process Configuration* phase, the Process Designer configures a Design Review *Community Manager*. The Community Manager creates the required roles of the *Community Members* and creates the relationships between the previously composed process, the created roles and the artifact templates (e.g. documents, checklists, presentations, etc.). The Community Manager provides those described relationships to the Process Designer. This concludes the Process Configuration activities.

Communities act as major conceptual abstraction as depicted in Fig. 6: the *Design Review Community* provides an information sharing workspace across peers (for *Project Manager*, *Project Members 1 and 2*, and *External Expert*). The community further works as a context platform for the instantiated process of a design review and supplies the necessary infrastructure for a team (the *Design Review Team*) to jointly execute a work item. Messages and notifications as well as distributed searches (e.g. for an artifact on in "valve design") can be sent via the community to all of its community members. Once a document is published

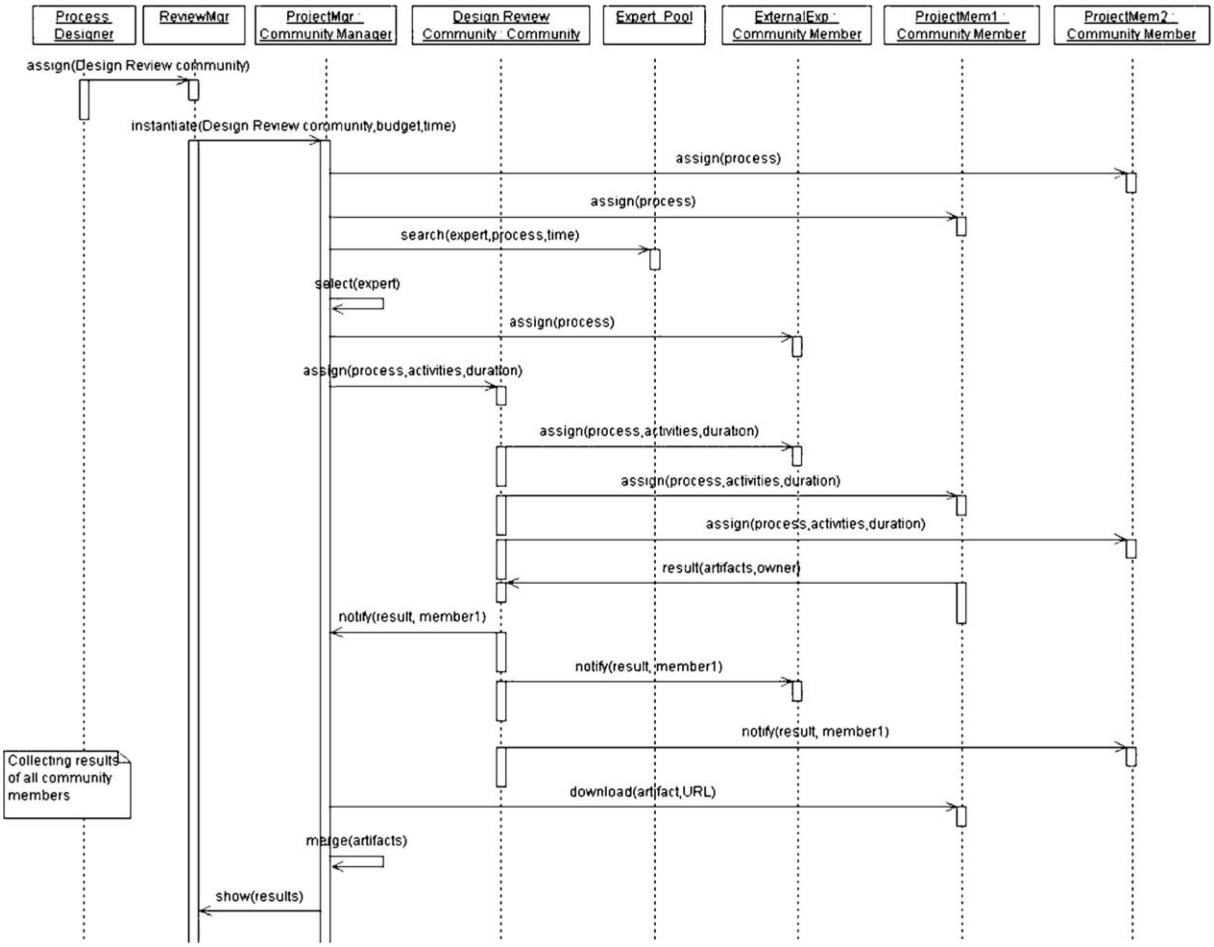

Fig. 6. Process composition and process configuration—operational phase.

by a community member as, for example, "available" or "updated" then an artifact retrieval is performed using a direct Web connection via the URL that was published (indicated as *download* (*artifact, URL*) in Fig. 6). Such retrievals can be done rather easily via the Web infrastructure.

