Inaugural lecture

Given on November 30th 2001
at the Technische Universiteit Eindhoven

making work flow

on the design, analysis and enactment of business processes

prof.dr.ir. W.M.P. van der Aalst

Introduction

Mr. Rector Magnificus, ladies and gentlemen,
Information technology has changed business processes within and between enterprises. More and more work processes are being conducted under the supervision of information systems that are driven by process models. Examples are workflow management systems such as Staffware, enterprise resource planning systems such as SAP and Baan, but also include many domain specific systems. It is hard to imagine enterprise information systems that are unaware of the processes taking place. Although the topic of business process management using information technology has already been addressed by consultants and software developers in depth, a more fundamental approach still is missing. Only since the nineties, scientists have started to work on the foundations of business process management systems. In this lecture I will address some of the scientific challenges we are facing and share some of my visions.

In the year 2000, I was appointed in both the department of Technology Management and the department of Mathematics and Computer Science. To avoid giving two talks, I will try to address the topic of this lecture from two scientific domains: computing science and operations management. You could argue that you are getting two talks for the price of one. You may also have noticed that this inaugural lecture is in English instead of Dutch. The reason is that many colleagues within the Information and Technology group, the Information Systems group, and the research school BETA do not speak Dutch. Therefore, I would consider it impolite to give this talk in Dutch.
The title of this lecture is 'Making work flow: on the design, analysis and enactment of business processes'. The goal is to show the relevance, architecture and Achilles heel of business process management systems. My definition of a business process management system is: *a generic software system that is driven by explicit process designs to enact and manage operational business processes*. The system should be process-aware and generic in the sense that it is possible to modify the processes it supports. The process designs are often graphical and the focus is on

structured processes that need to handle many cases. In the remainder of this talk, I will first put business process management and related technology in its historical context. Then, I will discuss models for process design. Since explicit models drive business process management systems, it is important to use the right techniques. Next, I will discuss techniques for the analysis of process models. I will argue that it is vital to have techniques to assert the correctness of workflow designs. Based on this I will focus on systems for process enactment, i.e., systems that actually make the 'work flow' based on a model of the processes and organizations involved. Finally, I will share my vision on future research, teaching, and cooperation.

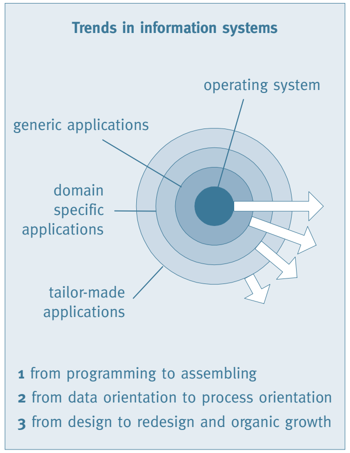
A historical perspective

"Only the wisest and stupidest of men never change."
Confucius

To show the relevance of business process management systems, it is interesting to put them into a historical perspective. Consider figure 1, which shows some of the ongoing trends in information systems. This figure shows that today's information systems consist of a number of layers. The center is formed by the operating system, i.e., the software that makes the hardware work. The second layer consists of generic applications that can be used in a wide range of enterprises. Moreover, these applications are typically used within multiple departments within the same enterprise. Examples of such generic applications are a database management system, a text editor, and a spreadsheet program. The third layer consists of domain specific applications. These applications are only used within specific types of enterprises and departments. Examples are decision support systems for vehicle routing, call center software, and human resource management software. The fourth layer consists of tailor-made applications. These applications are developed for specific organizations.

figure 1

Trends relevant for business process management.



In the sixties the second and third layer were missing. Information systems were built on top of a small operating system with limited functionality. Since no generic or domain specific software was available, these systems mainly consisted of tailor-made applications. Since then, the second and third layer have developed and the ongoing trend is that the four circles are increasing in size, i.e., they are moving to the outside while absorbing new functionality. Today's operating systems offer much more functionality. Database management systems that reside in the second layer offer functionality that used to be in tailor-made applications. As a result of this trend, the emphasis shifted from programming to assembling of complex software systems. The challenge no longer is the coding of individual modules but orchestrating and gluing together pieces of software from each of the four layers.
Another trend is the shift from data to processes. The seventies and eighties were dominated by data-driven approaches. The focus of information technology was on storing and retrieving information and as a result data modeling was the starting point for building an information system. The modeling of business processes was often neglected and processes had to adapt to information technology. Management trends such as business process reengineering illustrate the increased emphasis on processes. As a result, system engineers are resorting to a more process driven approach.
The last trend I would like to mention is the shift from carefully planned designs to redesign and organic growth. Due to the omnipresence of the Internet and its standards, information systems change on-the-fly. Few systems are built from scratch. In most cases existing applications are partly used in the new system. As a result, software development is much more dynamic.
The trends shown in figure 1 provide a historical context for business process management systems. Business process management systems are either separate applications residing in the second layer or are integrated components in the domain specific applications, i.e., the third layer. Notable examples of business process management systems residing in the second layer are workflow management systems [20, 22] such as Staffware, MQSeries, and COSA, and case handling systems such as FLOWer. Note that leading enterprise resource planning systems populating the third layer also offer a workflow management module. The workflow engines of SAP, Baan, PeopleSoft, Oracle, and JD Edwards can be considered as integrated business process management systems.

