10-2006

Service Pattern Discovery of Web Service Mining in Web Service Registry-Repository

Qianhui Althea LIANG
Singapore Management University, althealiang@smu.edu.sg

Jen-Yao CHUNG

Steven MILLER
Singapore Management University, stevenmiller@smu.edu.sg

Ouyang YANG
Singapore Management University, youyang@smu.edu.sg

DOI: https://doi.org/10.1109/ICEBE.2006.90

ICEBE '06: IEEE International Conference on E-Business Engineering, 24-26 October 2006, Shanghai: Proceedings
http://doi.ieeecomputersociety.org/10.1109/ICEBE.2006.90

Service Pattern Discovery of Web Service Mining in Web Service Registry-Repository

[1]Qianhui Althea LIANG, [2]Jen-Yao CHUNG, [1]Steven MILLER and [1]Yang OUYANG
[1]*School of Information Systems, Singapore Management University, Singapore*
althealiang@smu.edu.sg, stevenmiller@smu.edu.sg, youyang@smu.edu.sg
[2]*IBM T. J. Watson Research Center, USA*
jychung@us.ibm.com

Abstract

This paper presents and elaborates the concept of Web service usage patterns and pattern discovery through service mining. We define three different levels of service usage data: i) user request level, ii) template level and iii) instance level. At each level, we investigate patterns of service usage data and the discovery of these patterns. An algorithm for service pattern discovery at the template level is presented. We show the system architecture of a service-mining enabled service registry repository. Web service patterns, pattern discovery and pattern mining supports the discovery and composition of complex services, which in turn supports the application of web services to increasingly complex business processes and applications.

1. Introduction

As Web services technology [1] surges and evolves into the de facto platform for business integration, more and more businesses rely on the discovery of appropriate Web services developed by outside providers to fulfill their business goals. This reliance allows businesses to focus on mission critical task development while making use of existing routine functions in the form corresponding Web services. As a large amount of such functions become available by being published and registered as Web services, this reliance on the use of externally created Web services creates a major challenge in terms of discovering the correct services for distributed application development through service composition and process integration.

Discovery of suitable services or processes that satisfy the user's service needs has long been of great interests to both businesses and to the Web service research community and has been made a standardized function of any service registry conforming to the Web service model, such as UDDI [2]. Furthermore, the need for service discovery is further increased by automated service composition and business process integration. The expectation is to make new services or to create new processes by combining the core business processes with other required services existing out there on the Web.

The number of available services is growing large and still rapidly increasing. With the presence of a large number of services, the discovery of services for integration or composition purposes becomes a daunting task. Better supporting techniques are required that make better use of the large data set on the usage of Web services.

Meanwhile, the task of a service registry may not be limited to simple functions like accepting service registration, posting registration information, and answering queries of service requesters. Businesses look for automated solution generation for their integration problems. They look for decision making support. A registry can answer to these demands only if it holds all service related artifacts, a.k.a. registry-repository [3][4], and if it performs analysis on these artifacts. In particular, a registry can potentially improve Web service usage through integration enabling, if it maintains and analyzes the available service artifacts and their correlation patterns.

Service mining, a technology similar to web mining, has been proposed [5] in order to make use of the service artifacts in a registry-repository and to improve service discovery via analysis of service usage patterns. In this paper, we present the concept of patterns of service usage and their discovery for service mining and justify the idea in a service registry-repository setting. Patterns of services are studied in three hierarchical levels: user request level, template

level, and instance level. We also present the detailed algorithm for service pattern discovery at the template level. The rest of the paper is organized as follows. In section 2, we briefly introduce service mining and its usage in a service registry repository. Three levels of service usage patterns are defined and discussed respectively in section 3 and section 4. In section 5, we introduce the registry-repository architecture. Related work is described in section 6. Section 7 states the conclusions and Section 8 lists the references.

2. Service mining in service registry-repository

Service mining is defined as the automated discovery and analysis of how Web services are used in a collective way. It aims to discover services that meet the specified requirements.

