Designing effective workflow management processes

Eindhoven University of Technology
Department of Mathematics and Computing Science

Designing Effective Workflow Management Processes

by

M.C.A. van de Graaf and G.J. Houben

96/22

ISSN 0926-4515

Reports are available at:
http://www.win.tue.nl/win/cs

Computing Science Reports 96/22
Eindhoven, December 1996

Designing Effective Workflow Management Processes

M.C.A. van de Graaf
G.J. Houben

Eindhoven University of Technology[1]

Determining how a workflow management system supports the control of your business processes is not a one-time issue. It is an on-going process of adapting your controlling actions to changed circumstances. As a manager you can share the responsibility for this dynamic management process with your workflow management system. This report considers a number of aspects that address this shared responsibility and suggests, based on the experience of the authors, how effective workflow management processes are designed. By doing so, this report establishes a context for further research into the area of the design of workflow applications.

1. Introduction

Every organisation, whether it is an industrial organisation like a factory or it is an office organisation like a bank, an insurance company or a government office, has operational (primary and secondary) processes that are essential for the organisation's existence. For example in a bank one can expect to find a primary process that covers issuing loans, while an insurance company will probably have a primary process dedicated to handling claims made by their clients. One of the functions of an organisation's management is to control these operational processes. First, the management is responsible for a proper design of the operational processes. After the design, it must see to it that the operational processes are executed in such a way that the organisation's goals are achieved. The processes that support the management in this control function are called the control processes. In situations where the operational work processes involve the manipulation of information and thus the use of information systems, the control processes can be partially automated. This implies that a part of the support of the control function is realised by an information system. The approach called workflow management (WFM) aims at supporting management in the control function. It does so by using an information system to control the

[1] Authors address: Eindhoven University of Technology, Group Information Systems, PO Box 513, 5600 MB Eindhoven, the Netherlands; phone: (31) (40) 247 27 33; fax (31) (40) 246 39 92.

operational processes. In particular, a workflow management system (WFMS) is an information system that helps in practically controlling operational processes which are executed using information systems. While both industrial and office organisations need to control their operational processes, we focus in this paper on office organisations. Offices have operational processes for which the WFM approach is best suited, since the primary function of an office is to create and manipulate information. One therefore sees a focus of WFMSs to support office organisations.

As an organisation is a dynamic entity living in a rapidly changing environment, the control of its operational processes is highly dynamic itself. It is commonly understood that it is necessary for the management of an organisation to continuously adapt the operational processes to events happening in the processes' environment. However, the changing environment does not only have its effect on the operational processes, but also on the way in which the management executes the control of those processes: it has an effect on the control processes. Therefore, the control function and the control processes implementing this function, must be highly flexible too. Figure 1 shows the relation between the controlling and controlled processes embedded in the environment, based on [Mantz et al., 1990].

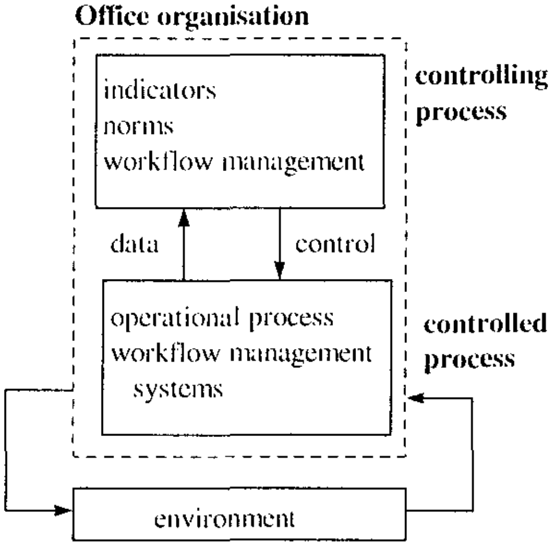

Figure 1: Controlling and controlled processes.

In order to efficiently implement this flexibility, we suggest that the controlling (management) processes should be considered as part of the entire process, in the same way as the controlled (operational) processes are considered. The flexibility of the entire process, consisting of controlled and controlling processes, acts as a measure for the flexibility of doing business: a business process should not be measured only by the quality of the operational processes. The controlling (control) processes build important tools for the management to run its business: the flexibility of the controlling processes reflect the competitive decisiveness of the business organisation.

When we consider current WFMSs, we can see that they do support the (controlled) operational processes. They are however designed for stable and structured processes, and they have proven to be suitable solutions for operational processes that are highly structured, in which large numbers of cases are handled and that remain unchanged during a long period of time. The success stories of WFM often relate to back-office processes for which the optimal process layout has been established. The consequence is that "traditionally" WFMSs are mainly used for production processes that can effectively and efficiently be specified in detail and in which the control function is stable enough to be executed by a machine. As state of the art WFMSs help management by providing the ingredients to implement the control processes, we exploit possibilities to enhance this function with delegated managerial tasks to automatically adjust to new situations. This means that we want the necessary flexibility to be an integral part of the control function.

Another aspect in which current WFMSs do not acknowledge the real needs of supporting office work processes, is the use of implicit information shared among office workers. For office workers the use of distributed implicit information is essential: it makes that the theoretically sound concept of exchanging work and documents can operate in practice. In typical front-office environments the operational work processes can be less structured and well-defined, and additional knowledge and information is needed by the workers to effectively perform their tasks. We coin this additional knowledge and information "implicit information": we mean by this that the information is implicitly and informally used in the process and therefore often not explicitly and formally specified.
Traditionally, a WFMS is a system that just exchanges explicit information between participants in the operational process. An ideal WFMS should be able to do more, i.e. it should be able to provide for more than a pure information logistics function. Thus the WFMS should be able to act as the connecting "glue" between the tasks of the operational process which are executed by the office workers. In this respect we suggest that WFMSs should have more of the capabilities of the tools meant to support workgroups. For example, modern groupware applications better support the use by office workers of the information that is implicitly part of the office work. A typical groupware application builds an information repository. Office workers know that all relevant information is available in that repository. Thus the repository can effectively serve the organisation's 'background memory'.

The two aspects of flexibility and implicit information are strongly related. Changes in the operational work processes imply by definition changes in the implicit information. Implicit information plays a significant role in controlling processes, as implicit information reflects the knowledge, experience and other qualities of the workers. In order to obtain and maintain efficient operational and control processes, one must not limit the attention to the information explicitly specified to be part of the actual work. Often in a WFMS the implicit information is left out of the scope and a high degree of structure and stability of the processes is assumed. Consequently, current WFMSs are not ideally suited to help an organisation in adapting to its dynamic environment. In contrast, modern groupware applications are easier adapted to new situations because they do not inhibit a high degree of structure, like a typical WFMS does. They therefore better address the flexibility and the need to extend the information involved in the organisation's process (both structured and unstructured). In this report we want to clarify the importance of these two aspects of the design of workflow management processes.

The consequence of the approach that we propose in this report is that if WFMSs are to be the tools of management for the future, they must be enhanced to derive the control over the operational processes as a function of its environment. This report gives an overview of the interaction between controlling (management) processes and controlled (operational) processes. We show the relationship between these processes: at a first level we can distinguish the elements that make up an operational process, at a second level these elements are composed into processes (currently controlled by WFMSs) and at a third level one can consider the meta-process of changing processes by adjusting them to match altered environments. By discussing the control parameters available in many of the state of the art WFMSs we clarify their insufficiencies in effectively controlling the business: most WFMSs only support the first two levels of operational work processes.