The idea to isolate the management of business processes in a separate component is consistent with the three trends identified. Business process management systems can be used to avoid hard-coding the work processes into tailor-made applications and thus support the shift from programming to assembling. Moreover, process orientation, redesign, and organic growth are supported. For example, today's workflow management systems can be used to integrate existing applications and support process change by merely changing the workflow diagram. Given these observations, I hope to have demonstrated the practical relevance of business process management systems. In the remainder of this lecture I will focus more on the scientific importance of these systems. Moreover, for clarity I will often restrict the discussion to clear cut business process management systems such as workflow management systems.
The early work on office information systems is an interesting starting point from scientific perspective. In the seventies, people like Skip Ellis [15], Anatol Holt [19], and Michael Zisman [33] already worked on so-called office information systems which were driven by explicit process models. It is interesting to see that the three pioneers in this area independently used Petri net variants to model office procedures. During the seventies and eighties there was great optimism about the applicability of office information systems. Unfortunately, few applications succeeded. As a result of these experiences both the application of this technology and research almost stopped for a decade. Consequently, hardly any advances were made in the eighties. In the nineties, there again was a huge interest in these systems. The number of workflow management systems developed in the past decade and the many papers on workflow technology illustrate the revival of office information systems. Today workflow management systems are readily available [22]. However, their application is still limited to specific industries such as banking and insurance. As was indicated by Skip Ellis it is important to learn from these ups and downs [16]. The failures in the eighties can be explained by both technical and conceptual problems. In the eighties networks were slow or absent, there were no suitable graphical interfaces, and proper development software was missing. However, there were also more fundamental problems: a unified way of modeling processes was missing and the systems were too rigid to be used by people in the workplace. Most of the technical problems have been resolved by now. However, the more conceptual problems remain.

figure 2

Good standards for business process modeling are still missing and even today's workflow management systems are too rigid.

Whirlwind (1953)
photo from the Timeline of Events on Computer History (© 2001 IEEE).

Architecture: 32 bit word length, duplex CPU, 75kips single address, no interrupts, 4 index registers, real time clock
Memory: magnetic core (4Kx64word) 6 micro-seconds cycle time; magnetic drum (150K word); 4 IBM Model 729 tape drives (~100K word each); parity checking
I/O: CRT display, keyboard, light gun, real time serial data (teletype 1300bps modem), voice line
Size: 60,000 vacuum tubes, 175,000 diodes, 13,000 transistors; CPU space 50x150 feet each; CPU weight 500,000 lbs; power consumption: 3 megawatts

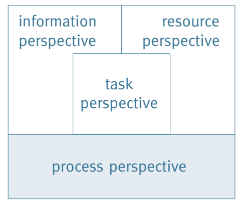
One of the great challenges of business process management systems is to offer both support and flexibility [6, 9, 21]. Today's systems are typically are too rigid, thus forcing people to work around the system. One of the problems is that software developers and computer scientists are typically inspired by processes inside a computer system rather than processes outside a computer. Figure 2 illustrates the typical mind-frame of people developing business process management systems. This photograph shows the Whirlwind computer, which was the first computer system to have a magnetic core memory (1953). It is interesting to mention that Whirlwind was developed by Jay Forrester who also developed the well-known Systems Dynamics approach [17]. Software engineers are typically trained in the architecture and systems software of computers like the Whirlwind and its successors. As a result, these engineers think in terms of control systems rather than support systems. This explains that few of the existing workflow management systems allow for the so-called implicit choice, i.e., a choice resolved by the environment rather than the system [8].
To summarize: I would like to state that, although the relevance of business process management systems is undisputed, many fundamental problems remain to be solved. In the remainder of this lecture I will try to shed light on some of these problems.

Models for process design

"Before we work on artificial intelligence why don't we do something about natural stupidity?"
Steve Polyak

Business process management systems are driven by models of processes and organizations. By changing these models, the behavior of the system adapts to its environment and changing requirements. These models cover different perspectives. Figure 3 shows some of the perspectives relevant for business process management systems [20]. The process perspective describes the control-flow, i.e., the ordering of tasks. The information perspective describes the data that are used. The resource perspective describes the structure of the organization and identifies resources, roles, and groups. The task perspective describes the content of individual steps in the processes. Each perspective is relevant. However, the process perspective is dominant for the type of systems addressed in this talk.
Many techniques have been proposed to model the process perspective. Some of these techniques are informal in the sense that the diagrams used have no formally defined semantics.

figure 3



Perspectives of models driving business process management systems.

These models are typically very intuitive and the interpretation shifts depending on the modeler, application domain, and characteristics of the business processes at hand. Examples of informal techniques are ISAC, DFD, SADT, and IDEF. These techniques serve well for the discussion of work processes. However, they are inadequate for directly driving information systems since they are incomplete and subject to multiple interpretations. Therefore, more precise modeling methods are required.

figure 4

WF-net.

