Computer Supported Cooperative Work and Petri Nets

Giorgio De Michelis
Department of Information Sciences
University of Milano
Via Comelico 39
Milano 20135 Italy

Clarence A. Ellis
Department of Computer Science
University of Colorado
Boulder, CO 80309-0430

Abstract

Computer Supported Cooperative Work (CSCW) has brought to the attention of computer professionals the role computer applications may play supporting collaboration, coordination and communication among people cooperating in a common task. In particular, people acting in accordance with a structured workflow, as a procedure or as a project plan, can be supported by a class of systems called workflow management systems, not only to play their role in it, but also to increase their awareness of the situation in which they are acting so that they can make better decisions when needed and overcome breakdowns finding new (exceptional) paths. Workflow management systems, in fact, facilitate the description, modeling, analysis, enactment, and coordination of (the) structured (component of) work processes. These systems assist and mediate communication, interaction, understanding, and synchronization among collaborating people and processes within organizations.

Petri Nets have played a major role in the development of the workflow management systems technology from its very beginning, in the age of office automation, since they have immediately appeared to the pioneers in the field as a good formal and graphical language to model workflows. Their role is assuming a greater importance within a CSCW perspective where new requirements for workflow management systems have been recognized. Workflow techniques, in fact, have existed for decades but, despite progress in many areas, intelligent, industrial strength workflow systems are not well established; the models themselves are too restrictive and the systems lack flexibility, built-in intelligence, distribution, and a comprehensive theoretical foundation.

This paper takes a look at the past, present, and potential future of workflow technology and of the role of Petri Nets in it. The authors reflect upon experiences building and deploying "office information systems" at Xerox PARC during the 1970s; progress on flexible groupware systems and models during the 1980s; and the state of the art in the 1990s. This document briefly discusses ongoing research, and work that needs to be done to prevent a repetition of the past failures.

Table of Contents

1. Introduction

From its very beginning, i.e. from the publication of the Doctoral Dissertation of Carl Adam Petri (1962), General Net Theory has contributed to Computer Science not only introducing a new theoretical framework for understanding and modelling computer behaviour but also opening a new perspective on computer and information systems. The perspective opened by the research on Petri Nets moved the focus of computer science on concepts as communication, concurrency, coordination that were largely disregarded in those years.

Computer applications at private firms and public institutions have been always oriented to increasing productivity, reducing cost and enhancing performances. While traditional information systems and the first office automation systems were focused on the automation of routinized activities, the diffusion of personal computers and their interconnection in local and, later, wide area networks has shifted the attention from the automation of human activities to their support through productivity and coordination tools. The new perspective has been further developed with the emergence of Computer Supported Cooperative Work (CSCW) that has emphasized the creative, cooperative dimensions of human work, as opposed to its routinization.

In the new perspective the coordination between people cooperating in a common performance as well as their communication while performing together have become the major aspects of work practices to be supported with computer applications. Also the plans, procedures and rules shaping routinized work in any organizational setting gain a new sense from this perspective: they are, in fact, not only prescriptions to be followed while executing a task, but also resources to be used to get awareness of the situation where a person is acting and to take decisions to overcome breakdowns and failures in the normal workflow.

Carl Adam Petri himself and his co-workers, in particular Anatol Holt, spent some time trying to develop a new vision on computer systems and information systems looking at them in terms of communication and concurrency between system components and paying particular attention to their usage conditions, arriving to sketch a new approach to human communication based on that understanding.
It is not surprising, therefore, that Petri Nets play an important role, at first, in the development of Office Information Systems and later in the development of Computer Supported Cooperative Work. General Net Theory research, in fact, offers a whole set of conceptual categories and a powerful theoretical framework for modelling the coordination of human activities. Moreover, both the theoretical framework and the

conceptual categories characterizing General Net Theory have been only partially exploited by current research on Computer Supported Cooperative Work and much remains to be done in order to fully exploit their potential.

In this paper we want to survey some of the contributions General Net Theory has given, is giving and can give in the future to the development of computer systems supporting cooperative work or, in other terms, of groupware systems.

The next Section surveys the main contributions Carl Adam Petri has given to communication and coordination modelling with his General Net Theory. Particular attention is paid to Communication Disciplines.
Section 3 briefly introduces the field of research and system development called Computer Supported Cooperative Work (CSCW), surveying its main areas, and analyzing how it conceives the workflow management systems that were already introduced within Office Automation;
Section 4 surveys how Net-Based Process Modelling has been applied within the first generation of Workflow Management Systems.
Section 5 discusses the limitations that have been observed within the currently available Workflow Management Systems and the research directions trying to overcome them. It will be shown how General Net Theory can play in the future an even greater role than in the past.
Section 6 presents in some details the research projects on the application of Petri Nets to workflow modelling under development at Colorado and Milano. They can be considered as examples of the current trends of the research in the field.
The last Section presents some concluding remarks.

2. General Net Theory, Communication and Process Modelling

When looking back to the past to reconstruct how some concepts currently of general usage have been conceived and developed the risk is always present that the reconstruction is biased by the point of view of the observer. We will try to avoid the above risk, presenting the main facts of our history in strict cronological order. Adopting this style of presentation, anyhow, we may misrepresent some links connecting them as well as reduce to one the many streams of research that intertwine during these years. Moreover we are aware that our choice of the relevant facts is partial and arbitrary. Finally, relevant facts will be recalled at different levels of detail.

Summing up, in this chapter we are not writing a history of the contributions of General Net Theory to the development of Computer Supported Cooperative Work but only offering to the reader some suggestions in order to make her aware of the fact that the categories and concepts she will see used within CSCW have antique roots and stimulate her to go back to some of the main papers of those years proposing a strong vision on the issues CSCW has posed to the attention of a vast part of the research community.

The definition of the Communication Disciplines by Carl Adam Petri (1977, 1977b) can be better understood if we recall the way in which he resumes twenty years and more of debate on the role of the computer and defines his own opinion on that subject (Figure 2.1).

We can distinguish among the images of the computer those that are influenced by their current use and those that reflect the aims of the ongoing research within computer science (in particular of its most ambitious sectors, as artificial intelligence): while the former tend to under-estimate its potential, the latter do not avoid to over-estimate it. Time passing is bringing both sides of the dispute to converge progressively towards an equilibrated evaluation, that Petri guess will be the consideration of the computer as a general medium for strictly organizable information flow. The reader should not forget that Petri's guess has been written twenty years ago, when really most computer professionals and researchers were far from imagining the evolution of computer science and technology of these years.

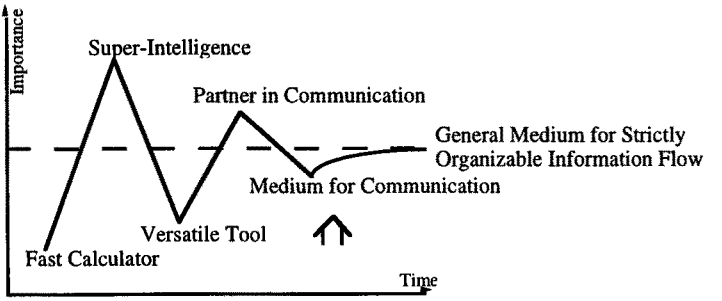


Figure 2.1

Petri's opinion on the role of the computer is strongly characterized by the emphasis he puts on communication and information flow. But, in order to exploit the intuition behind it, it is necessary to abandon the classical (naive) image of communication dominating our common sense, asserting that a communication medium has some functions (from the traditional ones –trasmitting, storing and disseminating– to the new ones originated by the computers – calculating and ordering) and that a 'good' communication medium must perform them quickly, reliably and at low cost (Figure 2.2).

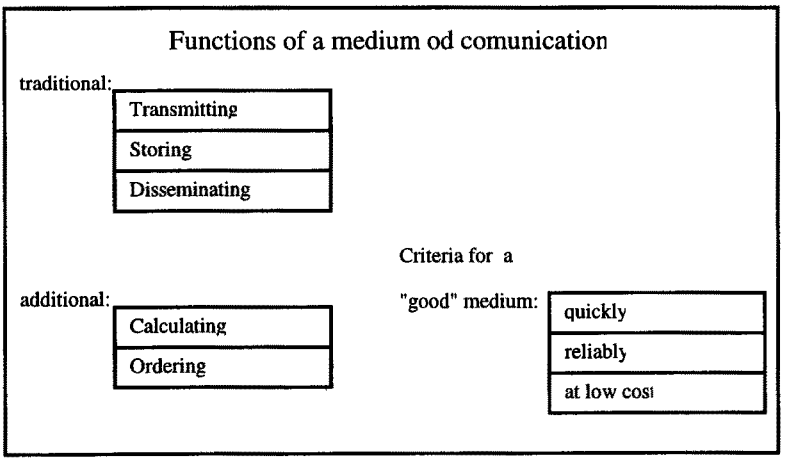


Figure 2.2

The latter, in fact, does not offer any hint to understand how the computer is changing human communications, making them possible to generate strictly organizable information flows. Organizing the information flow, or in other words managing the comunication, is today difficult and various mal-functions are frequent. If we analyze the latter, then we are provided with a long list of functions that should be performed by a 'good' communication medium. These functions are different from the traditional ones we have listed above, since they discipline the flow of the information within a network and not merely the information exchange. It is not casual that the new functions of communication media are called by Petri 'communication disciplines': they in fact are disciplines both being sub-areas of a science and restraints of behaviour.
Let us illustrate some of the 12 communication disciplines (Figure 2.3).

