On Web Services Workflow Mining

Robert Gombotz and Schahram Dustdar

Distributed Systems Group,
Vienna University of Technology, Austria
{gombotz, dustdar}@infosys.tuwien.ac.at

Abstract. With the ever growing importance of the service-oriented paradigm in system architectures more and more (business) processes will be executed using service-oriented systems. Therefore, we believe that the ability to discover processes in loosely-coupled systems is essential in system optimization. Firstly, we briefly describe our previously introduced idea of Web Services Interaction Mining (WSIM) and then direct our attention on mining for workflows in logs provided by SOA. We thoroughly examine strategies in other fields of mining for their applicability in SOA. After that, we discuss logging possibilities in service-oriented systems and analyze mining opportunities with regards to the provided logs. As a case study we present a service-oriented system and its logging features. We conclude with a demonstration of how we successfully applied existing process mining strategies on this system's logs and present the results of that mining in the form of workflow models.

1 Introduction

With the emergence of Web services (WS) as a widely accepted standard, service-oriented architectures (SOA) increasingly gain attention, both, in research and in industry. The use of Web services facilitates the integration of formerly independent systems. In the future, service-oriented systems can be expected to gain more and more importance in information technology (IT). Designing new systems based on the service-oriented paradigm allows for the development and the deployment of loosely-coupled systems which promise to be more flexible and more easily adaptable when business process requirements change over time. Such quickly evolving systems demand for sophisticated and powerful monitoring in order to ensure correct system behavior and to allow for optimizing system characteristics, like performance or usability. In other fields, mining techniques are applied to gain additional knowledge about systems and data. In this paper, we argue that mining may also be applied to service-oriented systems.

In our previous work we have introduced the idea of Web services Interaction Mining (WSIM) [1,2]. In WSIM we make use of the findings in the fields of data mining and process mining and apply them to the world of Web services and service-oriented architectures. Mining describes the act of examining and analyzing large amounts of (log) data with the ultimate goal of knowledge discovery (KD). Consequently WSIM attempts to apply mining techniques on logs

C. Bussler et al. (Eds.): BPM 2005 Workshops, LNCS 3812, pp. 216–228, 2006.

provided by Web services and SOA infrastructures. We begin by presenting an overview of WSIM to outline the cornerstones of our approach.

1.1 Fundamentals of Web Services Interaction Mining

We currently develop our Web Services Interaction Mining approach with regards to three levels of abstraction that represent three complementary "views" on Web services.

The lowest level of abstraction is the *Web service operational* level. On this level only one Web service is to be examined. Its standing within a service-oriented system is not yet considered. Characteristics of a WS to be examined on this level include, but are not limited to, execution time, service usage, or service availability. Mining on this level can be as detailed as to only analyze single operations of a given WS.

The second level of abstraction is the *Web service interactions* level. On this level the focus is still on a single Web service but with regards to its interactions with other services. In these interactions the given WS may act as either provider or consumer. Therefore, analyses on this level may deal with WS conversation, i.e., in what order are operations invoked by service requestors, or WS composition, i.e., which services are consumed by a given composite WS [3]. Furthermore, a thorough analysis of past interaction with a given WS can reveal dependencies in service-oriented systems and provide the information necessary for an impact analysis when changes to a WS are planned. The result of mining on this level can be an interaction graph as presented in [1].

The highest level of abstraction is the *Web services workflow* level. Here, we attempt to identify workflows, or (business) processes, that are implemented using the functionalities of Web services. In that sense WSIM is positioned one step before WS orchestration. WS orchestration allows for the definition of workflows in an orchestration language such as WSBPEL [4] and for a monitored execution of that workflow using a BPEL engine [5,6]. However, WSIM deals with service-oriented systems that do not yet apply WS orchestration tools. A future application of WSIM on the workflow level might therefore be to generate abstract workflow definitions of discovered processes and thereby facilitate the work of workflow designers.

This paper focuses on the discovery of Web services workflows. We want to emphasize that we are working in environments where no means of WS orchestration, e.g. BPEL, are used.

