IFIP Conference on E-commerce, E-business and E-Government (I3E 2003) Proc., pp. 549-559. Guarujá, SP, 21-24 September, 2003.

MODELING COORDINATION IN BUSINESS-WEBS

Alberto B. Raposo[1] Marco A. Gerosa Hugo Fuks
Computer Graphics Group (Tecgraf) *Software Eng. Lab. (LES)* *Software Eng. Lab. (LES)*
Computer Science Dept. *Computer Science Dept.* *Computer Science Dept.*
PUC-Rio – Brazil *PUC-Rio – Brazil* *PUC-Rio - Brazil*

Abstract: In the connected society an increasing part of the work conducted in companies involves others companies. Business-webs represent this tendency of putting together partners, suppliers, distributors, providers and customers in the business process. This multi-organizational collaboration, however, introduces some management difficulties. Groupware technology, which has advanced substantially in the study of workgroup coordination, has been developing concepts that may help in the development of e-business applications. This paper discusses how the principles of Groupware Engineering, more specifically the workflow and the coordination of interdependent tasks, foster e-business applications.

Key words: collaboration, groupware, workflow, business-web, software engineering

1. INTRODUCTION

A growing portion of work in companies and institutions no longer is conducted on an individual basis with a single person working alone until a job is finished. Increasingly, work is conducted on a collaborative basis. This trend is due partially to an increase in the complexity of tasks, which now

require multidisciplinary skills, and the need to involve different areas within a company working together to deliver goods and services to end-customers.

This trend toward collaborative work is noticed not only within companies, but also among different companies, which are reaggregating their values in what has been called business-webs (b-webs), one of the driving forces of the digital economy. A b-web is defined as "a distinct system of suppliers, distributors, commerce providers, infrastructure providers, and customers that use the Internet for their primary business communications and transactions" [1].

The creation of shared spaces and the exchange of information supported by groupware provides for distributed and decentralized collaborative work. The environments for sharing information and conducting commercial transactions enabled by e-business technologies (particularly by the b-webs) perfectly fit these notions of shared space and collaborative work. Therefore, e-business technologies should be developed keeping an eye on groupware achievements.

The objective of our research is the formulation of Groupware Engineering, aiming to identify the elements needed to develop collaborative applications. The collaboration model behind this approach is called the 3C model (Communication, Coordination, Cooperation), based on the work of Ellis et al. [2]. In this paper, coordination aspects of this collaboration model and their relation to e-business technologies are emphasized by means of the application of a workflow model to a b-web.

This paper is organized as follows: Section 2 presents the proposed Groupware Engineering approach and the 3C model; Section 3 shows the coordination model; Section 4 discusses how the coordination model and other concepts of Groupware Engineering may be applied to b-webs; finally, Section 5 presents the concluding remarks.

2. GROUPWARE ENGINEERING

Software Engineering, which has advanced substantially in the development of single-user applications and recently started addressing the human factor problem [3], fails to cope with the group aspects so needed in collaborative applications [4]. The formulation of Groupware Engineering, based on Software Engineering, enhanced by concepts originated from the fields of CSCW and CHI, seems suitable for developing e-business applications [5].

In order to put the groupware development cycle into context, the classic phases of software development [6] are shown in Figure 1 together with the topics that are being studied on this research project. The domain analysis

phase is elaborated on in this paper following a collaboration model based upon the concepts of communication, coordination and cooperation (Section 2.1).

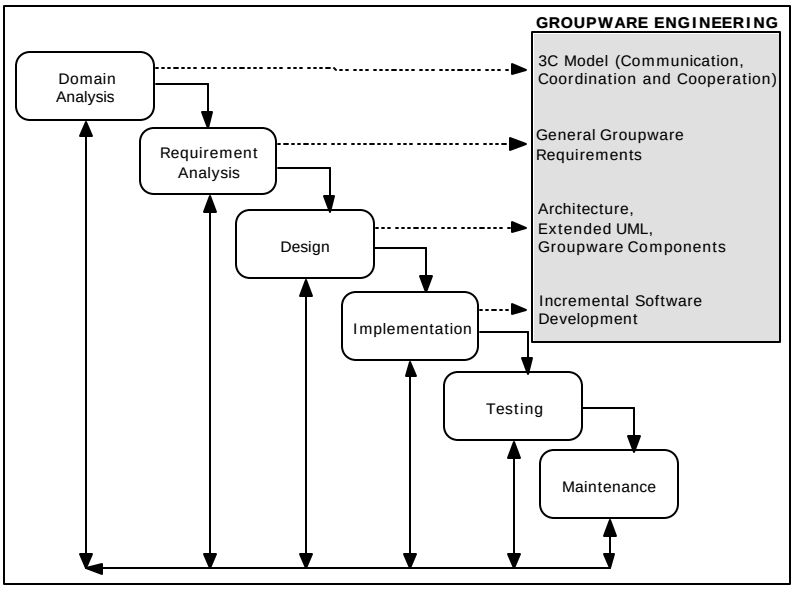

Figure 1. Classic cycle of software development in Software and Groupware Engineering

General groupware requirements are elicited in the requirement analysis phase [7]. Although very important for groupware development, these requirements seldom are clear enough to enable a precise specification of the system behavior. This is due to the fact that "we have only a sketchy knowledge of how people collaborate, and translating what we know into effective designs is difficult" [8]. For this reason, the Groupware Engineering proposal reinforces the use of design and implementation methodologies that accommodate the continuously evolutionary nature of groupware.