Functions of a medium of communication
("Communication Disciplines")

Synchronization	Identification
Addressing	Naming
Copying	Cancelling
Composition	Modelling
Authorizarion	Valuation
Delegation	Reorganization

Criterion for a "good" medium: perform these functions

Figure 2.3

Some of them can be considered as basic since they shape the basic communication phenomena in a network.
Synchronization is concerned with getting proper timing restraints for different activities. General Net Theory has based it on a partial ordering in terms of causality as opposed to ordering in terms of time.
Identification is concerned with well-known questions such as "identify the source of a letter" and with more sophisticated problems as proving the competence of agencies with regards to certain actions.
Addressing is concerned with describing routes or systems of paths through a net of channels and agencies.

Others are of higher type, since they involve people roles within the network.
Authorization is concerned with access rights, scheduling obligations and supervision rules.
Valuation is concerned with the scarcity of resources and their exchangeability. Values here do not depend on individual preferences but imply organizational restraints.
Re-organization is concerned with the rules through which a system can be changed without causing failures and/or disasters.

More on Communication Disciplines can be found in the two papers Petri has dedicated to them (Petri, 1977, 1977b) and in a paper co-authored by one of the two authors of this paper, where they are used to characterize the features of the prototype of a CSCW system (De Cindio et al., 1987b).

It is not surprising, therefore, that the work of Carl Adam Petri and his co-workers from its early years has assumed organized human activity among the cases to deal with in the development of General Net Theory. Early work on this matter has been done, besides Petri himself, by Anatol Holt (Meldman, Holt, 1971; Holt, 1979) whose interest on coordination of human acitivities dated from his previous work on system development at Univac. Holt recalls those years in his book (1997). Moreover, the Institute directed by Petri at GMD in Bonn (Germany), has also been frequently visited by various researchers interested in this subject (both the authors of this paper are among them) becoming a forum for the discussion of the problems arising with modelling organizational processes.

3. Workflow Management Systems from Office Automation to Computer Supported Cooperative Work

As anticipated in the Introduction an interesting class of tools that are "organizationally aware" are workflow management systems. Workflow systems are designed to assist groups of people in carrying out work procedures, and contain organizational knowledge of where work flows in the default case. Workflow Management Systems are defined as "systems that help organizations to specify, execute, monitor, and coordinate the flow of work items within a distributed office environment." (Bull, 1992) The system contains two basic components: the first component is the workflow model, which, on the one hand, enables administrators and analysts to define procedures and activities, analyze and simulate them, and assign them to people; on the other, allows the designers to write the program that is executed by the workflow execution module (see below) when a workflow is enacted. This component is sometimes called the "specification module"; usage of this module is typically completed before the flow of work tasks actually begins.

The second component is the workflow execution module (the workflow system) consisting of the execution interface seen by end users and the execution environment which assists in coordinating and performing the procedures and activities. It enables the units of work to flow from one user's workstation to another as the steps of a procedure are completed. Some of these steps may be executed in parallel; some executed automatically by the computer system. The execution interface is utilized for all manual steps, and typically presents forms on the electronic desktop of appropriate workers (users). The user fills in forms with the assistance of the computer system. Various databases and servers may be accessed in a programmed or ad-hoc fashion during the processing of any work step. Research in the work-flow area is challenging because these systems typically should be *dynamic* systems, *people* systems, *concurrent* systems, *semi-structured* systems, and *open* systems.

The history of workflow application in corporate America (Europe has played , in particular in early years, a minor role in the field) has been mixed; more systems have silently died than been successful (Bair, 1981). The 1960s and 1970s were the years of introduction of Office Automation Systems. The majority of these tools would not

qualify as workflow systems at all. For example, in 1964, IBM introduced its MTST magnetic tape typewriter with its cartridge storage which would allow one to edit and reprint a typed document. This was an early part of the word processor evolution which has been a significant part of the office productivity drive. Early workflow management systems (which we abbreviate to workflow systems) had no explicit workflow modeling and specification module; in fact this trait carries over to some of the workflow systems on the market today! Second generation workflow systems embed potentially complex specifications of the corporation's office procedures, detailing which procedure steps must precede which, and what data must be used in which steps. They also include a capability to perform various types of analyses and simulations of these procedures' specifications.

The 1970s were a time of wild optimism about the great beneficial effects upon productivity and effectiveness of this new technology. However, much of this optimism was unfounded. It was found that organizations succeed only if people creatively violate, augment, or circumvent the standard office procedures when appropriate. When these electronic coordinators were introduced into offices, people could no longer blatantly disobey the office procedures. In many cases, these systems led to ineffective organizations and technology rejection. Thus, the rigid systems of the 1970s tended to interfere with work routines rather than expedite them. Workflow was unsuccessful in the 1970s also because the technology was not available, because personal computers in the office were not socially accepted, because vendors were unaware of the requirements and pitfalls of group technology, and because networking was not commonly available. Technology availability and acceptance are very different today.

3.1. Workflow - The OfficeTalk Experience

The description of one of the most relevant projects ongoing in the area in those years - namely the Officetalk project, where one of the authors was heavily involved -. may be a good way to let the reader understand the direction of the research going on in the 1970s.

Officetalk-Z. This was an experimental office information system developed in the 1970s within the Office Research Group at Xerox PARC (Ellis, 1980). Officetalk was the first workflow system that provided a visual electronic desktop metaphor across an Ethernet network of end users' personal computers. It also provided a set of personal productivity tools for manipulating information, a forms paradigm, and a network environment for sharing information. It had no explicit model and no specification module; thus flow control information was embedded in officetalk code, or specified in an ad-hoc manner by the user. This system was created, evolved, and used extensively within the Xerox PARC research lab, and was also tested in selected sites outside of PARC. Goals of the Officetalk-Z project included flexibility, integration, reliability, resilience, and efficiency. A primary hypothesis was that "ordinary office workers" could effectively understand and utilize a system based upon an electronic desktop metaphor; thus there was a strong user interface emphasis. Note that at the time of construction, Personal computers and local area networks were just being invented inside of the Xerox PARC lab, and were not common (in fact unknown) to office workers. The system, which was implemented on a network of Alto computers using the BCPL language, attempted to provide an end user programming facility. This is a difficult challenge, and Officetalk was not successful in this. Thus the Officetalk

system, which was shipped without this feature, was actually named Officetalk-Z where the Z stood for version zero.

OfficeTalk-D. This extension to Officetalk graduated it from a forms manipulation system to a true second generation workflow system (Ellis, 1982). It added the Information Control Net (ICN) modeling framework and graphical language to officetalk, enabling the specification and simulation of procedures. At this evolutionary stage, the system had some knowledge of the organizational structure and procedures; also the system had a full fledged modeling and specification module. A primary hypothesis of the Officetalk-D system (The D stands for database oriented) was that ordinary users could understand and use an office information system based upon a paradigm of forms as windows onto a database. This means that if you change an item such as "customer address" on one form, then it is automatically changed on all forms. All information such as precedence among activities, roles and actors allowed to perform activities, and data repositories used by activities was stored on a database server. Novel features explored within Officetalk-D include an implementation of schedulers, observers, and alerters. The system implemented an interesting flexible activity binding capability, a distributed control mechanism, and an ADF (application design facility). For more information about Officetalk-D, see (Ellis, 1982).

OfficeTalk-P. A further, highly distributed extension to Officetalk was embodied in Officetalk-P (Ellis, 1979b), which used the implementation strategy, and the user metaphor of a system consisting of migrating processes. Thus, a user viewed her windows as intelligent forms which travelled from workstation to workstation to get their mission accomplished as specified by an ICN. Furthermore, the experimental system was constructed as forms processes, workstation processes, and overseer processes. (The P stands for process oriented.)

Backtalk Simulator. This prototype facility was built to control the network loading on individual Officetalk workstations [Nutt, 1979]. The Backtalk facility enabled one to configure a network of logical nodes to represent job stations in an office; some of the nodes were implemented by a workstation and a human user, while other nodes were a logical workstation with an algorithmic representation of how a human might execute certain classes of work. In the absence of a workflow model, Backtalk provided a facility for adding the algorithmic nodes into the network of workstations with human users, allowing one to experiment with a distributed system. For example, if a network of Officetalk workstations were to be introduced into an office, then each individual person would need to be trained how to do their work in the new automated environment; if all persons were to be trained at one time, then chaos would result, since each user is depending on other users to behave in the manner prescribed by the office procedure designer. Backtalk allowed one to automate all classes of workstations except the one being used to support the trainee; thus that person could learn how to use the system (in the absence of exceptional conditions!) without being dependent on the actions of another trainee.

With some delay with respect to the United States, some first prototypes of workflow management systems have been also developed in Europe. A citation must be done for the Domino system developed at GMD, Bonn (Germany), by Thomas Kreifelts and others (Kreifelts et al. 1984, 1991) which received much attention and has also been

transformed into a product. The Domino workflow specification module is based on a class of Petri Nets whose graphical representation has been changed in order to attach to places and transitions information about actors, intial and final actions, etc. and to gain readability.

