Document Workflow Optimization

H.M. Dortmans
Research & Development
Océ-Technologies bv
Po Box 101, 5900 MA Venlo,
The Netherlands
E-mail: hdo@oce.nl

L.J. Somers
Dept. of Math. and Comp. Science
Eindhoven University of Technology
Po Box 513, 5600 MB Eindhoven,
The Netherlands
E-mail: wsinlou@win.tue.nl

KEYWORDS
Workflow, Petri Nets, Structured Documents

ABSTRACT

This paper describes a new way of modeling and improving the workflow related to structured documents. A so called *Document Workflow Net* is defined. This executable model makes it possible to link document workflow to document structure. It is shown that this model can be used to simulate and to improve existing workflows.

INTRODUCTION

Workflow management is getting quite some attention nowadays, due to the need of modern businesses to improve and automate their business processes. According to the Workflow Management Coalition (WfMC 1995; WfMC 1999), workflow management is generally concerned with the automation of procedures where documents, information or tasks are passed between participants according to a defined set of rules, to achieve, or contribute to, an overall business goal.

Document Workflow Management focuses more specifically on the processing of documents, e.g. the creation of a technical document (set) by a collection of authors, or the scanning, image manipulations, and printing operations performed inside a document print center. Companies and in-house departments involved in document management and processing could improve their operation considerably when their current business processes could be optimized and automated.

Workflow management for structured documents has been the subject of other publications. Some publications (CIP4 2001; McClatchey et al. 1998) do not define a formal model. Others (Weitz 1998; Aalst 1997) do not take workflow improvement into account. We have taken the approach of defining an executable model, covering both document structure and process structure. Based on this model we show how to improve existing workflows.

By *not* considering the document as an atomic unit of information, but rather as a composite object, actually a hierarchical part structure, we can describe and simulate modern, real-world document production workflows. We will also show that the workflow as such can be improved by exploiting the parallelism that is implicitly present in such document structure. This could help document creation and production departments or businesses to decrease their turn-around time.

DOCUMENT WORKFLOW MODEL

Any workflow, whether dealing with the production of documents or not, can be described as a work breakdown structure, comprising a partially ordered set of atomic process steps, called *activities* or *tasks*. This structure can be formally modeled as has been shown in numerous publications, see e.g. (Ellis 1997; Aalst 1998).

Our model, however, describes not only the partial ordering of process steps, but also deals with the fact that most interesting documents have an internal structure. For example, a typical manual consists of chapters, where each chapter consists of sections. We model and exploit the fact that parts within this structure can often be processed concurrently.

Document Structure

Typical for interesting business documents (like user manuals, specification documents, etc.) is that they have an internal structure. For example, a typical manual consists of a number of chapters, where each chapter consists of sections. This structure is usually informally described in a *blueprint* before such a document is written and it is clearly visible afterwards in the table of contents of the completed document. In a manual production environment the basic components that make-up a document are often managed as reusable assets, using a suitable Content Management System.

In our model the Document Structure (DS) is the description of the hierarchical structure of a particular document-type. A DS is a rooted tree where each node represents a specific document-part-type (e.g. Chapter, Section)

A Document Structure is a directed graph where

- *one node, the root node (/), has no predecessor*
- *every other node has exactly one predecessor*
- *every node is labeled with a type identifier, i.e. a string identifying a document part type*

This definition implies that for every node there is a unique path from the root node to that node.

A DS Instance (DSI) is an instance of a particular DS where each node represents an instance of a document part type. For example, manual m could be an instance of document type M(anual) where chapter c1 of m is an instance of the part-type C(hapter). The DSI can typically be derived from a textual description, e.g. blueprint or table-of-contents, or a job definition (CIP4 2001) of the particular document that has to be produced.

Consider for example the case where the DS may describe that in general manuals contain a number of chapters and appendices, where chapters may be further divided into sections. The DS of a specific instance (m) of the manual document type could for instance define that there are in fact two chapters (c1, c2) and one appendix (a), where chapter c1 contains two sections (s11, s12) and chapter c2 also comprises two sections (s12, s22).

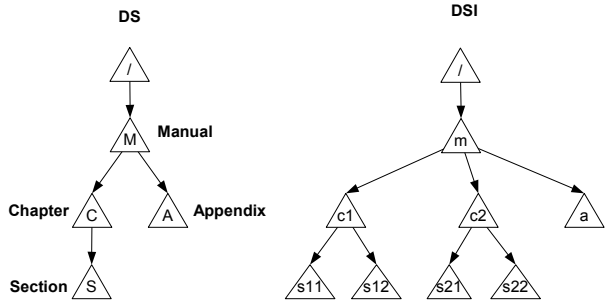

Figure 1 Document Structure of a Manual

Although not further used in this paper, we assume that each part, represent by a node in the DSI, is supposed to have a *state* that can be tested for resolving decisions during the execution of the workflow.

