# AN EVALUATION OF CASE HANDLING SYSTEMS FOR PRODUCT BASED WORKFLOW DESIGN

Irene Vanderfeesten, Hajo A. Reijers and Wil M. P. van der Aalst
*Department of Technology Management, Technische Universiteit Eindhoven*
*PO Box 513, 5600 MB Eindhoven, The Netherlands*

Keywords: Product Based Workflow Design, Case handling systems, Business Process Redesign, Business Process Management.

Abstract: *Case handling systems* offer a solution to the lack of flexibility and adaptability in workflow management systems. Because they are data driven they potentially provide good support for *Product Based Workflow Design* (PBWD). In this paper we investigate to which degree current case handling systems (FLOWer and Activity Manager) are able to support PBWD. This is done by elaborating the design process of a case from industry in both systems. From this evaluation we concluded that current case handling systems are not yet completely ready for supporting PBWD. Therefore, we recognize that better tool support is needed to make PBWD more suitable for practical use.

## 1 INTRODUCTION

In the past decades, process-orientation has gained a strong foothold in various fields, notably in the business management and information systems disciplines. This is illustrated by the emergence of process-oriented transformation approaches, like Business Process Redesign (BPR) (Davenport, 1993; Hammer & Champy, 1993), on the one hand and process-aware information systems, like workflow technology (van der Aalst & van Hee, 2002), on the other. With this rise, the historic focus on the *data* that is being processed within businesses settings - and by information systems in particular - has blurred. It should be remembered that during the 70s and 80s the majority of information systems development projects would start with a thorough data analysis, leading to conceptual data models, while nowadays similar projects typically start with mapping the business to be supported in the form of process models.

Recently, nothing short of a *data revival* has set in in the Business Process Management (BPM) community, bringing back attention for data aspects. This phenomenon can be distinguished in at least two places. Firstly, various problematic issues with workflow and BPM systems are being countered with the introduction of systems that put much more emphasis on the data that is being handled (e.g. case handling systems (van der Aalst & Berens, 2001; van der Aalst, Weske & Grünbauer, 2005)), in this way moving away from a purely control-flow centric perspective. Secondly, innovative BPR approaches are emerging that, rather counter-intuitively, take business data processing requirements as starting point for generating a new business process design (e.g. Reijers, Limam, & van der Aalst, 2003; Sun & Zhao, 2004).

In this paper, we will investigate to what extent synchronous movements towards a higher data awareness in the fields of (i) workflow management and (ii) business process design can mutually reenforce each other. In the recent past, we have worked on the development and application of the method of Product-Based Workflow Design (PBWD). This method takes a static description of an (information) product as a starting point to derive an improved process design. The idea to focus on the product instead of on an existing process when redesigning a process was introduced by (van der Aalst, 1999) and is based on a similar approach in manufacturing processes. Since its conception, this method has been worked out in some detail (Reijers, 2003; Reijers, Limam & van der Aalst, 2003; Reijers & Vanderfeesten, 2004) and has been successfully applied in industry in over a dozen of occasions. At the same time, the manual application of PBWD in



Vanderfeesten I., A. Reijers H. and M. P. van der Aalst W. (2007).
AN EVALUATION OF CASE HANDLING SYSTEMS FOR PRODUCT BASED WORKFLOW DESIGN.
In *Proceedings of the Ninth International Conference on Enterprise Information Systems - ISAS*, pages 39-46
DOI: 10.5220/0002356600390046

practice proves to be a time-consuming and error-prone affair. It is likely that the absence of automated tools to support the application of PBWD hinders the wider adoption of the method, despite its successes in bringing back cycle time and service times of actual business processes with 30% or more (Reijers, 2003). On the road to the development of PBWD support tools, it seems wise to consider some of the existing tools that could already deliver (partial) support for the application of PBWD. A notable candidate for such support would be current case handling technology. After all, just like traditional workflow management systems, case handling systems operate on the basis of a pre-defined process model. In contrast to workflow technology, however, case handling systems implement various data management features (van der Aalst, Weske & Grünbauer, 2005).