3.2. Groupware and CSCW

During the 1980s, there emerged, mainly in the United States, an anti-workflow thrust in the form of emerging collaborative systems which were explicitly "collaboration unaware." The term groupware, originally coined by Trudy and George Lenz, gained immense popularity during the 1980s. It has been defined as technology to coordinate and enhance the collaborative efforts of groups. The academic discipline which has grown up around groupware is entitled Computer Supported Cooperative Work (CSCW). There have been many attempts to define CSCW with different orientations. One of the most general and quoted is the following: CSCW is "an endeavor to understand the nature and requirements of cooperative work with the objective of designing computer-based technologies for cooperative work arrangements" (Schmidt, Bannon, 1992). The first international CSCW conference was held in 1986 in Austin Texas, and attended by 120 people (CSCW, 1986). This series of conferences has continued to grow in attendance –European conferences intertwine with American ones from 1989– and to provide a forum for many disciplines of interested researchers and developers to interact (CSCW, 1988, 1990, 1992, 1994, 1996; Bowers, Benford, 1991; ECSCW, 1991, 1993, 1995, 1997). Unlike the office automation conferences of the 1970s, these conferences have a healthy interdisciplinary attendance. Technologists are talking to social scientists are talking to organizational designers. Emphasis of the late 1980s in groupware was focussed upon organizationally unaware (non-workflow) types of groupware. Typical examples were electronic mail, conversation handlers, shared work-spaces, desktop video conferencing, and real-time distributed group document editors.

The workflow management systems developed within office automation were considered as systems unable to cope with the complexity of human collaboration, since they reduced work practices to routinized workflows. New paradigms for the analysis of work practices were proposed and discussed: the two most influential and controversial among them are the language/action perspective (Winograd, Flores, 1986) and the situated action perspective (Suchman, 1987). While the former emphasizes the communicative dimension of cooperative processes and analyzes through speech act theory the structure of conversations embedding in them the flow of actions, the second brings forth the situatedness of human work in terms of the community of practice cooperating on a common task, of the work-space they live in, of the experience they share. It has a particular relevance in the context of this paper: the reinterpretation of plans (i.e., of workflow models) proposed by Lucy Suchman, who claims that 'plans are resources for actions'. Suchman's remark suggests that workflow models are not only useful tools to automate the coordination of activities but also cognitive artifacts supporting their users to become aware of the context in which they are situated and to decide how to deal with the breakdowns that may occur in it.

While, as said above, the new perspectives emerging in the CSCW field were radically refusing the workflow management systems developed in the eighties, in the nineties we began to see workflow issues and studies (re)appear in CSCW conferences and

journals (Kreifelts et al., 1991; Medina-Mora et al., 1992; Kreifelts et al., 1993; Abbott, Sarin, 1994; Glance et al., 1996; Dourish et al., 1996; Prinz, Kolvenbach, 1996) and a large interest is accompanying the possibility of designing workflow management systems being truely 'resources for action' within cooperative work. In most cases, therefore, this is a more interdisciplinary endeavor, and the perspective is more of a computer science / social science interdisciplinary perspective. Moreover, Computer Science areas such as distributed systems, artificial intelligence, and visual languages have much to contribute provided they are coupled with a deep understanding of the social and organizational issues.

4. Net-Based Process Modelling and Workflow Management Systems

As it has been said in Section 3, the two main components of a Workflow Management System are: the specification module (*the workflow model*) and the execution module (*the workflow enactment system*). In this section we concentrate our attention on the first one. The workflow model specifies the (partial) order to be followed while executing the activities constituting the workflow of a (business and/or work) process. It is both a program to be executed by the execution module and a cognitive artifact supporting its users (designers, administrators and actors) in their understanding of the history and the current state of the process. On the one hand, it must, therefore, be executable by the execution module; on the other, it must allow its users to analyze and evaluate the main features of the workflow (conflicts and concurrency between activities; bottlenecks in the execution; potential deadlock situations; maximal and average execution time; ...) and to change it.

As it can be seen from the pioneeristic work of Meldman and Holt (1971) and from the Ph. D. thesis of Michael Zisman (1977), Petri Nets have been one first choice for many scholars trying to define a language for modelling any sort of organizational procedures: from legal systems to office procedures. In the seventies and the eighties there have been several proposals of process modelling laguages based on Petri Nets or on Petri Net related languages: Information Control Nets (ICN) by one of the authors (Ellis, 1979), Domino by Thomas Kreifelts and others at the GMD (Kreifelts, et al., 1984), Gameru by a group of the University of Milano comprising the other author (De Cindio et al., 1987) are some well known examples of these early applications of Petri Nets in the modelling of business and/or work processes. A different discourse is necessary for the work that Anatol Holt has initiated in the same years, developing, on the basis of and in contrast with Petri Nets, Coordination Mechanics and Diplans (Holt, 1979, 1988; Holt et al., 1983). Holt work –he has quite recently completed a book presenting a complete and rich account of his ideas (1997)– deserves specific attention that goes beyond the scope of this paper. We will, anyhow, come back on it later.

The reasons for the popularity of Petri Nets among the developers of process modelling tools are quite evident: a) their *graphical representation* allows the creation of a user-friendly interface; b) their explicit representation of *concurrency and conflicts* allows an unambiguous specification of complex workflows; c) the fact that Net-based models support both *execution* and *simulation* allows to link in an effective way a powerful specification module with an efficient execution module; d) their *mathematical properties* allow (semi-)automatic model correctness checks and the verification of other interesting properties.

It has to be underlined also that the application of Petri Nets to workflow modelling has not assumed the shape of a de-facto standard because any Net-based workflow model is based on a different class of Petri Nets and, frequently, modifies it with respect to many features, from the graphical representation to the basic components and mechanisms and to the operational semantics. We have therefore Workflow Models based on Elementary Nets and 1-safe Nets –e.g., Gameru, (De Cindio et al., 1987), Milano (Agostini et al., 1997), see also below Section 6.2–; on Place-Transition Nets (eventually with inhibitory arcs and/or other non-standard features) –e.g., Business-Procedure Nets (Van der Aalst, 1995)– and finally on various classes of High-Level Nets (Coloured, Predicate-Transitions and a variety of proposals for Obiect Nets) –e.g., Workflow Analyzer (Pinci, Shapiro, 1993), Ariadne (Simone et al., 1995).
A special mention is needed for the Workflow Analyzer developed at Meta Software (Cambridge, Massachusetts) by Robert Shapiro and his co-workers (Pinci, Shapiro, 1993) on the basis of the Coloured Petri Net based tool Design-CPN (Jensen, 1992; MSC, 1992). The Worflow Analyzer represents, in fact, an example of a very effective tool for simulating work-processes, evaluating their performances with different resource distributions and with different work-loads.

Finally, we have Workflow Models based on models derived from Petri Nets as ICN (see above) and Diplans (Holt, 1988, 1997). Some words are necessary at this point about Diplans. Diplans, in fact, are much more than a workflow modelling language, since they constitute a language for modelling coordination in a very general sense. Diplans are both a derivation of Petri Nets and something radically different from them. In his book (1997) Anatol Holt explains the modelling problems which conducted him to invent a new theoretical framework for the analysis and design of human coordination and offers a large collection of examples to illustrate it. Without trying to give in these page an account of a controversy involving fundamental categories for the understanding of systems and human behaviour, we recall that Diplans have been conceived to model explicitly coordination policies and not only strictly organizable information flows and/or coordinated behaviours.

The absence of any standard Workflow Net Model is a signal of the richness and variety of approaches lying behind them and, therefore, of the immaturity of the research in the field. We can expect that time passing will reduce them to a little subset of the many existing today, but there is no reason to think a unique standard will finally emerge. The services that a uniform standard can offer to the user community are therefore to be found, on the one hand, in those efforts devoted to creating general frameworks for the analysis of organizational processes as the Process Handbook under development at MIT (Malone et al., 1993; Lee et al., 1996), on the other, on specific devices letting different workflow management systems inter-operate as middleware based tools and/or Internet extensions (WMC, 1994).

5. Rethinking Workflow Management Systems

Workflow Management Systems are, today, a hot technology. In fact, they are considered the best-suited technology to radically reengineer business processes (Hammer, Champy, 1993) improving their performances and reducing their cost (White, Fischer, 1994). However, their market share has not grown in accordance with the expectations of technology analists. Up to now, no Workflow Management System is a best-selling product, while in the Groupware arena other products have gained wide popularity (e.g. Lotus Notes™).

Many observers and experts of the field agree that this is due to the weaknesses of most of the Workflow Management Systems offered today in the market (Abbott, Sarin, 1994). On the one hand, they have serious technological limitations: they are LAN based systems dealing with difficulties with geographically dispersed users; they are weakly integratable with the existing information systems (the legacy systems) as well as with other applications; they are difficult to design, despite the many claims that users should be able to design their own workflows. On the other, they do not provide some services users need: they do not provide any mechanism for integrating procedural and non-procedural work; they do not offer any support for external activities (e.g. paper work) and meetings; they do not allow flexible authorization and access control; they are not conceived for flexible and/or evolutionary process development.

Reacting to this situation, there is in these days a great variety of ongoing research projects aiming to overcome the above listed weaknesses. It is to early now to claim which one among them, if any, will emerge as the prototype of next generation Workflow Management Systems, but they deserve some attention.
In particular we consider very interesting and promising the prototypes aiming to increase the flexibility of the workflows they manage.

A first large group of these collects the workflow management systems under development in these years enhancing the flexibility of the workflow while it is under execution: Regatta (Swenson et al., 1994), FreeFlow (Dourish et al., 1996), GPSG (Glance et al., 1996). We can consider in this group also systems as Ariadne (Simone et al., 1995; Simone, Bandini, 1997), aiming to enhance the flexibility in the design of any coordination mechanism and not only of workflows.

A second group collects the workflow management systems enhancing the flexibility of a class of enacted instances of a workflow (Ellis et al., 1994; Agostini et al., 1997). Let us offer more details on the latter group, briefly sketching the research work the two authors are, separately, developing respectively at the University of Colorado at Boulder and at the University of Milano. The two research programs share a perspective oriented to the develoment of flexible and modifiable Workflow Management Systems and their using Net-based specification modules. They can be considered indicative, although not representative, of the Net-based research on the subject.