To provide instruments for the project phase, where the software is conceived in a manner that satisfies the requirements, toolkits and the concepts of groupware components, component architectures and UML language extensions are necessary [9], [10]. The component-based architecture is specially suited for multi-organizational systems, like the b-webs, because teams from different companies are able to develop components that assemble or increment the system.

In the implementation phase, the choice for an incremental process is natural, since collaborative systems are specially prone to failure [11] and they demand iterative evaluation during the development. This model of

implementation builds the software in small operational parts called "increments" [6], which are used for further evaluation of the requirements and fail or deficiency detection. After this cycle, a new increment is developed, normally incorporating the previous one.

2.1 The 3C Collaboration Model

Despite its advantages, collaboration requires additional effort to coordinate the members of a group. Without such coordination, much of the communication effort will not be taken advantage of during the cooperation. That is, for the group members to be able to operate together in a satisfactory manner it is necessary that the commitments that have been assumed during the participant interaction be carried out during joint work in shared space.

In order to make collaboration possible, information about what is happening is also required. This information is supplied through the awareness elements that capture and condense the information that has been collected through interaction between the participants. To become aware, in this context, is to acquire information through the senses about what is happening and what other people are doing. The diagram shown in Figure 2 summarizes the main concepts of the 3C model discussed above.

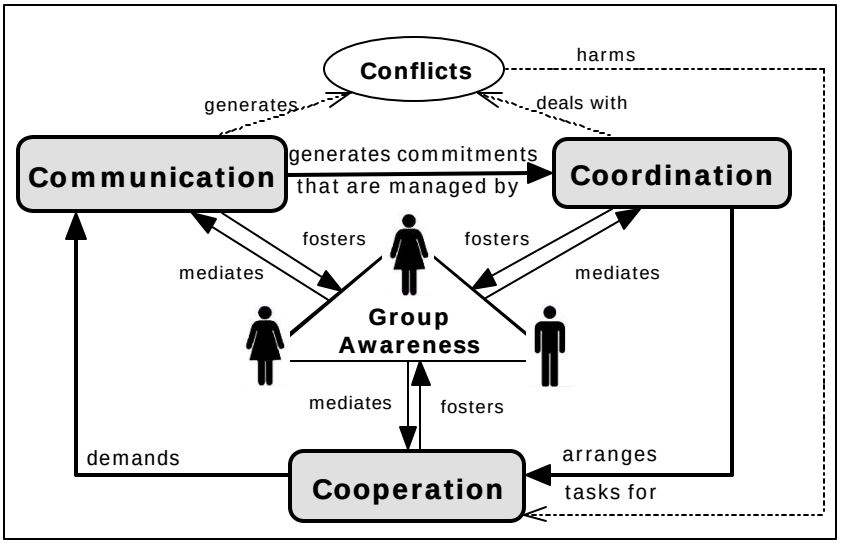

Figure 2. Overview of the 3C Collaboration Model

Despite the separation of these concepts for the purpose of analysis, it is not always possible to consider them monolithically, since they are

intimately dependent and inter-related. Details of the main elements of the diagram and their inter-relationships are presented below.

Communication

In collaboration it is important to ensure the understanding of the message in order to guarantee that the intention of the sender results in commitments assumed by the receiver, or by both. However, there is no way to check if the content that was received is equivalent to that which was sent, and if the receiver assimilated it. A communications failure thus would consist of disagreement between the intentions of the sender and the actions of the receiver who carries out the commitments.

The communication is conducted through expression elements that are available in the environment. The sender encodes the message using the available expression elements. The message is transmitted through the awareness channel. The receiver has access to the message through the awareness elements that are available in the environment.

Conversation for action generates commitments [12]. In order to ensure compliance with these commitments and the realization of collaborative work through the sum of individual labor, it is necessary to coordinate the activities. This coordination organizes the group in a manner that avoids the loss of communication and cooperation efforts and ensures that the tasks are carried out in the correct order, at the right time and in compliance with the restrictions and objectives [13].

Coordination

Some activities involving multiple individuals do not require formal planning. Activities linked to social relationships are well controlled by the so-called social protocol, characterized by the absence of explicit coordination between the tasks and by the confidence in the skills of the participants to mediate the interactions, as normally occurs during text chats and videoconferences (the coordination is culturally established and strongly dependent on mutual awareness). On the other hand, activities that are more directly aimed at work or business relationships require sophisticated coordination mechanisms in order to guarantee the success of the collaboration, as is the case of workflow systems and the b-webs.