The objectives of the paper can now be formulated as follows: (i) to determine whether the concepts of PBWD can be translated to the concepts of current case handling systems, (ii) to establish to what extent build-time features of case handling systems support the design of workflow models based on PBWD, and (iii) to find out how current case handling tools could be enhanced to support PBWD. Fulfilling these objectives could also be useful to determine the desirable features of a specifically tailored support tool for PBWD, i.e. without using current case handling systems.

The structure of this paper is as follows. In the next two sections, we will shortly review case handling systems and the PBWD method respectively, forming the fundaments of this paper. In Section 4, we will present our assessment of two existing case handling technologies, i.e. Pallas Athena's FLOWer and BPI's Activity Manager. To conclude the paper, we present the major implications from our assessment and directions for further research.

## 2 CASE HANDLING SYSTEMS

Traditional workflow and BPM systems are characterized by well-known limitations in terms of flexibility and adaptability (van der Aalst & Jablonski, 2000). These limitations can be associated with the dominant paradigm for process modelling found in these systems, which is almost exclusively activity-centric (Dumas, van der Aalst & ter Hofstede, 2005). The lack of flexibility and adaptability leads to many problems and inhibits a broader use of workflow technology. In recent years many authors have discussed the problem (van der Aalst & Jablonski, 2000; Agostini & De Michelis, 2000; Casati et al, 1996; Ellis & Keddara, 2000; Herrmann et al, 2000, Klein, Dellaroca & Bernstein, 1998 and 2000) and different solution strategies have been proposed. Basically, there are three ways to provide more flexibility:

- *Dynamic change* (Ellis & Keddara, 2000; Reichert & Dadam, 1998; Rinderle, Reichert & Dadam, 2004).
- *Worklets* (Adams et al, 2005; Staffware, 2003; Weske, 2001), and
- *Case handling* (van der Aalst & Berens, 2001; van der Aalst, Weske & Grünbauer, 2005).

The basic idea of *dynamic change* is to allow changes at run-time, i.e., while work is being performed processes may be adapted (van der Aalst & Jablonski, 2000; Ellis & Keddara, 2000; Reichert & Dadam, 1998; Rinderle, Reichert & Dadam, 2004). Clearly, dynamic change mechanisms can be used to support flexibility and adaptability.

A dynamic change may refer to a single case (i.e., process instance) or multiple cases (e.g., all running instances of a process). Both changes at the instance level and the type level may introduce inconsistencies, e.g., data may be missing or activities are unintentionally skipped or executed multiple times. A well-known problem is the "dynamic change bug" which occurs when the ordering of activities changes or the process is made more sequential (Ellis & Keddara, 2000). These issues have been addressed by systems such as ADEPT (Reichert & Dadam, 1998; Rinderle, Reichert & Dadam, 2004). Such a system can safeguard the consistency of a process. However, an additional complication is that the people changing the processes should be able to modify process models and truly understand the effects of a change on the *whole* process. In real-life applications, with hundreds of tasks, few people are qualified to make such changes.

*Worklets* (Adams et al, 2005) allow for flexibility and adaptability by the late binding of process fragments. Activities in a process are not bound to a concrete application or subprocess and only when they need to be executed a concrete application or subprocess is selected. YAWL (van der Aalst & ter Hofstede, 2005) is an example of a system that implements this idea. In YAWL activities may be handled by a worklet handler, this handler uses an extensible set of ripple-down rules to select the right worklet (i.e., a concrete application or subprocess). Similar ideas have been proposed by other authors

(e.g., Weske, 2001) and even implemented in commercial systems (cf. the Staffware extension that allows for process fragments (Staffware, 2003)). Although the worklets mechanism is easier to be used by end-users than most dynamic change mechanisms, the scope is limited and only particular forms of flexibility and adaptability can be supported.