5.1. Work at Colorado
The Boulder research group is studying models and systems that explore ideas of enhancement by incorporating concepts of goals and constraints into workflow. People are often hindered in a complex organization if they do not know the goals behind the specific tasks that they are doing. They ask "Why am I filling out this form in this manner?" Likewise a system to assist and augment these solutions must also have some awareness of the goals of the tasks. It must be able to answer the above question. Goals also provide a mechanism for synthesizing pieces of unstructured work within a structured procedure. By introducing new workflow modeling primitives, a mechanism is provided explicitly for exception handling and other activity that is not ordinarily captured by formalism.

The Boulder's philosophy is that the model and the system must co-evolve to handle the dynamic change and exception handling in a way which gives assistance (rather than hindrance) to the end user. Modeling should not be restricted to pre-execution, but should be a vehicle for continual system evolution as the organization evolves. In the midst of execution, the model can be useful for what-if scenarios, for question answering, and for making pro-active suggestions. Likewise the system should be available to the model so that pieces of a simulation can be evolved into enactment via incremental environmental relaxation.

Distributed systems issues are important within the architecture of the system and the model. The Boulder's group believes that the model should embody a language that can be used by people to discuss aspects of their work, and to instruct their workflow system when they want it to do specific tasks for them. Thus it is exploring issues of distributed scheduling, concurrency control, and visual languages. Within the editor for the model, new techniques for abstraction, for distribution, for construction, and for analyses are explored. The group leverages off of the past and current experience of its members; it builds prototypes to expand our intuition; and it validate its own work and prototypes by usage studies.

As an example of a Petri Net application in this field, we briefly describe work ongoing at the University of Colorado in the area of dynamic change within workflow systems. As organizations evolve, it is constantly necessary to implement, and adjust to change which occurs dynamically. The change is frequently time-critical, and unforeseen at the time of systems design. Dynamic change is a large and pervasive issue which surfaces within workflow systems, as well as within software engineering, manufacturing, and numerous other domains. Petri net models have recently been applied to explore and offer solutions to this ubiquitous problem.

The research at Colorado presents a formal definition of dynamic change, and a mathematical approach to its analysis. The Boulder's group stresses that this analysis is to be used interactively and synergistically, with end users mediating the social and organizational aspects of the changes. Some workflow changes are easy, some are difficult. It is typically easy to make an isolated change to the value of a variable in a database - this is considered "normal." Likewise, change of policy in many organizations is considered "normal," e.g. `Our future policy will no longer pay reimbursement for first class air travel.' These types of changes tend to be easier to implement than structural changes. If we consider a procedure as one type of structure within an organization, then change to that procedure is structural change. One company, when audited, found that they did not have sufficient separation of functional control within their procedures, and was required to make severe structural change that transcended the boundaries of many procedures. This is the type of complex change that Boulder's analysis can greatly assist.

This type of dynamic change can at times encounter "dynamic bugs" which would not appear within more static change. As an example of the type of "dynamic bug" problem that are addressed at Colorado, consider an office procedure for order processing within a typical electronics company. When a customer requests by mail, or in person, an electronic part, this is the beginning of a job (also called a work case.) A form is filled out by the order administrator; the job is sent to credit check, and then to shipping and then to billing, and then to archive. The shipping department will

actually cause the part to be sent to the customer; the billing department will see that the customer is sent a bill, and that it is paid.

Suppose that the organization decides to initiate the shipping and billing steps at the same time for speedier processing. This is an example of structural change because the structure of the procedure is changing. An even simpler structural change that can be analyzed is to move the billing step to take place before the shipping step - there could be many good reasons for wanting to do this. One way to do this change could be to delay and not process any new customer requests until after the change, and simultaneously, wait until all ongoing jobs are completed before making the change. This means that no jobs are in progress when the change is made. This strategy, called flushing the system, is safe, but quite costly - it might take years for the current jobs (perhaps thousands) to all reach completion, and this may delay thousands of new customers for an unacceptably long time. Another unpleasant strategy is to abort all jobs in progress. Another is to have the old version and the new version of the procedure simultaneously available. In fact, there are numerous variations of these strategies that are used, which have more or less safety. In the Colorado work (Keddara, et al., ?), we are concerned with making structural changes instantaneously and safely without flushing the system. This is the definition of dynamic change. In many situations, much can be gained if we can understand, and safely perform dynamic structural change. Typically, the more quickly we can convert all jobs to this change, the better.

A typical dynamic change problem occurs in the above example if the Jones work case is being processed by shipping at the time of the change. When shipping finishes with this job, it sends it to archive according to the instructions of the new procedure (since the new procedure was marked "effective immediately"). Thus Jones will not be billed for the part that he receives. If there are a large number of jobs being processed by shipping at the time of change, then a large number of customers will not be billed. This is a very simple example of a "dynamic bug;" many of these bugs are much more difficult to detect, and can have strange and insidious effects.

If correctness is defined by a set of criteria including "every job should go through shipping and billing (in any order)," then the procedure before the change is correct, and all jobs entering the system after the change will have correct behavior, but some jobs which enter the system before the change, and exit after the change may not be correct, although they strictly follow the changing procedure. Notice that this problem does not occur in static change strategies where we flush the system before change. If we view dynamic change from a programmer's perspective, it is equivalent to changing a computer program while it is running - programmers choose to stop the execution of the program and recompile while it is not executing. Because we are examining the behavior during the change, this problem is different from problems previously considered in systems reconfiguration literature, and term rewriting systems. General solutions to this are not available in the CIM and software engineering communities.

The above approach to analyzing change is mathematically detailed in publications elsewhere, and summarized in this paragraph. Given a specific dynamic procedural change, the procedure prior to the change is defined as the initial net and the procedure after the change as the final net. For each potential token configuration of the initial net, a token configuration of the final net is specified via a change mapping specified

as a graph grammar. Thus any job which is on any node of the initial net at the time of change is moved to the node(s) of the final net that are pre-specified by the change mapping. Thus every job has a new home after the change; the change can occur at any time.

Correctness is specified by a set of sequences of node labels (node labels are explained later; they may be labels such as billing, shipping, etc.) At completion, the execution of a job is said to be correct if its trace is one of the sequences in this set. It is shown that for certain primitive changes, jobs are always safe, and for other primitive changes jobs are unsafe. Given any change, we can construct one special net, called the synthesized normal form net, which contains both the initial and the final nets juxtaposed in such a way that all current jobs are mapped to their sequencing in the initial net, and all new jobs that enter the system go into the new part of the synthesized net. We show that, for any dynamic change, this synthesized change net, which can be optimized in various ways, maintains correctness.

In this example, Petri nets are an ideal tool to combine with graph grammars to explore the space of valid and correct dynamic changes. The PhD thesis of Karim Keddara is exploring this. The nets allow to give precise definitions to these terms. Furthermore, the generality of nets leads to observe that there are numerous semantics that can be applied, numerous categories of organizational structure, and numerous potential definitions of validity and correctness, which are useful and applicable to different real world situations.

5.2 .Work at Milano

In 1994 at the Cooperation Technology Laboratory one of the authors together with Alessandra Agostini, Maria Antonietta Grasso, and several students started the development of the prototype of a new CSCW system, called Milano (De Michelis, Grasso, 1994; Agostini et al., 1997). Milano is a CSCW platform supporting its users while performing within cooperative processes (De Michelis, 1995, 1996, 1997). Milano is based on a situated language-action perspective and, therefore, it offers to its users support to keep themselves aware of the history they share with the other actors of a cooperative process. It offers them a set of tools strictly integrated with each other to create with them that history; in particular, a multimedia conversation handler and a workflow management system. Without adding more details about the other components of Milano (the interested reader can find a more complete account on it in (Agostini et al., 1997)), let us spend some more words on its workflow management system and, in particular on its specification module.

The Milano workflow management system is a new generation workflow management system: its aim is to support not only its users while performing in accordance with the procedure described in its model, but also when they either need to follow an exceptional path or when they need to change the workflow model. The workflow model, therefore, within Milano is not only an executable code, but also a cognitive artifact. It is, in fact, an important part of the knowledge its different users (the activator of a workflow instance, the performer of an activity within it, the supervisor of the process where it is enacted and, finally, the designer of the workflow model) share while performing within a cooperative process.

The model, therefore, must not only support the execution of several workflow instances, but it must also support the enactment of any model change on all the ongoing instances (dynamic changes). On the other hand, its cognitive nature requires that a workflow model supports all its users to understand their situation, to make decisions, to perform effectively. The workflow model is not merely a program to be executed and/or simulated by the execution module with a graphical interface to make it readable by its users. Rather, it is a formal model whose properties allow the user to get different representations of the workflow, to compute exceptional paths from the standard behaviour, to verify if a change in the model is correct with respect to a given criterion and to enact safely a change on the ongoing instances.

For this reason, the specification module of the Milano workflow management system is based on the theory of Elementary Net Systems (Rozenberg, 1987; Thiagarajan, 1987). Let us call ENS the class of Elementary Net Systems. ENS, in fact, has some nice mathematical properties (it is possible to compute and classify forward- and backward-rolls linking their states; there is a synthesis algorithm from Elementary Transition Systems (ETS) to ENS (Nielsen et al., 1992); the morphisms in ENS (ETS) preserve some important behavioural properties) that appear suitable to get the above services. Moreover, since Milano is based on the idea that workflows must be as simple as possible, its workflow models constitute a small subcategory of ENS, namely Free-Choice Acyclic Elementary Net Systems, whose main properties are computable in polynomial time, allowing an efficient realization of the specification module.

