

**This is an electronic reprint of the original article.
This reprint *may differ* from the original in pagination and typographic detail.**

Author(s): Nurminen, Miika; Honkaranta, Anne; Kärkkäinen, Tommi

Title: ProcMiner: Advancing Process Analysis and Management

Year: 2007

Version:

Please cite the original version:

Nurminen, M., Honkaranta, A., & Kärkkäinen, T. (2007). ProcMiner: Advancing Process Analysis and Management. In Proceedings of the Workshop on Text Data Mining and Management (TDMM). April 15, Istanbul, Turkey (pp. 760-769). IEEE Computer Society. <https://doi.org/10.1109/ICDEW.2007.4401065>

All material supplied via JYX is protected by copyright and other intellectual property rights, and duplication or sale of all or part of any of the repository collections is not permitted, except that material may be duplicated by you for your research use or educational purposes in electronic or print form. You must obtain permission for any other use. Electronic or print copies may not be offered, whether for sale or otherwise to anyone who is not an authorised user.

ProcMiner: Advancing Process Analysis and Management

Miika Nurminen, Anne Honkaranta, Tommi Kärkkäinen

Faculty of Information Technology

University of Jyväskylä, Finland

minurmin@mit.jyu.fi, anne.honkaranta@it.jyu.fi, tka@mit.jyu.fi

Abstract

This paper contributes both to research and practice on process mining. Previous research on process mining has focused on mining patterns from event log files to generate process models. The process mining approach adopted in this paper is focused on producing patterns about process models, not the models themselves. The approach is demonstrated by ProcMiner – an explorative research prototype for management, consolidating, publishing, retrieving, and analyzing process models. Content-based document clustering is applied to process models represented as XML database in order to find topical groups from models. In practice, organizations face numerous challenges in managing their process models. The models may be heterogeneous or ambiguous. The modeling software may change over time or due to differences in departmental purchases. ProcMiner was used in quality system development initiative at the University of Jyväskylä. The findings support previous model engineering research, showing that multiple actions are needed to ensure consistency of process models, and to make them efficiently manageable.

1. Introduction

This paper proposes a new approach on process mining. Previous research on process mining [2] has concentrated on mining patterns from event log files to generate a process model. According to Madria *et al.* [21], web mining considers content, structure, and usage mining. By this categorization the area of traditional process mining may be classified as process *usage* mining, by which process models are produced from event log files.

The approach presented in this paper aims at process *content* mining, in which patterns about existing process models are produced. ProcMiner – an explorative research prototype for managing and analyzing process model information, realizes this approach. ProcMiner introduces a novel integration of existing techniques from multiple dis-

ciplines including business process management, text mining, structured document clustering, multichannel publishing, and database management. ProcMiner facilitates gathering process model information and producing novel combinations of information residing in the contents of the process models. Main features of the platform are the following:

1. A unique XML-based process markup language based on an intermediate object model that is convertible to many process representations.
2. Versatile process retrieval and publishing functionality.
3. Support to process mining by content-based document clustering by using ExtMiner [24], a platform for structured document retrieval and text mining.

Organizations utilize process models for various purposes, such as for process re-engineering to realize the benefits of enterprise systems [9], for re-organizing work, or for establishing a quality system. ProcMiner, for instance, has been used in quality system development at the University of Jyväskylä.

This paper is organized as follows. Section 2 describes previous research related to process mining. Section 3 introduces the ProcMiner application. Section 4 describes how ProcMiner was used in the process mining, modeling and development initiative in the Faculty of Information Technology, University of Jyväskylä. Section 5 concludes the paper and discusses the results.

2. Related Research

This section describes previous research on process management, process mining, and document retrieval.

2.1. Business Process Management

Business process is a specific ordering of work activities across time and place with a beginning and an end containing inputs and outputs [10]. Business processes typically

span multiple organizations both within and between enterprises [28].

Business process management (BPM) includes methods, techniques, and tools for the design, enactment, management, and analysis of operational business processes. It can be considered as an extension of *workflow management* (WFM) systems and approaches [1]. BPM is evolved from synthesis of *business process re-engineering* (BPR) and *total quality management* (TQM) [28]. BPR emphasizes radical (often one-time) process change, whereas TQM is focused on continuous, incremental improvement [10].

Business processes can be modeled as visual graphs (such as "swimlane diagrams" [28], BPMN [32] or UML activity diagrams [25]), using informal textual representations, or by using a modeling language (such as XPDL [31] or BPEL [3]). Also the Petri nets [1] or pi-calculus [29] have been proposed as formal basis for process modeling languages. MIT Process Handbook [22] is an example of a less formal, yet structured approach for process modeling.

Process model content may be varying: commonly it contains information of activities that form a chain of actions taken (usually depicted as circle or rectangle), information about ordering of the activities in the form of control flow (usually depicted as an arrow) and information about user roles, organizational actors, web services or other applications performing a task. The process model may also contain information flows, information repositories, information units (e.g. documents), and references to subprocesses. Both process content and "superstructure" (i.e. links between different processes) can be interpreted as a graph.

Given that the information contained in process models can be processed by utilizing software for data collection, grouping, and analysis, a great deal of information can be discovered for organizational development work. For example, the activities where certain actor is involved can be listed to see the range or work tasks with regard to the actor and to detect "hot spot" actors involved in many processes. One may also gather information about documents or other resources produced and consumed by certain activities. Process models may be analyzed manually or automatically by suitable software.