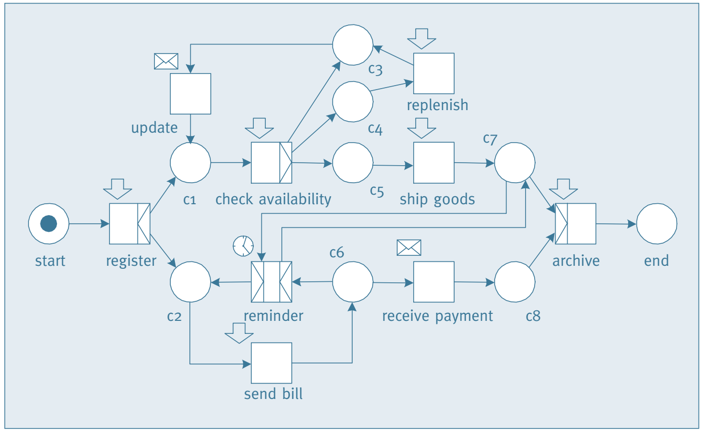


Figure 4 shows an example of an order handling process modeled in terms of a so-called workflow net [1]. Workflow nets are based on the classical Petri net model invented by Carl Adam Petri in the early sixties [24]. The squares are the active parts of the model and correspond to tasks. The circles are the passive parts of the model and are used to represent states.
In the classical Petri net the squares are named transitions and the circles places. A workflow net models the lifecycle of one case. Examples of cases are insurance claims, tax declarations, and traffic violations. Cases are represented by tokens and in this case the token in *start* corresponds to an order. Task *register* is a so-called AND-split and is enabled in the state shown. The arrow indicates that this task requires human intervention. If a person executes this task, the token is removed from place *start* and two tokens are produced: one for *c1* and one for *c2*. Then two tasks are enabled in parallel: *check availability* and *send bill*. Depending on the eagerness of the workers executing these two tasks either *check availability* or *send bill* is executed first. Suppose *check availability* is executed first. If the ordered goods are available, they can be shipped by executing task *ship goods*. If they are not available, either a replenishment order is issued or not. Note that *check availability* is an OR-split and produces one token for *c3*, *c4*, or *c5*. Suppose that not all ordered goods are available but the appropriate replenishment orders were already issued. A token is produced for *c3* and task *update* becomes enabled. Suppose that at that point in time task *send bill* is executed resulting in the state with a token in *c3* and *c6*. The token in *c6* enables two tasks. However, only one of these tasks can be executed. Task *receive payment* can be executed the moment the payment is received. Task *reminder* is an AND-join/AND-split and is blocked until the bill is sent and the goods have been shipped. Note that the reminder is sent after a specified period as indicated by the clock symbol. However, it is only possible to send a reminder if the goods have been actually shipped.
Assume that in the state with a token in *c3* and *c6* task *update* is executed. This task does not require human involvement and is triggered by a message of the warehouse indicating that relevant goods have arrived. Again *check availability* is enabled. Suppose that this task is executed and the result is positive. In the resulting state *ship goods* can be executed. Now there is a token in *c6* and *c7* thus enabling task *reminder*. Executing task *reminder* again enables the task *send bill*. A new copy of the bill is sent with the appropriate text. It is possible to send several reminders by alternating *reminder* and *send bill*. However, let us assume that after the first loop the customer pays resulting in a state with a token in *c7* and *c8*. In this state the AND-join *archive* is enabled and executing this task results in the final state with a token in *end*.
This very simple workflow net shows some of the routing constructs relevant for business process modeling. Sequential, parallel, conditional and iterative routing are present in this model. There also are more advanced constructs such as the choice between *receive payment* and *reminder*. This is a so-called *implicit choice* since it is not resolved by the system but by the environment of the system. The moment the bill is sent, it is undetermined whether *receive payment* and *reminder* will be the next step in the process. Another advanced construct is the fact that task *reminder* is blocked until the goods have been shipped. The latter construct is a so-called *milestone*. The reason that I point out both constructs is that many systems have problems supporting these rather fundamental process patterns [8].
Workflow nets have clear semantics. The fact that I'm able to play the so-called token game using a minimal set of rules shows the fact that these models are executable. None of the informal techniques mentioned before (i.e., ISAC, DFD, SADT, and IDEF) have formal semantics. Besides these informal techniques, there are many formal

techniques too. Examples are the many variants of process algebra [10] and state charts [18]. The reason we prefer to use a variant of Petri nets is threefold [1]:

- Petri nets are graphical and yet precise.
- Petri nets offer an abundance of analysis techniques.
- Petri nets treat states as first class citizens.

The latter point deserves some more explanation. Many techniques for business process modeling exclusively focus on the active parts of the process, i.e., the tasks. This is very strange since in many administrative processes the actual processing time is measured in minutes and the flow time is measured in days. This means that most of the time cases are in-between two subsequent tasks. Therefore, it is vital to model these states explicitly.
In recent years, UML (Unified Modeling Language, [12]) has become the de facto standard for software development. UML has four diagrams for process modeling. UML supports variants of statecharts and its activity diagrams are inspired by Petri nets. UML combines both good and bad ideas and can be considered semi-formal. Many colleagues are trying to provide solid semantics for UML. In my opinion, it would have been better to start with a solid foundation.