Let us introduce, in the following, the main definitions and facts about the workflow models of Milano and let us illustrate them with a small example. To avoid repetitions, we refer, for the main definitons on Elementary Net Systems and Elementary Transition Systems, to the contributions of Grzegorz Rozenberg and Philippe Darondeau in this volume.

As anticipated above, the specification module offers two different representations of a workflow model: the first one, called Workflow Net-Model, is based on Elementary Net Systems, while the second one, called Workflow Sequential-Model, is based on Elementary Transition Systems.

Definition 5.1 - Workflow Net Model
A Workflow Net-Model is an Elementary Net System, $\Sigma=(B,E,F,c_{in})$, such that the following hold:
a) Σ is structurally acyclic (there are not cycles in the graph);
b) Σ is extended Free-Choice (all conflicts are free).

The class of Workflow Net Models is called WNM.

Example 5.2
In Figure 5.1 it is presented the Workflow Net Model representing a simple order processing procedure (Ellis, Keddara, 1993).

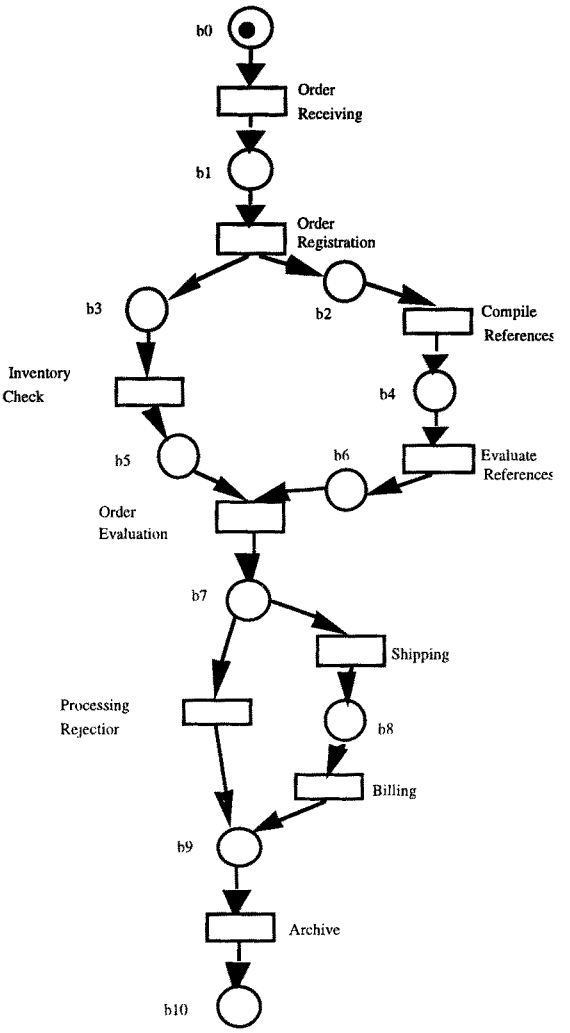


Figure 5.1

Definition 5.3 - Workflow Sequential Model
A Workflow Sequential Model is an Elementary Transition System $A=(S,E,T,s_{in})$, such that the following hold:
a) A is acyclic (there are not cycles in the graph);
b) A is well structured (all diamonds have no holes and the transitions with the same name are parallel lines in a diamond).
The class of Workflow Sequential Models is called WSM.

Example 5. 4
Figure 5.2 presents the Workflow Sequential Model of the Order Processing Procedure introduced in Example 5.1 (Figure 5.1)

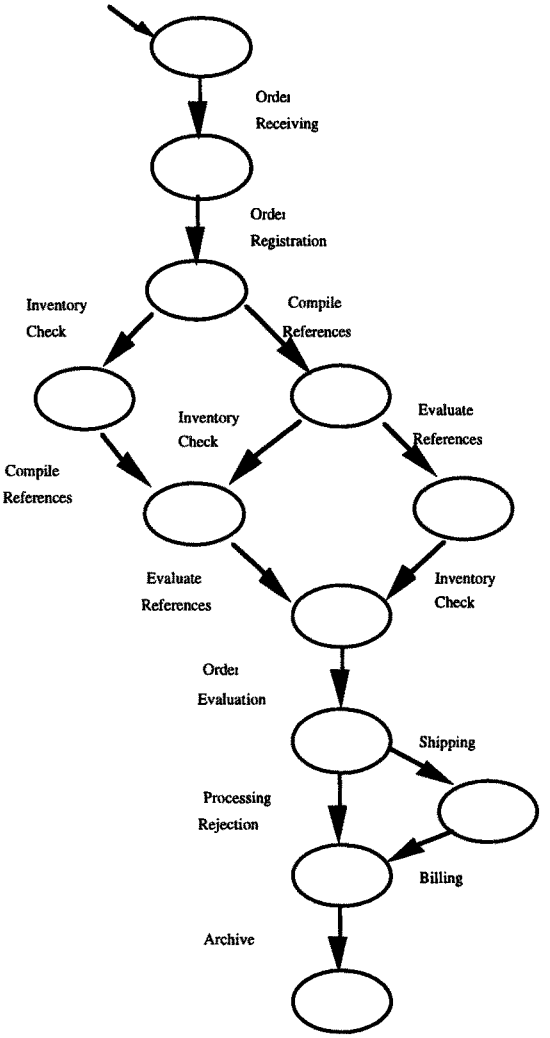


Figure 5.2

While the Workflow Net Model (Figure 5.1) is a local state representation making explicit, for example, the independence between the action of 'Inventory Check' and both 'Compile References' and 'Evaluate References', the Workflow Sequential Model (Figure 5.2) is a global state representation, where the path followed during the execution of an instance is made immediately visible.

It is well known that the sequential behaviour of an ENS can be represented as an ETS and, conversely, given an ETS it is possible to synthetize an ENS whose sequential behaviour is equivalent to the source ETS (Nielsen et al., 1992). It is easy to show that the above relation between ENS and ETS restricts itself to a relation between WNM and WSM. The algorithm to build the ENS corresponding to ETS is based on the computation of Regions (subsets of S uniformly traversed by action names). While

the algorithm presented in (Nielsen et al., 1992) generates a saturated ENS, having a place for each region of the source ETS, Luca Bernardinello (1993) has introduced a synthesis algorithm generating an ENS having a place for each Minimal Region of the source ETS, that is not a minimal representation of an ENS having the behaviour described in the source ETS but has some nice properties (e.g., it is contact-free and state-machine decomposable) making it very readable and well structured. We have therefore decided to normalize each WNM to its Minimal Regional representation and to associate to each WSM its minimal regional representation.

Fact 5.5
The sequential behaviour of a WNM can be represented as a WSM and conversely, given a WSM there is a WNM whose sequential behaviour is equivalent to it.

Proof outline
The proof is based on the fact that the sequential behaviour of an acyclic extended free-choice Elementary Net System is acyclic and well structured and, conversely, the (minimal) Regions of an acyclic well structured Elementary Transition System are such that the corresponding Elementary Net System is both acyclic and extended free-choice.

The synthesis algorithm for ENS has been proved to be NP-complete (Badouel et al., 1997), making it impossible to use it in real applications. The strong constraints imposed to WNM allow a rather efficient computation of Minimal Regions, so that it is usable in the specification module of the Milano Workflow Management System. Let us sketch the algorithm for the computation of the Minimal Regions of a WNM.

Algorithm 5.6
Let $A=(S,E,T, s_{in})$ be a Workflow Sequential Model. The following algorithm computes the minimal regions of A.

begin
$C := \{(S - \{ s_{in}\},\{ s_{in}\})\}$; $R := \emptyset$;
while $C \neq \emptyset$
  do $C := C - (S',r)$ with S' maximal;
  $E_r := \{e \mid \exists s \in S', e \text{ exits } s\}$;
  $E'_r := \{e \mid e \in E_r \text{ and } \exists s \in S' - r; e \text{ exits } s\}$
    if $E_r = \emptyset$ then $R := R \cup \{r\}$
      else if $E'_r = \emptyset$, then $R := R \cup \{r\}$;
        $C := C \cup \{(S'',r') \mid \exists e \in E_r, r' = \{s \mid e \text{ enters } s\}$
        and $S'' = \{s \mid s \in S' - (r \cup r')$ and s reachable
        from a state of $r'\}$
        else $C := C \cup \{(S'',r') \mid \exists e \in E'_r, r' = r \cup$
        $\{s \mid e \text{ exits } s\}$ and $S'' = S' - r'\}$
      fi
    fi
  od
end.

Example 5.7
Figure 5. 3 labels each state of a WSM with the Minimal Regions containing it. It is not difficult to see that the WNM of Figure 5.1 has a place for each of its Regions (it is therefore the result of the synthesis algorithm applied to the WSM of Figure 5.3) and that the WSM of Figure 5.2 is isomorphic to it.

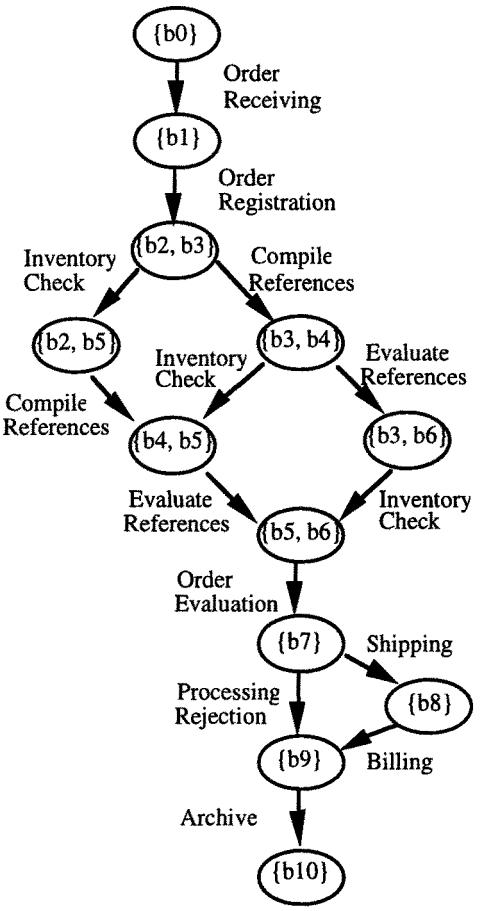


Figure 5.3

Fact 5.8
The algorithm given above is polynomial in the size of A (of its set of States, S).