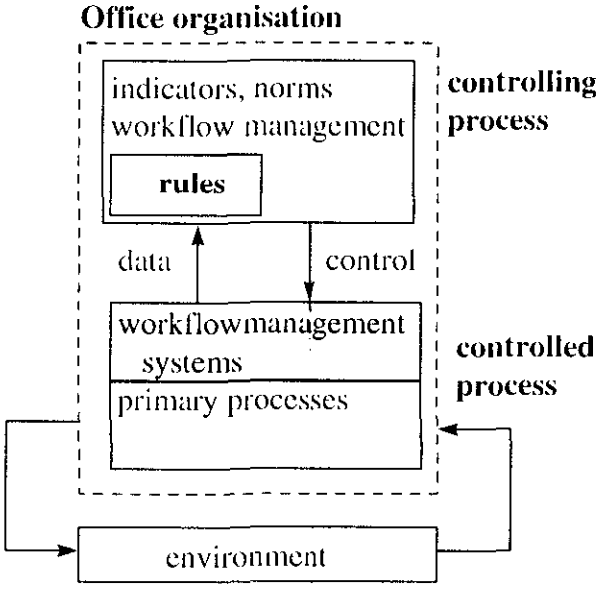

Figure 2: Semi-automated control over operational processes.

Based on experience we suggest in this report some general rules and guidelines for the design of an effective support system for the control processes. This part of the management function can be implemented in a WFM approach, as shown in Figure 2. We illustrate the implementation by placing the rules in a formal framework based on Petri Nets, thus realising the third level of work processes. This formalisation provides a way of better mastering the control function, while the framework builds a preliminary basis for more adaptive WFMSs.

Section 2 discusses the dynamics involved in the control function. The current approach to implementing control is the subject of Section 3, while its insufficiencies are discussed in Section 4. In Section 5 we give our approach and we discuss the consequences for next generation WFMSs. This report ends with a hint for further research in Section 7: it states a number of research issues in connection to the design of workflow processes and thus acts as a position paper on this topic. Section 7 contains a short conclusion.

2. Control dynamics

A modern organisation exists in a world of change. The organisation's environment changes and as a consequence the organisation itself is forced to adapt to changed circumstances.

The drivers for changes in an organisation emerge from both outside and inside the organisation.
Typical change drivers from the outside are a change in market demand, a change in government policy and competitor actions. Changes in channel interfaces are other examples of external change drivers:

- Suppliers can demand for example to use Electronic Data Interchange (EDI) as a new way to do business.
- Consumers also can demand for other way of interaction, like the concept of the Automatic Teller Machine (ATM) shows.
- While the possibility of Electronic Funds Transfers (EFT) via the Internet is technological feasible and some companies are not longer just testing this practice but actually providing real services, a whole new (world wide) market appears.

With this diversion of interfaces that change from manual to electronic, the office organisation has to be able to process this electronic information and be able to respond to altered interfaces.

Examples of internal change drivers are new performance metrics, changes in client management and the need to continuously fine-tune processes.

- For example, if one shifts from a functional-oriented organisation structure to a matrix based organisation, many new guidelines of doing business must be installed.
- Another example is the introduction of shared product databases to make information accessible to everyone that requires this information.
- Time is the new competitive advantage, and there exists a growing need to design business process that will meet the new time demands.
- As is shown in [Davidson, 1993], the introduction of new tools and methods results in a dominance in the business chain, i.e. enables new services to suppliers and customers.

The interaction between external and internal change drivers is recognised but not exploited any further in this report.

The actions taken by management lead to changes in the organisation. Management actions can be categorised into operational, tactical and strategical management functions [Hunger & Wheelen, 1993].

- Strategical management actions can result in totally new or dramatically changed processes. These actions happen infrequently.
- In the middle are the tactical management actions that have an impact on existing processes but are not as dramatic and infrequent as strategical actions.
- Operational management actions are narrow in scope (aimed at optimising tasks) and happen frequently to secure an efficient operation.

To incorporate the knowledge of operational and partially tactical managerial actions in WFMSs we suggest to make this implicit knowledge about controlling processes explicit by identifying a set of rules (see Figure 2). These rules are embedded into WFMSs to incorporate the controlling process in the primary process. Below some examples of these rules are listed.

> **Examples**
>
> *Typical examples of such managerial actions or causal effects of measurement and reaction are listed below. Notice that they fall in the category of operational and (partially) tactical management, as the scope is the existing process.*
>
> 1) *The frequency of checks on product integrity, i.e. quality control, is dynamic. In general the frequency is high (up to 100%) on the release of a product and is then gradually lowered to a certain minimum.*
> 2) *The structure of the process depends on the actual work in progress. For example, sometimes a parallel execution of tasks outperforms sequentially executed tasks. Related is resource management, which also depends on the actual work in progress. For example, during low workloads new employees get assigned difficult tasks in order to provide for learning experience. As the workload increases, this manner of work assignment is bypassed to ensure an efficient handling of operations.*
> 3) *Experienced employees need less information to support their tasks than a new employee does. Thus, the way in which tasks are performed is related to the experience of the workers.*
> 4) *The structure of a process can depend on certain conditions: for example, there can exist a specific customer profile that requires extra attention during a related marketing campaign.*
> 5) *If the process consist of many different procedures, the employees are forced to review their work and look for the best suitable procedure to handle the work. This is the case when many exceptions to the standard process are expected.*

Besides these managerial actions , implicit information shared by employees is an important aspect in the relationship between controlling and controlled processes. This falls into the category of "mutual adjustment", identified in [Mintzberg, 1989]. The explicit management of processes, handled by WFMSs, consider information that is explicitly part of the given process to be handled. However, the information implicitly available in the office, for example experience built with similar cases or guidelines verbally issued by senior staff, are equally important in day to day's operations. Some of the explicitly defined rules are based upon the assumption of implicit knowledge, like the difference between new and experienced employees. Below some examples of implicit knowledge are stated.

Examples

1) *Knowledge and experience: not all employees can or may use the same information. While in industrial settings employees can be perfectly considered as elements of a class, office environments are less strict and often demand a more elaborate approach to qualifications of employees.*
2) *Responsibilities: besides actually performing tasks, such as handling a document, employees in an office need to distribute responsibilities in order to guarantee a proper management of the office work. While the distribution of responsibilities is an important management tool, most WFM approaches do not consider this distribution an explicit part of the process. This results in a reduction in the ability to efficiently adjust the control process in reaction to changes in the environment.*
3) *Communication: Office workers do not just pass on documents to each other, but they also use common implicit knowledge, informal or implicit communication, informal directives and lots of other implicit or informal information that exists primarily in the minds of the employees. An example of such knowledge is a specific customer history profile shared by some office workers.*

We will address possible extensions of current WFMSs to incorporate the functions of control identified above. Therefore we list the current functionalities of WFMSs in the next section in order to situate our extensions.

3. Control in WFMSs

Current WFMS implementations reflect a particular realisation of an operational process: the implementation is based on one (stable) layout of the process. The control function, the control process, is materialised by the setting of a number of control parameters: once the layout of the operational process is known the control parameters are set.

According to the theory of the dynamics of information systems [Ramackers and Verrijn-Stuart, 1991], the dynamics regarding WFMSs can be defined as zeroth, first and second order. We start by presenting the zeroth-order dynamics: these are the atomic elements of a WFMS that can be described in isolation. After that we describe the first-order dynamics, where the atomic elements are grouped in a specific setting to build a process. The changes that occur in the process are called second-order dynamics. The different levels of dynamics are listed in Figure 3.

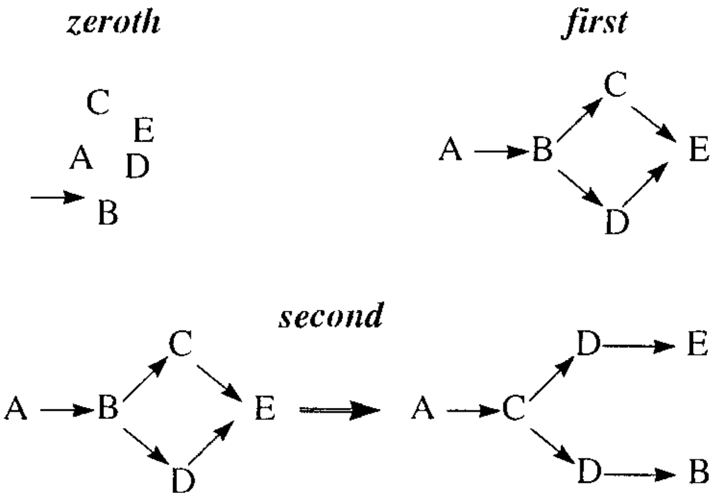

Figure 3: Zeroth-, first- and second-order dynamics.