Techniques for process analysis

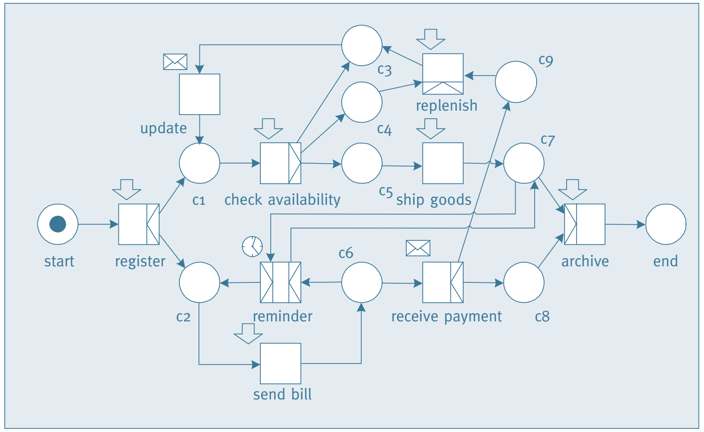
"From the errors of others, a wise man corrects his own."
Syrus

Business process management systems allow organizations to change their processes by merely changing the models. The models are typically graphical and can be changed quite easily. This provides more flexibility than conventional information systems. However, by reducing the threshold for change, errors are introduced more easily. Therefore, it is important to develop suitable analysis techniques. However, it is not sufficient to just develop these techniques. It is as least as important to look at methods and tools to make them applicable in a practical context. Traditionally, most techniques used for the analysis of business processes, originate from operations research. All students taking courses in operations management will learn to apply techniques such as simulation, queueing theory, and Markovian analysis. The focus mainly is on *performance analysis* and less attention is paid to the correctness of models. *Verification* and *validation* are often neglected. As a result, systems fail by not providing the right support or even break down [2, 26]. Verification is needed to check whether the resulting system is free of logical errors. Many process designs suffer from deadlocks and livelocks that could have been detected using verification techniques. Validation is needed to check whether the system actually behaves as expected. Note that validation is context dependent while verification is not. A system that deadlocks is not correct in any situation. Therefore, verifying whether a system exhibits deadlocks is context independent. Validation is context dependent and can only be done with knowledge of the intended business process.
To illustrate the relevance of validation and verification and to demonstrate some of the techniques available, we return to the workflow net shown in figure 4. This workflow process allows for the situation where a replenishment is issued before any payment is received. Suppose that we want to change the design such that replenishments are delayed until receiving payment. An obvious way to model this is to connect task *receive payment* with *replenish* using an additional place *c9* as shown in figure 5. Although this extension seems to be correct at first glance, the resulting workflow net exhibits several errors. The workflow will deadlock if a second replenishment is needed and something is left

behind in the process if no replenishments are needed. These are logical errors that can be detected without any knowledge of the order handling process. For verification, application independent notions of correctness are needed. One of these notions is the so-called *soundness property* [1]. A workflow net is sound if an only if the workflow contains no dead parts (i.e., tasks that can never be executed), from any reachable state it is always possible to terminate, and the moment the workflow terminates all places except the sink place are empty. Note that soundness rules out logical errors such as deadlocks and livelocks.

figure 5

An incorrect WF-net.

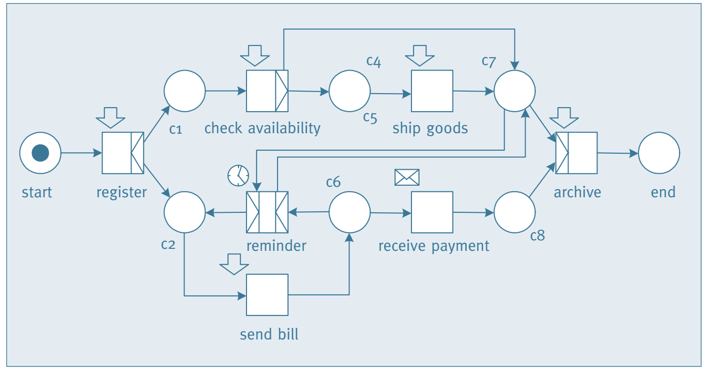


The notion of soundness is applicable to any workflow language. An interesting observation is that soundness corresponds to liveness and boundedness of the short-circuited net [1]. The latter properties have been studied extensively [25, 13]. As a result, powerful analysis techniques and tools can be applied to verify the correctness of a workflow design. Practical experience shows that many errors can be detected by verifying the soundness property. Moreover, Petri net theory can also be applied to guide the designer towards the error. Soundness does not guarantee that the workflow net behaves as intended. Consider for example, the workflow net shown in figure 6. Compared to the original model, the shipment of goods is skipped if some of the goods are not available. Again this may seem to be a good idea at first glance. However, customers are expected to pay even if the

goods are never delivered. In other words, task *receive payment* needs to be executed although task *ship goods* may never be executed. The latter error can only be detected using knowledge about the context. Based on this context one may decide whether this is acceptable or not. Few analysis techniques exist to automatically support this kind of validation. The only means of validation offered by today's workflow management systems is gaming and simulation.

figure 6

A sound but incorrect WF-net.