How a Web service is described is essential to service mining. If Web services are defined and described in a machine understandable manner, their discovery will be a much easier job. Adopting OWL-S as the service description language, knowledge of Web services consists of four parts, which are listed below [5]:

1. *Service Profile information,* such as service provider's contact information and service operation input and output information
2. *Service grounding information,* such as the protocol used to interact with the service
3. *Service constraint information,* such as the conditions that limit the use of the service
4. *Service usage data,* such as patterns associated with the use of the service

Service mining is meant to discover all four types of knowledge. In particular, goals of service mining are as the followings:

1. *When constructing a new service for a business process by service composition, predict the correct service functions to select.*
2. *When building such services, predict the good-profiled service partners to collaborate with.*
3. *Optimize composite service execution for performance issue due to the scalability of the composite service*
4. *Classify services*

The purpose of service mining is to discover the patterns of service usage. *Service patterns* are defined as specific ways that Web services (or service operations) are used repeatedly by a group of users with similar properties as well as that they co-occur and are correlated to and interacted with each other. In order to analyze in details what service patterns are and why they matter, we define three hierarchical levels from three different aspects as: user request level, template level and instance level. To be more specific, user request level mainly concerns about the users, such as the information about the users and how a user submit a request. By analyzing the user's data, we are able to connect the users' concerns and interests with related Web services. At template level, we focus our study on the abstract information of Web services, such as service interfaces, operations, messages etc. The data only relate to the usage in terms of certain Web service abstracts, not individual service providers. At instance level, we study service providers of individual simple services or simple processes. The data shall be related to various service providers (or the service/binding element that contains the actual URL of the service provider). At this level, our subjects are the Web service providers in the real world. We investigate what kind of constraints of certain Web services are published by certain organizations and how different service providers with different constraints tend to work together.

Two main research fields have important influences on service mining i.e. Web mining [6][7] and workflow mining [15]. Service mining differs from both mining techniques in various ways. Web content mining operates on raw Web-based information, such as HTML pages, which may not be well-organized. However, service mining works on Web services registered in a service registry, which are well-categorized. This helps make the search more efficient. Services are semantically more complicated than Web content, which allows mining services to be carried out in a more sophisticated way. Process mining refers to the effort to discover workflow patterns in a given set of log data [15], where a workflow is defined as an ordered set of activities that are performed to accomplish a complex task. In service mining, we care in what patterns users use services as well as how Web services correlate or co-occur in a meaningful unit of business activity. Process mining techniques may be useful to specialize service mining. In order to make the study of service mining general, we have not incorporated the ideas in process mining.

Service registry repository holds all the service description documents, including WSDL [9], OWL-S [10], BPEL4WS [11] and WSCI [12] documents. WSDL and OWL-S are used to describe what the Web service is like and how it works. BPEL4WS provides a language for the formal specification of business

processes and business interaction protocols. WSCI is an XML-based interface description language that describes the flow of messages exchanged by a Web service participating in choreographed interactions with other services.

3. Three levels of service usage data

The concept of service usage partially follows that of Web usage in Web mining. Web usage mining is the automatic discovery of user access patterns from Web servers. Similar to Web usage, service usage implies how users request for Web services and the patterns they usually follow. Furthermore, service usage concerns not only about the user access patterns but also the web services composition or business process integration patterns. The Web service composition patterns refer to how different services are correlated and interacted with each other in a composite service template and business process integration patterns refer to how individual processes are combined to satisfy business goals. Service usage can be divided into three levels as user request level, template level and instance level.

3.1. User request level

At user request level, service usage is usually related to how a user submits a request. At this level, all services and processes are treated as autonomous units. Usage data about services are collected and analyzed at the outmost business process application level without referring to components and their interactions. In other words, how complex they are and whether they are provided by a single provider or multiple providers, are transparent to this level of study. This level of usage concerns about how composite services are used by the users. In this level, the usage data have the following aspects:

- *User usage frequency*: Within a certain period, how many times the same user uses similar processes or composite service.
- *User group usage frequency:* Within a certain period, how many users from the same geographical area use similar processes or composite services.
- *Usage frequency* : Within a certain period, how many users use similar composite services.
- *Price range*: For a sequence of usage of the same (similar) composite service(s) or business processe(s), what the price range is.
- *Security level :* What security level the collection of services from the same group of providers has.
- *QoS guarantee level*: What QoS guarantee level the collection of services from the same group of providers has.