3.1. Zeroth-order WFMS dynamics

The elements that makeup a WFMS are postulated here, together with a formal approach. This approach builds on the theory of Petri Nets, which has good properties to serve as modelling and analysis technique for business processes [Aalst, Hee and Houben, 1994]. We use this technique because it is easily comprehensible and the model has good graphical abilities to capture business processes.

Most of today's WFMSs offer the possibility to specify elements as the ones listed below. These are also identified by the Workflow Management Coalition [WFMC, 1994].

Task
Every process is constructed from a set of task. A task has a fixed input, a predetermined transformation function and delivers an output. Every task is assigned to a particular set of resources who are responsible to create the output. According to the theory of Petri Nets, a task is a transition, graphically depicted with a square.

State
Work that is in between tasks has a certain state assigned to it. These states are the traditional "trays" that hold (partially finished) work. The advantage of a state based description of workflow, is that tasks can be defined on states. As will be shown in our examples, the usage of states is crucial to enhance WFMSs with elaborated control mechanisms. In the theory of Petri Nets, a state is graphically depicted by a circle.

Case
A case is the work that flows from task to task via the states. Tasks can start as soon as they have the proper case assigned to them. Cases represent the explicitly defined information, available on forms or in databases. They are graphically depicted by a "token" in Petri Net theory. Every case (token) is always in a certain state (circle) and those tasks (squares) that have all the proper cases assigned to their input can start task. When a case is transformed it is moved from the input to an output state, thus materialising the flow.

Resource
A resource can be anything that is needed to transform a case task. In office organisations they are mostly humans with specific skills. In today's WFMSs they can be assigned to groups and roles. For example, a person can be a secretary for one task and a manager in the other. Other resources are applications or physical entities, like forms.

Priority and Time
Cases can get a different priority assigned to them. The WFMS can sort cases for humans according to these priorities. Unfortunately, most priorities do not match the meaning of "priority" that is required. For example, some WFMSs assign a figure to a case ranging from 0..10. This is however unrelated to the organisational usage of priorities (what does the number mean, how is it assigned and can it be altered?). Time aspects are new in today's process controls. WFMSs have some excellent features to define time-based actions on particular states.

We present a small example of these basic elements, to illustrate the usage of them in a WFMS.

Example
Suppose we model a small process that has three tasks; 'start', 'work' and 'end'. When the start task completes its work, it is placed in stated 'ready'. When this state is reached the task 'work' can start, which will bring the process in state 'finished'. This enables the last task 'end' to start its execution. The case is a simple form with only one amount on it. The cases are sorted in such a manner that the highest is always on top and ready to be executed by the task. Further it is stated that a person named 'Claude' executes both the start and the work tasks, while 'Smith' only takes care of the 'end' task. A zeroth-order description is given in Figure 4.

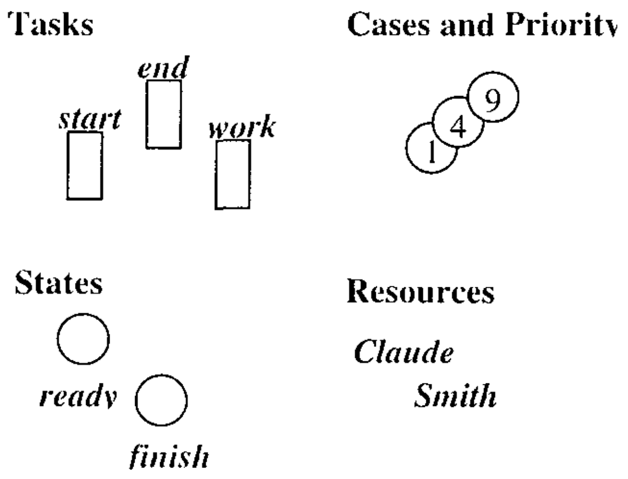

Figure 4: Zeroth-order description.

The identification of the tasks, states, cases, priority rules and resources are all part of a management function, i.e. to provide the necessary elements to construct an effectively performing process. The next management task is to combine them to create a value-added process. This means that the tasks are sequenced logically and that it results in an output that has meaning and value to customers.

3.2. First-order WFMS dynamics

When the elements of the zeroth-order dynamics are grouped together to support a process, the process gets (formally) defined. The relationships between tasks define the sequence in which the cases flow through the process. A similar skill is required in the design-phase of a WFMS. Typically, there is no single method to combine the zeroth-order dynamics of WFMSs into a first-order definition. The usage of Petri Nets helps to structure this design-phase. In this case, every process is designed according to the formal flow-relations of Petri Nets. Some commercial WFMS also opted for a formal approach, like COSA (Software-Ley) or Flowpath (Bull).

A special issue in the first-order dynamics of WFMSs is that explicitly defined logistical rules are defined. These rules enable parallel or alternative tasks to happen. Parallel work is facilitated by a so-called 'and-split' and is combined by an 'and-join'. When an task has a choice of outputs, i.e. it selects a next task, it is defined by an 'or-split'. The alternative paths can be combined with an 'or-join'. In our examples we will use these terms and we refer to the work of the Workflow Management Coalition [WFMC, 1994] for more details.

Example

When the example situated in the previous section is carefully read, the first-order dynamics can be found. It states which tasks and states are combined to *form the process. Figure 5 shows the first-order description of the example process.*

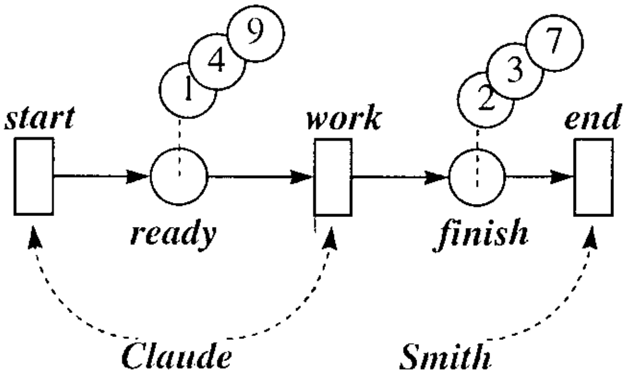

Figure 5: First-order description of the example.

This example only shows sequential tasks, no parallel or alternative tasks are taken here.

4. Insufficiencies in today's WFMSs

While WFMSs have proven to be excellent tools in describing zeroth-order dynamics of processes and in building working solutions by combining them into full first-order dynamics, they lack every support for management to adjust this first-order dynamic description of the process when needed. Thus, they lack the support of second-order dynamics that change the zeroth- and/or first-order dynamics of a processes.

When a WFMS is used in practise, this is the common way control is implemented:
1) Measuring the executions of the processes on defined indicators in the WFMS.
2) These results are interpreted by management and compared to the norms.
3) Feedback or steering actions are given by management, through the adjustment of zeroth or first-order changes in the WFMS.

This control function, in detail analysed in [Grant and Higgins, 1996], leads to the phenomenon that all the process descriptions are stored in massive databases that can be used by management to analyse the operations. The result of such an analysis is that the management takes actions that are reflected in the control parameters of the WFMS. A major disadvantage of this approach is that the amount of process data is so large that it is difficult to efficiently analyse that data, and thus to efficiently improve the process.

At this stage we see a number of things happen in practice. For example:

1) Every adjustment of the process leads to a new version of the WFMS operating for new cases: in the extreme situation every single case gets its own controlled process.
2) Instead of controlling a process with the help of a WFMS, the management cannot cope with the massive amounts of data that it cannot comprehend. Therefore, it cannot make an effective use of the WFMS and possibly excludes its help in the future.