Automatic information gathering from process models is not easy in practice. There may be variations among models [30] which hinder the model information analysis, model reuse, and process similarity analysis. The processes may also be defined by using a modeling tool that does not allow model-level integration to other systems. The expressive power and formal basis of different modeling languages varies [1]. On one hand, the popular formal process modeling languages BPEL and XPDL support information interchange between systems in a detailed level. Yet they may be too complex for end-user driven business process modeling. On the other hand, some tools advertised as "process

modeling" software may turn out to be essentially drawing programs. They do not provide enough information for formal analysis and composition of the models, preventing automated process improvement or execution in a workflow engine.

2.2. Workflow and Process Mining

The goal of workflow (or process) mining is to reverse the process of constructing the workflow model on design phase and then configuring it to some process-aware system [2]. Workflow mining assumes that there is some process data available, such as event log files. These workflow logs are used to construct a process specification which adequately models the behavior registered. Business process intelligence (BPI) [15] is related to workflow mining, but aims at supporting business-level execution analysis, predictions, and process re-design by analyzing process logs based on existing process specifications.

Previous research on applying clustering to readily available process models is scarce. An early application by Ellmer & Merkl [12] focused on clustering plain text descriptions about software processes, in order to facilitate software model reuse. Osterweil [26] suggested that process comparison could produce classification models. This idea has been applied to produce generic processes and reusable software components that implement these models using classification [23]. Huang *et al.* [17] have developed a similarity measure suitable for process graphs. Furthermore, Klein & Bernstein [19] have suggested that semantic process descriptions could be applied to compute similarity of natural language elements, combined with taxonomic reasoning and graph-theoretic similarity.

Adopting the categorization scheme for web mining suggested by Madria *et al.* [21], workflow mining or BPI can be classified as process *usage* mining and structural similarity measures as process *structure* mining, whereas the approach presented in Sect. 4.1 is an application of process *content* mining. The concept of process mining is adopted for the approach on discovering knowledge from the process model data. Contrary to process usage mining, the mining results are patterns *about* process models, not the models themselves.

2.3. Structured Document Retrieval and Clustering

Structured documents provide more information and possibilities for process analysis and fine-tuning the clustering than just plain-text descriptions. Interpreted as a directed graph, it is possible to represent the process models as structured documents, e.g. by using XML [6]. ProcMiner uses the unique XML-based process modeling language designed for semiformal, operational process models, whose




ProcMiner uses metric clustering algorithms, such as hierarchical clustering [33]. Process clustering within process mining produces a generalization hierarchy -like process structure that is separate from manually modeled super- and subprocesses and may provide novel insights about the processes supporting model reuse, maintainability, and information architecture design.

ProcMiner is an explorative research prototype for managing, consolidating, publishing, retrieving, and analyzing process model information based on text mining techniques. Process models can be grouped based on content similarity using ExtMiner [24], a platform for structured document retrieval and text mining.

The concept of process ontology [5] is essential for process discovery and service interoperability, yet tight coupling of ontology and organization-specific process models might unnecessarily complicate the modeling. Process ontology should be independent of the process modeling lan-

3.1. Background

The first process modeling project, PROMI (project I) was carried out by one modeler using wall-diagrams and Microsoft Excel. The project focused on modeling information flows within the faculty using the genre-based method proposed by Karjalainen *et al.* [18]. An example of a project I process model is illustrated in Fig. 1.

Microsoft Excel - Prosessikvaasioiden tulos									
A	B	C	D	E	F	G	H		
1	Prosessin nimi: GRAADUPROSESSI								
2	Seur	VAIHEN KUIVAUS	MISTA (rooli)	MITÄ (tietä)	MIHIN (rooli)	LOMAKKEET	Linkkilä-prosessit		
3	...								
4	...								
5	37								
6	Jos TKT:n laitos, niin opiskella jättää vämmä tulokseen koirana kappaleena laitojen kansaan ja täydästä osptomomakkeen laitojen laittamissa tuloksempankin vevv-sivulle		OPIKSELLIA	Tutkielema ja osptomomake	LAITOISEN KANSILIA	Sogimustomake	Maturuetti		
7	2	Ohjaaja tekee ehdotuksen kahdoksi tarkastajaksi	OHIAAJA	Ehdotus tarkastajaksi	AMANNIENSI		Linkkilä-prosessit		
8	3		AMANNIENSI	Tieto tuloksesta ja määä tarkastajaksi	LAITOISEN JOHTO				
9	4	Laitekon jake tuotuu tuloksesta ja määä silä kakei tarkastajia	LAITOISEN JOHTO	Piaotus tarkastajaksi	AMANNIENSI		Tarkastajan vevv-sivulle		
10	5		AMANNIENSI	Piaotus tarkastajaksi	TARKASTAJAT				
11	6	Ohjaaja kalle määä 9	AMANNIENSI	Piaotus tarkastajaksi	OPIKSELLIA				
12	6	Jos TKT:n laitos, niin opiskella jättää vämmä tulokseen koirana kappaleena laitojen kansaan ja täydästä osptomomakkeen laitojen laittamissa tuloksempankin vevv-sivulle	OPIKSELLIA	Tutkielema ja osptomomake	LAITOISEN KANSILIA	Sogimustomake	Maturuetti		
13	7	Ohjaaja tekee ehdotuksen kahdoksi tarkastajaksi (ja usein poe, jos luvumäärän ohjauet vevv vähiten FM tuloksen suostanetta)	OHIAAJA	Ehdotus tarkastajaksi	LAITOISEN VARAJOHTAJA		Linkkilä-prosessit		
14	8		LAITOISEN VARAJOHTAJA	Piaotus tarkastajaksi	TARKASTAJAT		Tarkastajan vevv-sivulle		
15	8		LAITOISEN VARAJOHTAJA	Piaotus tarkastajaksi	OPIKSELLIA				
16	8		LAITOISEN VARAJOHTAJA	Piaotus tarkastajaksi	LAITOISEN KANSILIA				
17	9	Opiskella laiteetta kopio tuotuu tuloksesta jättä vevv-sivulle	OPIKSELLIA	Tutkielema-kopio	TARKASTAJAT				
18	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
19	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
20	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
21	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
22	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
23	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
24	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
25	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
26	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
27	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
28	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
29	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
30	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
31	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
32	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
33	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
34	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
35	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
36	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
37	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
38	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
39	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
40	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
41	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
42	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
43	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
44	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
45	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
46	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
47	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
48	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
49	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
50	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
51	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
52	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
53	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
54	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
55	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
56	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
57	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
58	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
59	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
60	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
61	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
62	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
63	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
64	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
65	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
66	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
67	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
68	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
69	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
70	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
71	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
72	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
73	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
74	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
75	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
76	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
77	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
78	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
79	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
80	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
81	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
82	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
83	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
84	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
85	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
86	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
87	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
88	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
89	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
90	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
91	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
92	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
93	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
94	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
95	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
96	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
97	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
98	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
99	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
100	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
101	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
102	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
103	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
104	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
105	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
106	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
107	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
108	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
109	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
110	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
111	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
112	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
113	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
114	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
115	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
116	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
117	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
118	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
119	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
120	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
121	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
122	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
123	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
124	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
125	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
126	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
127	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
128	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
129	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
130	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
131	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
132	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
133	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
134	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
135	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
136	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
137	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
138	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
139	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
140	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
141	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
142	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
143	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
144	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
145	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
146	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
147	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
148	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
149	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
150	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
151	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
152	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
153	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
154	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
155	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
156	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
157	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
158	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
159	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
160	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
161	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
162	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
163	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
164	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
165	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
166	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
167	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
168	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
169	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
170	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
171	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
172	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
173	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
174	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
175	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
176	9	Tarkastajat antavat lauseutensa kappaleena kuitteina							
177	9	Tarkastajat antavat lauseutensa kappaleena kuitteina	</						