3.2. Template level

A service template is a flow of component services or business processes whose final output can satisfy the users' complex need or business goal. At this level, service usage concerns about how the components correlate. For example, a service may often co-occur with another service; one service may often be followed by another service. Analyzing such data can help us to find out the association patterns between different Web services. By finding out such patterns through recording and analyzing available service data, we can more easily construct a composite Web service or an integrated process than searching large quantities of un-related Web services in the registry repository. A composite Web service is a conglomeration of existing Web services to offer a new value-added service. It coordinates a set of services as a cohesive unit of work to achieve common goals.

At this template level, the association rules and hierarchy patterns among the component services are analyzed. There can be different relationships among Web services encompassed by one process. In this paper, we simply classify the relationships into two main classes as *interact with* and *orthogonal*. Service S_A interacts with service S_B when service S_A depends on service S_B in some way or vise versa and their executions have to be arranged in a prescribed way. Service S_A is orthogonal with Service S_B when Service S_A and Service S_B can be executed either sequentially or in parallel and are independent of each other.

3.3. Instance level

At instance level, service usage concerns the constraints of service providers. Constraints are restrictions that limit what services look like, and whether and how services can function. By recording the constraints of related Web services, we are going to find out how services with different constraints tend to work together.

4. Service patterns at template level

The discovery of service patterns is a key aspect of the service mining. Algorithms and techniques from several research domains, such as data mining, machine learning, statistics and pattern recognition shed light on service pattern discovery. The goal of the pattern analysis is to extract interesting rules or patterns from the output of the usage pattern data collection process.

4.1. Transactions of Web services

Before we can discuss pattern discovery of service operations, we have to define transactions of Web services. In database management, database transactions refer to a sequence of database operations that satisfy Atomicity, Consistency, Isolation and Durability, a.k.a. ACID property. In data mining, a transaction is made of all the merchandises that a customer purchased in one visit to the store.

Unlike traditional definitions of transactions, *transactions of Web services* for service usage pattern discovery are not defined rigorously. What form a transaction can be very flexibly described as any collection of service operations that are invoked as to perform a meaningful business activity of interest. Under certain a study requirement, a transaction may better be as simple as an operation, while under others, as complex as an entire integrated business process. In between these two extremes, sometimes, transactions are required to be atomic processes [10] that satisfy the ACID constraints. Other times, long-run business activities [10] are of the study interests.

4.2. Service association rules

We can examine a set of transactions of Web services in order to draw some conclusions like the following: if service operation S_A is invoked or expected to be invoked in a transaction of Web services, it is likely that S_B will also be invoked in the same transaction. If we define *Service Operation Set* as a collection of Web service operations and *invocation of a service operation set* as invocation of all operations in the service operation set, the previous conclusion can be generalized into: if operation set I_A is invoked (expected to be invoked) in this transaction, it is likely operation set I_B will also be invoked in the same transaction. Therefore, an association rule of Web services has the following format:

$$I_A \Rightarrow I_B$$

Now we go on to the discovery of service association rules. Here are several definitions related to service operation set. *Support* of an operation set is defined as the fraction of transactions of Web services that contain all the operations in the operation set. *Frequent service operation set* is defined as *sup*(operation set) >= min_support, where min_support is the minimum acceptable support value for a given requirement. A simple algorithm to find frequent service operation sets is listed in List 1. This algorithm is similar to the algorithm in [13] to identify frequent item set in data mining:

List 1

1. FOR each operation {
2. Check if it's a frequent operation set;
3. k = 1;
4. REPEAT {
5. FOR each new frequent operation set I_k with k items {
6. Generate all operation sets I_{k+1} with k+1 operations, I_k is a subset of I_{k+1};
7. From k+1 operation sets I_{k+1}, filter out operation sets among whose subsets, at least one is not a frequent operation set;
8. Process relevant artifacts of all corresponding services once and check if the left I_{k+1} operation sets are frequent;
9. k=k+1; }
10. } UNTIL no new frequent operation sets are identified

With "support of operation sets", we can define "support of service association rules." The support of an association rule of Web services $I_A \Rightarrow I_B$ is the support of operation set $I_A \cup I_B$. The confidence of the rule is the ratio of number of transactions that contain I_A and number of transactions that contain $I_A \cup I_B$, or *sup*($I_A \cup I_B$)/*sup*(I_A). A simple algorithm similar to what is described in [13] can be used to discovery service association rules whose support is greater than *min_support* and whose confidence is greater than *min_confidence*.