The remainder of this paper is structured as follows. In Section 2 we examine other fields of mining and discuss their relevance for workflow mining in service-oriented architectures. In Section 3 we analyze logging possibilities in SOA and propose a scale of levels of logging as well as the mining opportunities on these levels. Section 4 describes a service-oriented system we have implemented as a motivating example. In Section 5 we describe the mining experiments that were performed in our case study. A conclusion is given in Section 6.

2 Related Work

In this section we present the current state-of-the-art in other fields of research which have influence on Web services Interaction Mining, the most relevant of which are Web usage mining (WUM) and process mining.

2.1 Web Usage Mining

With the ever growing importance of the World Wide Web (WWW) many researchers have directed their attention to developing means of managing, analyzing, and understanding the vast amounts of data provided on and by the web. These efforts are centered around applying data mining strategies to the content of Web sites, Web server log records, etc. Depending on the data that is analyzed and the desired outcomes of these analyses, web mining can be further broken down into sub-categories: *Web content mining*, *Web structure mining* and *Web usage mining* [7]. Web usage mining, and possibly Web content mining provide methods and approaches that can be integrated and used in Web services Interaction Mining.

Web usage mining is concerned with discovering Web access patterns in Web server logs in order to analyze user browsing behavior [8,9]. Most Web servers provide logging features, which record one log entry for each request received by the server. Such a log entry typically contains the following information [10,11]: the IP-address of the requesting host, a timestamp, the request line, e.g. `"GET /index.html HTTP/1.0"`, and the HTTP status code returned to the client. Another important element a log entry may contain is the *referer* (sic). This is an identifier a client may include in his request indicating where he was referred from when requesting this resource, i.e., where the link was he clicked to request the current resource. Also, if user authentication is required to access a web site, the username is included in all log records of requests sent by that user.

In order to discover access patterns it is first necessary to group individual requests into user sessions, a process called session reconstruction. A session describes a user's visit to a Web site from opening the first page until the site is left. Once sessions have been identified, WUM mines in these sessions for reoccurring patterns. The knowledge gained is used to optimize or customize Web sites. Session reconstruction can be done by collecting requests sent from a given host and applying time constraints [12]. Such time-oriented heuristics pose limitations on the duration of the entire session (h1) and on the duration of a stay on one individual page (h2). The values for h1 and h2 are often fixed, e.g., to 30 and 10 minutes, respectively. Some argue that these values should be flexible and should also incorporate the structure and content of a web site [13,14].

2.2 Process Mining

Workflow mining or *process mining* (the terms may be used interchangeably), describes the effort to discover workflow patterns in a given set of log data. A

workflow is a reoccurring, ordered set of *activities* that are performed in order to fulfil a higher-level task. Each individual execution of that workflow is called an *instance* of the workflow, or a *case*. The goal of process mining is to analyze a so-called *workflow execution log* in order to construct a *workflow model* that best describes *all* recorded instances of that workflow. Log entries must include a workflow identifier and a case identifier, so that the recorded events can be mapped to the according workflow and case.

The greatest challenge in process mining is the construction of models for complex processes. Workflows may contain alternative flows, concurrencies, or loops [15]. Such complex patterns pose challenges to the development of efficient mining algorithms. In [16] van der Aalst et al. present the current state of process mining. In [17] the InWoLvE process mining system is presented which claims to be able to discover a wide range of process models. In [18] an algorithm for "mining exact models of concurrent workflows" is introduced. More issues are discussed in [19,20,15].

2.3 Applicability of Web Usage Mining and Process Mining in WSIM

Web usage mining will have an impact on our work of developing methods to discover workflows in WS-related logs. Comparable to the browsing through Web sites, Web service interactions can also be seen as isolated request-response transactions, embedded in larger-scale sessions made up of numerous such transactions. The records one can keep of WS interactions are somewhat similar to records found in Web server logs. Especially the scenarios addressed in [13,8], i.e., mining in Web server log records in the absence of session information, pose the same problems as the situation we are faced with in WSIM. However, the values used in time-oriented algorithms for user session reconstruction, namely h1 and h2, may be suitable in human user interaction scenarios, where page requests often do happen in intervals of >1 minute. However, in service-oriented systems which are designed for machine-machine interactions, the time elapsed between two interactions may be far less. Therefore, new metrics will have to be found when applying WUM approaches in WSIM. Also, the structure of the system to be examined will have an impact of the values assumed for h1 and h2, which is also one of the key statements in [13]. For example, in a system that performs a workflow of ordering products, time heuristics similar to those in Web usage mining may be suitable. On the other hand, in a system for chemical simulations, in which certain services compute parts of computationally intensive equations, interactions might happen in intervals of only a few milliseconds. Therefore, there are limitations to the possibilities of applying WUM in service-oriented systems.