The process modeling was continued from administrative perspective by INFORYSÄ (project II) using Microsoft Visio. The project focused on modeling content production processes at the University of Jyväskylä in general. Many of the processes described by the project were also addressed in project I, but from a different perspective.

KAARI (project III) was carried out simultaneously using Microsoft Visio, too. Its goal was to define processes related to student information at the University of Jyväskylä in general. Most of the processes described by the project were also addressed in project I, but from a different perspective and in faculty level. Fig. 2 illustrates the process model notation used in projects II and III.

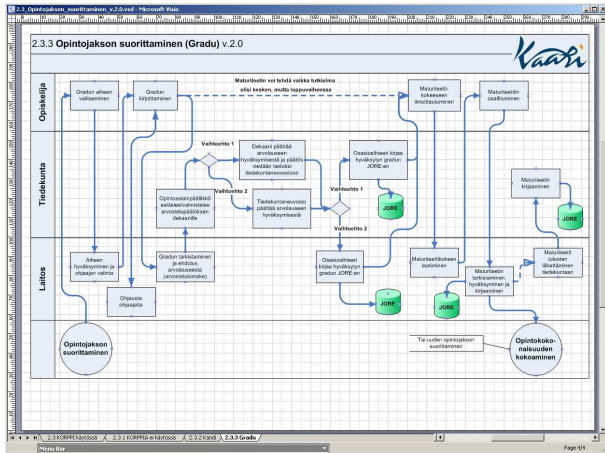


Figure 2. An example of a process description (master's thesis review) in project III (similar notation was used in project II).

At the time of quality system implementation, the Faculty of Information technology faced the similar challenges on process model management than many other organizations. There were problems on utilizing the process models. Similar to findings by Soffer & Hadar [30], the models were found to be imprecise and ambiguous. The same process, such as *production of master's thesis* was modeled in slightly different ways in different projects. The process models were also stored in differing formats, and the models had not been maintained after their creation.

The problematic situation provided an option for developing solutions for process model management, as well as process modeling data for text and process mining research that was carried out on the Faculty as well. ProcMiner is the result of this explorative and constructive research.

3.2. ProcMiner Requirements and Architecture

The general requirements for ProcMiner include the following:

- **Semiformal, understandable process models.** Like MIT Process Handbook, ProcMiner focuses on organizing knowledge, not on simulating performance [22]. Semiformal process modes lack executable logic

and precise interpretation, but contain enough information for analysis. Yet they are almost as understandable as informal, textual process descriptions.

- **Portability.** ProcMiner can be extended with import and export filters to other process modeling software. ProcMiner uses natively an XML-based format that allows easy specification of processes without the need to use specialized tools and contains all information related to a process in a single location, including process metadata and workflow specification.
- **Maintainability.** Maintainability was considered to be of key importance because of the experiences in projects I-III. For example, when Visio was used for modeling, the models did not have any centralized document or role list, so the modeler was left with considerable responsibility on maintaining the integrity of models and their attributes.
- **Retrieval and publishing capabilities.** Structured retrieval and publishing functionalities are a basic requirement for any process management system. For example, activities or documents that certain actor is involved can be listed, thus assessing the work load for actors or prospects for simplifying a process.
- **Process mining.** Contrary to traditional notion of process mining, ProcMiner allows extracting knowledge from process models in the early phase of process management efforts.

The architecture of ProcMiner is depicted in Fig. 3. Boxes represent systems and components, solid arrow denote component dependencies, dashed arrows denote data flows.