An interesting technique to support validation is the inheritance of dynamic behavior. Inheritance can be used as a technique to compare processes. Inheritance relates subclasses with superclasses [11]. A workflow net is a subclass of a superclass workflow net if certain dynamic properties are preserved. A subclass typically contains more tasks. If by hiding and/or blocking tasks in the subclass one obtains the superclass, in which the subclass inherits the dynamics of the superclass[1]. The superclass can be used to specify the minimal properties the workflow design should satisfy. By merely checking whether the actual design is a subclass of the superclass, one can validate the essential properties. Consider for example figure 7. This workflow net describes the minimal requirements the order handling process should satisfy. The tasks *register, ship goods, receive payment* and *archive* are mandatory. Tasks *ship goods* and *receive payment* may be executed in

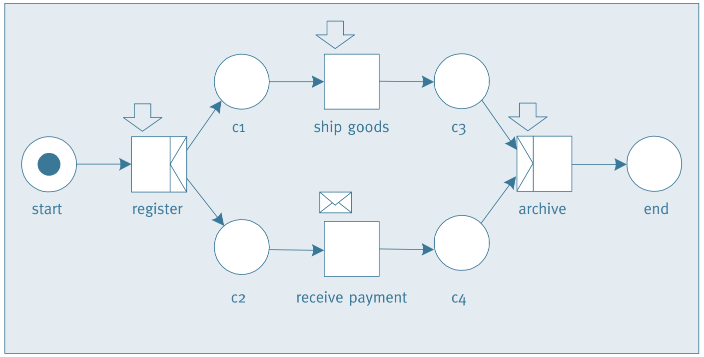
1 We have identified four notions of inheritance. In this lecture, we only refer to life-cycle inheritance.

parallel, but should be preceded by *register* and followed by *archive*. The original order handling process shown in figure 4 is a subclass of this superclass. Therefore, the minimal requirements are satisfied. However, the order handling process shown in figure 6 is not a subclass. The fact that task *ship goods* can be skipped demonstrates that not all properties are preserved.

Inheritance of dynamic behavior is a very powerful concept that has many applications. Inheritance-preserving transformation rules and transfer rules offer support at design-time and at run-time [5]. Subclass-superclass relationships also can be used to enforce correct processes in an E-commerce setting. If business partners only execute subclass processes of some common contract process, then the overall workflow will be executed as agreed.

figure 7

A superclass WF-net.



It should be noted that workflows crossing the borders of organizations are particularly challenging from a verification and validation point of view [4]. Errors resulting from miscommunication between business partners are highly disruptive and costly. Therefore, it is important to develop techniques and tools for the verification and validation of these processes.

figure 8

A screenshot showing the verification and validation capabilities of Woflan.

Few tools aiming at the verification of workflow processes exist. Woflan [29] and Flowmake [26] are two notable exceptions. We have been working on Woflan since 1997. Figure 8 shows a screenshot of Woflan. Woflan combines state-of-the-art scientific results with practical applications [3, 7, 29, 31]. Woflan can interface with leading workflow management systems such as Staffware and COSA. It can also interface with BPR tools such as Protos. Workflow processes designed using any of these tools can be verified for correctness. It turns out that the challenge is not to decide whether the design is sound or not. The real challenge is to provide diagnostic information that guides the designer to the error. Woflan also supports the inheritance notions mentioned before. Given two workflow designs, Woflan is able to decide whether one is a subclass of the other. Tools such as Woflan illustrate the benefits of a more fundamental approach.

Systems for process enactment

"If the automobile had followed the same development cycle as the computer, a Rolls-Royce would today cost $100, get one million miles to the gallon, and explode once a year, killing everyone inside."
Robert Cringely

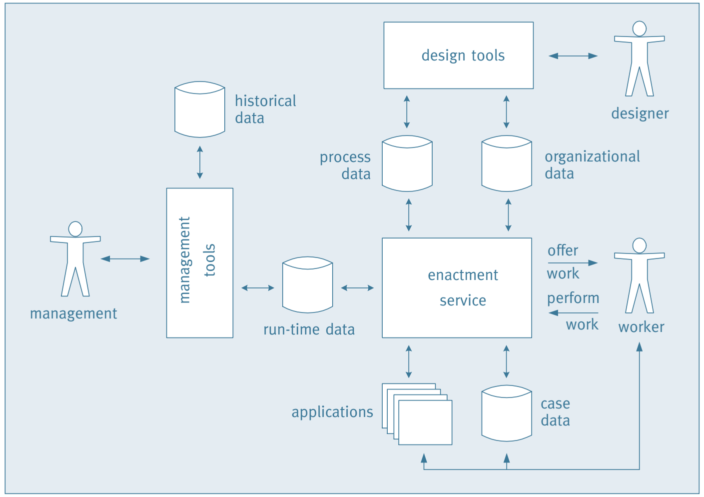
Progress in computer hardware has been incredible. In 1964 Gordon Moore, predicted that the number of elements on a produced chip would double every 18 months[2]. Up until now, Moore's law still applies. Information technology has also resulted in a spectacular growth of the information being gathered. The commonly used term 'information overload' illustrates this growth. It is estimated that for each individual, i.e., child, man, and woman, 250 megabytes of data are gathered each year [23]. The Internet and the World-Wide-Web have made an abundance of information available at low costs. However, despite the apparent progress in computer hardware and information processing, many information systems leave much to be desired. Typically, software contains errors and people need to work around the system to get things done. These observations justify the use of solid models and analysis techniques, as discussed before.
Thus far, the focus of this lecture has been on the design and analysis of work processes. Now it is time to focus on the systems to enact these work processes. Figure 9 shows the typical architecture of a business process management system. The designer uses the design tools to create models describing the processes and the structure of the organization. The manager uses management tools to monitor the flow of work and act if necessary. The worker interacts with the enactment service. The enactment service can offer work to workers and workers can search, select and perform work. To support the execution of tasks, the enactment service may launch various kinds of applications. Note that the enactment service is the core of the system deciding on 'what', 'how', 'when' and 'by whom'. Clearly, the enactment service is driven by models of the processes and the organizations.