Another disadvantage in the approaches of Web usage mining that itdoes not go as far as attempting to construct complete process models. The construction of such a model, however, is the main goal of WSIM on the workflow level. This goal is clearly shared with process mining. A major advantage of process mining is the availability of powerful algorithms, which can clearly be of use in WSIM research.

A draw-back of process mining is, in our opinion, the restrictive assumptions made about the richness of information available in workflow execution logs [16].

From our research we conclude that WSIM is located between WUM and process mining. Web usage mining attempts to find frequent traversal path patterns from a limited amount of log information. Process mining, on the other hand, tries to construct complete workflow models from very rich log data. The log data available in WSIM is comparable to that in Web server logs. However, the more steps are taken, and the earlier WSIM is considered in the development process of a service-oriented system, the richer the log information can become. Therefore, WSIM on the workflow level should both (a) apply Web usage mining techniques on service-oriented systems that provide little log information and (b) present methods for the development of more sophisticated logging in service-oriented systems, so that these systems may later be mined for exact workflow models with the assistance of process mining tools. Consequently, the quality of workflow models that can be discovered using WSIM will depend on the quality and richness of execution log data provided by the underlying system.

3 Web Service Logging

In this section we examine and formalize the logging possibilities in service-oriented architectures. The levels of logging vary in the richness of the information that is logged and in the additional development effort that is needed when implementing the respective features.

Level 1: Standard HTTP-server logging

The most commonly used log formats provided by Web servers are the *Common Log Format* and the *Combined Log Format* [10,11,21]. The log entry recorded in Apache Tomcat when a request is sent to a WS ExampleService may look as follows:

```
127.0.0.1 - - [15/Mar/2005:19:50:13 +0100]
"POST /axis/services/ExampleService HTTP/1.0" 200 819 "-" "Axis/1.1"
```

The log entry contains the requestor's IP address, a timestamp, the request line, the HTTP code returned by the server, i.e., 200 for OK, the size of the returned resource, and the User-Agent, i.e., Axis/1.1. The empty element, i.e. "-", indicates that no referer-information is available. Such log records allow for tracking of the service consumer, determining which service is called how often (but not which operation of the service), or analyzing service failure rates.

This level of logging can be achieved by simply enabling the respective Web server's logging facilities. The service-oriented environment is in no way affected.

Level 2: Logging of complete HTTP requests and responses

Alternatively to Web server logging, an HTTP listener may be used to record all traffic directed to a given port, which allows for logging of the complete HTTP

request and response traffic. This way the involved SOAP messages are also recorded since they are sent as part of the HTTP messages.

Achieving logging on level 2 requires an HTTP level logger that listens to a given port A and redirects to another port B. Also, it is necessary to reconfigure the HTTP server from the original port A, where service calls are expected to be received, to port B, to which the HTTP logger redirects. The service-oriented system is not affected. An existing open-source HTTP listener is Apache TCP Tunnel/Monitor [22].

Level 3: Logging at WS container level

On level 3 the logging is done inside the WS container, e.g., Apache Axis [23]. Unlike on levels 1 and 2, the logging is more flexible and can be configured, e.g., by only monitoring and logging certain WS. Also, HTTP requests not directed at Web services are ignored. The logging itself is achieved using SOAP intermediaries. In Apache Axis such intermediaries are called *handlers*, and they can be added to each deployed Web service's request and/or response flow.

This level of logging can be reached by developing SOAP intermediaries that log all SOAP traffic and integrating them into the service-oriented system. The WS themselves are not affected.