In practice, however, it is not always clear what should be left to the social protocol and what must have a coordination mechanism associated with it. The ideal is that collaborative systems do not impose rigid standards of work or of communication. Another big challenge upon proposing coordination mechanisms for group work is to make them sufficiently flexible so they adjust to the dynamics of the interaction among the participants as well as avoid conflicts.

Conflicts may occur as a result of problems of communication or awareness or through differences in the interpretation of a situation or a subject of interest [14]. Coordination must deal with the conflicts that harm the group, such as competition[2], disorientation, problems of hierarchy, diffusion of responsibility, etc. [15].

In order for there to be coordination, awareness information is essential for transmitting changes in plans and to help carry out the commitments that have been assumed. The members of the group has to understand how the work of their partners is getting along: what was done, how it was done, what needs to be done until it is finished, what are the preliminary results, etc. [16], so as to avoid unnecessary duplication of effort during the cooperation process.

Cooperation

Communication and coordination, although vital, are not enough: "it takes shared space to create shared understandings" [17]. Cooperation is the joint operation of the members of a group within a shared space that strives to accomplish tasks that are managed through coordination. Individuals cooperate by producing, manipulating and organizing information, building and refining cooperation objects such as documents, spreadsheets, charts, etc. In order to work with these objects, the members of the group count on expression elements. Awareness elements supply information about the changes made in the shared space.

The designer of a virtual environment must anticipate what awareness information is relevant, how it can be obtained or generated, where the awareness elements are necessary, how to present them and how to give individuals control over them. Excessive information can cause overload and complicate the flow of the collaboration. To avoid overload, it is necessary to balance the need to supply information with that of preserving attention to the work.

The 3C model, elaborated during the domain analysis phase (see Figure 1), forms the basis for the following phases of Groupware Engineering. For example, the requirements are separated into communication, coordination and cooperation ones. In the implementation phase, the system may be developed over a general collaboration framework, that implements functionalities common to all services of a groupware. Coupled to this framework there are communication, coordination and cooperation frameworks. Each groupware tool (component) is coupled in one of these frameworks according to its functionality.

The next section details one of the central aspects of collaboration, which is particularly important in the e-business arena: coordination.

3. DEFINING AND MODELING COORDINATION

Coordination, in a broad sense, may be defined as the activities responsible to ensure the effectiveness of the collaborative work. In this broad sense, coordination is a synonym of what has been called articulation work, defined as "a set of activities required to manage the distributed nature of cooperative work" [18]. Among the activities of articulation work, the identification of the objectives of the group work, the mapping of these objectives into tasks, the participants' selection, the distribution of tasks among them, and the coordination (in a narrow sense, as it will be seen below) of tasks execution can be mentioned.

Coordination, in a narrow definition, is "the act of managing interdependencies between activities performed to achieve a goal" [19]. In this sense, coordination is the most important part of the articulation work because it represents the dynamic aspect of articulation, demanding renegotiation almost continuously during a collaborative effort.

Coordination, in its broad definition, is essential to any kind of collaboration. In spite of that, in its narrowest definition, coordination does not need to be explicitly implemented for the computer-supported realization of some kinds of collaborative activities, such as those related to social relations, whose activities generally are well coordinated by the valid social protocol.

On the other hand, there is a large group of activities that require sophisticated coordination mechanisms in order to be efficiently supported by computer systems. In this kind of activity, tasks depend on one another to start, to be performed, and/or to end. This kind of coordination is essential in business processes and is generally implemented by means of workflow management systems.

A workflow is a system that helps "to automate the processing of policies and procedures in an organization" [20]. More formally, the Workflow Management Coalition (WfMC) defines a workflow as "a process definition that consists of a network of activities and their relationships, criteria to indicate the start and termination of a process, and information about the individual activities, such as participants, associated IT applications and data, etc." [21]. Two important contributions of workflow technologies are the separation of the business process logic from the implementation of the activities and the connection of independent activities, allowing the migration "from islands of automation to system support for the overall business process" [22].

The increasing globalization of the world economy created the necessity of interorganizational workflows, which cross a single organization boundary, being composed of several organizations working cooperatively

(as is the case of the b-webs). The necessity of cooperation among different organizations raises many problems, which we can separate into two main classes, those related to the definition of a joint communications infrastructure and those related to the coordination of cooperative workflows.

The first class of problems deals with the integration of heterogeneous software environments in a common communications infrastructure. These problems can be roughly summarized by the difficulties of establishing an "interoperability contract" [23] among the cooperative organizations. This "contract" must establish, among other things, which workflow engines within an organization are capable of interoperating with which engines within other organizations, the transport technology, the communication protocol, security policies and exception handling. The XML (eXtensible Markup Language) is rapidly becoming the *lingua franca* for these "contracts". The WfMC is currently working on standards addressing these issues [24].