We differentiate two types of service associations,

i.e. *Interaction association* and *orthogonal co-occurrence association*. Interaction association, which includes control flows of "uses" and "depends-on" and data flows, indicates that whenever a business process interacts with operation S_A (or operation set I_A), it likely also interacts with service S_B (or operation set I_B). For example, whenever "Bank Loan" process interacts with "risk assessment", it will very likely interact "credit checking". An orchestration scheme (implemented by BPEL) or P2P choreography scheme (implemented by WSCI) dictates whether a component service operation or sub-process is directly interacting with another component service operation or indirectly through an integration engine. Orthogonal co-occurrence, which involves no dependency due to data flow or control flow, tells that service operations orthogonally co-occur in the same transaction of Web services. Separate association rules can be defined for both types of associations.

Category-hierarchy or (inheritance relation) shall be imposed on association rules. When a rule is true for a service operation, the rule is also true for any service operation or process that is on the higher positions in the inheritance hierarchy.

4.3. Sequential service patterns

A sequence of transactions of Web services can be easily transformed to a sequence of service operations by substituting each transaction with one possible invocation sequence of its operations on a sequential process engine. In other words, the sequence describes the order of operations in terms of invocation precedence. This allows us to identify frequently arising service invocation patterns over the time line.

The order of performing a number of sub-functions within a sequence of business processes can be naturally mapped to a sub-sequence of service operations in the set of operations invoked by the corresponding business processes. A *subsequence* of a sequence of operations is defined as any operation sequence that consists of a subset of the operations in the original sequence and that its order does not conflict the order in the original sequence. *Support of the sequence s* is defined as the percentage of the sequences that have *s* as their subsequence.

For example, as in Figure 1, the process of approving/denying a home mortgage loan application can be separated into 3 stages: application, lender verification and closing. Application stage can again be decomposed into 3 sequential steps, *Sales Contract Analysis*, *Earnest Money Deposit Verification,* and *Home Inspection Report Generation*. Lender verification stage can be decomposed to the following sequential steps: *Property Appraisal, Credit Report, Verify Employment and Assets, Verify Housing Payments, Establish Loan-to-Value Ratio*, and *Approval of Mortgage Insurer*. If the loan is approved, *Transaction Generation* followed by *Closing Statement Mailing* make up the closing stage. If "application", "lender verification" and "closing" correspond to three individual transactions i.e. T_1, T_2 and T_3 and if all steps of T_1, T_2 and T_3 comprise an operation set $I_{1\_2\_3}$, both {*Property Appraisal, Credit Report, Verify Housing Payments*} and {*Approval of Mortgage Insurer*, *Transaction generation*, *Closing statement mailing* } are subsets of the operation set $I_{1\_2\_3}$.