Level 4: Logging client activity

Logging on levels 1 through 3 involves provider-side logging only. Such scenarios are comparable to the situations addressed by Web usage mining where interaction with a single, central server is assumed. On level 4 logging should also be done on the consumer side, e.g., when a WS is a composite service and itself makes calls to other WS possibly deployed on other hosts. Level 4 logging, therefore, greatly expands the opportunities of workflow mining, since a system's activities as a client are also recorded. If a workflow spans over multiple hosts or systems, such interactions may be discovered if level logs are available.

Apache Axis allows for the use of client-side handlers which are invoked whenever the Axis API is used to consume services [23]. The system itself is not affected when level 4 logging is to be implemented.

In [24,25] the authors present a sophisticated logging architecture for service-oriented systems. Their logging facilities reach level 4 logging on the scale we propose. Some of the authors' goals they wish to reach using the log architecture are also addressed by WSIM on the operational level, such as service execution time and Web service usage.

Level 5: Providing for process information

As stated in subsection 2.3, successful mining for exact workflow models requires workflow information in log records. We have also shown that such information is not available in conventional Web server logs. One of the goals of WSIM is to make service-oriented systems fit for process mining by providing suitable logs.

Table 1. Summary of logging features and mining opportunities

Level	Logged information	Logging facilities	Mining opportunities
1	- consumer IP - invoked WS - timestamp - HTTP status code	- Web server	- service utilization - consumer tracking - failure rates
2	- level 1 + - SOAP request & response - timestamps	- HTTP listener & logger	- level 1 + - WS execution time - analysis of SOAP messages
3	- invoked WS & operation - SOAP request & response - timestamps	- WS container - SOAP handlers	- level 2
4	- level 3 + - consumer-side activity	- WS container - SOAP handlers	- level 3 + - client-side activity
5	- level 4 + - workflow information	- level 4 + - Web services	- level 4 + - Web services workflows

Therefore, the highest level of logging in SOA must provide for both a workflow identifier and a case identifier in each interaction that is logged. This in turn requires for additional design and implementation considerations when implementing service-oriented architectures. To be precise, Web services must "cooperate" in a sense that they are able to receive and forward information regarding the workflow they are currently part of. The exchange of that information between Web services can be done by using SOAP headers. Web services should process the workflow information and forward it to other WS it consumes in the process. If such measures are taken when designing Web services the recorded logs will allow the mining of exact models of complex workflows. Our future work includes the development of an API that facilitates the implementation of Web services which allow for level 5 logging.

4 Motivating Example

In this section we present a motivating example showing possible applications of WSIM. Although only simple, the example has been fully implemented and it demonstrates, on a small scale, what could be done with WSIM on a much larger scale in the not so distant future. We first present the scenario and then describe the mining process in Section 5.

The sample service-oriented system we have developed is a collection of 13 WS which are shown in Figure 1. We assume to have ownership over host 1, i.e., access to its logs, while host 2 and 3 are third-party owned.

The workflow is started by a client application making a call to the OrderService (WSA), which triggers the execution of an application implementing the workflow by using the other Web services. The DataValidationService (WSB) is called to check if the customer is registered and if the given product is available

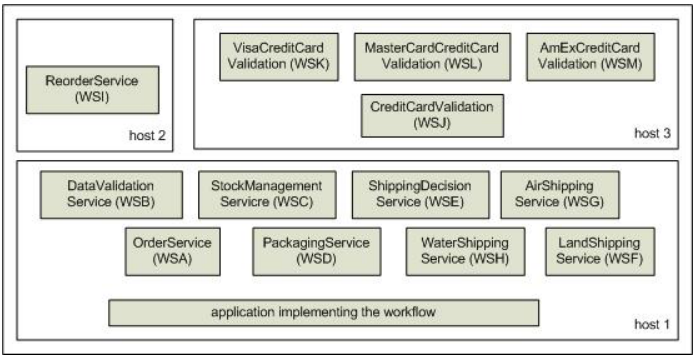

Fig. 1. Case Study Service oriented system

in the desired quantity. If the product is not available the StockManagementService (WSC) is called, which in turn makes a call to the ReorderService (WSI) on host 2. Furthermore, the DataValidationService (WSB) calls a CreditCardValidation (WSJ) on host 3. The CreditCardValidation-WS makes, depending on the company in the credit card information supplied, a call to either the Visa-, the MasterCard-, or the AmEx- Web service. Once order data and customer data have been successfully validated, the PackagingService (WSD) is invoked. When completed, the ShippingDecisionService (WSE) is called which determines the method of shipping and finally invokes one of the ShippingServices. This completes the workflow.