Process Structure

Petri-nets are considered well suited for modeling and simulation of all kinds of concurrent processes. Also the use of Petri-nets for modeling the work routing aspects of workflows can easily be justified (Aalst 1998):

- A clear and precise formal definition.
- Intuitive graphical representation
- Can model causality, choice, parallelism, iteration.
- Lots of research results, algorithms and proofs.

A classical Petri-net as introduced by Carl Adam Petri (Petri 1962; Peterson 1981; Reisig 1985) is a directed bipartite graph with two types of nodes: *places* and *transitions*. The nodes are connected via directed *arcs*. Connections between two nodes of the same type are not allowed.

A Petri-net is a triple (P,T,F) where:

- *P is a finite set of places,*
- *T is a finite set of transitions ($P \cap T = \varnothing$),*
- $F \subseteq (P \times T) \cup (T \times P)$ is a set of arcs (representing the flow relation between places and transitions)

A place *p* is called an *input place* of a transition *t* iff there exists a directed arc from *p* to *t*. Place *p* is called an *output place* of transition *t* iff there exists a directed arc from *t* to *p*.

At any moment of time a place is supposed to contain zero or more *tokens*. The state of the Petri-net, referred to as its *marking M*, is the current distribution of tokens over places: $M : P \rightarrow N$ (where N denotes the set of natural numbers).

Petri-nets can be illustrated graphically: places are represented by circles, transitions by rectangles and tokens as small dots within the places.

Petri-nets have well-defined *simulation semantics*:

- *A transition t is said to be enabled to fire iff each of its input places contains at least one token.*
- *If transition t fires, then t consumes one token from each of its input places and puts one token in each of its output places.*

In this paper we use a special kind of Petri-nets, the so called Workflow Nets (Aalst 1998). Workflow Nets have been used with success for describing, simulating and analyzing realistic workflow processes.

A Workflow-Net (WN) is a Petri-net that:

- *has two special places: i and o. Place i is a source place, i.e. has no input arcs, and place o is a sink place, i.e. has no output arcs;*
- *is strongly connected if we add an extra transition t* to it which connects place o with i.*

The workflow-net is initiated by putting a token in place i and is ready when a token occurs in place o. The second condition guards that there are no dangling transitions or places. They are all on some path from place i to place o.
We assume that the following routing primitives as defined in (WfMC1999) are sufficient for document workflows:

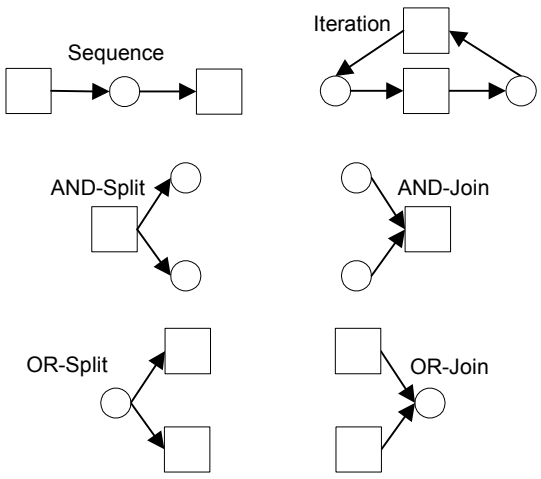

Figure 2 Workflow Routing Primitives

Note that a place in the net can be both part of an OR-Split and part of an OR-Join. Likewise any transition can be both part of an AND-Split and an AND-Join. Iteration is in fact a special combination of an OR-Join and an OR-Split.
In this paper we require document workflows to be *well structured* (Aalst 1996; Aalst 1998). We believe that this leaves us enough power to model and simulate most interesting documentation processes.

A Workflow-Net WN is well-structured if and only if:

- *For any pair of nodes x and y of $\overline{WN}$ (i.e. WN extended with an extra transition that connects place o to place i) and for any pair of elementary paths C_1 and C_2 leading from x to y: if the set of common nodes of C1 and C2 is {x, y} then C1 = C2.*

In other words, in a well-structured WN there is only one unique path from a transition to a place or from a place to a transition. This means that flows that are split by an OR-split, are joined by an OR-join. Likewise, flows split by an AND-split are joined by an AND-join.