The other class of problems–those related to the coordination of cooperative workflows–appears in a higher abstraction level. Here, it is assumed that the communications environment is well defined by the partners and the major concerns are related to the coordination of interdependent activities. In order to deal with this kind of problem in single or multi-organizational workflows, we developed a coordination model, which will be presented in the following.

3.1 The Coordination Model: Task Interdependencies

In the context of this work, a collaborative activity is defined as a coordinated set of tasks realized by multiple actors in order to achieve a common goal. Thus, a task, either atomic or expressed as a group of subtasks, is one of the building blocks of any collaborative activity. A group of subtasks could be considered to be a task when it presents no external interdependencies, that is, no interdependencies with another task that does not belong to the group. This definition of task enables the modeling of collaborative activities using several abstraction levels (see Figure 3), which facilitates the coordination specification and management.

Interdependency is a key concept in the coordination theory–if there are no dependencies between tasks to be performed in a collaborative effort, there is nothing to coordinate. The approach task/interdependency is a step toward giving flexibility to coordination mechanisms, which is crucial to further use of this kind of mechanism.

One of the advantages of the separation task/interdependency is the possibility of altering coordination policies by simply altering the

coordination mechanisms for the interdependencies, without the necessity of altering the core of the collaborative system. Additionally, interdependencies and their coordination mechanisms may be reused. It is possible to characterize different kinds of interdependencies and identify the coordination mechanisms to manage them, creating a set of interdependencies and respective coordination mechanisms capable of encompassing a wide range of collaborative applications [25].

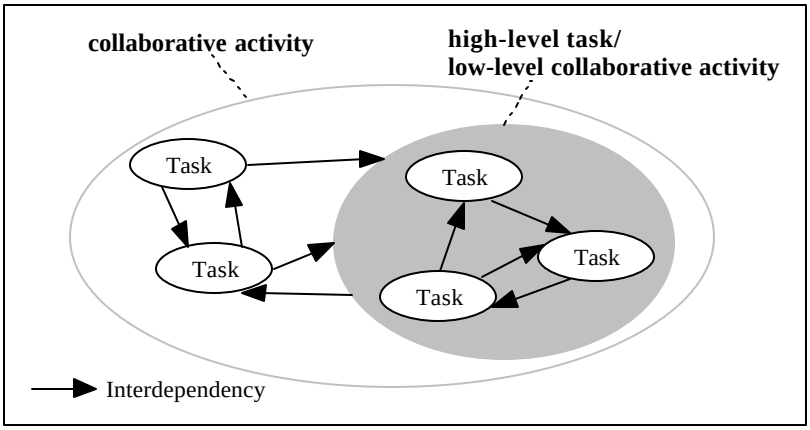

Figure 3. Hierarchical model of tasks and collaborative activities

According to the coordination model by Ellis and Wainer [26], there are two levels of coordination, one related to the activity level (temporal–the sequencing of tasks that make up an activity) and the other related to object level (resource–"how the system deals with multiple participants' sequential or simultaneous access to some set of objects").

Temporal interdependencies

Temporal interdependencies establish the relative order of execution between a pair of tasks. The set of temporal interdependencies of the proposed model is historically based on temporal relations defined by J. F. Allen [27], [13]. He proved that there is a set of seven primitive and mutually exclusive relations that could be applied over time intervals[3]. The adaptation of Allen's primitives to the context of collaborative activities takes into account that any task T will take some time (from i to f) to be performed.

However, the merely descriptive characteristic of Allen's temporal relations allows for different interpretations of a single interdependency. For example, suppose that tasks A and B are associated by the *equals* relation, one of Allen's temporal interdependencies, which establishes that both

intervals start and finish at the same instant. In this situation, what should the coordination mechanism do when task A is ready to begin, but not task B? Should it block the execution of task A until task B is ready (passive interpretation), or should it force the start of task B to guarantee that the interdependency will be respected (active interpretation)? In a different situation, if it is said that task A occurs before task B, what should be done when task B is ready but not task A? Should the coordination mechanism block task B until the end of task A, or should it allow the execution of task B, blocking future executions of task A (which would violate the relation)? For all of these reasons, it was necessary to make some adaptations to Allen's basic relations [28].

The first adaptation deals with active and passive interpretations, as discussed above, by means of two operators: *enables* and *forces*. The *enables* operator represents the passive interpretation, while *forces* represents the active one. These operations may be applied on the initial and final instants of each interdependent task. Additionally, these extreme points have two states, *ready* and *concluded*, indicating, respectively, that the task is ready to start (or finish) and that it has already started (or finished). These states are used in the first operand, indicating that it will enable or force the second operand before (ready) or after (concluded) its own execution.