*Case handling* is another paradigm for supporting flexible and knowledge intensive business processes. The concept of *case handling* offers a solution to the lack of flexibility in traditional workflow systems (van der Aalst, Weske & Grünbauer, 2005). Case handling is supporting knowledge intensive business processes and focuses on what *can* be done instead of on what *should* be done. To support this, a case handling system is much more data driven than a workflow system. The central concept for case handling is the *case* and not the routing of work or the activities. The case is the product that is manufactured in the process based on the data that is processed. The core features of case handling are (van der Aalst & Berens, 2001; van der Aalst, Weske & Grünbauer, 2005):

- to avoid context tunneling by providing all information available (i.e., present the case as a whole rather than showing just bits and pieces),
- to decide which activities are enabled on the basis of the information available rather than the activities already executed,
- to separate work distribution from authorization and allow for additional types of roles, not just the execute role,
- to allow workers to view and add/modify data before or after the corresponding activities have been executed (e.g., information can be registered the moment it becomes available).

These core features of case handling are supported by systems such as FLOWer (Pallas Athena, 2006). Other systems such as BPi's Activity Manager (GYATA BPi, 2006) only support some of these features. Unlike dynamic change and worklets, case handling provides implicit flexibility, i.e., there is no need to change a process model or to select a particular worklet. Moreover, as the list of core features suggests, case handling takes a broader perspective by also incorporating aspects as work distribution and information collection.

# 3 PBWD

Product Based Workflow Design, or in short PBWD, (van der Aalst, 1999; Reijers, 2003; Reijers, Limam & van der Aalst, 2003; Reijers & Vanderfeesten, 2004) is a *revolutionary* approach to workflow process design. It is revolutionary because a clean-sheet of paper is taken to design the complete process from scratch. Rather than the activities and the workflow process itself, it takes the processing of data and the workflow end product as the central concepts. This approach has several advantages that are described in (Reijers, 2003; Vanderfeesten, van der Aalst & Reijers, 2005). The most important advantage is that PBWD is *rational*. In the first place because the product specification is taken as the basis for a workflow design, each recognized information element and each production rule can be justified and verified with this specification. As a consequence there are no unnecessary tasks in the resulting workflow. Secondly, the ordering of (tasks with) production rules themselves is completely driven by the performance targets of the design effort.

The workflow product is represented by a *Product Data Model* (PDM), i.e. a network structure of the components of the product. The approach of PBWD is very similar to the way in which manufacturing processes are structured. This will be explained in more detail in the remainder of this section.

Section 3.1 shortly describes the similar concepts in manufacturing, while Section 3.2 subsequently elaborates on the important concepts of PBWD. Finally, Section 3.3 introduces an industry case as an example of PBWD, which is used throughout the assessment of the two concrete systems, as summarized in Section 4.

## 3.1 Bill-Of-Material (BOM)

In manufacturing, often a static representation of the product is used to organise the assembly lines. Figure 1 shows such a representation for the assembly of a car. A car is made of 4 wheels, a chassis, and an engine. The structure of the assembly line can be derived from the picture as follows: first, the four wheels and the chassis are put together, resulting in a subassembly product. Next, the final assembly takes place by putting the subassembly product and the engine together. The result is a car. The representation of the product and its parts is referred to as the Bill-Of-Material (BOM) (Orlicky, 1972) and is also used in information systems, e.g.

MRP- and ERP-systems for production planning and control.
Manufacturing and service-oriented processes have a lot in common (Platier, 1996), e.g. process management in both domains focuses on the routing of work and the allocation of work to resources. Because of these similarities it was considered worthwhile to explore the applicability of some concepts from the field of manufacturing to administrative and information intensive processes (referred to as *workflow processes*). The PBWD method derives a process model from the structure of an (administrative) product. This product structure is represented with a PDM, as explained in the next section.

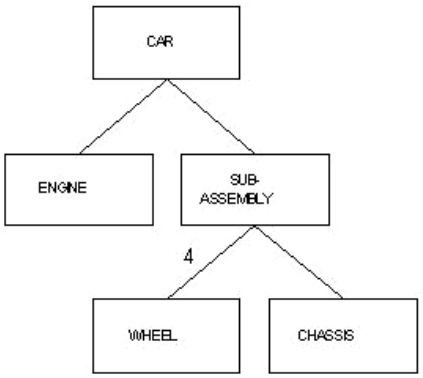


Figure 1: The Bill of Material (BOM) of a car.