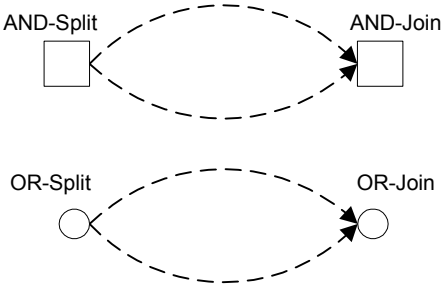

Figure 3 Well-Structured Split and Join

Well-structured nets have desirable dynamical properties (Aalst1996). They can be checked for "soundness", i.e. lifeness, boundedness, and proper termination, in polynomial time. A sound, well-structured net is safe, i.e. each place never contains more then one token. This represents in real life, that a condition holds or does not hold, or a resource is available or not available. Furthermore, a well-structured net is by definition a proper hierarchical nesting of subnets, that we will call blocks.

A *Block* within a well-structured Workflow Net is:

- *a single node, or*
- *the complete Workflow Net, or*
- *a subnet where all arcs entering the subnet do so only at one particular OR-Join or AND-Join node and all arcs exiting the subnet do so only at one particular OR-Split or AND-Split node*

A block is either a single node, the complete Workflow Net, or any subnet contained within an AND-Split/AND-Join pair or an OR-Split/OR-Join pair, including Iteration subnets.
It can easily be proven that blocks can be safely abstracted, refined, or replaced without invalidating the well-structuredness of the WN. A block can be replaced by another block. Each single node can be refined by replacing it with a block. Blocks can be recursively decomposed in smaller blocks. A block can be abstracted by replacing it by a single place or transition node.

Document Workflow Nets

We have shown how to describe the structure of a particular document type as a Document Structure (DS). The business process related to this particular document-type, can be described as a Workflow Net (WN).

In order to model, simulate and improve the workflow related to the creation and production of structured documents we introduce a new kind of net that we will call Document Workflow Net (DWN).

A Document Workflow Net is a graph comprising:

- *a Document Structure DS, where each node is labeled with a unique identifier,*
- *a WorkflowNet WN, where each place is labeled with a reference to a node in the DS, and a reference to an activity-type.*

By means of the labeling, we are linking WN-transitions to DS-nodes. These links represent the fact that performing an activity involves a particular document-part and vice-versa each document-part could be operated upon in one or more activities at different points of time.
Each activity-type label in the DWN denotes a reference to an atomic workflow operation, such as e.g. 'Write', 'Review', 'Publish', 'Print'.

Following example is a DWN for the manual DS that we have described earlier.

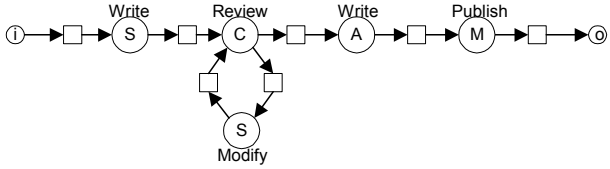

Figure 4 Document Workflow Net for a Manual

This DWN describes that first all sections of the manual are written. Then the complete chapter will be reviewed. If not ok, certain sections will be modified. When the review is positive, appendices will be written and finally the manual is published.
For a specific DS, using the same example, we can expand above DWN into a DWN instance. An instance of the DWN is generated by replacing each reference to node in the DS by (one or more) references to nodes in the DSI.

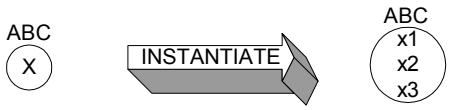

Figure 5 Instantiation

For instance, the type C is substituted by its instances (c1, c2), the actual chapters of the manual. This type substitution is equivalent to place refinement, i.e. replacing a place by a subnet with one entry place (OR-Join) and one exit place (OR-Split) as shown in figure 6. Because of the balanced use of OR-Join and OR-Split we safeguard that the substitution maintains the well-structuredness of the net.
Note that each place in the DWN in fact can be considered a small subnet in which the activity is linked to a transition (see top of figure 6), which is done in most other papers. The graphical notation we use in the DWN, however, turns out to be quite helpful in keeping drawings concise and in explaining algorithms.