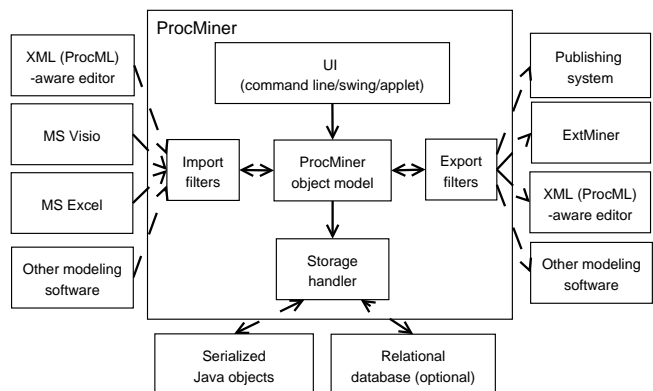


Figure 3. ProcMiner architecture.

The process management environment consists of tools for importing and transforming the process model data using ProcML format. Facilities for storing, combining, and

analyzing process data are provided. ProcMiner utilizes a three-tier architecture that provides a complete separation of user interface from process model logic (i.e. object model) and data storage (i.e. storage handler). This separation allows updating the process model structure and queries with virtually no need to update the storage code. For intermediate storage, process model can be serialized using standard Java object serialization mechanism, or optionally, to a relational database. ProcMiner can be used with a command-line interface (mainly for filters), Swing-based desktop interface (process mining and model manipulation), or with an applet-enhanced web UI (retrieval and publishing). Publishing system uses Graphviz (<http://www.graphviz.org/>) for automatic layout of process graphs and L^AT_EX (<http://www.latex-project.org/>) for typesetting a printable "handbook" of the processes.

ProcMiner utilizes ExtMiner software developed in earlier structured document mining project [24]. The internal representation of documents and queries on ExtMiner is based on a field-based index. Index terms, links, metadata and other custom elements such as headers are represented as vectors and indexed as fields that can vary based on document characteristics. ExtMiner can be used as an interface for a search engine for both full-text and field-based structural queries and multiple clustering algorithms. It uses Apache Digester (<http://jakarta.apache.org/commons/digester/>) for XML parsing and Apache Lucene (<http://lucene.apache.org/>) as the search engine backend. ProcMiner and ExtMiner have been programmed with the Java language, and licensed as open source applications. In addition, the server side of the publishing system is partly implemented with PHP.

3.3. Object Model and ProcML

Object model is the core of ProcMiner. ProcMiner object model works as an intermediate format facilitating conversions between multiple modeling languages. Fig. 4 illustrates the metamodel of the intermediate object model as UML class diagram.

It is well known that process modeling is hard – even at the informal level – and requires communication of tacit knowledge. Semiformal models are easier to understand than formal ones and require less modeling effort. Because the object model contains information about documents and roles in the processes, genre-based communication analysis method [18] could be used for detecting candidates for formalization and automation. Even without explicit formalization, semiformal models can be used for disseminating knowledge about work procedures among staff.

Because of the diversity in earlier process modeling projects, the object model in ProcMiner was designed to be as adaptable to different representations as possible. For

example, project I focused on data flows, whereas projects II and III focused on activities. Despite these differences in modeling paradigms, the metamodel must be flexible enough to account both of them. Similar to MIT Process Handbook, ProcMiner allows a wide variety of data from diverse sources. In this respect ProcMiner and Process Handbook have an advantage over more formal approaches because many alternatives can co-exist in the system [22]. However, this flexibility requires that process modeler is supported by explicit conventions on how to use the object model for a specific modeling goal.

There are two key features in ProcMiner object model that presumably have no equivalent concept in other popular process modeling languages. First is the separation of process and process instance. *Process* is an abstract specification of the general characteristics and metadata related to a certain business process, including its customer, owner, pre- and postconditions and so on. *Process instance* is an organization-specific model of the process with additional metadata (like implementer) and process specification graph. This way, it is easy to compare departmental process realizations and unify them incrementally, if desired. Another possibility is to model separate process instances for current and objective states of the process. Process instances can be inherited (or included), enabling maximal model reuse but allowing department-specific adaptations. Process instances should not be confused with process and workflow *execution*, also referred as process instances by some authors (cf. [26]).

Another novel concept in ProcMiner object model is the notion of *abstraction levels*. Process model can be interpreted as a multilevel graph, where each level adds more elements or overrides elements in the upper level. Top-level should specify only one linear sequence that represents a successful execution of the process. However, real-life processes are rarely straightforward but contain exceptions and branches. Abstraction levels are a way of decomposing the process to its essential components, providing both "executive summary" for management and more details for the actors involved in the process. For example, a level 0-description might involve only "high-level" organizational actors such as *faculty* and *department* that are further defined as *amanuensis*, *head of academic affairs*, and *dean*. Abstraction levels are related to use case levels described by Cockburn [8], and detail levels proposed by Sharp & McDermott [28]. However, in ProcMiner object model, all the levels are integrated in a single model. Representations of varying abstraction can be produced automatically.