In addition to the example workflow we have implemented logging facilities which reach level 5 on our proposed scale, i.e., the Web services are able to receive, process, and forward workflow information. The SOAP enabling software used was Apache Axis [23], deployed on an Apache Tomcat servlet engine [21]. The logging is done by SOAP handlers on both the client and the server side which are invoked before and after the invocation of each WS. Therefore, all request and response messages are logged. Each log record is of the format

```
uniqueID - mappingID- targetWS - msgType - SOAPmessage - timestamp
```

where mappingID is an identifier mapping a SOAP request to its corresponding response, targetWS is the URL of the invoked service, msgType is the type of message, i.e., request or response, and SOAPmessage is the complete SOAP message that was sent.

5 The Mining Process

In our experiments the workflow was executed several times so that every possible flow of the process was performed at least once, e.g., in at least one instance of the workflow products were unavailable and the ReorderService had to be called. Once a sufficient amount of data was collected the log records were preprocessed to a format more suitable for data mining. For example, the recorded

SOAP messages were analyzed for workflow information in their headers and the invoked method was retrieved. Also, corresponding log records, i.e., request and corresponding response, were combined into one log record, so that one call to a Web service was represented by one record. The log format after data pre-processing was as follows:

```
workflowID - instanceID - targetWS - operation - SOAPrequest -
requestTimestamp - SOAPresponse - responseTimestamp
```

5.1 Scenario

In order to demonstrate the benefits of logging on level 5 consider the following scenario. The formerly independent hosts 1 - 3 in our case study are now under ownership of one virtual organization. Such a situation might occur when companies merge or when the independent hosts were previously managed by separate departments and are now put under the control of one central IT-department or outsourced. In such a situation it is beneficial when workflow information is available for processes that are performed using services deployed on (formerly) independent hosts.

5.2 Data Pre-processing for ProM

In our experiments we used ProM as the process mining tool [26]. ProM requires an input file in XML format which contains the information concerning processes and their instances. It should be noted that ProM works with event-based data in order to be able to discover complex workflow patterns. This means that not activities are recorded but rather events. The simplest events regarding an activity are `start` and `complete`. Therefore, the Web service execution logs have to be transformed into an event-based view. The XML format of a ProM - input file is roughly the following:

The root element is the <`WorkflowLog`> element and it has a number of <`Process`> sub-elements, each encapsulating execution data of *one* process, or workflow. A <`Process`> element has a number of <`ProcessInstance`> child elements. A <`ProcessInstance`> has numerous <`AuditTrailEntry`> child elements. An <`AuditTrailEntry`> represents one log record and contains an identifier of the activity, the event-type and a timestamp.

The pre-processed logs described in the previous subsection were traversed for log records belonging to a given workflow, i.e., "orderprocess" in our case. Because ProM requires log entries to be sorted by process instances, a list of instances was constructed, which was then traversed until all instances were processed. As an activity identifier we used the values `targetWS-operation` of the pre-processed logs. Furthermore, the receiving of a SOAP request was taken as the `start` event of an acivity, the sending of the SOAP response was considered the `complete` event. Also, since both, provider- and consumer-side interactions were logged, care had to be taken that interactions were only considered once, in cases of provider and consumer residing on the same host.

5.3 Mining Results

The procedure of creating a ProM - compatible workflow log was performed several times with varying amounts of log data available. In one session only the logs collected at host 1 were used. This scenario may very well appear in the real world when system owners decide to upgrade their infrastructure to a BPEL - enabled environment.