## 3.2 Product Data Model (PDM)

The product of a workflow process can be an insurance claim, a mortgage request, a social benefits grant, etc. Similar to a BOM, a PDM of this product can be made. However, the building blocks are not the physical parts that have to be assembled, but the data elements (e.g. name, birth date, amount of salary, type of insurance and register of holidays) that have to be processed to achieve new data.

Figure 2 contains a small and simple example, comparable to the simple BOM of the car in Figure 1. It describes the decision whether an applicant is allowed for a training to become a helicopter pilot (see also Reijers, 2003). Persons that want to become a helicopter pilot should meet some requirements: they should be healthy, their eye-sight should be excellent, they should pass a psychological assessment, and they should not have been rejected in the previous two years. The figure shows that the final decision whether a person can become a helicopter pilot (data element a) is dependent either on the data elements (b) and (c), or on (f), or on (d). In reality, these different combinations reflect the different conditions under which certain operations can be executed. In case there is a result of a recent suitability test (d), this information directly determines the outcome (a). Also, in case the value for the quality of eye-sight of the applicant is bad (f) this directly leads to a rejection (a). In the other cases, the results of both a psychological (b) and a physical test (c) are needed. One level lower, the physical test (c) consists of the results for the quality of reflexes (e) and for the quality of eye-sight (f).

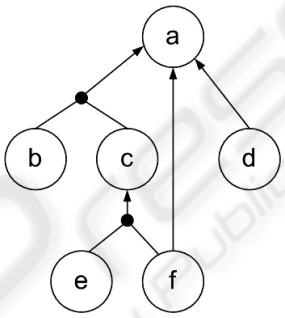


Figure 2: The product data model which represents the decision on the suitability to become a helicopter pilot. The meaning of the elements is as follows: (a) decision for suitability to become a helicopter pilot, (b) psychological fitness, (c) physical fitness, (d) latest result of suitability test in the previous two years, (e) quality of reflexes, (f) quality of eye-sight.

The *data elements* of the PDM are depicted as circles. The *operations* on these data elements are represented by arcs. The arcs are 'knotted' together when the data elements are all needed to execute the particular operation. Compare, for instance, the arcs from (b) and (c) leading to (a) on the one hand, to the arc from (d) leading to (a) on the other in
Figure 2. In the latter case only one data element is needed to determine the outcome of (a), while in the case of (b) and (c) both elements are needed to produce (a).

The helicopter pilot example, which we discussed here, is very small. Typically, in industry the PDMs are much larger; possibly containing hundreds of data elements. Based on such a PDM, a workflow process model can be obtained by grouping data elements and operations into activities (see also Reijers, 2003; Reijers & Vanderfeesten, 2004), as will be illustrated in the next section.

## 3.3 The GAK Case

In this section we introduce a case from industry as a motivating example. This example is used in the assessment of the two contemporary case handling systems, as described in Section 4. The subject of the case study is the GAK agency (currently known as UWV) which is responsible for awarding unemployment benefits in the Netherlands. The process in question deals with the decision that the GAK has to make on whether or not to grant such benefits once a request has been received. The typical factors that should be taken into account are the reason for the applicant to have become unemployed, the length of the period that the previous job was held, and the coverage regulations.
The PDM for the GAK case is shown in Figure 3. A detailed description of the case and of the data elements can be found in (Reijers, 2003). The next section describes how we have assessed the process of design in two contemporary case handling systems based on the GAK PDM. For this assessment we have used the process model as it was manually derived from the PDM in earlier work. Because of space limitations we can not show the resulting process model here. However, it can be found in (Reijers, 2003).