2 Moore (founder of Intel), commenting on the growth of the microelectronics industry in 1964, noted a doubling of the number of elements on a produced chip once every 12 months. For a decade, that meant a growth factor of approximately 1000. Today, when Moore's Law is quoted, the time constant typically quoted is 18 months. However, some argue that a constant of 24 months is more appropriate.

By merely changing these models the system evolves and adapts. This is the ultimate promise of business process management systems.

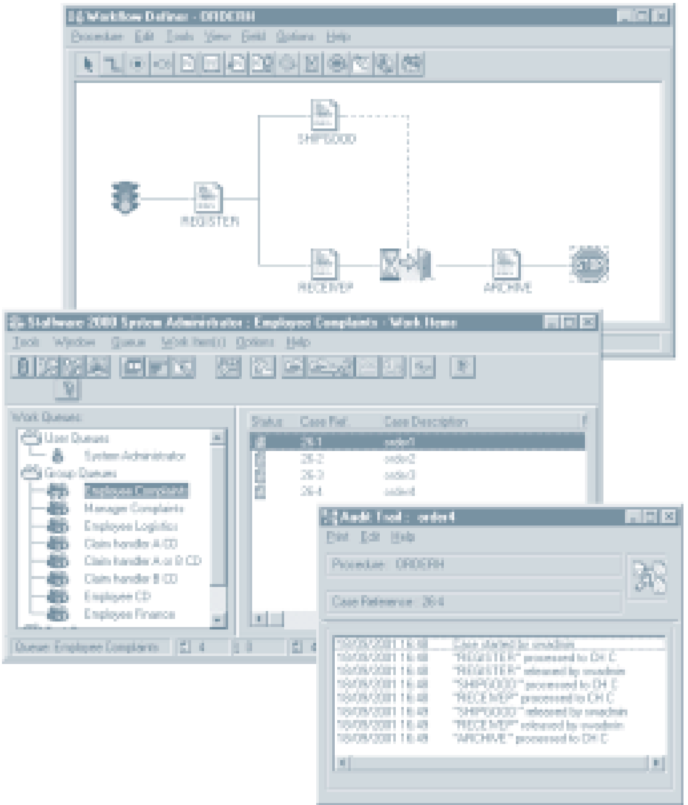
figure 9

The architecture of a business process management system.



Today's workflow management systems have an architecture consistent with figure 9. Consider for example the screenshots of Staffware shown in figure 10. Staffware is one of the leading workflow management systems. The top window shows the design tool of Staffware defining a simple workflow process. Work is offered through so-called work queues. One worker can have multiple work queues and one work queue can be shared among multiple workers. The window in the middle shows the set of available work queues (left) and the content of one of these work queues (right). The bottom window shows an audit trail of a case. The three windows show only some of the capabilities offered by contemporary workflow management systems. It is fairly straightforward to map these windows onto the architecture. In other processes-aware information systems, such as for example enterprise resource planning systems, one will find the architecture shown in figure 9 embedded in a larger architecture.

figure 10



The Graphical
Workflow Definer,
Work Queue,
and Audit Trail
of Staffware.

Despite the acceptance of process-aware information systems, the current generation of products leaves much to be desired. To illustrate this, we focus on the current generation of workflow management systems. I will use figure 9 to identify five problems.
First of all, there is a lack of good standards for workflow management. There is no good standard, for example, for exchanging process models. Existing formats [22] have no clearly defined semantics and fail to capture many routing constructs [27]. Current standards for workflow