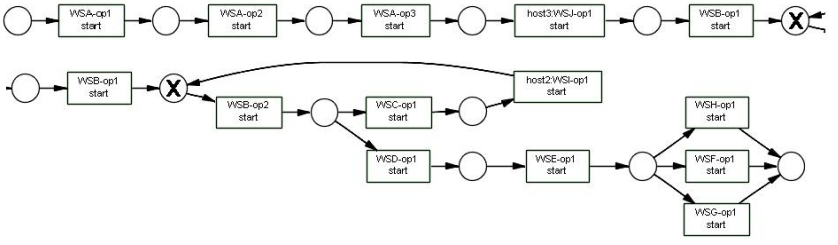

Fig. 2. Process model of host 1

The log data first had to be cleaned from interactions which were recorded twice, i.e., by both provider-side and consumer-side handlers. For all interactions with third party hosts the consumer-side entries made at host 1 were used to capture these interactions. From the information extracted from the logs the workflow model in Figure 2 was constructed using ProM. The workflow model turned out to be complete and correct.

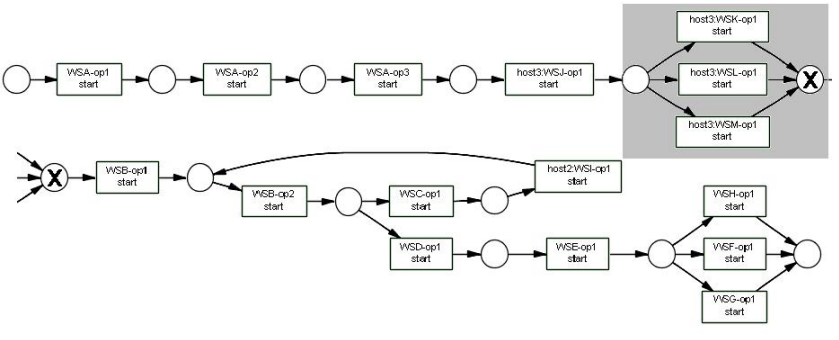

Fig. 3. Process model of all hosts

Note that the names of WS and operations were abbreviated in Figures 2 and 3 and only start-events were considered in order for the models to be displayable here. The abbreviations correspond to those used in Figure 1. Furthermore, the original images created by ProM had to be "cut in half". The circles marked X are where the two halves should be joined.

In another experiment, we considered all logs collected on the three hosts. This scenario is imaginable when formerly independent hosts come into ownership of one virtual organization. Again, the data was pre-processed and cleaned from double recorded interactions. The workflow model in Figure 3 was constructed which now includes the workflow performed on host 3. Figure 3 demonstrates the benefits from providing logs at a level 5 of our scale. The portion of the model highlighted is the additional knowledge gained from including logs from host 3 in the mining process.

6 Conclusion and Future Work

In this paper we have presented our approach of Web services Interaction Mining (WSIM) with a focus on mining for Web services workflows. We have examined related work, of which we specifically point out Web usage mining and process mining. Since the possibilities of process mining in SOA seem limited using traditional logging facilities like Web server logs, we have presented a classification of service-oriented systems by the richness of the log data they provide. Depending on the level of logging, different methods of WSIM may be applied to a system. If level 5 logging is implemented, a service-oriented system may be mined for complete workflow information using already existing process mining tools such as ProM. As a case study we presented such a system and described the process of processing the log data to create logs understandable by ProM. We also showed the results achieved by mining the logs for workflows.

Since level 5 logging cannot be assumed in many service-oriented systems, our future work will be directed at developing strategies and algorithms for WSIM in systems which provide less log information. Specifically, we will examine WUM strategies and adapt them to the needs when mining in logs provided by service-oriented systems.

Our current research effort goes in two directions. On the one hand, we examine the possibilities of workflow mining in environments with limited log information. In such environments workflow mining may be impossible, which is why we currently focus on discovering frequent interaction patterns rather than complete workflows. On the other hand, we are examining strategies to generate workflow information in incomplete logs which may later allow for workflow mining. The most promising strategy at the moment seems to be that of session reconstruction in service-oriented systems.