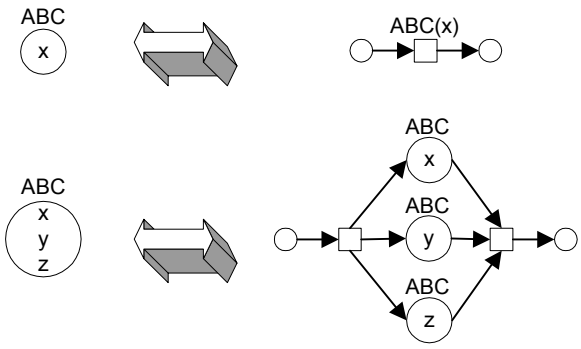

Figure 6 Graphical Notation

Simulation Semantics

Simulation semantics of Document Workflow nets is based on Petri-net transition firing rules, but extended with an additional rule. This "*Mutex Rule*" is defined as follows:

- *A node in the Document Structure can not be processed by more than one activity at a time*
- *Two nodes p and q in the Document Structure cannot be processed simultaneously when the path from the root node to p contains q, or the path from the root node to q contains p*

This rule prevents concurrent processing of a document part. It also prevents multiple document parts to be processed concurrently, when they have a 'built-from' or 'part-of relation'.

WORKFLOW IMPROVEMENT

Document Workflow Nets can not only be simulated based on the above described simulation semantics. We can also use the 'Mutex Rule' to remove unnecessary ordering of process steps, allowing more activities to be enabled concurrently. We will outline some simple but effective improvement steps.

Improving a Document Workflow Net

Given a particular DWN, we can generate an improved DWN as follows:

- *For each transition t that has exactly one input place a and one output place b, check the Mutex-Rule for the DS nodes referenced by a and b. When it turns out that there is no need for mutual exclusive treatment of these nodes, perform the following steps:*
- *Starting from a follow the flow downstream, not stepping into blocks and not exiting the current block, until a place y is found which label references a DS node that is to be handled mutual exclusive with the node that is referenced by a. If no such place is found use the last place encountered in the current block as y.*
- *Generate a new transition with a as input place and y as output place.*
- *Starting from b follow the flow upstream, not stepping into blocks and not exiting the current block, until a place x is found which label references a DS node that is to be handled mutual exclusive with the node that is referenced by a. If no such place is found use the last place encountered in the current block as x.*
- *Generate a new transition with x as input place and b as output place.*
- *Remove t*

By applying these rules we get the following improved DWN for the manual example of figures 1 and 4.

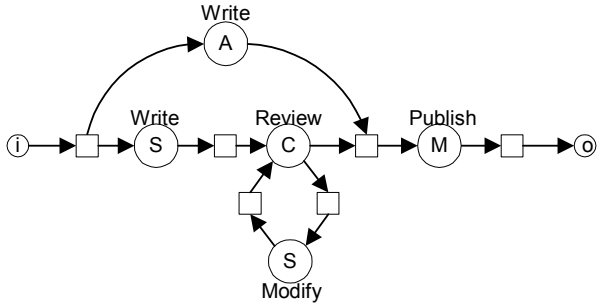

Figure 7 Improved Document Workflow Net

In this new DWN we see that the original transition between 'Review(C)' and 'Write(A)' is removed. Now appendices (A) could be written at the same time that sections (S) are written and chapters (C) are reviewed. Of course, this improvement can only be exploited in practice when enough human resources are available.

Improving a Document Workflow Net Instance

After having improved the Document Workflow Net, by only using information from the DS, we can even go a step further, at least for a particular document instance, by using information from the DSI. Let us look at the instantiated, improved DWN of the manual what this means.

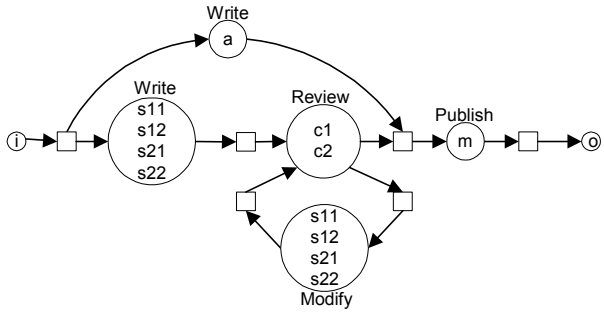

Figure 8 Instantiated Document Workflow Net

The DSI of manual m, which is a specific instance of the DS of all manuals M, shows that instances s11 and s12 of S only have a relation with instance c1 of C, and not with instance c2, etc. We can use this information to improve the Document Workflow Net a step further for this particular document instance.

An improvement rule, *Transition Splitting,* for instantiated DWN's can be outlined as follows:

- *For each transition t that has exactly one input and one output place, check the Mutex-Rule for each pair of node-references p and q, where p is taken from the input place and q is taken from the output place*
- *Remove t and replace it by a new transition for all combinations of groups of input and output node-references that need to be processed mutual exclusively*

Figure 9 illustrates this Transition Splitting rule for a transition with an input place with labels (x,y,z) and an output place with labels (m,n) where m is mutex with x, y (and not with z) and n is only mutex with z.