Consider, for example, two tasks Ta and Tb, with initial and final points ia, ib, fa and fb. The interdependency Ta *starts* Tb, which establishes that both tasks start simultaneously, may be extended into different interpretations:

ia (ready) enables ib AND ib (ready) enables ia – this statement indicates the passive situation, in which the tasks will start their execution only when both are ready (i.e., Tb will be enabled to start only when Ta is ready to start, and vice-versa), but neither will force the execution of the other.

ia (ready) forces ib – in this situation, when Ta is ready to begin, Tb is forced to start, indicating a master/slave active interdependency (similarly, Tb could be considered the master if *ib (ready) forces ia*).

ia (ready) forces ib AND ib (ready) forces ia – active interdependency with no master (the beginning of each task will force the beginning of the other).

In spite of the operators *enables* and *forces*, there are undefined situations remaining. Such a situation occurs, for example, in Ta *before* Tb. After Ta and Tb have been finished, how should the coordination mechanism proceed if Tb wants to start again? Should it allow its execution, since Ta has already been executed (one to many relationship), or should it make Tb wait until Ta is executed again (one to one relationship)? A similar doubt arises for Ta

during Tb, i.e., how many times Ta is allowed to execute during a single execution of Tb?

In order to deal with such situations, it was necessary to include an optional parameter for the *enables* operator. This parameter indicates the number of times a condition (first operand) enables the event (second operand).

For example, to define that Ta *before* Tb allows the maximum of three executions of Tb for each execution of Ta, the following statements are used *fa (concluded) enables[3] ib*, indicating that, after each execution of Ta, Tb is allowed to execute up to three times. It is also possible to define that there is no restriction on the number of times a task may be executed after or during another (equivalent to define the parameter as infinite).

In order to enhance the flexibility of the model, it is also necessary to create the *blocks* and *unblocks* operators that, respectively disable and re-enable the execution of an event (second operand) when the state of the first operand is reached[4]. The use of these operators, for example, allows for a new interpretation of Ta *before* Tb:

ib (concluded) blocks ia – in this case, there is a restriction in the execution of Ta, which may not be executed anymore if Tb has already started its execution. There is no restriction on the execution of Tb (Tb does not have to wait for the execution of Ta, as would happen with the situation given by *fa (concluded) enables ib*).

Resource management interdependencies

Resource-related interdependencies may be represented by combinations of temporal relations. For example, if two tasks, Ta and Tb, may not use the same resource simultaneously, it is possible to define a "not parallel" dependency as the following statement, *ia (ready) blocks ib AND fa (concluded) unblocks ib AND ib (ready) blocks ia AND fb (concluded) unblocks ia*. However, besides being prone to deadlocks, this possibility ignores the notion of resource, which is quite important in the context of workflows and collaborative activities. Therefore, it is not sufficient to treat the problem of task interdependencies as a temporal logic problem. Moreover, considering resource management dependencies independently of temporal ones, a more flexible model is created, allowing the designer to deal with each kind of dependency separately.

Resource management interdependencies in the proposed model are complementary to temporal ones and may be used in parallel to them. This kind of interdependency deals with the distribution of resources among the

tasks. Three basic resource management dependencies were defined elsewhere [13].

Sharing – a limited number of resources must be shared among several tasks.
Simultaneity – a resource is available only if a certain number of tasks request it simultaneously. It represents, for instance, a machine that may only be used with more than one operator.
Volatility – indicates whether, after the use, the resource is available again. For example, a printer is a non-volatile resource, while a sheet of paper is volatile.

Each of the above interdependencies requires parameters indicating the number of resources to be shared, the number of tasks that must request a resource simultaneously and/or the number of times a resource may be used (volatility).

In the following section, we are going to show how the presented coordination model can be used to model the execution of a typical b-web, a value chain.

4. A BUSINESS-WEB EXAMPLE

The b-web typology comprises agoras, aggregations, alliances, value chains and distributive networks. Zooming into the value chain type of b-web, its main organization is called context provider, which "structure and direct the b-web network to produce a highly integrated value proposition" [1]. Other participants of this type of b-web may do everything else: manufacturing, delivering, on-site customer services, etc.

Value chains are further differentiated between routine production and shop production. While the former is product-centric and goods are designed for mass markets and production efficiency, the latter supports custom solutions where activities are not routine and are driven entirely by demand, that is, the end-customer is the one that triggers the value-creating process.

Cisco Systems is the quintessential example of shop production value chain b-web. Using Cisco's Configuration Tool on the company's web page, the end-customer receives guidance to prepare its order while all kinds of discounts and other services are being offered. Only after selling the good does Cisco make it. But, in reality, Cisco will coordinate the production process instead of actually making it.

Cisco plays the coordinator role in this shop production value chain b-web. Using the available configuration tool customers communicate their

orders, triggering the cooperation cycle among manufacturers, assemblers, distributors, component suppliers and the sales channels. The cooperation object itself is the computer (or solution) that will be shipped to the end-customer. Figure 4 illustrates how the Cisco b-web fits the 3C model.