References

1. Gombotz, R., Dustdar, S., Baina, K.: Web Services Interaction Mining. Technical Report TUV-1841-2004-16, Vienna University of Technology (2004) http://www.infosys.tuwien.ac.at/Staff/sd/papers/TUV-1841-2004-16.pdf.
2. Gombotz, R., Baina, K., Dustdar, S.: Towards Web Services Interaction Mining Architecture for e-commerce Applications Analysis. In: Proceedings of the Conference on E-Business and E-Learning. (2005)

3. Alonso, G., Casati, F., Kuno, H., Machiraju, V.: Web Services - Concepts, Architectures and Applications. Springer-Verlag (2004)
4. OASIS: OASIS Web Services Business Process Execution Language (WSBPEL) TC (2004) http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsbpel.
5. ActiveBPEL: ActiveBPEL Engine - Open Source BPEL Server (2004) http://www.activebpel.org.
6. Oracle: Oracle BPEL Process Manager (2004) http://www.oracle.com/technology/products/ias/bpel/index.html.
7. Patricio Galeas: Web Mining (2005) http://www.galeas.de/webmining.html.
8. Wu, K., Yu, P., Ballman, A.: SpeedTracer: A Web usage mining and analysis tool. IBM Systems Journal **37** (1998)
9. hypKNOWsis: hypKNOWsis ... making sense of your data (2004) http://www.hypknowsys.org.
10. Apache Software Foundation: Log Files - Apache HTTP Server (2005) http://httpd.apache.org/docs-2.0/logs.html.
11. Microsoft TechNet: IIS Logging Server Activity (2004) http://www.microsoft.com/technet/archive/IIS3/iischp7.mspx .
12. Spiliopoulou, M., Mobasher, B., Berendt, B., Nakagawa, M.: A Framework for the Evaluation of Session Reconstruction Heuristics in Web Usage Analysis. INFORMS Journal of Computing, Special Issue on Mining Web-Based Data for E-Business Applications (2003)
13. Zhang, J., Ghorbani, A.A.: The Reconstruction of User Sessions from a Server Log Using Improved Time-oriented Heuristics. In: Proceedings of the Second Annual Conference on Communication Networks and Services Research (CNSR'04). (2004)
14. Berendt, B., Mobasher, B., Nakagawa, B., M.Spiliopoulou: The Impact of Site Structure and User Environment on Session Reconstruction in Web Usage Analysis. In: Proceedings of the 4th WebKDD 2002 Workshop at the ACM-SIGKDD Conference on Knowledge Discovery in Databases. (2002)
15. van der Aalst, W., Weijters, A.: Process mining: a research agenda. Computers in Industry **53** (2003) 245–264
16. van der Aalst, W., van Dongen, B., Herbst, J., Maruster, L., G.Schimm, Weijters, A.: Workflow Mining: A Survey of Issues and Approaches. Data and Knowledge Engineering **47** (2003) 237–267
17. Herbst, J., Karagiannis, D.: Workflow mining with InWoLvE. Computers in Industry **53** (2003) 245–265
18. Schimm, G.: Mining exact models of concurrent workflows. Computers in Industry **53** (2003) 265–281
19. Pinter, S., Golani, M.: Discovering workflow models from activities' lifspans. Computers in Industry **53** (2003) 283–296
20. Cook, J., Du, Z., Chongbing, L., Wolf, A.: Discovering models of behaviour for concurrent workflows. Computers in Industry **53** (2003) 297–319
21. Apache Software Foundation: The Jakarta Site - Apache Jakarta Tomcat (2004) http://jakarta.apache.org/tomcat/.
22. Apache Software Foundation: Apache SOAP Documentation: User's Guide (2004) http://ws.apache.org/soap/docs/.
23. Apache Software Foundation: WebServices - Axis (2004) http://ws.apache.org/axis/.
24. Serra, S., Campos, L., Pires, P.: A Data Mart Approach for Monitoring Web Services Usage and Evaluating Qualtity of Services. In: XVIII Brazilian Sysmposium of Data Bases. (2003)

25. Serra, S., Campos, M., Pires, P., Campos, L.: Monitoring E-Business Web Services Usage through a Log Based Architecture. In: Proceedings of the IEEE Interantional Conference on Web Services (ICWS'04. (2004)
26. Technische Universiteit Eindhoven: Process Mining Reserach (2004) `http://www.processmining.org`.