Proof outline
The number of elements we can put in C lies between |S| and $2.\sqrt{|S|}$. Moreover each step of the algorithm requires at most one observation of each element of S.

The efficiency of Algorithm 5.6 grants that the switch between the two representations of a workflow model (namely WNM and WSM) can be computed whenever necessary,

so that there are no constraints imposing a particular representation to the user. The problems related to the graphical visualization of the two representations (e.g., multi-dimensional diamonds will appear as intricate and difficult to read graphs) are not considered in this context. They are taken into account within the framework of a system for the visualization of graph-based models (Bertolazzi et al., 1995).

The reader may object that the constraints imposed to WNM (WSM) are too strong so that the actors are forced to follow very rigid prescriptions. This is not true, since the actors, whenever they can not act in accordance with the model, can jump (either forward or backward) to another state from which execution can progress again. The freedom in the choice of the states that may be reached through jumps is not constrained by the model but can be constrained in accordance with the rules of the organization where the workflow is modelled. The actors are supported in the choice of an authorized jump by the possibility of computing and classifying composed paths in the graph.

Without entering into irrelevant technical details, let us present a simple example where it is assumed that the organization allows two different classes of jumps: *strongly linear jumps* (moving in the WNM only one token) not requiring any type of authorization and *weakly linear jumps* (cancelling two or more tokens and writing one token in the WNM) requiring the authorization of the process initiator, i.e. of the person responsible for the execution of the procedure.

Example 5.9
Let an instance of the order processing procedure presented in Figures 5.1, 5.3 be in the state $\{b_2, b_5\}$ (Figure 5.4, a). Then the allowed strongly linear jumps move the process either to the state $\{b_2, b_3\}$ or to the state $\{b_5, b_6\}$ (Figure 5.4, b).

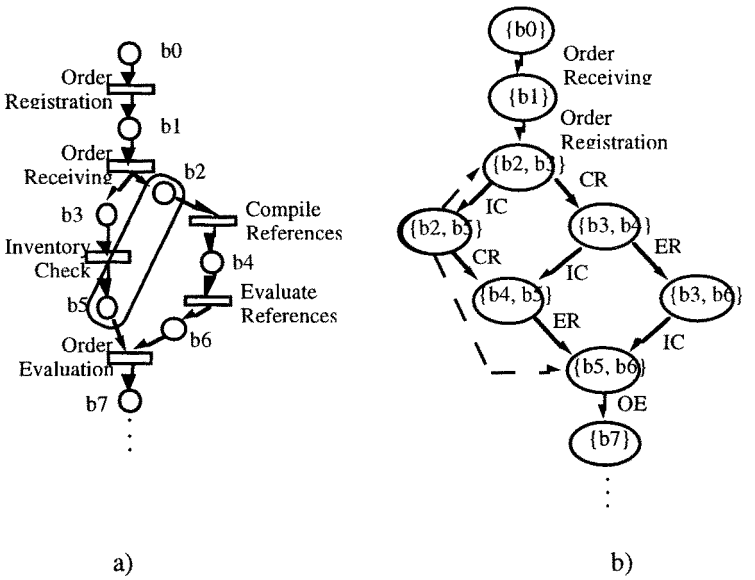


Figure 5.4

From the same state $\{b_2, b_5\}$ (Figure 5.5, a) weakly linear jumps may move the process to the following states: $\{b_0\}$, $\{b_1\}$, $\{b_7\}$ (Figure 5.5, b).

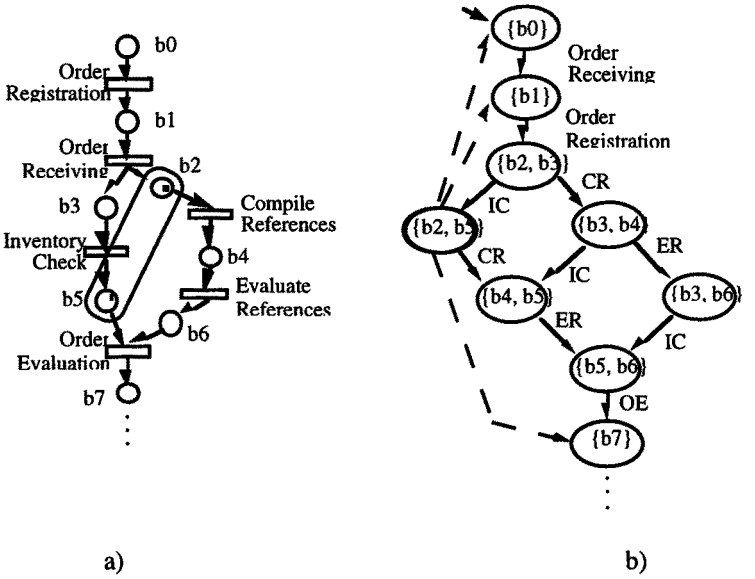


Figure 5.5

The modelling framework constituted by the couple (WNM, WSM) is therefore offering various services to its various categories of users. Actors, initiators, administrators and designers can choose between WNM and WSM to have the most effective visualization of the workflow model with respect to their current interest; actors and initiators can analyze the context in which a breakdown occurs choosing how to solve it.

Administrators and/or designers receive from the above modelling framework also some relevant services with respect to their responsibility on the model and on its changes. They can, in fact, define a minimal critical specification (see Definition 5.10, below) that must be satisfied by the adopted workflow model and by all its changes, using it as a reference to guide changes. In this case the theoretical framework supports them with the automatic verification of the correctness of changes, that is based on the properties of the morphisms between WNMs (WSMs). Moreover, the framework allows them to enact the change on all the already ongoing instances of the workflow, moving to the new model all the instances that are in a safe state and postponing the enactment of the change of the instances that are in an unsafe state until they reach a safe one (for the definition of safe and unsafe states see Definition 5.11, below).

These services are based on the fact that the class constituted by a minimal critical specification together with all the workflows that are correct with respect to it is closed under the morphisms induced by the action-labels, and on the distinction between safe and unsafe states with respect to a given change given by the composition of morphisms and inverse morphisms (being the morphisms induced by E injective and total in WSM (WNM) they always admit inverse).

Let us explain the above claim with some simple examples, that assume that any workflow model must have the same set of action labels as its minimal critical specification and that only changes not modifying the set of action labels are allowed.

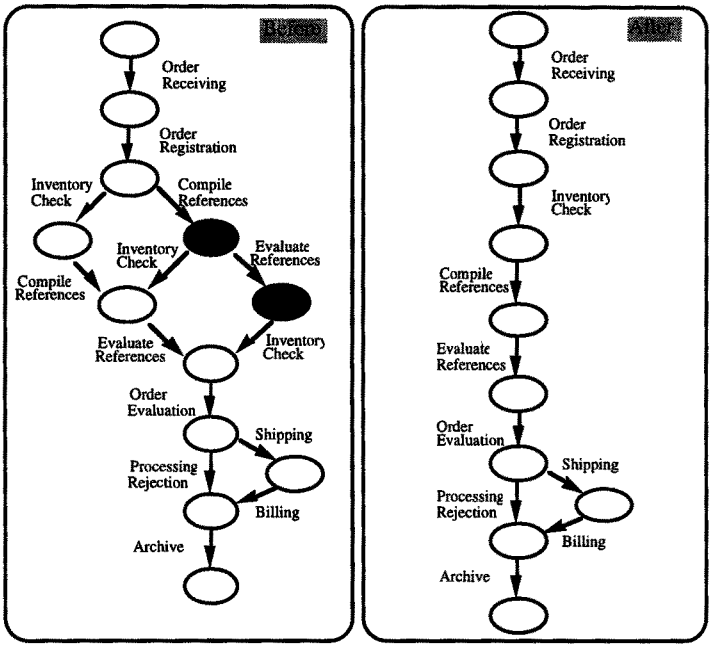
Definition 5.10 - Minimal critical specification
A WSM, A = (S, E, T, s_{in}), is correct with respect to a minimal critical specification MCS = (S', E, T', s_{in}') if and only if the morphism induced by E, g:S —> S', is injective and total.

As its name evokes and its definition grants, a minimal critical specification is less constraining than any workflow model correct with respect to it, i.e. it admits a larger class of behaviours. Whenever no minimal critical specification is given, it can be assumed that the n-dimensional diamond representing the sequential behaviours of the workflow where all the n actions labels are concurrent is the implicit minimal critical specification to be taken into account.

Definition 5.11 - Unsafe states with respect to a change
Let A=(S, E, T, s_{in}) be a WSM and A'=(S', E, T', s_{in}') be the a WSM being the effect of a change on it. Let both, A and A', be correct with respect to the minimal critical specification, MCS = (S", E, T", s_{in}"). Let, finally, g: S—>S" and g':S'—>S" be, respectively, their morphisms on MCS induced by E: then S - $g^{-1}(g'(S'))$ is the set of *unsafe* states of A with respect to the given change. If a state is not unsafe with respect to a change, then it is *safe* with respect to it.