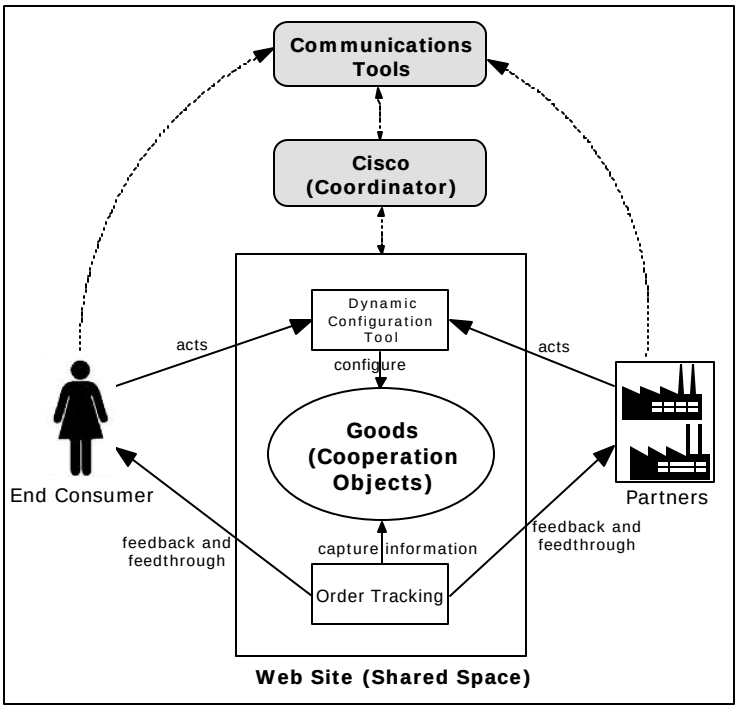

Figure 4. Collaboration model instantiated for Cisco Systems value chain b-web.

From the coordination point-of-view, it is possible to use the interdependencies presented in the previous section to model the b-web workflows. In order to illustrate that, we present an example of a simple transaction in a hypothetical value chain b-web, involving four independent participants, a customer, the context provider (like Cisco Systems), a producer, and a distributor. The workflow of this environment is represented in Figure 5, stressing aspects of the relation between the context provider and its partners. In this figure, tasks are described in the ovals and their relations in the hexagons. Dotted arrows indicate interdependencies and normal arrows, workflow transitions. The word *or* in the workflows indicates alternative paths (only one of them is followed). The absence of *or* indicates parallel paths. Letters *a* and *b* near the inter-related tasks are used to identify them in the directives inside the hexagons.

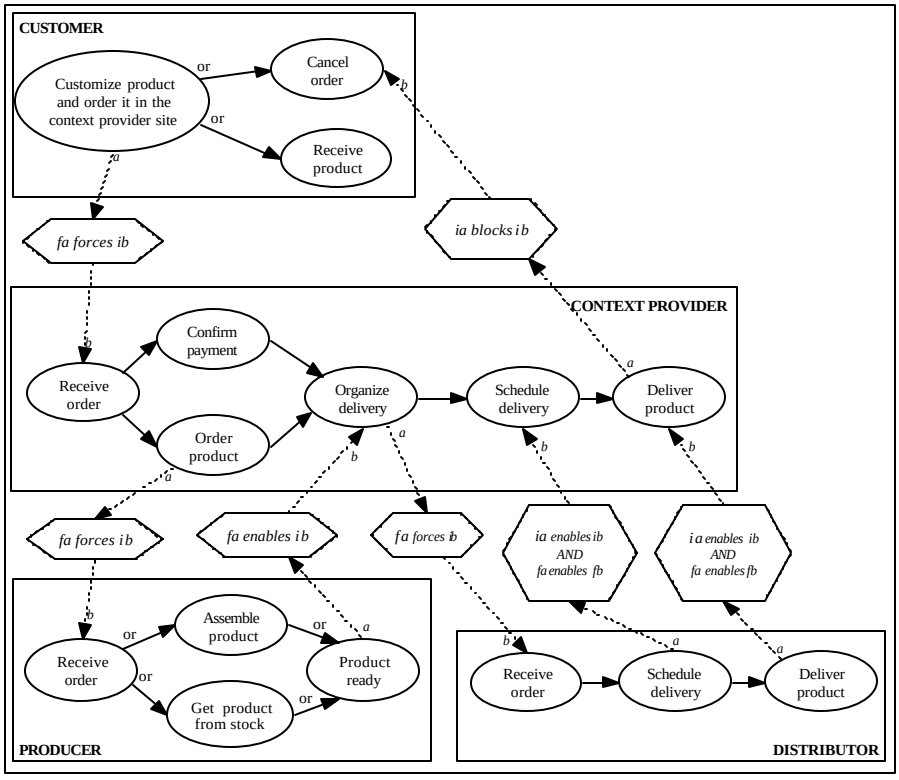

Figure 5. Workflow of a typical transaction in a value chain b-web.

In the workflow of Figure 5, the customer must start the process by customizing and ordering the product in the context provider web site. This task starts the context provider's workflow (the end of the consumer's task *forces* the beginning of the context provider's workflow). The customer then waits for the product or cancels the order. However, the order can only be cancelled if the product has not been delivered yet. This defines the relation between tasks deliver product (in the context provider workflow) and cancel order (in the customer workflow). The relation establishes that the beginning of the delivery *blocks* the order's cancellation.