management are incomplete, inconsistent, at the wrong abstraction level, and mainly driven by the commercial interests of workflow vendors.
Second, the expressive power, i.e., the ability to represent complex work processes, of the current generation of workflow management systems is insufficient. We have evaluated 12 workflow management systems using a set of desirable workflow patterns. This evaluation revealed that even the leading workflow management systems do not support more than half of these patterns [8, 32]. As an example, consider the workflow process shown in figure 4. Few systems are able to handle the implicit choice and milestone construct identified before.
A third problem is the lack of understanding of how people actually work [28, 30]. Work processes are more than the ordering of tasks. Work is embedded in a social context. A better understanding of this context is needed to make systems socially aware as well.
The fourth problem is the limited support for workflow analysis. As indicated before, there are powerful techniques for workflow analysis. However, few systems embed advanced analysis techniques [27]. Besides model-based verification, validation and performance analysis, new types of analysis are possible. The combination of historical and run-time data on the one hand and workflow designs on the other, offers breathtaking possibilities. Historical data can be used to obtain stochastic data about routing and timing. Using run-time data to reconstruct the current state in a simulation model allows for *on-the-fly simulation*. Simulation based on the current state, historical data, and a good model offers high-quality information about potential problems in the near future. Historical data can also be used for *workflow mining*. The goal of workflow mining is to derive process models from transaction logs.
Finally, many technical problems remain. Some of these problems can be resolved using Internet-based technology and standards. However, many problems related to the integration of components and long-lived transactions remain unsolved.
Research should focus on the five problems just mentioned. Note that the problems require input from multiple disciplines including computing science, operations management, and social sciences. It appears that the research school BETA can provide an ideal platform for such an endeavour.

Outlook

"A scientific truth does not triumph by convincing its opponents and making them see the light, but rather because its opponents eventually die and a new generation grows up that is familiar with it."
Maxwell Planck

In this lecture, I have described some of the challenges related to the design, analysis and enactment of business processes. My personal goal is to solve some of the problems identified. More specifically, I would like to continue to work on the following topics:

- workflow flexibility
- dynamic change
- case handling
- workflow mining
- business patterns
- inheritance of behavior
- component-based systems
- cross-organizational support
- proclets
- data-driven process design
- architectures for process-aware systems
- and our research prototypes Woflan and XRL/flower.

To conduct high-quality research on these topics, it is important to create an environment that on the one hand allows for a truly scientific attitude but on the other also values interaction with industry. It is also important that in such environment, researchers have access to industry-standard software packages and support to build research prototypes. One of the initiatives in this direction is the Eindhoven Digital Laboratory for Business Processes [14]. Another challenge is the development of a high-quality program for the new Bachelor and Master's structure. The new structure provides the opportunity to improve the quality and alignment of our education. Moreover, it also enables joint efforts of both departments towards a Master of Business Information Systems.

Thanks

Finally, I would like to thank a number of people. I am grateful to all my fellow scientists whose beautiful papers inspired me to pursue an academic career. I thank my current and former colleagues in both the I&T and IS groups. In particular, I would like to thank Eric Verbeek for his work on Woflan, Twan Basten for his work on inheritance, and Jaap Wessels and Kees van Hee for their support while writing my PhD thesis. Kees van Hee has always been a source of inspiration and it will be a pleasure to continue working with him. I would also like to thank colleagues from other universities that I had the pleasure of working with. In particular, I would like to thank Arthur ter Hofstede, Akhil Kumar, Jörg Desel, Stefan Jablonski, Amit Sheth, Andreas Oberweis, Rüdiger Valk, Daniel Moldt and Skip Ellis. Finally, I would like to thank my parents, friends and family. I would like to thank my wife Karin for her loving support and Anne and Willem for being the moon and sunshine in our daily life.

References

1 W.M.P. van der Aalst. The Application of Petri Nets to Workflow Management. *The Journal of Circuits, Systems and Computers*, 8(1):21–66, 1998.

2 W.M.P. van der Aalst. Formalization and Verification of Event-driven Process Chains. *Information and Software Technology*, 41(10): 639–650, 1999.

3 W.M.P. van der Aalst. Woflan: A Petri-net-based Workflow Analyzer. *Systems Analysis - Modelling - Simulation*, 35(3):345–357, 1999.

4 W.M.P. van der Aalst. Loosely Coupled Interorganizational Workflows: Modeling and Analyzing Workflows Crossing Organizational Boundaries. *Information and Management*, 37(2): 67–75, March 2000.

5 W.M.P. van der Aalst and T. Basten. Inheritance of Workflows: An Approach to Tackling Problems Related to Change. *Theoretical Computer Science*, 2001 (to appear).

6 W.M.P. van der Aalst, J. Desel, and A. Oberweis, editors. *Business Process Management: Models, Techniques, and Empirical Studies*, volume 1806 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin, 2000.

7 W.M.P. van der Aalst and A.H.M. ter Hofstede. Verification of Workflow Task Structures: A Petri-net-based Approach. *Information Systems*, 25(1):43–69, 2000.

8 W.M.P. van der Aalst, A.H.M. ter Hofstede, B. Kiepuszewski and A.P. Barros. Advanced Workflow Patterns. In O. Etzion and P. Scheuermann, editors, *7th International Conference on Cooperative Information Systems (CoopIS 2000)*, volume 1901 of *Lecture Notes in Computer Science*, pages 18–29. Springer-Verlag, Berlin, 2000.

9 W.M.P. van der Aalst and S. Jablonski, editors. *Flexible Workflow Technology Driving the Networked Economy*, Special Issue of the International Journal of Computer Systems, Science and Engineering, volume 15, number 5. CRL Publishing Ltd, 2000.

10 J.C.M. Baeten and W.P. Weijland. *Process Algebra*, volume 18 of *Cambridge tracts in theoretical computer science*. Cambridge University Press, Cambridge, 1990.