5. Conclusions and future work

In this paper we described a three-layer software architecture for distributed and mobile collaborative (DMC) systems, which provides *mobility of context* to its group members. This architecture defines a foundation for the flexible integration of Collaborative Systems (such as Workflow Management, Groupware or Business Process Modeling) with teamwork services that support distributed and mobile collaboration. Mobility, connectivity and process configuration are based on specified teamwork services that exploit peer-to-peer principles for data sharing supported by client-server structures in contexts of persistency handling. This DMC architecture enables use cases such as information sharing and notification of availability (of resources), expert search combined with searching and inviting people for synchronous communication (e.g. chat, video/telephone conference); information retrieval about resources and their profiles (e.g. users, artifacts, processes and their meta-data), or community establishment and management. Future work includes the implementation of additional business-specific services and service configuration facilities based on our current prototype software system. Results from the ongoing end-user evaluation at the industrial partner will be used to further refine the design and integrate it into the prototype implementation.

Acknowledgements

The MOTION project was supported by the European Commission in the IST Program (IST-1999-11400). We thank the MOTION team at the Distributed Systems Group: Engin Kirda, Pascal Fenkam, Gerald Reif, and Mehdi Jazayeri.

References

[1] L. Bass, P. Clements, R. Kazman, Software Architecture in Practice, Addison-Wesley, Reading, MA and London, 1998.
[2] G.A. Bolcer, Magi: an architecture for mobile and disconnected workflow, IEEE Internet Computing May and June (2000) 46–54.
[3] J. Bosch, Design and Use of Software Architectures: Adopting and Evolving a Product-line Approach, Addison-Wesley, 2000.
[4] C. Bussler, The role of B2B protocols in inter-enterprise process execution, in: Proc. TES 2001, Computer Science Lecture Notes, Springer Verlag, 2001, pp. 16–29.
[5] C. Bussler, Enterprise-wide workflow management, IEEE Concurrency 7 (3) (1999) 32–43.
[6] F. Casati, M. Sayal, M.C. Shan, Developing e-Services for composing e-services, in: Proc. CaiSE 2001, Computer Science Lecture Notes, Springer Verlag, 2001, pp. 171–186.
[7] Q. Chen, M. Hsu, U. Dayal, Peer-to-Peer Collaborative Internet Business Servers, HP-Labs Technical Working Paper HPL-2001-14, 2001.
[8] S.J. Chinn, G.R. Madey, Temporal representation and reasoning for workflow in engineering design change review, IEEE Transactions on Engineering Management 47 (4) (2000) 485–492.
[9] N. Craven, D.E. Mahling, Goals and processes: a task basis for projects and workflows, in: Proc. COOCS International Conference, Milpitas, CA, USA, 1995.
[10] V. Christophides, R. Hull, A. Kumar, Querying and splicing of XML workflows, in: Proc. CoopIS 2001, Computer Science Lecture Notes, Springer Verlag, 2001, pp. 386–403.
[11] G. Cugola, E. Di Nitto, A. Fuggetta, The JEDI event-based infrastructure and its application to the development of the OPSS WFMS, IEEE Transactions on Software Engineering 27(9) 827–850.
[12] U. Dayal, M. Hsu, R. Ladin, Business process coordination: state of the art, trends, and open issues, in: Proc. 27th VLDB Confererence, Roma, Italy, 2001.
[13] S. Dustdar, H. Gall, Architectural concerns in distributed and mobile collaborative systems, in: Proc. 11th Euromicro Conference on Parallel Distributed and Network based Processing (PDP 2003), February, Genoa, Italy, IEEE Computer Society Press, 2003.
[14] C.A. Ellis, G.J. Nutt, Office information systems and computer science, Computing Surveys 12 (1) (1980) 27–60.
[15] C.A. Ellis, K. Keddara, G. Rozenberg, Dynamic change within workflow systems, in: Proc. COOCS International Conference, Milpitas, CA, USA, 1995.
[16] C.A. Ellis, S.J. Gibbs, G.L. Rein, Groupware: some issues and experiences, Communications of the ACM 34 (1) (1991) 39–58.
[17] P. Fenkam, DUMAS: Dynamic User Management System, Master's Thesis, Technical University of Vienna, Distributed Systems Group, June 2000.