The context provider, after receiving an order, starts two parallel activities: payment confirmation and contact with the producer. This contact starts the producer's workflow, in the same way the consumer started the context provider's workflow (*fa forces ib*). After the payment confirmation and the ordering, the context provider must wait for the product to be ready. This determines the relation that when the producer concludes the product, task organize delivery in the context provider's workflow is *enabled.* We do not use the *forces* operator here because it is possible that the product

becomes ready before the payment confirmation, and in this case, the delivery must wait, although enabled by the producer.

When the context provider is ready to organize the delivery, it starts the distributor's workflow, which has a close relationship to the rest of the context provider's one. This relationship is expressed by means of the *equals* relationship (*ia enables ib AND fa enables fb*), meaning that the tasks occur almost simultaneously in the distributor and in the context provider.

The above example used only temporal interdependencies between tasks, but resource management dependencies could also be used in a straightforward way. For example, the context provider could have a stock, which defines an alternative route to that of contacting the producer. The task of getting the goods from the stock would have a volatility dependency, indicating that the stock would eventually finish.

It is necessary to reinforce that the presented example represents a single part of the whole b-web model. We did not stress all details for the modeled scenario (for instance, the consequences of the consumer's cancellation, such as refund, devolution of the goods to the producer, and so on). Its main goal is to show how the coordination model may be used in a practical situation.

One of the advantages of using the coordination model to represent the workflow of b-webs is that, by means of a formal mathematical analysis (for example, using Petri Nets [13]), it is possible to anticipate and test the behavior of the interorganizational environments before their implementation. From the directives of the coordination model, it is also possible to define a direct mapping to construct the adequate software coordination mechanisms, as shown elsewhere [28].

5. CONCLUSION

At least potentially, collaboration can produce better results than individual work. In the e-business arena, more than a potential for better results, collaboration is a necessity. This necessity appears in the early stages of designing the software infrastructure, a task that usually requires collaboration among different professionals to deal with its several aspects, and remains during the e-business operations (B2B, B2C, C2B and C2C are collaborative activities, as their names clearly indicate).

B-webs are partner networks of producers, suppliers, service providers, infrastructure companies, and customers linked via digital channels. The b-web inter-enterprise way of doing business is a good representative of the need for computational support for collaboration in the workplace.

This paper focuses on the coordination aspects of Groupware Engineering, which are essential to the development of e-business

applications. It is important to reinforce, however, that coordination is just one aspect of this approach. Communication, cooperation and awareness are other aspects that must be also considered when developing useful groupware.

The coordination of interdependent tasks in interorganizational environments is a problem that should be addressed to ensure the effectiveness of the cooperation among organizations. The separation between activities and dependencies, as presented in the coordination model and the utilization of reusable coordination mechanisms are steps towards this goal.

As next steps of this research, the coordination model will be mapped to a language for the coordination specification (for example, an XML-based description of the tasks and interdependencies), and the coordination mechanisms will be implemented by software components (for example, Petri Nets may be used to model such mechanisms [28]).

ACKNOWLEDGEMENTS

The authors are financed by individual grants awarded by the Brazilian National Research Council (CNPq): Hugo Fuks nº 303055/02-2, Alberto Barbosa Raposo nº 305015/02-8 and Marco Aurélio Gerosa nº 140103/02-3.

Thanks to Prof. Carlos J. P. Lucena, Head of the Software Engineering Lab. (LES) / Catholic University of Rio (PUC-Rio), and Prof. Marcelo Gattass, Head of the Computer Graphics Group (Tecgraf) / PUC-Rio, which is a group mainly funded by Petrobras, Brazilian Oil & Gas Company.

NOTES

1. Contact author: Alberto B. Raposo (abraposo@tecgraf.puc-rio.br). Tecgraf/PUC-Rio, Rua Marquês de São Vicente, 225,Gávea, Rio de Janeiro, RJ, Brazil, 22453-900. Tel. +55 21 2512-5984.
2. Although from the groupware point-of-view competition is seen as harmful for the collaborative activity, it is sometimes a reality in e-business systems. The term *coopetition* is associated to the b-web concept, a market space in which organizations both cooperate and compete with one another [1].
3. A time interval is characterized by two events, which in turn are associated to time instants. The first event is the starting (initial) time of an interval A, denoted here ia. The other event is the ending (final) time of the same interval, denoted fa, always with ia < fa. Allen's relations between time intervals, A and B, are: equals (*ia=ib* and *fa=fb*), starts (*ia=ib* and *fa < fb*), finishes (*ia>ib* and *fa=fb*), before (*fa < ib*), meets (*fa = ib*), overlaps (*ia < ib*, *ib < fa* and *fa < fb*), during (*ia> ib* and *fa < fb*).