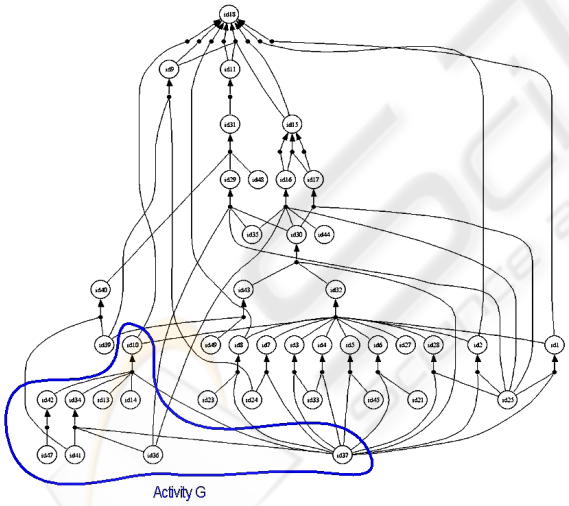


Figure 3: The PDM for the GAK case.

Designing a process model from a PDM mainly comes down to grouping data elements and operations in a smart way, considering several context constraints and requirements on the structure (e.g. the processing order should be determined such that the expected number of additional work at any point in the process is minimized for the average case). As an illustration, we have indicitated in Figure 3 the part of the GAK PDM that corresponds to activity G in the resulting process model (i.e. G contains the data elements *id10, id13, id14, id34, id36, id37, id41, id42*, and *id47* and their corresponding operations.)

# 4 ASSESSMENT

As was explained in the introduction, workflow management systems focus on the control-flow perspective, while case handling systems are more data-driven. Because of their focus on data, case handling systems may provide support for PBWD. In order to investigate their possibilities and potential support for PBWD, we have selected two case handling systems:

- *FLOWer* is a case handling system developed by Pallas Athena (Pallas Athena, 2006). It consists of a number of components, of which FLOWer Studio is the graphical design environment. FLOWer Studio is used at build-time to define case definitions consisting of activities, precedences, data objects, roles and forms.
- *Activity Manager* by BPi is an "add-on" that can be used in combination with a workflow management system, such as COSA and Staffware (Kaan, Reijers & van der Molen, 2006). For demonstration purposes also a stand-alone version can be used. In this research we used this stand-alone version because it is easier to manage. Activity Manager combines the structure and control of a workflow management system with the flexibility of case handling. It imports the process model from the workflow management system via a database and provides the means to further define the activities in this model by elaborating the operations.

When considering the PBWD method in detail, we think a system that supports this method in a proper way should at least provide for the following:

- a means to define and view the product structure.
- a way to define and view the content of each activity (in terms of data elements and their relationships).
- proper support for the process of designing a process model based on the PDM (for example, it should give the designer some freedom to

play around with different designs and groupings of operations and data elements).

In (Vanderfeesten, van der Aalst & Reijers, 2006) we have elaborated in detail on the way in which PBWD can be used to design a process model in FLOWer and Activity Manager, describing all steps taken to translate the PDM into the process model. It is illustrated with a series of screenshots for both systems (Vanderfeesten, van der Aalst & Reijers, 2006). The focus in both assessments is on the process of designing and defining the process model based on the PDM[i]. In general, the following steps should be taken and supported by the system to get from a PDM to a process model:

1. The PDM must be translated to the specific system. This means that either the data elements or the operations (or both) must be mapped on concepts in the system and subsequently be specified.
2. The activities must be defined as groups of data elements and/or operations. There must be an easy way to transfer an operation or data element from one activity to another, as a way of exploring various designs. Also, the correct order of activities must be defined, because precedence relationships should be respected.
3. The process model must be finalized with for instance information on resources, conditions, or activity duration.

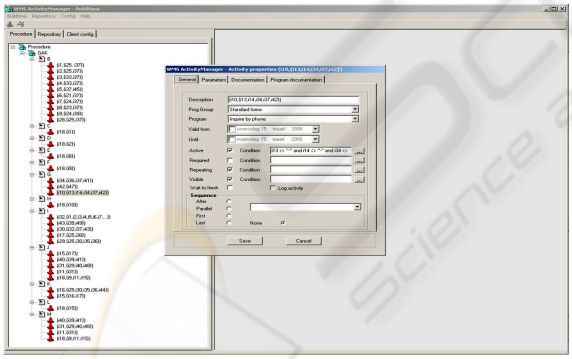

Figure 4: Screenshot of the design environment of Activity Manager. Note that on the left-hand side all activities are summarized and their content is shown. For example, the content of activity G corresponds to the data elements and operations indicated in Figure 2. The data elements are represented by their identifiers (e.g. "id29") and operations are represented by tuples (e.g. (id1,{id25, id37})) with one output element and a set of one or more input elements. For a more elaborate explanation we refer to (Vanderfeesten, van der Aalst & Reijers, 2006).