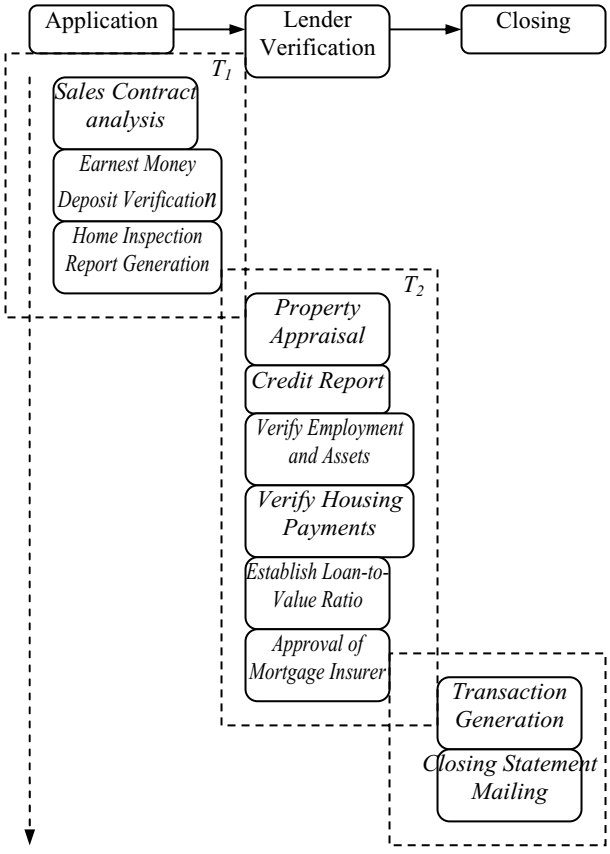

Figure 1: Process of home mortgage application

5. System Architecture

We have developed an architecture for registry-repositories to allow service and process artifacts compilation and service pattern discovery and analysis. A registry-repository not only is a registry to accept publications of services but also serves as a mass storage to maintain and catalog artifacts of services and processes throughout their lifecycles, for example, service or process description documents, process execution logs and process engineering specifications. A registry repository provides accesses to these artifacts. Most of all, it discovers implicit knowledge of services and their usage, such as service usage patterns, and uses the analysis result of the knowledge for process integration or process re-engineering.

Figure 2 shows the registry-repository architecture. The registry-repository has a collection of artifact repositories. Each such repository holds one type of artifacts or similar types of architects. For example, all WSDL documents can be stored in one repository and all OWL-S documents can be put in another. A metadata and catalog repository is connected to all artifact repositories to facilitate catalog and metadata management services of the contents in the repositories underneath it. These services are provided by various functional components of the registry to be discussed shortly. The registry also includes storages for the logs of executed business process instances, which make up one important information source for service analysis. The registry repository primarily 1) offers registration service for businesses to publish services, 2) responds to service queries of service requesters, 3) provides storage for all service-related artifacts, 4) discovers and analyzes service usage patterns and 5) supports automated business process integration and engineering. These five functions can be grouped as three groups 1) 2) 3), 4), and 5). The first two functions listed above are standard functions unanimously expected and implemented by service registries in current SOA. Due to the complex nature of services and versatility of service information, service related artifacts are also stored and managed via the third function. The fourth function makes use of data in both repositories and process execution logs and is introduced to discover possible usage patterns that can help to better use the available services. The last function allows new processes to be integrated or created upon existing processes.

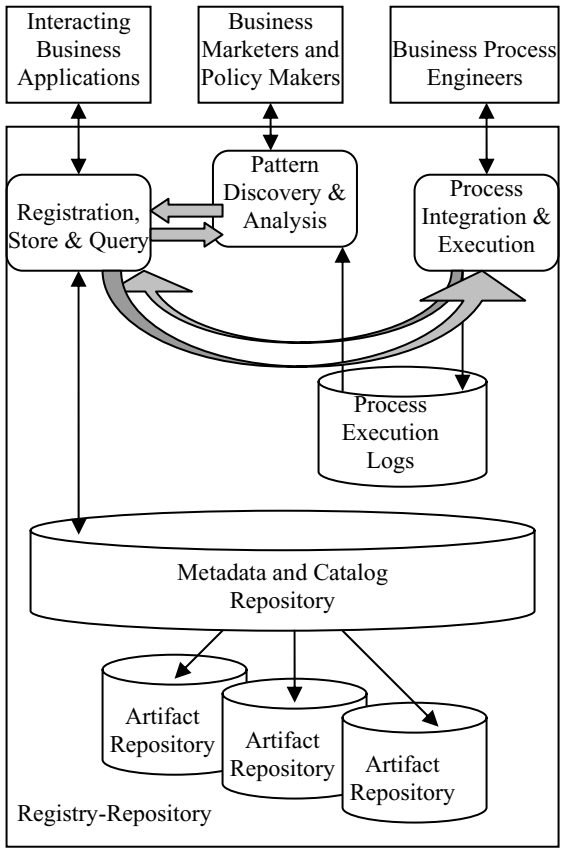

Figure 2: Pattern-Discovery enabled Registry-Repository Architecture

Registration information and stored service artifacts are used to discover and analyze service usage patterns. In addition, the process execution logs provided by Process Integration and Execution component have to be analyzed for usage pattern discovery. Process integration relies on pattern discovery and analysis results to improve integration performance. The integrated processes will be deposited into the repository to enhance service artifacts. Events during the executions will be written to the logs. Different functions of registry repository will be used by different roles in businesses and for different purposes. The registration, query and store function is mostly used by interacting business partners. The pattern discovery function is interesting to marketers and business strategy makers to produce better business strategies. The process integration and execution function is mostly useful to business engineers for business re-engineering.