11 T. Basten and W.M.P. van der Aalst. Inheritance of Behavior. *Journal of Logic and Algebraic Programming*, 47(2):47–145, 2001.

12 G. Booch, J. Rumbaugh, and I. Jacobson. *The Unified Modeling Language User Guide*. Addison Wesley, Reading, MA, USA, 1998.

13 J. Desel and J. Esparza. *Free Choice Petri Nets*, volume 40 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, Cambridge, UK, 1995.

14 Eindhoven Digital Laboratory for Business Processes. http://www.tm.tue.nl/it/research/edlbp.

15 C.A. Ellis. Information Control Nets: A Mathematical Model of Office Information Flow. In *Proceedings of the Conference on Simulation, Measurement and Modeling of Computer Systems*, pages 225–240, Boulder, Colorado, 1979. ACM Press.

16 C.A. Ellis and G. Nutt. Workflow: The Process Spectrum. In A. Sheth, editor, *Proceedings of the NSF Workshop on Workflow and Process Automation in Information Systems*, pages 140–145, Athens, Georgia, May 1996.

17 J.W. Forrester. *Industrial Dynamics*. MIT Press, Cambridge, MA, 1968.

18 D. Harel. Statecharts: A Visual Formalism for Complex Systems. *Science of Computer Programming*, 8:231–274, 1987.

19 A.W. Holt. Coordination Technology and Petri Nets. In G. Rozenberg, editor, *Advances in Petri Nets 1985*, volume 222 of *Lecture Notes in Computer Science*, pages 278–296. Springer-Verlag, Berlin, 1985.

20 S. Jablonski and C. Bussler. *Workflow Management: Modeling Concepts, Architecture, and Implementation*. International Thomson Computer Press, London, UK, 1996.

21 M. Klein, C. Dellarocas, and A. Bernstein, editors. *Adaptive Workflow Systems*, Special Issue of Computer Supported Cooperative Work, 2000.

22 P. Lawrence, editor. *Workflow Handbook 1997, Workflow Management Coalition*. John Wiley and Sons, New York, 1997.

23 P. Lyman and H. Varian. How Much Information. http://www.sims.berkeley.edu/how-much-info.

24 C.A. Petri. *Kommunikation mit Automaten*. PhD thesis, Institut für instrumentelle Mathematik, Bonn, 1962.

25 W. Reisig and G. Rozenberg, editors. *Lectures on Petri Nets I: Basic Models*, volume 1491 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin, 1998.

26 W. Sadiq and M.E. Orlowska. Analyzing Process Models using Graph Reduction Techniques. *Information Systems*, 25(2):117–134, 2000.

27 A.P. Sheth, W.M.P. van der Aalst, and I.B. Arpinar. Processes Driving the Networked Economy: ProcessPortals, ProcessVortex and Dynamically Trading Processes. *IEEE Concurrency*, 7(3):18–31, 1999.

28 L.A. Suchman. Office Procedures as Practical Action: Models of Work and System Design. *ACM Transactions on Office Information Systems*, 1(4):320–328, 1983.

29 H.M.W. Verbeek, T. Basten and W.M.P. van der Aalst. Diagnosing Workflow Processes using Woflan. *The Computer Journal*, 44(4):246–279, 2001.

30 T. Winograd and F. Flores. *Understanding Computers and Cognition: A New Foundation for Design*. Ablex, Norwood, 1986.

31 Woflan Home Page. http://www.tm.tue.nl/it/woflan.

32 Workflow Patterns Home Page. http://www.tm.tue.nl/it/research/patterns.

33 M.D. Zisman. *Representation, Specification and Automation of Office Procedures*. PhD thesis, University of Pennsylvania, Warton School of Business, 1977.

Curriculum Vitae

Prof.dr.ir. Wil van der Aalst (1966) is a full professor of Information Systems and head of the section of Information and Technology of the department of Technology Management at the Technische Universiteit Eindhoven. He is also a part-time full professor at the Computing Science section at the department of Mathematics and Computer Science at the same university.
He holds an MSc in Computing Science (1988) and a PhD in Mathematics (1992) awarded by the Technische Universiteit Eindhoven. From 1992 until 1999 he worked as an assistant/associate professor for the department of Mathematics and Computer Science at the Technische Universiteit Eindhoven, where he supervised the SMIS research group from 1996 until 1999.
From 1993-1998 he also worked as a part-time consultant for Bakkenist. He has been a visiting professor to several universities including the Universität Karlsruhe (5 months), the University of Georgia (5 months) and the University of Colorado (8 months).
Wil van der Aalst directs the Eindhoven Digital Laboratory for Business Processes (EDL-BP) and is a fellow and management team member of the research institute BETA. His research interests include information systems, simulation, Petri nets, process models, workflow management systems, verification techniques, enterprise resource planning systems, computer supported cooperative work, and interorganizational business processes.

Colofon

Productie:
Communicatie Service
Centrum TU/e

Fotografie cover:
Rob Stork, Eindhoven

Ontwerp:
Plaza ontwerpers,
Eindhoven

Druk:
Drukkerij Lecturis,
Eindhoven

ISBN: 90-386-1272-9