From our evaluation we can conclude that it was not at all straightforward to follow these general steps in both systems[ii]. Therefore, we feel the systems do not match all requirements that were stated above. For example, they both did not provide a way to represent the product structure. In both systems it is possible to somehow define data elements (in FLOWer this was easier than in Activity Manager), but the concept of operations and their dependencies is less clear to capture with these systems. Since operations are the main building blocks for activities, the lack of a clear notion of operations in the tool might hamper the design process.

Moreover, Activity Manager does not provide the means to start with defining the data elements. First, the order of activities has to be established in this system. This means that there is less freedom in grouping operations to activities. On the other hand, it was possible to map all concepts from the standard workflow terminology (i.e. process, activity, operation, and data element) to concepts in Activity Manager. In principle, this mapping should allow for a smoother embedding of PBWD within the Activity Manager.

In comparison, FLOWer could not map all workflow concepts (there was no equivalent for an operation), but it was possible to easily define, view and change the content of an activity. A more extensive discussion on these two tools can be found in (Vanderfeesten, van der Aalst & Reijers, 2006).

# 5 CONCLUSION

In this paper we have investigated to what extent current case handling systems are able to support PBWD by evaluating FLOWer and Activity Manager. Both systems still put some emphasis on the control-flow of the process, despite of their innovative focus on data. However, in FLOWer we really can start reasoning from the PDM (i.e. by starting with the definition of data elements and their dependencies). This provides the opportunity to really focus on the grouping of data elements instead of on the definition of activities directly. By putting groups of data elements on one form and playing around with these combinations it is possible to compose activities based on the data and operations of the PDM instead of first defining the activities and afterwards determining what should be done in these activities.

By contrast, BPi's Activity Manager is considerably more process driven than data driven, as it starts from the definition of a process model. Of

course, this follows from the fact that Activity Manager is "added on" to a workflow system, which only allows Activity Manager to further specify the process structure already given. Because of this, it is not possible to directly design a process model which is purely based on a PDM. The user needs to have a good understanding of how the activities are organized and what the content of each activity should be. This means that the process of designing a process model based on the PDM should then be done outside the tool, in such a way that the result (i.e. the activities including their operations) can be implemented in the system. This violates our third requirement, i.e. that the tool itself should provide some support in the design process. Taking this design perspective we can remark that FLOWer offered the best assistance in creating a process model based on the product structure.

Looking at the evaluation from a conceptual viewpoint, we can conclude that both systems do not (yet) provide a facility to display the PDM as a hierarchical structure. Therefore, this would be a nice extension in order to use these systems as PBWD support tools. However, all concepts of the PDM and PBWD could be mapped to concepts in Activity Manager, while FLOWer is able to represent all concepts except for the operations.

This evaluation shows that current case handling systems, and thus current workflow technology in general, are not yet completely ready for PBWD. The research challenge now is to develop good support for applying this method in practice. The first contribution of this assessment is an overview of how existing systems can be improved to support PBWD. In close cooperation with suppliers of case handling systems we will further investigate the opportunities of using their systems. Secondly, we have learned some lessons for the development of specific tools for PBWD support. It seems to be important to (i) display and edit the PDM in the tool, and (ii) to somehow circumvent direct relations from activities to data elements. Finally, future work will focus on the discovery and collection of data elements and their relationships (i.e. the derivation of a PDM). At this point in time, the ProM import framework for process mining (van der Aalst et al, 2003) already supports mining based on data elements (Guenther & van der Aalst, 2006). On a general level, this research shows that current workflow technology is *not* neutral towards the kind of process design. Even data-focused technology, such as case handling systems, still needs some control-flow information right from the start of the design process.