A detailed architecture for Web service pattern discovery in a registry-repository is also developed by us as depicted in Figure 3.

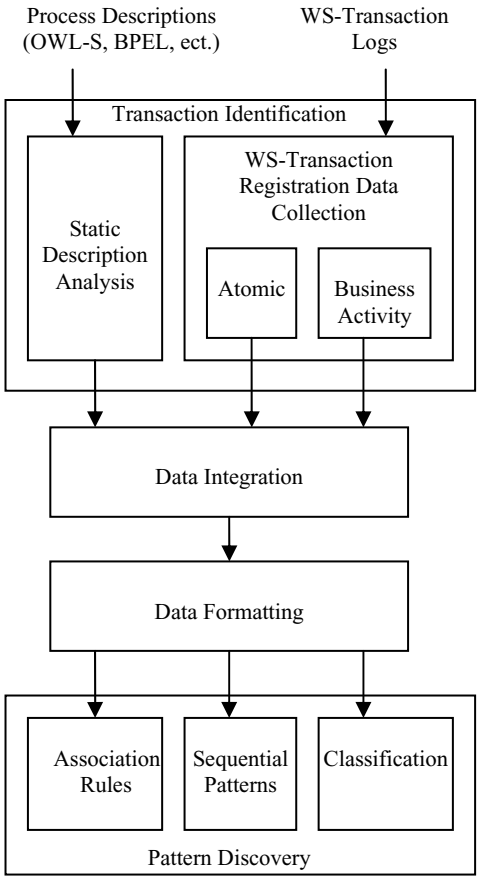

Figure 3: Service Mining Architecture in Registry-Repository

The architecture divides the Web service pattern discovery process into four main stages, i.e. transaction identification, data integration, data formatting and pattern discovery. According to the previous definition of transactions of Web services as collections of service operations that form meaningful units of business activities, different sources can be used to identify transactions. Business process descriptions in process description languages, such as BPEL4WS and OWL-S, provide process components and their interaction interfaces. On the other hand, process instance execution logs contain event traces about transactional services during their executions. Both information sources shall be input to identify transactions of interest. Static description analysis will take process descriptions and produce transactions constrained by business process definitions. Dynamic data of both atomic and business activity transactions are processed. Output from all three analysis channels will go through data integration so that different transaction sources are all merged to one source. Integrated transaction data are then formatted to satisfy different pattern discovery purposes, including association rule discovery, sequential pattern discovery and classification.

6. Related Work

Valuable research results have been reported in web mining and workflow mining. Especially, the idea of workflow mining has aroused interests to some researchers. Agrawal and D.Gunopulos [14] proposed an algorithm for mining workflow logs early in 1998. Recent years have seen exiting progress on this research. W.M.P. van der Aalst et al. [15] use the term process mining technique in developing a method of distilling a structured process description from a set of real executions. They present an algorithm to extract process models from the even logs which is proved to successfully mine any workflow represented by a so-called SWF-net and represent it in terms of a Petri net. Ricardo Silva, Jiji Zhang and James G.Shanahan [16] present an algorithm for learning workflow graphs from data which for the first time makes use of a coherent probability model. They describe the models of work as abstract representations of typical process instances modeling the causal and probabilistic dependencies among tasks. Since the usual representation of work processes is workflow graphs, they try to figure out the graph of work models.

Presently, with the development of Web services, there is a trend of utilizing the techniques in web mining and workflow mining in service-oriented world. Sami Bhiri et al. [17] propose a concept of transactional patterns for specifying flexible and reliable composite Web services. Transactional patterns extend from workflow patterns and integrate the advanced transactional models. In their work, they concentrate on the transaction properties for composition and synchronization of component Web services which are called Transactional Composite Services. Similar to our work, they defined several dependencies between Web services and also the control flows and transactional flows. The transactional patterns can be used for Web services composition.