The ProcMiner system has much in common with MIT Process Handbook with some exceptions. ProcMiner process model allows the use of documents and roles in processes. It also allows the use of branches in process steps. In Process Handbook, any element can be inherited in a non-

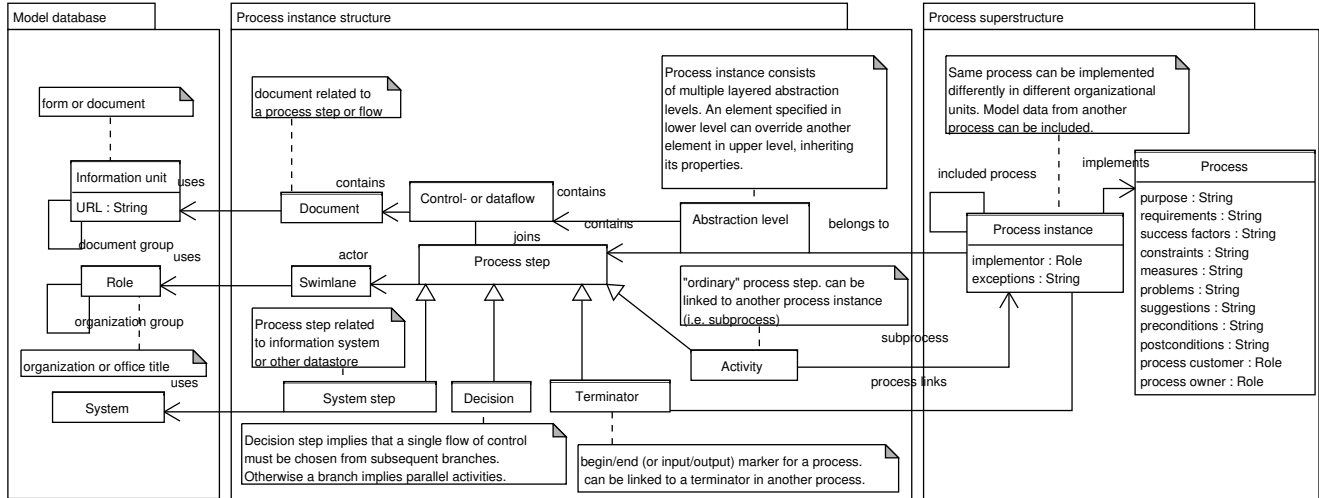


Figure 4. ProcMiner object model.

monotonic way, in which at each (sub)class in the class hierarchy, or any inherited property value may be overridden with another value, or simply cancelled [5]. ProcMiner allows any process to be inherited by allowing values to be overridden but not cancelled. However, in ProcMiner it is possible to inherit only a subset of the parent process, i.e. elements contained in a specified abstraction level.

ProcMiner uses XML-based process modeling language "ProcML" that works as the native serialization format for ProcMiner object model. Reading and writing ProcML is implemented as filters in ProcMiner. ProcML syntax is defined as XML DTD. The language is designed for ease of expressivity for input of multilevel graph-based data without the need to use specialized graphical tools, as is practically the case with BPEL, for example. Further, organizational role hierarchy cannot be sufficiently expressed with BPEL. XPDL is better suited for descriptive process modeling, because it supports documents and roles. However, because of the complex routing and implementation-specific concepts both BPEL and XPDL were considered to be too complex and low-level for descriptive, end-used driven process modeling. Finally, expressing the novel modeling concepts, like abstraction levels, in BPEL or XPDL would have been laborious for modeler and would have required extensive conversions to work in a natural way with the object model. However, should a process be automated, a one-way conversion and further manual formalization to an executable workflow language might be necessary.

3.4. Retrieval and Publishing

Processes can be retrieved using full-text or metadata field-specific search, as well as browsing by document, role or information system lists that show all the processes

where the given modeling entity is located. As special cases for retrieval there was a special interest to retrieve "hot spot" actors that are present in many processes, as well as detecting data flows and incremental changes for certain documents that are present in complex processes.

All process metadata and graphical information is retrievable from the same object model, facilitating process maintenance. This way there is no need to maintain and keep in sync separate model and metadata documents, which is the case when process graph is modeled using a drawing program and process metadata is contained in a separate document (e.g. projects II-III). Updating data allows replacing obsolete modeling elements and combining duplicated elements when necessary. This is often the case with heterogeneous process data.

ProcMiner's publishing system produces a HTML-based "process portal" that contains a search engine, process descriptions, as well as process-, document-, role-, and information systems trees or lists. Process description documents contain both textual and graphical representation with automatic flowchart-like layout generated by Graphviz. This twofold notation is required, because different users prefer different representations. Process documents are produced semidynamically with PHP scripts and a Java applet. For printing, a PDF-based "handbook" is generated using XSL Transformations [7] and \LaTeX that includes selected processes augmented with a back-of-book index of documents, roles, and information systems.

3.5. Process Mining Using Clustering

The process mining process supported by ProcMiner encompasses all phases of the KDD process. Selected process data is preprocessed using ProcMiner's import filters, which

unify representation formats to ProcML format. Textual content on XML document fragments is then transformed to extended vector model, suitable for clustering by ExtMiner [24] application. Finally, the results are evaluated.

1. The **selection** phase involves selecting and converting input model data to a manageable representation that can be consumed by ProcMiner input filters. The activities of this phase depend on input format and can often be implemented as part of an input filter. However, it is useful to distinguish activities performed by external systems from activities performed by ProcMiner on the conceptual level. Selection phase might involve fetching data from the web using a crawler or a web service, using conversion or data cleaning utilities, or formatting using XSLT.
2. Process model datasets are consolidated to a common representation in the **preprocessing** phase using import filters. During preprocessing, document and actor classes are added to the data model based on elementary string comparison procedure. If different models feature an actor with same label, it is assumed to be an instance of the same actor class. It is also possible to use an existing process model, where the new processes are merged to.
3. After the process models are imported to ProcMiner, they can be reviewed and, if necessary, modified by the user. For example, synonymous role or document names could be merged and corrections could be made to process superstructure. For further processing, process models are **transformed** to ProcML using an export filter. Resulting XML files are input data for ExtMiner. During the indexing stage, terms are stemmed, stop-words are eliminated and *tf*-weighting [4] is applied to the terms. A custom *indexformer* [24] tailored for ProcML is used in ExtMiner. The field-based index consists of full-text content, external metadata about the original dataset (if data was imported from multiple repositories) and the following fields parsed from predetermined XML structures in each process model file: *process title*, *process actors*, and *process documents*.
4. In the **data mining** phase, documents representing process models are clustered. The similarity measure used in searching and clustering is by default the *cosine similarity*, i.e. the "angle" between the document vectors [4]. If multiple fields are used in addition to content-only data, the overall similarity can be calculated with a weighted linear sum, where each component of a sum represents a single field [14]. The similarity measure can be used by any metric clustering algorithm.