# ACKNOWLEDGEMENTS

This research is supported by the Technology Foundation STW, applied science division of NWO and the technology programme of the Dutch Ministry of Economic Affairs. We gratefully acknowledge the technical assistance from Pallas Athena and Gyata BPI.

# REFERENCES

Aalst, W.M.P. van der, 1999. On the automatic generation of workflow processes based on product structures. *Computers in Industry*, 39, pp. 97-111.

Aalst, W.M.P. van der, Berens, P.J.S., 2001. *Beyond workflow management: product-driven case handling*. In Ellis, S., Rodden, T., and Zigurs, I., editors, International ACM SIGGROUP *Conference on Supporting Group Work* (GROUP 2001), pp. 42-51, ACM Press, New York.

Aalst, W.M.P. van der, Hee, K.M. van, 2002. *Workflow management: models, methods, and systems*. MIT press, Cambridge, MA.

Aalst, W.M.P. van der, Hofstede, A.H.M. ter, 2005. YAWL: Yet Another Workflow Language. *Information Systems*, 30 (4), pp. 245-275.

Aalst, W.M.P. van der, Jablonski, S., 2000. Dealing with workflow change: identification of issues and solutions. *International Journal of Computer Systems, Science, and Engineering*, 15 (5), pp. 267-276.

Aalst, W.M.P. van der, Dongen, B.F. van, Herbst, J., Maruster, L., Schimm, G., and Weijters, A.J.J.M., 2003. Workflow mining: a survey of issues and approaches. *Data and Knowledge Engineering*, 47 (2), pp. 237-267.

Aalst, W.M.P. van der, Weske, M., Grünbauer, D., 2005. Case handling: a new paradigm for business process support. *Data and Knowledge Engineering*, 53 (2), pp. 129- 162.

Adams, M., Hofstede, A.H.M. ter, Edmond, D., and Aalst, W.M.P. van der, 2005. *Facilitating flexibility and dynamic exception handling in workflows*. In: Belo, O., Eder, J., Pastor, O., and Falcao e Cunha, J., editors, *Proceedings of the CAiSE'05 Forum*, pp. 45-50, FEUP, Porto, Portugal.

Agostini, A., De Michelis, G., 2000. *Improving flexibility of workflow managment systems.* In: Aalst, W.M.P. van der, Desel, J., and Oberweis, A., editors, *Business Process Management: models, techniques and empirical studies*, volume 1806 of Lecture Notes in Computer Science, pp. 218-234, Springer-Verlag, Berlin.

Casati, F., Ceri, S., Pernici, B., and Pozzi, G., 1996. Workflow Evolution. In: *Proceedings of ER '96*, pp. 438-455, Cottubus, Germany.

Davenport, T.H., 1993. *Process innovation: reengineering work through information technology*. Harvard Business School Press, Boston.

Dumas, M., Aalst, W.M.P. van der, Hofstede, A.H.M. ter, 2005. *Process-aware information systems: bridging people and software through process technology*. Wiley & Sons.

Ellis, C.A., Keddara, K., 2000. *A workflow change is a worklfow*. In: Aalst, W.M.P. van der, Desel, J., and Oberweis, A., editors, *Business Process Management: models, techniques and empirical studies*, volume 1806 of Lecture Noted in Computer Science, pp. 201-217, Springer-Verlag, Berlin.

Guenther, C.W., Aalst, W.M.P. van der, 2006. *Mining activity clusters from low-level event logs*. BPM report, http://is.tm.tue.nl/staff/wvdaalst/BPMcenter/-reports.htm .

GYATA BPi, 2006. GYATA BPi website. http://www.-gyatabpi.com, retrieved on November 15, 2006.

Hammer, M., Champy, J., 1993. *Reengineering the corporation*. Nicolas Brealey Publishing, London.

Herrmann, T., Hoffmann, M., Loser, K.U., and Moysich, K., 2000. *Semistructured models are surprisingly useful for user-centered design*. In: De Michelis, G., Giboin, A. Karsenty, L., and Dieng, R., editors, *Designing Cooperative Systems* (Coop 2000), pp. 159-174, IOS Press, Amsterdam.