The first scenario, new adjustments mean new versions, is not a desired end:

- A WFMS has every case wander through the process according to some pre-set route, thus passing along the tasks performed by the employees. In order to do this in an efficient and effective manner the WFMS must be able to steer the cases along the right route. This in its turn requires that the routes and their selection criteria remain fairly stable. When the routes change frequently the WFMS does not have the time to settle in a stable state where the operations can be managed efficiently: a certain learning time is needed. It is clear that new routes imply a loss of efficiency.
- Giving every case its own process means that every time one has to go to the phases of designing and implementing a new process. This is not only a costly affair, but also an affair that is not free of possible errors.
- New versions of a WFMS imply that several versions at a time are participating in the control of the operational processes. This makes it more difficult to control the entire process, as the different versions can cause problems in co-ordination and synchronisation. Moreover, it then becomes an issue to consider the effect that versions have on each other: it can be expected that different control actions can interfere in an unforeseen manner.

Another undesired scenario is the second one, where management is not capable of interpreting the control data.

- If all aspects of the process's execution are stored into databases for control purposes, the result is a very detailed perspective of the operations. Too many data to monitor quickly leads to an "analysis paralysis" situation; management is so pre-occupied with monitoring that the real function, that of proper control, is forgotten.
- Using a WFMS as a mere measuring-apparatus implies that management is responsible for the execution of all control functions that have their effect in the WFMS. However, WFMSs can play a role in the execution of explicitly defined managerial actions, thus incorporating feedback.

There is an alternative approach, which acknowledges the fact that part of the control function can be automated. This means that managerial tasks are truly delegated to a WFMS where possible. The amount of data to analyse is then minimised and controlling the process and its adaptation becomes a mature WFMS function.
In the next section we argue that this alternative option is the way to go. We give rules and guidelines for implementation strategies for that option.

5. Implementing second-order dynamics in WFMSs

In Section 2 we listed some managerial actions that should be incorporated into WFMSs, together with informal information shared among office employees. Both execute a form of control in the execution of the process. Here we show how the zeroth- and first-order elements postulated in Section 3 provide for a second-order dynamic change of the process in the supporting WFMSs.

5.1. Augmenting the scope with managerial rules

The scope of control is stretched by explicitly incorporating managerial rules like the ones listed in Section 2. These are only some illustrative examples from our experience and the opportunity for manufacturers and office organisations lies in the full exploitation managerial rules into WFMSs.

Examples

The frequency of checks on product integrity, i.e. quality control, is dynamic. In general the frequency is high (up to 100%) on the release of a product and is then gradually lowered to a certain minimum.

A "check" is an task that is executed by a special resource on some specific case. Instead of assigning every case to this task this assignment is enhanced. The two functions added are a) statistics and b) a time-interval. This means that the process is designed with the check as a regular task, but it is recognised that it will be skipped for a number of cases in the future. This is exemplified in Figure 6. It shows an 'or-split' that assigns work to either 'check' or 'normal' based on a statistical function, denoted p and $(1.0\text{-}p)$ at time t_1. When the situation has changed, e.g. quality has improved due to expected learning of the organisation, the statistical function of t_2 is activated. Notice that instead of assigning fixed time-intervals one can also use a percentage of errors detected to derive the statistical split. When $p = 1.0$ all cases are assigned to the check task, and for example, when $q=0.1$ only one out of ten is checked.

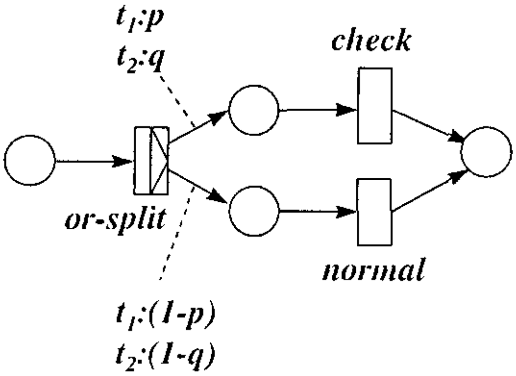

Figure 6: Time-based statistical or-split.

The structure of the process depends on the actual workload. For example, sometimes a parallel execution of tasks outperforms sequentially executed tasks. Related is resource management, which also depends on the actual workload. For example, during low workloads new employees get assigned difficult tasks in order to provide learning. As the workload increases, this manner of work assignment is bypassed to ensure an efficient handling of operations.

The structure of the tasks in a process depends on the workload of the process. This dimension is covered by assigning thresholds in the workload and assigning different scenario's when thresholds are passed. A threshold can be found with the help of simulation. The example in Figure 7 defines two alternative execution paths, based on a single threshold.

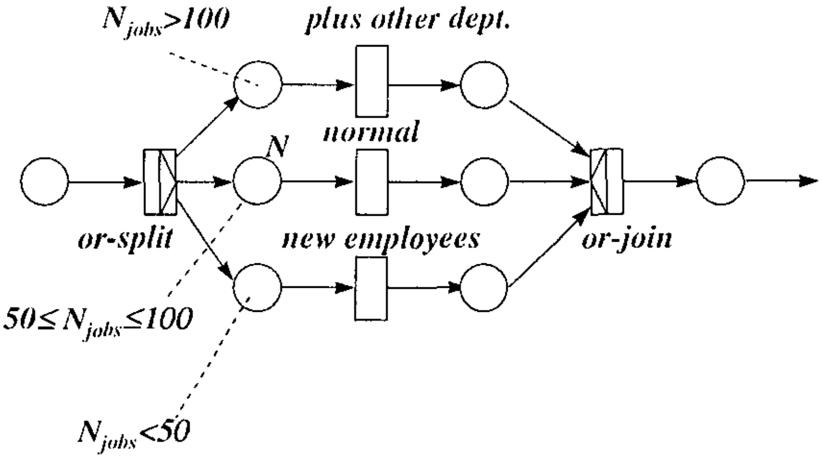

Figure 7: Workload dependent tasks.

This example shows three possible execution paths. The choice of case assignment is based on the current workload *N* on the normal task and its resources (not shown in the figure). As a remark we state that normally these tasks each are composed of a number of tasks, i.e. these models are hierarchical. The different work assignments can therefore enable other ways to conduct business. In the example, cases are send to 'new employees' when the workload is low and to an 'other department' when the cases reached a maximum of (in this case) hundred.

Experienced employees need less information to support their tasks than a new employee does. Thus, the way in which tasks are performed is related to the experience of the workers.
An advantage of WFMSs is that it provides all the necessary information to help new employees do their case. Once they are assigned to a new task they have access to information which tells them how to execute this task. When they master the task they can exclude the help of the WFMS and execute tasks on their own.

The structure of a process can depend on certain conditions: for example, there can exist a specific customer profile that requires extra attention during a related marketing campaign.
Sometimes the operational process holds information that is required for additional purposes, like administration or marketing. Therefore, a mechanism must be installed to "tap into" the operational process like an eavesdropper. The operational process is not to be disturbed by it. A way to handle this is to assign special states that allow easy access by other processes. Those familiar with programming languages translate this to global versus local objects: the global states can be accessed by many.