5. Clustering results are assessed in the **evaluation** phase. Process clustering produces a new hierarchy or partitioning in addition to decomposition and (asserted) inheritance dimensions. Automatically generated clustering model can be compared to these asserted models and might reveal new connections previously unknown to the modeler. For example, different processes might have a similar subprocess that might have been modeled twice. If similar terms are used in modeling, the subprocesses should end up to same cluster and be potential candidates for model reuse by inheritance.

Process mining functionality in ProcMiner is currently limited to content-based analysis. However, since processes are modeled as graphs, support for structured data mining is an obvious prospect for additional development. If a structured similarity measure was used, processes could be classified based on the graph structure, or by typical patterns in the process descriptions.

4. Evaluation

This section describes the use of ProcMiner in the constructive research during the process mining, modeling and development initiative at the University of Jyväskylä.

4.1. Clustering Heterogeneous Process Data

Process clustering was tested for comparing the existing process models from the three projects. Microsoft Excel and Visio files were converted from their binary formats to Office 2003 XML. Nonessential data was filtered from XML files using XSLT. In addition to process graphs, all the document and role data was imported and consolidated to ProcMiner. Visio models did not contain any organization- or document hierarchy – they had to be fetched implicitly from the modeling elements.

The statistics of the processes, documents and roles modeled in projects I-III separately and after the consolidation are summarized in Table 1. Relatively few common entities were identified using a purely string equality -based comparison. However, the models shared substantially more elements with the same meaning but different labels.

The dataset was clustered using full-text based similarity information using group average hierarchical clustering algorithm. The advantage of the method is that it requires no input parameters and produces clusters of superior quality compared to other hierarchical clustering algorithms, such as the minimum link method [33]. The numerous clusters produced by the hierarchical clustering algorithm were manually combined into larger process clusters. The clustering results are illustrated in Fig. 5.

Table 1. Processes modeled in projects I-III
(* After consolidation).

Project	Modeling tool	Processes	Roles	Docs
Project I	MS Excel	15	67	103
Project II	MS Visio	10	75	53
Project III	MS Visio	13	26	26
Total	ProcMiner	38	167*	178*

The clustering results were somewhat surprising: it was expected that process clustering would reveal some kind of general topic-based structure, shared by processes modeled by different projects. On the contrary, the processes were clustered almost entirely according to the original modeling projects. Naturally, the *original project* metadata field was not used in clustering.

A possible explanation for this distribution may be the large amount of features vs. samples – a total of 566 index terms were extracted from only 38 process documents and used in the similarity computation. Besides, project I was modeled on department-level and projects II and III on university-level. However, this doesn't explain that also projects II and III ended up mostly in different clusters.

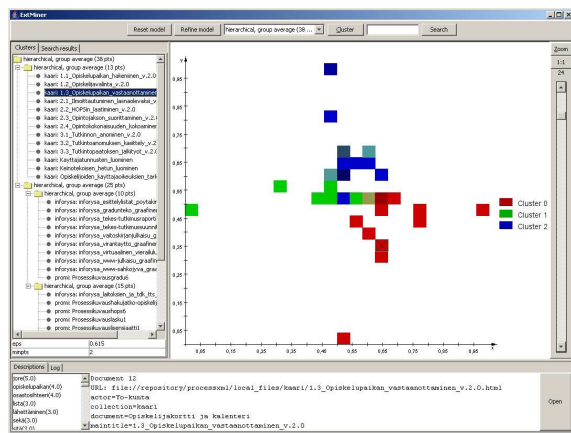


Figure 5. a view of the clustered process descriptions in ExtMiner.

Because of the simple term-based clustering, the subtle differences in terminology and phrasing conventions used in the projects influenced considerably the clustering results, even if the process topic was the same. There were some exceptions for processes that contained slightly specialized vocabulary, since most of the other processes were concerned about studying, research activities and content production. It should also be noted that hierarchical clustering is affected by the order of documents [33], i.e. by

varying the document input order the algorithm might produce slightly different clusters.

The clustering result supports the notion that different people usually present different models given the same domain [30]. Uncontrolled, these variations may be harmful for model information analysis, model reuse, and process similarity analysis. Organizations need to take multiple actions to ensure consistency of terms and the quality of process models to make them efficiently usable. To achieve this, a corporate taxonomy or ontology should be developed, combined with close coordination between modeling teams. After the initial modeling effort is completed, a process "caretaker" should be assigned to keep the models consistent whenever the models need to be updated. In practice, tool support is required for large process structures.

4.2. XML Modeling and Process Portal

Parallel to the unsatisfactory process clustering experiment, new processes were modeled manually, partially accounting existing process models. By Fall 2006, the faculty-specific model database contained 152 process descriptions. Of those, 85 descriptions represent actual processes, the rest being process groups or context-dependent subprocesses. In addition, 46 document types, 86 organizational roles and 13 information systems were listed. During the project, 3 different modelers have been modeling the processes with ProcML using a DTD-aware XML editor and ProcMiner publishing system.