Robert Gombotz et al. [18] present a Web services interaction mining architecture (WSIM) for analyzing interactions between Web service consumer and provider. They also utilize the web usage mining and process mining in their work and developed the Web services interaction mining regards to three levels of abstraction which are Web services workflows, Web

services Interactions and Web service Operations. What's more, in each level they present a normative format of log entries. Their work could potentially improve the manageability of a Web service or an entire service-oriented system. In their later work, they precede the idea of Web services interaction mining with a focus on mining for Web services workflows and present a motivating example showing possible applications of WSIM.

7. Conclusions

The paper defines service usage data at three different levels: user request level, template level and instance level. Discovery of service usage data at the template level is discussed extensively. A service-pattern-discovery enabled registry-repository architecture is investigated in the paper. The aim of this research is to provide the concepts and methods for detailed and systematic analysis of service patterns such that automated process integration can be enabled and improved.

In this paper, simple algorithms for service pattern discovery at the template level are discussed. More sophisticated algorithms and complete solutions are yet to be investigated. Service usage patterns at the user request level and instance level need to be further developed, because such results will have an important impact on business process creation and related strategy formation. In terms of data collection and experiments, we hope to deploy the architecture in a public domain so more real service usage data can be collected and the performance of our algorithms can be empirically evaluated.

8. References

[1]. Web Services Architecture, http:// www.w3.org/TR/ws-arch/, 2004.

[2]. UDDIV3, http://www.oasis-open.org/committees/uddi-spec/doc/tcspecs.htm#uddiv3, 2004.

[3]. Using an SOA Registry Repository, www.sun.com/ products/soa/registry/soa_registry_wp.pdf, 2005.

[4]. SOA Enterprise Patterns, Services, Orchestration, and Beyond, Dragos Manolescu and Boris Lublinsky, to be published by Morgan-Kaufmann Publishers in 2007.

[5]. Q. Liang, S. Miller and J. Chung, "Service Mining for Web Service Composition", by IEEE International Conference on Information Reuse and Integration (IEEE IRI-2005), Las Vegas, Nevada, USA, 2005.

[6]. O. Etzioni, The World Wide Web: quagmire or gold mine? Communications of ACM, Nov. 1996.

[7]. R. Kosala and H. Blockeel, Web mining research: A survey, SIGKDD Explorations, 2(1), pages 1 - 15, 2000.

[8]. Wu, K., Yu, P. and Ballman, A., "SpeedTracer: A Web usage mining and analysis tool," IBM Systems Journal 37, 1998.

[9]. WSDL, Web Services Description Language 2.0. http://www.w3.org/TR/wsdl20/, 2006.

[10]. OWL-S: Semantic Markup for Web Services, http://www.daml.org/services/owl-s/1.1/overview/, 2004.

[11]. BPEL4WS, Specification: Business Process Execution Language for Web Services Version 1.1. ftp://www6.software.ibm.com/software/developer/library/ws-bpel.pdf, 2003.

[12]. WSCI, Web Service Choreography Interface (WSCI) 1.0. http://www.w3.org/TR/wsci/, 2002.

[13]. R. Ramakrishnan, J. Gehrke, Database Management Systems (Hardcover), McGraw-Hill Science/ Engineering/Math 3 edition, 2002.

[14]. R.Agrawal, D.Gunopulos, and F.Leymann, "Mining process models from work-flow logs," Proc of 6th international Conference on Extending Database Technology, pages 469-483, 1998.

[15]. Wil van der Aalst, Ton Weijters and Laura Maruster, "Workflow Mining: Discovering Process Models from Event Logs," IEEE Transactions on Knowledge and Data Engineering, vol. 16, no. 9, pp. 1128-1142, Sept. 2004.

[16]. Ricardo Silva, Jiji Zhang and James G.Shanahan, "Probabilistic Workflow Mining," Proceeding of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining (KDD '05), Chicago, Illinois, USA, 2005.

[17]. Sami Bhiri, Olivier Perrin and Claude Godart, "Extending workflow patterns with transactional dependencies to define reliable composite Web services," Proceedings of the Advanced International Conference on Telecommunications and International Conference on Internet and Web Applications and Services, 2006.

[18]. Robert Gombotz and Schahram Dustdar, "On Web Services Workflow Mining," BPM2005 Workshops, LNCS 3812, 2005.