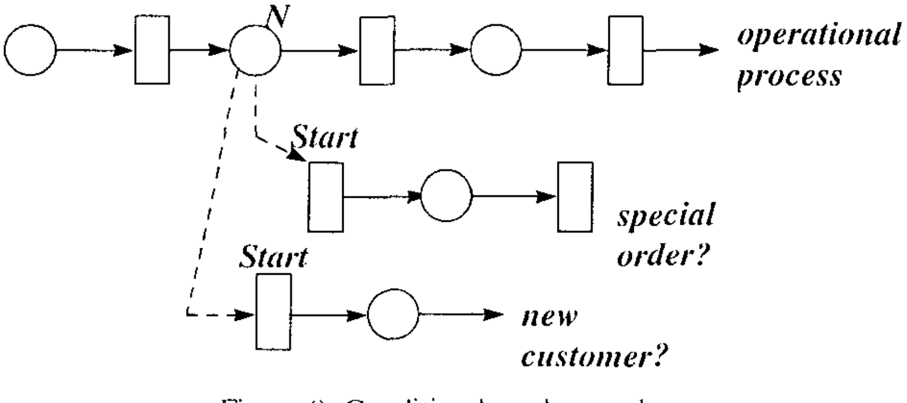

Figure 8: Condition based procedures.

In the example stated in Figure 8 each case that is in state *N* is "screened". Based on some conditions it can be necessary to start other (unrelated) procedures. The arrows are dotted because the case is not transferred to the procedures but a new instance is made for new procedures. This way many small procedures can be constructed that start when needed.

> *If the process consist of many different procedures, the employees are forced to review their work and look for best suitable procedures to handle the work. This is the case when many exceptions to the standard process are expected.* The usage of WFMSs is limited to structured processes. However, many situations could benefit from a WFMS, but have a more unstructured base; they are composed of small structured processes that do not have an a priori defined connection. To be able to manage this situation whilst using the advantages of WFMSs, the employees must be equipped with a range of optional processes. These optional processes can be started by an employee after the task has been completed and new (partially predictable) actions have to be taken. This approach can be taken instead of merging all possibilities into one massive and uncontrollable process definition. The result is a set of small procedures that can be grouped together for specific means, like the previous example showed.

The situation dependent rules implied by the above mentioned examples, translate into the following general rule:
Augment the scope of the control function to include second-order dynamics. By a carefully designed process architecture that implements significant aspects of second-order dynamics, an efficient division in responsibilities is possible between management and support system. The management actions that can be automated are embedded in the system, while it is efficiently possible to leave the remainder of the management function a responsibility for the management.

By implementing managerial rules within the WFMS, the amount of information for management to monitor becomes minimal. The controlling of the process and adjustment of its layout becomes a mature management function. The operational aspects are separated from the tactical ones: the automated WFMS covers the operational aspects, while the human attention can be focused on the higher level management aspects.

5.2. Augmenting the scope of control with implicit information

We have argued earlier that it is a good approach to augment the scope of the entire process controlled by the WFMS such that a large part of the control process is managed by the automated system. A related aspect concerns implicit or informal information. By this we mean the information that is usually not part of the logistical perspective of the process, but that is implicitly used in connection with the execution of office work. We give examples of that information that can be exploited for automated support by a WFMS.

Knowledge and experience

Not all employees can or may use the same information. While in industrial settings employees can be perfectly considered as elements of a class, office environments are less strict and often demand a more elaborate approach to qualifications of employees. To perform a given task an office worker uses first of all information that is specifically related to the case on-hand. A typical property of office work is that besides the specific information, general information is used.

> **Example**
> *Examples of general information used in the execution of office work are general guidelines and directives that specify how the organisation wants to work independently of specific cases. Such general recipes are often not considered in the implementation of a WFMS. For the design of a flexible controlling process these recipes are however found to be important. A possible implementation of a process is the combination of different technologies. Combining for example the internet with a classical WFMS could offer a parallel track of formal and implicit knowledge.*

> **Example**
> *Another example of general information is background knowledge and experience that relate to the general way of working: as office tasks often involve mental tasks (e.g. decision making) the actual performance is strongly related to the person actually doing it and the background he or she is using: a novice can often only use the specific information on hand, while a background of knowledge and experience can help to obtain a better performance (both in quality and time).For example, build a history of the tasks executed by an employee. If similar cases appear, an experienced employee can automatically be selected by a WFMS. Elaborate rules can define when one is 'experienced'.*

Based on our experiences we argue that knowledge and experience result in information that should be considered in the design of the process. It does not mean that the information should be implemented in full detail. However, those aspects that directly relate to the performance and the flexibility should be taken into account when designing the automated and non-automated parts of the controlling processes.

> **Example**
> *To effectively manage an operational process, one wants to have information on the availability of knowledge. A large part of managing the process is about allocating the right people (and knowledge) to the cases on-hand. For this reason, management needs to have an actual and correct perspective of the knowledge available.*

Responsibilities

Besides actually performing tasks, such as handling a document, employees in an office need to distribute responsibilities in order to guarantee a proper management of the office work. While the distribution of responsibilities is an important management function, most WFMS design methods do not consider this distribution an explicit part of the process. On example that does is [Medina-Mora et al., 1992]. Usually all the information flows is considered in one (logistical) perspective: the documents flowing through the organisation's tasks, the resources are flowing to and from the tasks and the (managerial) responsibilities are flowing between the resources to help control the process. While these flows are important, from the point of view of the control function these flows should be considered differently. If we look at responsibilities we can see that they involve being informed, approving, initiating and other tasks that are not directly related to the flow of documents. These (responsibility) tasks which consume a lot of time and energy within an office, can and should play an important role in the controlling process.

> **Example**
> *An important feature that would want to see in an environment of knowledge workers, is the ability to verify the commitments that are made (e.g. with clients) about the work in progress. If an office worker is not available or some kind of communication breaks down, one wants to be able to check what goal the work was trying to achieve: one can imagine that it is important to know which commitments are given by the organisation to their clients. That additional information can then be used (by the management) to take appropriate actions (e.g. notify the client of a delay or substitute the worker).*

The usual approach of considering these responsibility tasks in the same way as all other tasks decreases the ability to quickly adjust the control process without tampering with the document flows. We feel that the ability to efficiently adjust the control process in reaction to changes in the environment, should imply a carefully designed implementation of responsibilities.

Communication

Office workers do not just pass on documents to each other, but they also use common implicit knowledge, informal communication, informal directives and lots of other implicit or informal information that exists primarily in the minds of the employees: this information can be considered as the "mind" of the office. Generally, an office needs such a mind to be able to function properly. This information implies a pattern of communication that is facilitating, but not part of the flow of work. Just as the responsibilities, this facilitating communication has a significant impact on the workflow and should hence be treated accordingly in the design and the implementation of the WFMS.

> **Example**
> *The use of informal communication makes an organisation really operate. While in an ideal situation the process lay-out defines all aspects of the process, in practice a number of assumptions are made (about the way the organisation operates and the role of the informal communication). It is not*

> *always feasible to try to implement the informal communication. However, without the informal communication the organisation would not operate as efficiently. The most illustrative example is the communication that takes place at the coffee-machine: there the office workers meet and informally exchange information (which formally would not have been exchanged).*

The design should acknowledge the role this communication plays and the consequences for the ability to change. Currently, we can see cases where a groupware approach is more suited than a strict WFMS approach, since groupware sometimes better acknowledges the different aspects of communication in an office. The lesson to be learned here is to carefully decide on the right need to model communication and on the right way of implementing.

These aspects result in the following general rule:
Managing the workforce is more than making sure that the right number of people is available in the office: office workers in most cases are led by implicit or informal information that is available within an office. It is vital for an efficient office that this implicit information, and specially changes in it, are an integral part of the process which is managed by the WFMS.

6. Further research

The previous sections contain a number of guidelines for the design of workflow processes. These guidelines specially aim at the implementation of information systems (WFMS) that effectively support the management in its control function. From the point of view of research this leads to a number of questions.