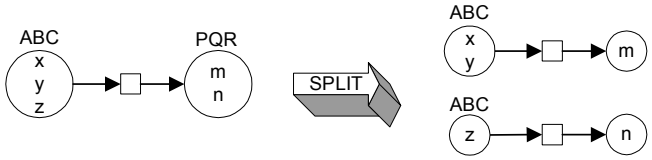

Figure 9 Transition Splitting

Applying the splitting rule to the instantiated DWN of figure 8 yields the following, even more parallelized DWN instance:

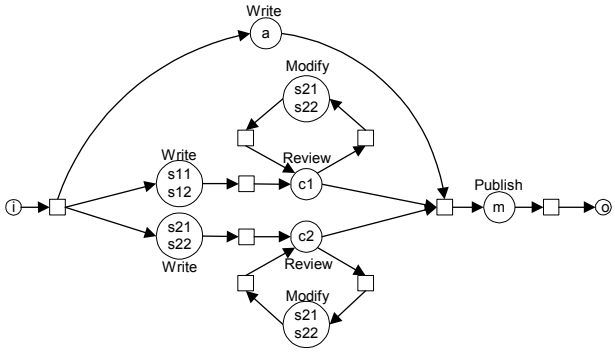

Figure 10 Improved Document Workflow Net Instance

CONCLUSIONS

In this paper we have introduced Document Workflow Nets to describe workflows that manipulate structured documents. By linking the description of the structure of a business document with the Petri-net description of the workflow we can better describe and simulate the workflow of realistic document production workflows, where documents are constructed and manipulated not as atomic objects, but as hierarchical combinations of (reusable) parts. To enable realistic simulation and analysis, the normal Petri-net simulation semantics is extended with a mutex-rule that prevents simultaneous operations on mutually interdependent parts of a structured document.

Furthermore, an algorithm has been outlined that, given a Document Workflow Net, can generate an improved one. The algorithm increases concurrency of activities in order to support increased operational productivity.

The results of this work, possibly extended with the concepts of "time", can be used to simulate and compare existing versus improved document workflows in real cases. They can also be used to build specialized workflow management software for document creation, imaging and production.

REFERENCES

Aalst W.M.P. van der. 1996. "Structural Characterizations of Sound Workflow Nets". Computing Science Reports 96/23, Eindhoven University of Technology, Eindhoven.

Aalst W.M.P. van der. 1997. "Designing workflows based on product structures." In *Proceedings of the ninth IASTED International Conference on Parallel and Distributed Computing Systems*, K. Li, S. Olariu, Y. Pan, and I. Stojmenovic (Eds.), IASTED/Acta Press, Anaheim, 337-342.

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

CIP4. 2001. "JDF Specification". International Cooperation for Integration of Processes in Prepress, Press and Postpress, Zurich, Switzerland, <http://www.cip4.org>.

Ellis, C.A. and G.J. Nutt. 1997. "Modelling and Enactment of Workflow Systems." In *Application and Theory of Petri Nets 1993*, M. Ajmone Marsan, (Eds.), Volume 691 of Lecture Notes in Computer Science, Springer-Verlag, Berlin, 1-16.

McClatchey R., Kovacs Z., Estrella F., Le Goff J-M., Chevenier G., Baker N., Lieunard S., Murray S., Le Flour T., Bazan A., "The Integration of Product Data and Workflow Management Systems in a Large Scale Engineering Database Application", In: *Proceedings of the 1998 International Database Engineering and Applications Symposium*, Cardiff, UK, 296-302 (Jul).

Peterson J.L. 1981. *Petri Net Theory and the Modelling of Systems*. Prentice-Hall, Englewoods Cliffs, New Jersey.

Petri C.A. 1962. "Kommunikation mit Automaten". PhD thesis (in German), Institut fur Instrumentelle Mathematik, Bonn.

Reisig W. 1985. "Petri nets. An introduction". In W. Brauer, G. Rozenberg, and A. Salomaa (Eds.), *EATCS, Monographs on Theoretical Computer Science volume 4*. Springer Verlag, Berlin.

Weitz W. 1998. "Combining Structured Documents with High-level Petri-nets for Workflow Modeling in Internet-based Commerce", *International Journal of Cooperative Information Systems,* 7(4), 275-296.

WfMC. 1995. "The Workflow Reference Model", Document Number WFMC-TC-1003, Issue 1.1, Workflow Management Coalition, Winchester, Hampshire, UK <http://www.wfmc.org>

WfMC. 1999. "Workflow Management Coalition Terminology and Glossary". Document Number WFMC-TC-1011, Issue 3.0, Workflow Management Coalition, Winchester, Hampshire, UK <http://www.wfmc.org>.