S - $g^{-1}(g'(S'))$ contains all the states not having an image in S', and therefore moving an instance being in one of them to the changed model is impossible since we can not find univocally the state in which it will be after the change. Moreover, any choice we do for it, does not allow a correct completion of the process.

Example 5.12
Let the WSM of Figure 5.6, b, be the effect of a change of the WSM of Figure 5.6, a. Then the two shaded states of the first WSM are its only unsafe states with respect to the given change.

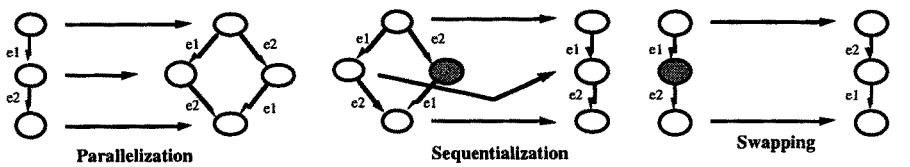


a) b)

Figure 5.6

Example 5. 13

Figure 5.7 presents the three patterns of change allowed by our theoretical framework: *parallelization*, making two sequential action labels concurrent (Figure 5.7, a); *sequentialization*, creating a sequence with two concurrent action labels (Figure 5.7, b); *swapping*, inverting the order of two sequential action labels (Figure 5.7, c). The shaded states represent unsafe states.



a) b) c)

Figure 5.7

The class of changes introduced above is quite small. An extension of the allowed changes may be obtained weakening the condition that the minimal critical specification contains all the action labels of any workflow model correct with respect to it, to the one imposing only that its action labels are contained in the set of action labels of any workflow model correct with respect to it.

Finally, a precise definition of action-label refinement within the above theoretical framework will further extend the class of changes supported by the specification module of the Milano workflow management system.

6. Conclusion

Observing the successes and failures of today's workflow systems suggests a research agenda needed to successfully realize the next generation. Workflow systems must be open systems and must have high interoperability capabilities. These requirements are being more and more addressed by some of the aggressive workflow vendors. The asynchronous groupware paradigm needs to be merged with the real time groupware paradigm, so that the workflow system can connect distributed groups for decision making and assist joint problem solving. Studies have shown that a large amount of time is spent on exception handling and problem solving and fire fighting. Research is needed to develop truly helpful systems that enhance rather than impede people's unstructured work capabilities and habits.

Two promising technologies for the future are agent [Veloso, 1997] technologies (both autonomous agents and dependent user agents), and full immersion virtual reality (VR). Using this technology, end users immerse themselves in VR environments, and then they are able to simultaneously view and manipulate shared data, artifacts, and group context (Nutt, 1997). Certain organizational problems can be alleviated using virtual conference rooms, intelligent autonomous critics and other agents, and subservient workflow technologies. The large scale architectures that support this vision of the virtual corporation of the future have not yet emerged. These systems must be user driven and dynamically changeable and evolutionary. A user must be able to move seamlessly from a single user environment to a multi-user distributed environment easily and at will. Much research needs to be done in distributed systems, social and organizational modeling, high level secure network protocols, and distributed object oriented large scale technology, among many other areas, before the above vision can be truly realized.

Opportunity exists for a leap forward in productivity, effectiveness, and satisfaction when workflow systems successfully incorporate and utilize knowledge of goals, constraints, and the social and organizational context into which they are embedded. Structured procedural work frequently has unstructured components. The mechanisms to help people do their necessary problem solving and exception handling are not available in today's work-flow systems. Several research institutions are researching basic workflow issues which must be addressed for this vision to become a reality.

The set of issues and problems are challenging, but approachable. The time seems appropriate for this work because many contemporary organizations employ personal computers, workstations, and networks; also many are expressing a strong interest in workflow (Dyson, 1992). The Gartner group has predicted that workflow will be one of the primary areas of organizational productivity enhancement in the 1990s, and will mature around the year 2000 if some significant inhibitors can be overcome (Leung, 1992).

In summary, workflow systems consist of modeling components and enactment systems components. The models must enable expression of goals, temporal

constraints, dynamic change, and exception handling. The systems must enable execution of dynamic, goal-based models; provide coordination and assistance to users at each opportunity; and take advantage of distributed computer systems technology. The method to be employed for future enhancement of workflow must include theoretical framework creations, prototyping efforts, methodology development, studies of work in organizational settings, and learning by deployment of systems.

7. Acknowledgments

This work was partially supported by NSF grant #IRI-9307619 and by EC within the Esprit Project DESARTE #21870.
Giorgio De Michelis has been lecturing about CSCW and Petri Nets at the Advanced Course in Petri Nets at Dagstuhl (Germany) in Fall 1996 and within the International Conference on Application and Theory of Petri Nets at Toulouse (France) in Summer 1997. His contribution to this paper is based on the course materials he prepared for those occasions.
The authors thank Alessandra Agostini and Luca Bernadinello for their careful reading of some earlier versions of the manuscript.

8. References

[Abbott, Sarin, 1994] Abbott, K. R., Sarin, S. K. Experiences with Workflow Management: Issues for The Next Generation. In: [CSCW, 1994],pp. 113-120.

[Agostini et al., 1994] Agostini, A., De Michelis, G., Grasso, M. A., Patriarca, S. *Reengineering a business process with an innovative Workflow Management System: a Case Study.* Collaborative Computing, 1.3,1994, pp.163-190.

[Agostini et al., 1994b] Agostini, A., De Michelis, G., Patriarca, S., Tinini, R. *A Prototype of an Integrated Coordination Support System.* Computer Supported Cooperative Work. An International Journal, 2.4, 1994, pp. 209-238.

[Agostini et al., 1997] Agostini, A., De Michelis, G., Grasso, M. A. Rethinking CSCW systems: the architecture of Milano. In: [ECSCW, 1997], pp. 33-48.

[Badouel et al., 1995] Badouel, E., Bernardinello, L., Darondeau, P. *The synthesis problem for Elementary Net Systems is NP-Complete.* Theoretical Computer Science, 1997 (to appear).

[Bair, 1981] Bair, J. Office Automation Systems: Why some work and others fail. In. *Proceedings of the Stanford Office Automation Conference*, Stanford University Center for Information Technology, June 1981.

[Bernardinello, 1993] Bernardinello, L. Synthesis of Net Systems. In: *Application and Theory of Petri Nets*, LNCS 691, Springer Veralg, Berlin, 1993, pp. 89-105.

[Bertolazzi et al., 1995] Bertolazzi, P., Di Battista, G., Liotta G. *Parametric Graph Drawing.* IEEE Trans.on Software Engineering, 1995.

[Bowers, Bernford, 1991] Bowers, J., Bernford, S. (Eds.) *Studies in Computer Supported Cooperative Work.* North Holland. Amsterdam, 1991.

[Bowers et al., 1995] Bowers, J., Button, G., Sharrock W. Workflow from Within and Without: Technology and Cooperative Work on the Print Industry Shopfloor. In: [ECSCW, 1995], 1995, pp. 51-66.

[Brauer et al., 1987] Brauer, W., Reisig, W., Rozenberg, G. (Eds.) *Petri Nets: Central Models and Their Properties.* LNCS 254, Springer Verlag, Berlin, 1987.

[Brown, Duguid, 1991] Brown, J. S., Duguid, P. *Organizational Learning and Communities of Practice: a unified View of Working, Learning and Innovation.* Organization Science, 2.1, 1991, pp. 40-56.

[Bullen, Bennett 1990] Bullen, C. V., Bennett J. L. Learning from User Experience with Groupware. In: [CSCW, 1990], pp. 291-302.
[Bull, 1992] Bull L. P. M. *FlowWorks, The Bull Workflow Product - Architectural Design and Functional Specifications.* Bull, Paris, 1982.
[CSCW, 1986] *Proceedings of the Computer Supported Cooperative Work Conference.* MCC, Austin, 1986.
[CSCW, 1988] *Proceedings of the 2nd Computer Supported Cooperative Work Conference.* ACM, New York, 1988.
[CSCW, 1990] *Proceedings of the 3rd Computer Supported Cooperative Work Conference.* ACM, New York, 1990.
[CSCW, 1992] *Proceedings of the 4th Computer Supported Cooperative Work Conference.* ACM, New York, 1992.
[CSCW, 1994] *Proceedings of the 5th Computer Supported Cooperative Work Conference.* ACM, New York, 1994.
[CSCW, 1996] *Proceedings of the 6th Computer Supported Cooperative Work Conference.* ACM, New York, 1996.
[Curtis et al., 1992] Curtis, B., Kellner, M.I., Over J. *Process Modelling.* Communications of the ACM, 35.9, 1992., pp. 75-90
[De Cindio et al., 1987] De Cindio, F., De Michelis, G., Simone, C. GAMERU, a language for the analysis and design of human communication pragmatics within organizational systems, In: *Advances in Petri Nets 87*, Springer Verlag, Berlin, 1987, pp. 21-44.
[De Cindio et al., 1987b] De Cindio, F., De Michelis, G., Simone, C. The Communication Disciplines of Chaos. In: K. Voss, H. J. Genrich, G. Rozenberg (Eds.), *Concurrency and Nets*, Springer, Berlin, 1987, pp. 115-140.
[De Michelis, 1995] De Michelis, G. Computer Support for Cooperative Work: Computers between Users and Social Complexity. In: [Zucchermaglio et al., 1995], pp. 307-330.
[De Michelis, 1996] De Michelis, G. Work Processes, Organizational Structures and Cooperation Supports: Managing Complexity. In: D. Brandt, T. Martin (Eds.) *Automated Systems Based on Human Skills,* Pergamon, New York, 1996, pp.3-12. Also in: *Annual Reviews in Control,* Pergamon, New York, 1997 (to appear).
[De Michelis, Grasso 1994] De Michelis, G., Grasso, M. A. Situating conversations within the language/action perspective: the Milan conversation Model. In: [CSCW, 1994], pp. 89-100.
[Dourish et al., 1996] Dourish, P., Holmes, J., Mc Lean, A., Marqvardsen, P., Zbyslaw A. Freeflow: Mediating Between Representation and Action in Workflow Systems. In: [CSCW, 1996], pp. 190-198.
[ECSCW, 1991] *Proceedings of ECSCW'91.* Kluver, Dordrecht, 1991.
[ECSCW, 1993] *Proceedings of ECSCW'93.* Kluver, Dordrecht, 1993.
[ECSCW, 1995] *Proceedings of ECSCW'95.* Kluver, Dordrecht, 1995.
[ECSCW, 1997] *Proceedings of ECSCW'97.* Kluver, Dordrecht, 1997.
[Ellis, 1979] Ellis, C. Information control nets: a mathematical model of office information flow. In: *Proc. of the 1979 ACM Conf. on simulation, measurement and modeling of computer systems,* ACM Press, New York, 1979.
[Ellis, 1982] Ellis, C. Office Talk-D: An Experimental Office Information System. In: *Proceeding of the 1st Conference on Office Information Systems.* ACM Press, New York, 1982, pp. 131-140.
[Ellis et al., 1979] Ellis, C., Gibbons, R., Morris, R. Office Streamlining. In: N. Naffah (Ed.) *Integrated Office Systems-Burotics.* North-Holland, Amsterdam, 1979, pp. 111-125.