Kaan, K., Reijers, H.A., and Molen, P. van der, 2006. *Introducing case mangagement: opening workflow management's black box*. In S. Dustdar, J.L. Fiadeiro, and A. Sheth, editors, *Proceedings of the 4th International Conference Business Process Management (BPM 2006*), volume 4102 of Lecture Notes in Computer Science, pp. 358-367, Springer Verlag, Berlin, 2006.

Klein, M., Dellarocas, C., and Bernstein, A., editors, 1998. *Proceedings of the CSCW-98 Workshop Towards Adaptive Workflow Systems*, Seattle, Washington.

Klein, M., Dellarocas, C., and Bernstein, A., editors, 2000. Adaptive Workflow Systems, volume 9 of *Special issue of the journal of Computer Supported Cooperative Work*.

Orlicky, J.A., 1972. *Structuring the bill of materials for MRP*. Production and Inventory Management, pp. 19-42.

Pallas Athena, 2006. Pallas Athena website. http://www.pallas-athena.com, retrieved on November 15, 2006.

Platier, E.A.H., 1996. *A logistical view on business processes: BPR and WFM concepts* (in Dutch). PhD thesis, Eindhoven University of Technology, Eindhoven, the Netherlands.

Reichert, M., Dadam, P., 1998. ADEPTflex: supporting dynamic changes of workflow without loosing control. *Journal of Intelligent Information Systems*, 10 (2), pp. 93-129.

Reijers, H.A., 2003. *Design and control of workflow processes: business process managment for the service industry*, volume 2617 of Lecture Notes in Computers Science, Springer-Verlag, Berlin.

Reijers, H.A., Limam, S., and Aalst, W.M.P. van der, 2003. Product-based workflow design. *Journal of Management Information Systems*, 20 (1), pp. 229-262.

Reijers, H.A., Vanderfeesten, I.T.P., 2004. *Cohesion and coupling metrics for workflow process design*. In: Desel, J., Pernici, B., and Weske, M., editors, *International Conference on Business Process management* (BPM 2004), volume 3080 of Lecture Notes in Computer Science, pp. 290-305, Springer-Verlag, Berlin.

Rinderle, S., Reichert, M., and Dadam, P., 2004. Correctness criteria for dynamic changes in workflow systems: a survey. *Data and Knowledge Engineering*, 50 (1), pp. 9-34.

Staffware, 2003. *Staffware Process Suite Version 2 – White Paper*. Staffware PLC, Maidenhead, UK.

Sun, S.X., Zhao, J.L., 2004. *A data flow approach to workflow design*. In: Proceedings of the 14th Workshop on Information Technology and Systems (WITS'04), pp. 80-85.

Vanderfeesten, I., Aalst, W.M.P. van der, and Reijers, H.A., 2005. *Modelling a product based workflow System in CPN Tools*. In: Jensen, K., editor, Proceedings of the 6th workshop on the practical use of coloured Petri Nets and CPN Tools (CPN 2005), volume 576 of DAIMI, pp. 99-118, Aarhus, Denmark.

Vanderfeesten, I., Reijers, H.A., and Aalst, W.M.P. van der, 2006. *Product Based Workflow Design with Case Handling Systems*. Beta Working Paper, no. 189, Technische Universiteit Eindhoven, Eindhoven. Available from http://www.tm.tue.nl/beta/ .

Weske, M., 2001. *Formal foundation and conceptual design of dynamic adaptations in a workflow management system*. In: Sprague, R., editor, Proceedings of the thirty-fourth annual Hawaii International Conference on Systems Science (HICSS-34). IEEE Computer Society Press, Los Alamitos, California.

[i] Note that the process of designing and defining a process model based on a PDM is different from the common way in practice to design a process model. Instead of using a subjective workshop setting (i.e. interviews, observations, etc.) to discover the process model, a more objective approach is used starting from the product structure.

[ii] A detailed description of the translation of the PDM to a process model in both systems can be found in (Vanderfeesten, van der Aalst & Reijers, 2006), including two series of screenshots for the two systems.