At the moment, ProcMiner has only XML interface for updating process graphs. An alternative, graphical interface using QPR ProcessGuide is under development. When modeling semiformal processes, the actual technique used for data input is not critical from the point of the modeler. More difficult is to understand and communicate the knowledge related to a process. It is not an essential difference if the model is expressed in Visio or ProcML – except that the latter allows efficient analysis and maintenance of the processes. Even if graphical tool may be easier to use for a beginner, user's attention may be drifted to irrelevant aspects, such as manual layout of the models.

Process portal (see Fig. 6) was used extensively by all project stakeholders including the developer, modeler, steering group, and faculty staff. Public, searchable process repository allows organization-wide transparent reviews and feedback. Process models must be communicated throughout the organization to be effective and to fuel continuous improvement. A formal "process improvement process" was defined as a part of the other processes, containing guidelines for process modeling, inspection, deviation, and evolution. After the initial modeling, processes were approved in two stages: first, the steering group inspected the processes in a detailed way, then a larger group

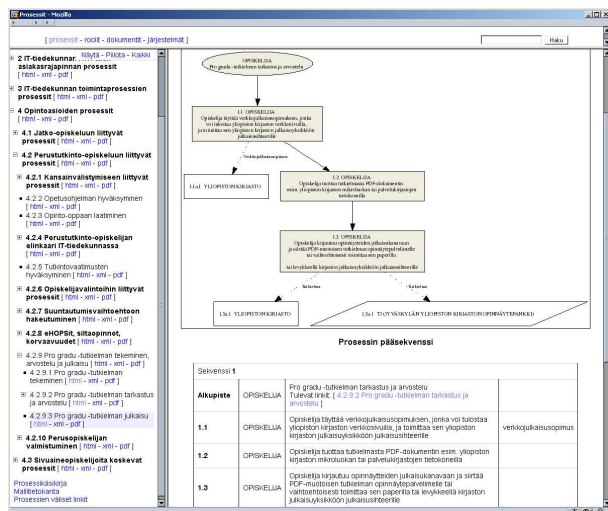


Figure 6. An example of a generated process report (publication of master's thesis).

representing the whole staff inspected and approved the processes in a formal review. Publishing system was used for distributing printed handbooks for stakeholders.

Published process models have proved to be useful as a centralized repository of work instructions, scattered earlier to different unit-level web pages. Using the process portal, everything can be found from the same structure. In addition to being purely a process repository, the portal works as a document reference storage. Documents can be physically located in unit-specific storages, or ideally, content management systems, but the view in document tree allows quick hyperlinks to document templates. Overall, process portal and ProcMiner publishing system work as a solid basis for an organization-wide searchable process handbook that is a central part of an enterprise-wide quality system.

5. Conclusion and Future Work

This paper described the novel content-based process mining approach. The approach was demonstrated by ProcMiner, an explorative research prototype for managing and analyzing process model information based on text mining techniques. ProcMiner utilizes ExtMiner, a platform for clustering and retrieving structured documents, for process data mining. A new XML-based process modeling language was developed to consolidate the process models and to support maintainable process modeling. The complexity of process data management in real-life settings was illustrated by describing process development at the University of Jyväskylä. Development on process model analysis, information gathering, retrieval and publishing was described.

An application of process mining was demonstrated by applying explorative document clustering to the process models with ProcMiner. Based on the key features described in Sect. 1, ProcMiner supports process analysis and management as follows:

1. The unique XML-based process markup language ProcML, based on the intermediate object model, has been successfully applied for modeling new processes and consolidating process data from diverse sources.
2. Process retrieval and multichannel publishing simplifies organization-wide applicability and communication of process descriptions both in modeling and implementation stages.
3. Structured document clustering may facilitate business process development by providing an independent view to the process subject areas. However, in order to achieve useful clustering results, the processes should be modeled using standard, consistent terminology.

ProcMiner provides many interesting aspects for future research, such as process decomposition to multiple levels of abstraction, as well as content-based process mining. For improving process data consistency ProcMiner should be enhanced with additional process consolidation functionality; such as detecting multiple connotations inferring to the same actor. Process data analysis should be appended with the use of structural metrics or similarity measures. The XML-based process modeling language shows potential but needs to be cross-analyzed with other process modeling languages. Process mining approach has potential for facilitating business process management.

References

- [1] W. M. P. van der Aalst. Business process management demystified: A tutorial on models, systems and standards for workflow management. In J. Desel, W. Reisig, and G. Rozenberg, editors, *Lectures on Concurrency and Petri Nets*, pages 1–65. Springer, 2003.
- [2] W. M. P. van der Aalst, B. F. van Dongena, J. Herbst, L. Maruster, G. Schimm, and A. J. M. M. Weijters. Workflow mining: A survey of issues and approaches. *Data Knowledge Engineering*, 47(2):237–267, November 2003.
- [3] T. Andrews, F. Curbera, H. Dholakia, Y. Goland, J. Klein, F. Leymann, K. Liu, D. Roller, D. Smith, S. Thatte, I. Trickovic, and S. Weerawarana. Business process execution language for web services version 1.1. Technical report, BEA, IBM, Microsoft, SAP, Siebel, 2003.
- [4] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison-Wesley, 1999.
- [5] A. Bernstein and B. Grosz. Beyond monotonic inheritance: Towards semantic web process ontologies. Working paper, University of Zurich, Department of Informatics, 2003.