- *How do WFM tools corporate second-order dynamics?*
  We have argued that second-order dynamics need to be considered, but at the same time we have seen that the existing WFM tools do not support this function. Facilities to manage second-order dynamics should make it possible to effectively specify the second-order dynamics (in order to obtain an effective control process). There is however another aspect: the facilities should also make it possible to deal with the practical consequences of changing operational processes: the communication between the information systems involved, can change on-the-fly.
- *How can WFM tools deal with implicit information?*
  Current WFMSs concentrate on the logistical perspective: perhaps after doing Business Process Redesign, the lay-out of the process considers the formal and explicit knowledge. Modern groupware technology enables knowledge workers to use a common platform for the exchange of information (on demand). Practice shows that these approaches do not really satisfy the combined need of considering both formal and informal information. It is feasible to expect an integrated approach to better satisfy that need.
- *What is the proper method for designing control processes?*

We have demonstrated the need to adequately consider the relationship between control process and controlled process. In order to design a real-sized workflow application, the different aspects should be considered. This process of considering all relevant aspects should find a right balance between the needs of the business process and the possibilities that (available) technologies offer.

- *What is the influence of different types of WFMSs to the design method?*
  The technologies to implement workflow applications differ significantly. In practice one can observe that different technologies suit different types of processes with e.g. back-office, front-office and knowledge workers. In order to make the proper choices in the design process, characteristic properties of the different technologies and of the different application types need to be identified. This should lead to a mapping of technologies to business application types.
- *How should a control process (and its behaviour) represent the continuous nature of a management process?*
  We have argued that second-order dynamics need to be incorporated into the design of the control process. The continuous nature of managing an operational process has specific characteristics that seem to be hard to describe using standard process modelling tools. For a valid description of the second-order dynamics new approaches seem to be necessary. This need should be met by next-generation WFM tools.
- *How do the discrete and the continuous design functions of a WFMS relate?*
  Every design of new business processes follows certain stages. In this report we focused on the design of the continuous functions of a WFMS, i.e. incorporate control and implicit information. However, when we listed these functions we identified "a process" beforehand. How is this process constructed? What characteristics does this basic process have in terms of continuous functions?
- *How can tools support the design of workflow (control) processes?*
  Some of the questions stated here imply that designers are not yet able to easily design a workflow (controlled) process. Current WFM tools focus on the operational part of the control process: they make sure that the work is properly routed within the organisation. There is also a need for tools that support the designers in their process of designing effective workflow applications. We think of on-the-fly correction of design errors, using a repository of functions (that play a role in operational or control process), etc.

7. Conclusion

In this report we have argued that today's workflow management systems (WFMSs) should be enhanced with control functions that are to date part of the managerial domain. From this domain we have extracted explicit managerial rules and informal information as essential control mechanisms for operational processes. We have stated a number of research questions that give a context for further research in this direction.

Elaborated control implemented in a WFMS takes a burden of management, that is overloaded with data about the way the processes are functioning. This way the

WFMS functions not only as a measuring-apparatus but uses the explicitly defined rules to provide feedback to itself. From this feedback it produces altered process definitions or altered definitions of the elements that makeup the process, like resource management or the use of time-based tasks.
The described new situation implies a shift in focus for manufacturers of WFMSs. Not only do they have to realise a complete description of processes, they have to place themselves in the place of management and ask themselves which information they would need to manage the process and how they would react on this information. If these rules are made explicit into WFMS design, it introduces an intelligent version of today's straightforward implementations. In this way a first step is made to truly delegate parts of management control to the WFMS.

The result is an intelligent process description that adapts itself to carefully defined changes in the environment. Once installed it runs accordingly, which leaves management with the task to monitor the effectiveness of the organisation at large in stead of continuously adjusting processes in the small. This leads to a more strategic management of processes, with a more effective control of second order dynamics.

8. Literature

Aalst, W.M.P., K.M. van Hee, G.J. Houben, Modelling and analysing workflow using a Petri Net based approach, Computer Supported Co-operative Work, PETRI NETS and related formalisms, ed. G. De Michelis, C. Ellis and G. Memmi, Workshop within the 15th Int. Conference on Application and Theory of Petri Nets, 1994, pg. 31-50.

Davidson, W.H., Beyond re-engineering: The three phases of business transformation, IBM Systems Journal, Vol. 32, No.1, 1993, pg. 65-79.

Grant, R.A., C. Higgins, Computerized Performance Monitors as Multidimensional Systems: Derivation and Application, ACM Transactions on Information Systems, Vol. 14, No. 2, April 1996, pg. 212-235.

Hunger, J.D., T.L. Wheelen, Strategic Management, Addison-Wesley Publishing company, 1993.

Mantz, E.A., J.T.H.C. van Lieshout, F.A.P. van der Zijden, Practical cases of information policy and information planning, Informatie, 32, nr. 1, 1990, pg. 23-31 (in Dutch).

Medina-Mora, R., Winograd, T., Flores, R., Flores, F., The action Workflow Approach to Workflow Management Technology, CSCW Proceedings, November 1992, pg. 281-288.

Mintzberg, H., Mintzberg on Management, The Free Press, 1989.

Ramackers, G.J., A.A. Verrijn-Stuart, First and Second order dynamics in Information Systems, in Information Systems Modelling II, ed. H.G. Sol and K.M. van Hee, Elsevier Science Publishers B.V., North-Holland, 1991.

Workflow Management Coalition, Workflow Reference Model, Technical Report, Workflow Management Coalition, Brussels, 1994.

Computing Science Reports

Department of Mathematics and Computing Science
Eindhoven University of Technology

In this series appeared:

93/01	R. van Geldrop	Deriving the Aho-Corasick algorithms: a case study into the synergy of programming methods, p. 36.
93/02	T. Verhoeff	A continuous version of the Prisoner's Dilemma, p. 17
93/03	T. Verhoeff	Quicksort for linked lists, p. 8.
93/04	E.H.L. Aarts J.H.M. Korst P.J. Zwietering	Deterministic and randomized local search, p. 78.
93/05	J.C.M. Baeten C. Verhoef	A congruence theorem for structured operational semantics with predicates, p. 18.
93/06	J.P. Veltkamp	On the unavoidability of metastable behaviour, p. 29
93/07	P.D. Moerland	Exercises in Multiprogramming, p. 97
93/08	J. Verhoosel	A Formal Deterministic Scheduling Model for Hard Real-Time Executions in DEDOS, p. 32.
93/09	K.M. van Hee	Systems Engineering: a Formal Approach Part I: System Concepts, p. 72.
93/10	K.M. van Hee	Systems Engineering: a Formal Approach Part II: Frameworks, p. 44.
93/11	K.M. van Hee	Systems Engineering: a Formal Approach Part III: Modeling Methods, p. 101.
93/12	K.M. van Hee	Systems Engineering: a Formal Approach Part IV: Analysis Methods, p. 63.
93/13	K.M. van Hee	Systems Engineering: a Formal Approach Part V: Specification Language, p. 89.
93/14	J.C.M. Baeten J.A. Bergstra	On Sequential Composition, Action Prefixes and Process Prefix, p. 21.
93/15	J.C.M. Baeten J.A. Bergstra R.N. Bol	A Real-Time Process Logic, p. 31.
93/16	H. Schepers J. Hooman	A Trace-Based Compositional Proof Theory for Fault Tolerant Distributed Systems, p. 27
93/17	D. Alstein P. van der Stok	Hard Real-Time Reliable Multicast in the DEDOS system, p. 19.
93/18	C. Verhoef	A congruence theorem for structured operational semantics with predicates and negative premises, p. 22.
93/19	G-J. Houben	The Design of an Online Help Facility for ExSpect, p.21.
93/20	F.S. de Boer	A Process Algebra of Concurrent Constraint Programming, p. 15.
93/21	M. Codish D. Dams G. Filé M. Bruynooghe	Freeness Analysis for Logic Programs - And Correctness, p. 24
93/22	E. Poll	A Typechecker for Bijective Pure Type Systems, p. 28.
93/23	E. de Kogel	Relational Algebra and Equational Proofs, p. 23.
93/24	E. Poll and Paula Severi	Pure Type Systems with Definitions, p. 38.
93/25	H. Schepers and R. Gerth	A Compositional Proof Theory for Fault Tolerant Real-Time Distributed Systems, p. 31.
93/26	W.M.P. van der Aalst	Multi-dimensional Petri nets, p. 25.
93/27	T. Kloks and D. Kratsch	Finding all minimal separators of a graph, p. 11.
93/28	F. Kamareddine and R. Nederpelt	A Semantics for a fine λ-calculus with de Bruijn indices, p. 49.
93/29	R. Post and P. De Bra	GOLD, a Graph Oriented Language for Databases, p. 42.
93/30	J. Deogun T. Kloks D. Kratsch H. Müller	On Vertex Ranking for Permutation and Other Graphs, p. 11.