4. Although useful, blocking situations should be carefully used, since they could create deadlocks.

REFERENCES

1. Tapscoot, D., Ticoll, D., and Lowy, A. Digital Capital: Harnessing the Power of Business Webs. Harvard Business School Press, USA, 2000.
2. Ellis, C. A., Gibbs, S. J., and Rein, G. L. Groupware – Some Issues and Experiences. Communications of the ACM 34, 1, 38-58. 1991.
3. DeMarco, T., and Lister, T. Peopleware: Productive Projects and Teams. Dorset House Publishing, USA, 1999.
4. Grudin, J. Groupware and Social Dynamics: Eight Challenges for Developers. Communications of the ACM 37, 1, 92-105. 1994.
5. Fuks, H., Raposo, A.B., and Gerosa, M.A. Engineering Groupware for E-Business. First Seminar on Advanced Research in Electronic Business (EBR'2002), pp. 78-84. 2002.
6. Pressman, R. Software Engineering: A Practitioner's Approach. 3rd ed. McGraw-Hill, USA, 1992.
7. Schmidt, K., and Rodden, T. Putting it all Together: Requirements for a CSCW Platform. In: The Design of Computer Supported Cooperative Work and Groupware Systems, D. Shapiro, M. Tauber, and R. Traunmüller (Eds.), pp. 157-176. North Holland, Holland, 1996.
8. Gutwin, C., and Greenberg, S. The Mechanics of Collaboration: Developing Low Cost Usability Evaluation Methods for Shared Workspaces. IEEE 9th International Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE'2000), pp. 98-103. 2000.
9. Stiemerling, O., and Cremers, A.B. The EVOLVE Project: Component-Based Tailorability for CSCW Applications. AI and Society 14, 120-141. 2000.
10. Tietze, D. A. A Framework for Developing Component-based Co-operative Applications. Ph. D. Dissertation, Computer Science, Technischen Universität Darmstadt, Germany, 2001.
11. Grudin, J. Why groupware applications fail: Problems in design and evaluation. Office: Technology and People 4, 3, 245-264. 1989.
12. Winograd, T., and Flores, F. Understanding Computers and Cognition. Addison-Wesley, USA, 1987.
13. Raposo, A.B., Magalhães, L.P., and Ricarte, I.L.M. Petri Nets Based Coordination Mechanisms for Multi-Workflow Environments. International Journal of Computer Systems Science & Engineering 15, 5, 315-326. 2000.
14. Putnam, L.L., and Poole, M.S. Conflict and Negotiation, Handbook of Organizational Communication: An Interdisciplinary Perspective, pp. 549-599. Newbury Park, USA, 1987.
15. Salomon, G., and Globerson, T. When Teams do not Function the Way They Ought to, Journal of Educational Research, 13, 1, 89-100. 1989.
16. Dourish, P. and Belloti, V. Awareness and coordination in shared workspaces, Proceedings of Computer Supported Cooperative Work (CSCW'92), pp. 107-114. 1992.
17. Schrage, M. No more teams! Mastering the dynamics of creative collaboration. Currency Doubleday, USA, 1995.
18. Schmidt, K., and Bannon, L. J. Taking CSCW Seriously – Supporting Articulation Work. Computer Supported Cooperative Work – An International Journal, 1, 1-2, 7-40. 1992.

19. Malone, T. W., and Crowston, K. What is Coordination Theory and How Can It Help Design Cooperative Work Systems? Proceedings Computer Supported Cooperative Work (CSCW'90), pp. 357-370. 1990.
20. Khoshafian, S., and Buckiewicz, M. Introduction to Groupware, Workflow, and Workgroup Computing. John Wiley & Sons, USA, 1995.
21. Workflow Management Coalition. Workflow standard – Terminology & glossary. Technical Report WFMC-TC-1011, Version 2.0. 1996.
22. Klingemann, J. Goal-Based Execution of Flexible Workflows. GMD Research Series, no. 12. 2001.
23. Anderson, M. Workflow Interoperability – Enabling E-Commerce. Workflow Management Coalition White Paper. 1999.
24. Workflow Management Coalition. Workflow Standard – Interoperability Wf-XML Binding. Document Number WFMC-TC-1023. 2001.
25. Malone, T. W., and Crowston, K. The Interdisciplinary Study of Coordination. ACM Computing Surveys, 26, 1, 87-119. 1994.
26. Ellis, C. A., and Wainer, J. A Conceptual Model of Groupware. Proceedings of Computer Supported Cooperative Work (CSCW'94), pp. 79-88. 1994.
27. Allen, J. F. Towards a General Theory of Action and Time. Artificial Intelligence, 23, 123-154. 1984.
28. Raposo, A. B., and Fuks, H. Defining Task Interdependencies and Coordination Mechanisms for Collaborative Systems. In Cooperative Systems Design, M. Blay-Fornarino et al. (Eds.), pp 88-103. Frontiers in Artificial Intelligence and Applications, vol. 74, IOS Press, Holland. 2002.