[Ellis et al., 1991] Ellis, C., Gibbs, S. J., Rein, G. L. *Groupware: some issues and experiences.* Communications of the ACM, 34,1, 1991, pp. 39-58.
[Ellis et al., 1995] Ellis, C, Keddara, K., Rozenberg, G. Dynamic Change within Workflow Systems. In: *Proceedings of the Conference on Organizational Computing Systems.* ACM Press, New York, 1995, pp. 10-21.
[Ellis, Keddara, 1993] Ellis, C., Keddara, K. *Dynamic Change within Workflow Systems.* University of Colorado Technical Report, July 1993.
[Glance et al., 1996] Glance, N., Pagani, D. S., Pareschi, R. Generalized Process Structure Grammars (GPSG) for Flexible Representations of Work. In: [CSCW, 1996], pp. 180-189.
[Hammer, Champy, 1993] Hammer, M., Champy, J. *Reengineering the Corporation,* Harper Business, New York, 1993.
[Holt, 1979] Holt,. A. W. Net Models of Organizational systems in Theory and Practice, In: C. A. Petri (Ed.), *Ansaetze zur Organisationstheorie Rechnergestuerze Informations-systeme*, Oldenbourg, Muenchen, 1979, pp. 39-62.
[Holt, 1988] Holt, A. W. *Diplans: A new language for the study and implementation of coordination.* ACM Trans. Office Information Systems, 6.2, 1988, pp. 109-125.
[Holt, 1997] Holt, A. W. *Organized Activity, and its Support by Computer.* Kluwer, Dordrecht, 1997.
[Holt et al., 1983] Holt, A. W., Ramsey, H. R., Grimes, G. D. *Coordination system technology as the basis for a programming environment.* Electrical Communication 77.4, 1983, pp. 307-313.
[Jensen, 1992] Jensen, K. *Coloured Petri Nets.* Springer Verlag, Berlin, 1992.
[Kreifelts et al., 1984] Kreifelts, T., Licht, O., Seuffert, P, Woetzel, G. DOMINO a system for the specification and automation of cooperative office processes. In: *Proc. EUROMICRO'84*, North Holland, Amsterdam, 1984, pp. 33-41.
[Kreifelts et al., 1991] Kreifelts, T., Hinrichs, E., Klein, K.H., Seuffert, P., Woetzel, G. Experiences with the DOMINO Office Procedure System. In: [ECSCW, 1991], pp. 117-130.
[Kreifelts et al., 1993] Kreifelts, T., Hinrichs, E., Woetzel, G. Sharing To-do Lists with a Distributed Time Manager. In: [ECSCW, 1993], pp. 31-46.
[Lee et al., 1996] Lee, J., Yost, G. and the PIF Working Group. *The PIF Process Interchange Format and Framework.* University of Hawaii, Tech. Report, 1996.
[Malone, Crowston, 1990] Malone, T. W., Crowston, K. What is coordination theory and how can it help design cooperative work systems? In: [CSCW, 1990], pp. 357-370.
[Malone et al., 1993] Malone, T. W., Crowston, K., Lee, J., Pentland, B. Tools for Inventing Organizations: Towards a handbook of organizational processes. In: *Proc. of 2nd IEEE Workshop on Enabling Technologies Infrastructure for Collaborative Enterprises*, New York, IEEE, 1993.
[Medina Mora et al., 1992] Medina-Mora, R., Winograd, T., Flores, R., Flores, F. The Action Workflow Approach to Workflow Management Technology. In: [CSCW, 1992], pp. 281-297.
[Meldman, Holt, 1971] Meldman, J. A., Holt, A. W. *Petri nets and legal systems.* Jurimetrics Journal, 12.2, 1971, pp. 65-75.
[MSC, 1992] Meta Software Corporation, *Design/CPN User's Manual.* MSC, Cambridge, 1992.
[Nielsen et al., 1992] Nielsen, M., Rozenberg, G., Thiagarajan, P.S. *Elementary Transition Systems.* Theoretical Computer Science, 96/1, 1992.
[Nutt, 1997] Nutt, G. Resource Management for a Virtual Planning Room. In; *1997 Interantional Workshop on Multimedia Information Systems*, Como, 1997, pp. 17-27.

[Petri, 1962] Petri, C. A. *Kommunikation mit Automaten*, Rheinisch-Westfaelisches Institut fuer Instrumentelle Mathematik and der Universitaet Bonn, Schrift Nr. 2, 1962. Also: *Communication with Automata*, Griffiss Air Force Base, New York, RADC-TR-65-377, Vol. 1, Suppl. 1, 1966 (English Translation).

[Petri, 1977] Petri, C. A., Communication Disciplines. In: B. Shaw (Ed.), *Computing System Design. Proc. of the Joint IBM University of Newcastle upon Tyne Seminar*, Sep. 1976, University of Newcastle upon Tyne, 1977, pp. 171-183.

[Petri, 1977b] Petri, C. A., Modelling as a Communication Discipline. In: H. Beilner, E. Gelenbe (Eds.), *Measuring, Modelling and Evaluating Computer Systems*, North Holland, Amsterdam, 1977, pp. 435-449.

[Pinci, Shapiro, 1993] Pinci, V. O., Shapiro, R. M. *Work Flow Analysis*. MSC, Cambridge, 1993

[Prinz, Kolvenbach, 1996] Prinz, W., Kolvenbach, S. Support for Workflows in a Ministerial Environment. In: [CSCW, 1996], pp. 199-208.

[Rozenberg, 1987] Rozenberg, G. Behaviour of Elementary Net Systems. In: [Brauer et al. 1987], pp. 60-94.

[Schmidt, Bannon, 1992] Schmidt, K., Bannon, L. *Taking CSCW Seriously: Supporting Articulation Work*. Computer Supported Cooperative Work. An International Journal, 1.1-2, 1992, pp. 7-40.

[Simone et al., 1994] Simone, C., Divitini, M., Schmidt, K. A notation for malleable and interoperable coordination mechanisms for CSCW systems. In: N. Comstock et al. (Eds.) *COOCS'95. Conf. on Organizational Computing Systems*, ACM Press, New York, 1995, pp. 44-54.

[Simone, Bandini, 1997] Simone, C., Bandini, S. Compositional features for promoting awareness within and across cooperative applications. In *Group'97*, ACM Press, New York, 1997 (to appear).

[Strong, 1988] Strong, D.M. *Design and Evaluation of Information Handling Processes*. Ph.D. Dissertation, Carnegie Mellon University, School of Business, June 1988.

[Suchman, 1987] Suchman, L. A. *Plans and Situated Actions. The Problem of Human-Machine Communication*. Cambridge University Press, Cambridge, 1987.

[Thiagarajan, 1987] Thiagarajan, P. S. Elementary Net Systems. In: [Brauer et al. 1987], pp. 26-59.

[Van der Aalst, 1995] Van der Aalst, W. P. M. *A class of Petri Nets for modeling and analyzing business processes*. CS-TR 95/26, Eindhoven University of Technology, Eindhoven, 1995.

[Veloso, 1997] Veloso, M. (Ed.) *Proceedings of the ProTem/SNF Workshop on Intelligent Agents*, Porto Allegre, 1997.

[White, Fischer, 1994] White, T. E., Fischer, L. (Eds.) *The Workflow Paradigm*, Future Strategies, Alameda, 1994.

[Winograd, Flores, 1986] Winograd, T., Flores, F. *Understanding Computers and Cognition*. Ablex, Norwood, 1986.

[WMC, 1994] Workflow Management Coalition, *Coalition Overview*. TR-WMC, Brussels, 1994

[Zisman, 1977] Zisman, M. D. *Representation, Specification and Automation of Office Procedures*, PhD Thesis, University of Pennsylvania, The Wharton School, Philadelphia, 1977.

[Zucchermaglio et al., 1995] Zucchermaglio, C., Bagnara, S., Stucky, S. (Eds.) *Organizational Learning and Technological Change*. Springer Verlag, Berlin, 1995.