93/31	W. Körver	Derivation of delay insensitive and speed independent CMOS circuits, using directed commands and production rule sets, p. 40.
93/32	H. ten Eikelder and H. van Geldrop	On the Correctness of some Algorithms to generate Finite Automata for Regular Expressions, p. 17.
93/33	L. Loyens and J. Moonen	ILIAS, a sequential language for parallel matrix computations, p. 20.
93/34	J.C.M. Baeten and J.A. Bergstra	Real Time Process Algebra with Infinitesimals, p.39.
93/35	W. Ferrer and P. Severi	Abstract Reduction and Topology, p. 28.
93/36	J.C.M. Baeten and J.A. Bergstra	Non Interleaving Process Algebra, p. 17.
93/37	J. Brunekreef J-P. Katoen R. Koymans S. Mauw	Design and Analysis of Dynamic Leader Election Protocols in Broadcast Networks, p. 73.
93/38	C. Verhoef	A general conservative extension theorem in process algebra, p. 17.
93/39	W.P.M. Nuijten E.H.L. Aarts D.A.A. van Erp Taalman Kip K.M. van Hee	Job Shop Scheduling by Constraint Satisfaction, p. 22.
93/40	P.D.V. van der Stok M.M.M.P.J. Claessen D. Alstein	A Hierarchical Membership Protocol for Synchronous Distributed Systems, p. 43.
93/41	A. Bijlsma	Temporal operators viewed as predicate transformers, p. 11.
93/42	P.M.P. Rambags	Automatic Verification of Regular Protocols in P/T Nets, p. 23.
93/43	B.W. Watson	A taxomomy of finite automata construction algorithms, p. 87.
93/44	B.W. Watson	A taxonomy of finite automata minimization algorithms, p. 23.
93/45	E.J. Luit J.M.M. Martin	A precise clock synchronization protocol,p.
93/46	T. Kloks D. Kratsch J. Spinrad	Treewidth and Patwidth of Cocomparability graphs of Bounded Dimension, p. 14.
93/47	W. v.d. Aalst P. De Bra G.J. Houben Y. Komatzky	Browsing Semantics in the "Tower" Model, p. 19.
93/48	R. Gerth	Verifying Sequentially Consistent Memory using Interface Refinement, p. 20.
94/01	P. America M. van der Kammen R.P. Nederpelt O.S. van Roosmalen H.C.M. de Swart	The object-oriented paradigm, p. 28.
94/02	F. Kamareddine R.P. Nederpelt	Canonical typing and Π-conversion, p. 51.
94/03	L.B. Hartman K.M. van Hee	Application of Marcov Decision Processe to Search Problems, p. 21.
94/04	J.C.M. Baeten J.A. Bergstra	Graph Isomorphism Models for Non Interleaving Process Algebra, p. 18.
94/05	P. Zhou J. Hooman	Formal Specification and Compositional Verification of an Atomic Broadcast Protocol, p. 22.
94/06	T. Basten T. Kunz J. Black M. Coffin D. Taylor	Time and the Order of Abstract Events in Distributed Computations, p. 29.
94/07	K.R. Apt R. Bol	Logic Programming and Negation: A Survey, p. 62.
94/08	O.S. van Roosmalen	A Hierarchical Diagrammatic Representation of Class Structure, p. 22.
94/09	J.C.M. Baeten J.A. Bergstra	Process Algebra with Partial Choice, p. 16.

94/10	T. verhoeff	The testing Paradigm Applied to Network Structure. p. 31.
94/11	J. Peleska C. Huizing C. Petersohn	A Comparison of Ward & Mellor's Transformation Schema with State- & Activitycharts, p. 30.
94/12	T. Kloks D. Kratsch H. Müller	Dominoes, p. 14.
94/13	R. Seljée	A New Method for Integrity Constraint checking in Deductive Databases, p. 34.
94/14	W. Peremans	Ups and Downs of Type Theory, p. 9.
94/15	R.J.M. Vaessens E.H.L. Aarts J.K. Lenstra	Job Shop Scheduling by Local Search, p. 21.
94/16	R.C. Backhouse H. Doornbos	Mathematical Induction Made Calculational, p. 36.
94/17	S. Mauw M.A. Reniers	An Algebraic Semantics of Basic Message Sequence Charts, p. 9.
94/18	F. Kamareddine R. Nederpelt	Refining Reduction in the Lambda Calculus, p. 15.
94/19	B.W. Watson	The performance of single-keyword and multiple-keyword pattern matching algorithms, p. 46.
94/20	R. Bloo F. Kamareddine R. Nederpelt	Beyond β-Reduction in Church's $\lambda\rightarrow$, p. 22.
94/21	B.W. Watson	An introduction to the Fire engine: A C++ toolkit for Finite automata and Regular Expressions.
94/22	B.W. Watson	The design and implementation of the FIRE engine: A C++ toolkit for Finite automata and regular Expressions.
94/23	S. Mauw and M.A. Reniers	An algebraic semantics of Message Sequence Charts, p. 43.
94/24	D. Dams O. Grumberg R. Gerth	Abstract Interpretation of Reactive Systems: Abstractions Preserving $\forall$CTL*, $\exists$CTL* and CTL*, p. 28.
94/25	T. Kloks	$K_{1,3}$-free and W_4-free graphs, p. 10.
94/26	R.R. Hoogerwoord	On the foundations of functional programming: a programmer's point of view, p. 54.
94/27	S. Mauw and H. Mulder	Regularity of BPA-Systems is Decidable, p. 14.
94/28	C.W.A.M. van Overveld M. Verhoeven	Stars or Stripes: a comparative study of finite and transfinite techniques for surface modelling, p. 20.
94/29	J. Hooman	Correctness of Real Time Systems by Construction, p. 22.
94/30	J.C.M. Baeten J.A. Bergstra Gh. Ştefanescu	Process Algebra with Feedback, p. 22.
94/31	B.W. Watson R.E. Watson	A Boyer-Moore type algorithm for regular expression pattern matching, p. 22.
94/32	J.J. Vereijken	Fischer's Protocol in Timed Process Algebra, p. 38.
94/33	T. Laan	A formalization of the Ramified Type Theory, p.40.
94/34	R. Bloo F. Kamareddine R. Nederpelt	The Barendregt Cube with Definitions and Generalised Reduction, p. 37.
94/35	J.C.M. Baeten S. Mauw	Delayed choice: an operator for joining Message Sequence Charts, p. 15.
94/36	F. Kamareddine R. Nederpelt	Canonical typing and Π-conversion in the Barendregt Cube, p. 19.
94/37	T. Basten R. Bol M. Voorhoeve	Simulating and Analyzing Railway Interlockings in ExSpect, p. 30.
94/38	A. Bijlsma C.S. Scholten	Point-free substitution, p. 10.