- [6] T. Bray, J. Paoli, C. M. Sperberg-McQueen, E. Maler, and F. Yergeau. Extensible markup language (XML) 1.0 (third edition). Technical report, W3C Recommendation, 2004.
- [7] J. Clark. XSL transformations (XSLT) version 1.0. Technical report, W3C Recommendation, 1999.
- [8] A. Cockburn. *Writing Effective Use Cases*. Addison-Wesley, 2000.
- [9] N. P. Dalal, M. Kamath, W. J. Kolarik, and E. Sivaraman. Toward an integrated framework for modeling enterprise processes. *Commun. ACM*, 47(3):83–87, 2004.
- [10] T. H. Davenport. *Process Innovation – Reengineering Work through Information Technology*. Harvard Business School Press, 1993.
- [11] J. Dörre, P. Gerstl, and R. Seiffert. Text mining: finding nuggets in mountains of textual data. In U. Fayyad, S. Chaudhuri, and D. Madigan, editors, *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 398–401. ACM Press, 1999.
- [12] E. Ellmer and D. Merkl. Classifying software process models based on natural language descriptions. In R. Wagner and H. Thoma, editors, *Seventh International Workshop on Database and Expert Systems Applications, DEXA '96*, pages 320–325. IEEE Computer Society, 1996.
- [13] U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth. From data mining to knowledge discovery in databases. *AI Magazine*, 17:37–54, 1996.
- [14] E. A. Fox, G. L. Nunn, and W. C. Lee. Coefficients of combining concept classes in a collection. In *Proc. of the 11th Annual Int. ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 291–307. ACM Press, 1988.
- [15] D. Grigori, F. Casati, M. Castellanos, U. Dayal, M. Sayal, and M.-C. Shan. Business process intelligence. *Computers in Industry*, 53(3):321–343, April 2004.
- [16] J. Han and M. Kamber. *Data Mining: Concepts and Techniques*. Morgan Kaufmann, 2000.
- [17] K. Huang, Z. Zhou, Y. Han, G. Li, and J. Wang. An algorithm for calculating process similarity to cluster open-source process designs. In H. Jin, Y. Pan, and N. Xiao, editors, *Grid and Cooperative Computing - GCC 2004 Workshops: GCC 2004 International Workshops, IGKG, SGT, GISS, AAC-GEVO, and VVS. Proceedings*, pages 107–114. Springer, 2004.
- [18] A. Karjalainen, T. Päiväranta, P. Tyrväinen, and J. Rajala. Genre-based metadata for enterprise document management. In *HICSS '00: Proceedings of the 33rd Hawaii International Conference on System Sciences-Volume 3*, page 3013. IEEE Computer Society, 2000.
- [19] M. Klein and A. Bernstein. Searching for services on the semantic web using process ontologies. In *Searching for Services on the Semantic Web using Process Ontologies. in The First Semantic Web Working Symposium (SWWS-1)*, 2001.
- [20] R. W. Luk, H. Leong, T. S. Dillon, A. T. Chan, W. B. Croft, and J. Allan. A survey in indexing and searching XML documents. *Journal of the American Society for Information Science and Technology*, 53(6):415–437, 2002.
- [21] S. K. Madria, S. S. Bhowmick, W. K. Ng, and E.-P. Lim. Research issues in web data mining. In *Data Warehousing and Knowledge Discovery, First International Conference, DaWaK '99*, pages 303–312, 1999.
- [22] T. W. Malone, K. Crowston, J. Lee, B. T. Pentland, C. Del-larocas, G. M. Wyner, J. Quimby, A. Bernstein, G. A. Herman, M. Klein, C. S. Osborn, and E. O'Donnell. Tools for inventing organizations: Toward a handbook of organizational processes. In T. W. Malone, K. Crowston, and G. A. Herman, editors, *Organizing Business Knowledge. The MIT Process Handbook*. MIT Press, 2003.
- [23] H. Mili, G. B. Jaoude, E. Lefebvre, and G. Tremblay. Going beyond MDA: Business process modeling for software reuse. In *Proceedings of the Workshop on Legacy Transformation: Capturing Business Knowledge from Legacy Systems - OOPSLA'2004*, 2004.
- [24] M. Nurminen, A. Honkaranta, and T. Kärkkäinen. ExtMiner: Combining multiple ranking and clustering algorithms for structured document retrieval. In *International workshop on Integrating Data Mining, Databases and Information Retrieval (IDDI'05), Proceedings of the 16th International Workshop on Database and Expert Systems Applications*, pages 1036–1040. IEEE Computer Society, 2005.
- [25] OMG. Unified modeling language specification v2.0. Technical Report 05-07-04, Object Management Group, 2005.
- [26] L. J. Osterweil. Software processes are software too, revisited: an invited talk on the most influential paper of icse 9. In *ICSE '97: Proc. of the 19th international conf. on Software engineering*, pages 540–548. ACM Press, 1997.
- [27] G. Salton, A. Wong, and C. S. Yang. A vector space model for automatic indexing. *Commun. ACM*, 18(11):613–620, 1975.
- [28] A. Sharp and P. McDermott. *Workflow Modeling*. Artech House, 2001.
- [29] H. Smith and P. Fingar. Workflow is just a pi process. *BP-Trends*, January 2004.
- [30] P. Soffer and I. Hadar. Reusability of conceptual models: The problem of model variations. In K. Siau, T. Halpin, and J. Krogstie, editors, *Eighth CAiSE/IFIP8.1 International Workshop on Evaluation of Modeling Methods in Systems Analysis and Design (EMMSAD'03)*, pages 126–134, 2003.
- [31] WfMC. Workflow management coalition workflow standard: Process definition interface – xml process definition language. Technical Report WfMC-TC-1025, Workflow Management Coalition, 2005.
- [32] S. White. Business process modeling notation (bpmn) version 1.0. Technical report, BPMI.org, 2004.
- [33] P. Willett. Recent trends in hierarchic document clustering: A critical review. *Information Processing & Management*, 24(5):577–597, 1988.