[18] F. Flores et al., Design of systems for organizational communication, ACM Transactions on Office Information Systems 6 (2) (1988).
[19] G. Li, JXTA: a network programming environment, IEEE Internet Computing 5 (3) (2001) 88–95.
[20] Groove Networks, 2001. Available from <http://www.groove.net>.
[21] J. Grudin, Why CSCW applications fail, in: Proc. ACM CSCW'88 Conference, August 1988.
[22] A. Hausleitner, S. Dustdar, Caramba—Ein Java basiertes Multimedia Koordinationssystem, in: Silvano Maffeis, et al. (Eds.), Erfahrungen mit Java. Projekte aus Industrie und Hochschule, dPunkt-Verlag, Heidelberg, 1999.
[23] M. Hauswirth, M. Jazayeri, A component and communication model for push systems, in: Proc. ESEC/FSE 99-Joint 7th European Software Engineering Conference (ESEC) and 7th ACM SIGSOFT International Symposium on the Foundations of Software Engineering (FSE-7), Toulouse, France, September 1999, pp. 20–38.
[24] M. Jazayeri, A. Ran, F. van der Linden, Software Architecture for Product Families: Principles and Practice, Addison-Wesley, 2000.
[25] E. Kafeza, D.K.W. Chiu, I. Kafeza, View-based contracts in an e-service cross-organizational workflow environment, in: Proc. TES 2001, Computer Science Lecture Notes, Springer Verlag, 2001, pp. 74–88.
[26] E. Kirda, G. Reif, P. Fenkam, H. Gall, G.P. Picco, G.A. Cugola, Web-based peer-to-peer architecture for collaborative nomadic working, in: 10th IEEE Workshops on Enabling Technologies: Infrastructures for Collaborative Enterprises (WETICE), Boston, MA, USA, June 2001.
[27] K. Klöckner, BSCW: cooperation support for distributed workgroups, in: Proc. 10th European Workshop on Parallel and Distributed Processing PDP2001, Las Palmas, IEEE Computer Society Publications, Los Alamitos, 2001, pp. 277–282.
[28] R. Krithivisan, A.S. Helal, BizBuilder—an e-services framework targeted for Internet workflow, in: Proc. TES 2001, Computer Science Lecture Notes, Springer Verlag, 2001, pp. 89–102.
[29] P.B. Kruchten, The 4+1 view model of architecture, IEEE Software 12 (6) (1995) 4250.
[30] M.P. Papazoglou, M.A. Jeusfeld, Distributed, interoperable workflow support for electronic commerce, in: Proc. TREC 1998, Computer Science Lecture Notes, Springer Verlag, 1998, pp. 192–204.
[31] J. Puustjärvi, H. Laine, Supporting cooperative inter-organizational business transactions, in: Proc. DEXA 2001, Computer Science Lecture Notes, Springer Verlag, 2001, pp. 836–845.
[32] T. Schal, Workflow Management Systems for Process Organizations, Springer, New York, 1996.
[33] Workflow Management Coalition (WfMC), Workflow Management Specification Glossary, 1995.
[34] L. Zeng, B. Benatallah, A.H.H. Ngu, On demand business-to-business integration, in: Proc. CoopIS 2001, Computer Science Lecture Notes, Springer Verlag, 2001, pp. 403–714.

Schahram Dustdar is an Associate Professor at the Distributed Systems Group, Information Systems Institute, Vienna University of Technology (TU Wien). He received his M.Sc. (1990) and Ph.D. degrees (1992) in Business Informatics (Wirtschaftsinformatik) from the University of Linz, Austria. In April 2003 he received his Habilitation degree (Venia docendi in Angewandte Informatik) with his work on *Process-aware Collaboration Systems—Architectures and Coordination Models for Virtual Teams*. His work experience includes several years as the founding head of the Center for Informatics (ZID) at the University of Art and Industrial Design in Linz (1991–1999), Austrian project manager of the MICE EU-project (1993–1997), and director of Coordination Technologies at the Design Transfer Center in Linz (1999–2000). While on sabbatical leave he was a post-doctoral research scholar (Erwin–Schrödinger scholarship) at the London School of Economics (Information Systems Department) (1993 and 1994), and during 1998 a visiting research scientist at NTT Multimedia Communications Labs in Palo Alto, USA. Since 1999 he works as the co-founder and chief scientist of Caramba Labs Software AG (CarambaLabs.com) in Vienna, a venture capital funded software company focused on software for collaborative processes in teams. Caramba Labs was nominated for several (international and national) awards: World Technology Award in the category of Software (2001); Top-Startup companies in Austria (Cap Gemini Ernst and Young) (2002); MERCUR Innovationspreis der Wirtschaftskammer (2002). He has published over 50 scientific papers as conference-, journal-, and book-contributions. He has published three academic books as well as one professional book. His latest book (co-authored with H. Gall and M. Hauswirth) is on software architectures for distributed systems (2003), Springer-Verlag. More information is available at: http://www.infosys.tuwien.ac.at/Staff/sd

Harald Gall is an Associate Professor at the Vienna University of Technology (TU Wien). He was a senior researcher in the EU Project ARES and concerned with architecture recovery for program families. Recently, he was a senior researcher in the EU project MOTION to develop a platform for mobile teamwork support for large and distributed enterprises. His research interests are in distributed systems with focus on software architectures, reverse engineering, long-term software evolution, object-oriented technologies, and mobile computing. He has co-authored three books, Rechnernetze nach OSI (Addison-Wesley, 1993), Objektorientiertes Reverse Engineering (Springer, 1995), and Software Architekturen für Verteilte Systeme, Springer, 2003. Harald Gall received the Ernst-Denert Software Engineering Award of the German Computer Society (GI) for his work on object-oriented reverse engineering in 1994. He is a member of IEEE and the Austrian Computer Society (ÖCG). More information is available at: http://www.infosys.tuwien.ac.at/Staff/hg