94/39	A. Blokhuis T. Kloks	On the equivalence covering number of splitgraphs, p. 4.
94/40	D. Alstein	Distributed Consensus and Hard Real-Time Systems, p. 34.
94/41	T. Kloks D. Kratsch	Computing a perfect edge without vertex elimination ordering of a chordal bipartite graph, p. 6.
94/42	J. Engelfriet J.J. Vereijken	Concatenation of Graphs, p. 7.
94/43	R.C. Backhouse M. Bijsterveld	Category Theory as Coherently Constructive Lattice Theory: An Illustration, p. 35.
94/44	E. Brinksma J. Davies R. Gerth S. Graf W. Janssen B. Jonsson S. Katz G. Lowe M. Poel A. Pnueli C. Rump J. Zwiers	Verifying Sequentially Consistent Memory, p. 160
94/45	G.J. Houben	Tutorial voor de ExSpect-bibliotheek voor "Administratieve Logistiek", p. 43.
94/46	R. Bloo F. Kamareddine R. Nederpelt	The λ-cube with classes of terms modulo conversion, p. 16.
94/47	R. Bloo F. Kamareddine R. Nederpelt	On Π-conversion in Type Theory, p. 12.
94/48	Mathematics of Program Construction Group	Fixed-Point Calculus, p. 11.
94/49	J.C.M. Baeten J.A. Bergstra	Process Algebra with Propositional Signals, p. 25.
94/50	H. Geuvers	A short and flexible proof of Strong Normalazation for the Calculus of Constructions, p. 27.
94/51	T. Kloks D. Kratsch H. Müller	Listing simplicial vertices and recognizing diamond-free graphs, p. 4.
94/52	W. Penczek R. Kuiper	Traces and Logic, p. 81
94/53	R. Gerth R. Kuiper D. Peled W. Penczek	A Partial Order Approach to Branching Time Logic Model Checking, p. 20.
95/01	J.J. Lukkien	The Construction of a small CommunicationLibrary, p.16.
95/02	M. Bezem R. Bol J.F. Groote	Formalizing Process Algebraic Verifications in the Calculus of Constructions, p.49.
95/03	J.C.M. Baeten C. Verhoef	Concrete process algebra, p. 134.
95/04	J. Hidders	An Isotopic Invariant for Planar Drawings of Connected Planar Graphs, p. 9.
95/05	P. Severi	A Type Inference Algorithm for Pure Type Systems, p.20.
95/06	T.W.M. Vossen M.G.A. Verhoeven H.M.M. ten Eikelder E.H.L. Aarts	A Quantitative Analysis of Iterated Local Search, p.23.
95/07	G.A.M. de Bruyn O.S. van Roosmalen	Drawing Execution Graphs by Parsing, p. 10.
95/08	R. Bloo	Preservation of Strong Normalisation for Explicit Substitution, p. 12.
95/09	J.C.M. Baeten J.A. Bergstra	Discrete Time Process Algebra, p. 20
95/10	R.C. Backhouse R. Verhoeven O. Weber	Math∫pad: A System for On-Line Prepararation of Mathematical Documents, p. 15

95/11	R. Seljée	Deductive Database Systems and integrity constraint checking, p. 36.
95/12	S. Mauw and M. Reniers	Empty Interworkings and Refinement Semantics of Interworkings Revised, p. 19.
95/13	B.W. Watson and G. Zwaan	A taxonomy of sublinear multiple keyword pattern matching algorithms, p. 26.
95/14	A. Ponse, C. Verhoef, S.F.M. Vlijmen (eds.)	De proceedings: ACP'95, p.
95/15	P. Niebert and W. Penczek	On the Connection of Partial Order Logics and Partial Order Reduction Methods, p. 12.
95/16	D. Dams, O. Grumberg, R. Gerth	Abstract Interpretation of Reactive Systems: Preservation of CTL*, p. 27.
95/17	S. Mauw and E.A. van der Meulen	Specification of tools for Message Sequence Charts, p. 36.
95/18	F. Kamareddine and T. Laan	A Reflection on Russell's Ramified Types and Kripke's Hierarchy of Truths, p. 14.
95/19	J.C.M. Baeten and J.A. Bergstra	Discrete Time Process Algebra with Abstraction, p. 15.
95/20	F. van Raamsdonk and P. Severi	On Normalisation, p. 33.
95/21	A. van Deursen	Axiomatizing Early and Late Input by Variable Elimination, p. 44.
95/22	B. Arnold, A. v. Deursen, M. Res	An Algebraic Specification of a Language for Describing Financial Products, p. 11.
95/23	W.M.P. van der Aalst	Petri net based scheduling, p. 20.
95/24	F.P.M. Dignum, W.P.M. Nuijten, L.M.A. Janssen	Solving a Time Tabling Problem by Constraint Satisfaction, p. 14.
95/25	L. Feijs	Synchronous Sequence Charts In Action, p. 36.
95/26	W.M.P. van der Aalst	A Class of Petri nets for modeling and analyzing business processes, p. 24.
95/27	P.D.V. van der Stok, J. van der Wal	Proceedings of the Real-Time Database Workshop, p. 106.
95/28	W. Fokkink, C. Verhoef	A Conservative Look at term Deduction Systems with Variable Binding, p. 29.
95/29	H. Jurjus	On Nesting of a Nonmonotonic Conditional, p. 14
95/30	J. Hidders, C. Hoskens, J. Paredaens	The Formal Model of a Pattern Browsing Technique, p.24.
95/31	P. Kelb, D. Dams and R. Gerth	Practical Symbolic Model Checking of the full μ-calculus using Compositional Abstractions, p. 17.
95/32	W.M.P. van der Aalst	Handboek simulatie, p. 51.
95/33	J. Engelfriet and JJ. Vereijken	Context-Free Graph Grammars and Concatenation of Graphs, p. 35.
95/34	J. Zwanenburg	Record concatenation with intersection types, p. 46.
95/35	T. Basten and M. Voorhoeve	An algebraic semantics for hierarchical P/T Nets, p. 32.
96/01	M. Voorhoeve and T. Basten	Process Algebra with Autonomous Actions, p. 12.
96/02	P. de Bra and A. Aerts	Multi-User Publishing in the Web: DreSS, A Document Repository Service Station, p. 12
96/03	W.M.P. van der Aalst	Parallel Computation of Reachable Dead States in a Free-choice Petri Net, p. 26.
96/04	S. Mauw	Example specifications in phi-SDL.
96/05	T. Basten and W.M.P. v.d. Aalst	A Process-Algebraic Approach to Life-Cycle Inheritance Inheritance = Encapsulation + Abstraction, p. 15.
96/06	W.M.P. van der Aalst and T. Basten	Life-Cycle Inheritance A Petri-Net-Based Approach, p. 18.
96/07	M. Voorhoeve	Structural Petri Net Equivalence, p. 16.
96/08	A.T.M. Aerts, P.M.E. De Bra, J.T. de Munk	OODB Support for WWW Applications: Disclosing the internal structure of Hyperdocuments, p. 14.
96/09	F. Dignum, H. Weigand, E. Verharen	A Formal Specification of Deadlines using Dynamic Deontic Logic, p. 18.
96/10	R. Bloo, H. Geuvers	Explicit Substitution: on the Edge of Strong Normalisation, p. 13.
96/11	T. Laan	AUTOMATH and Pure Type Systems, p. 30.
96/12	F. Kamareddine and T. Laan	A Correspondence between Nuprl and the Ramified Theory of Types, p. 12.
96/13	T. Borghuis	Priorean Tense Logics in Modal Pure Type Systems, p. 61

96/14	S.H.J. Bos and M.A. Reniers	The I^2C-bus in Discrete-Time Process Algebra, p. 25.
96/15	M.A. Reniers and J.J. Vereijken	Completeness in Discrete-Time Process Algebra, p. 139.
96/16	P. Hoogendijk and O. de Moor	What is a data type?, p. 29.
96/17	E. Boiten and P. Hoogendijk	Nested collections and polytypism, p. 11.
96/18	P.D.V. van der Stok	Real-Time Distributed Concurrency Control Algorithms with mixed time constraints, p. 71.
96/19	M.A. Reniers	Static Semantics of Message Sequence Charts, p. 71
96/20	L. Feijs	Algebraic Specification and Simulation of Lazy Functional Programs in a concurrent Environment, p. 27.
96/21	L. Bijlsma and R. Nederpelt	Predicate calculus: concepts and misconceptions, p. 26.