A Model Driven Service Identification Approach For Process Centric Systems

Vishal Dwivedi, Naveen Kulkarni
SETLabs, Infosys Technologies Ltd
{ Vishal_Dwivedi, Naveen_Kulkarni}@infosys.com

Abstract

The raison d'être for web-services is that they could be used to compose new services. As in process centric systems, wherein blocks of software implement a functional domain through a set of linked activities, a SOA based system modeled through web-services, requires business services (of low granularity) which implement various business functions. Such business services could not only ensure a high degree of re-use but could also help in realizing service oriented architecture in its true sense. Although today quite a few approaches for the problem exist, but most of them are bottom up solutions, or mainly focus on domain decomposition and composite service formation. Not many of them have yet utilized the full potential of business processes, which form the backbone of any enterprise. In this work we define what a business service would be like, vis-a-vis its corresponding business function, and how it could be realized through data, utility, information and other services. We present our service identification approach which utilizes process maps and service hierarchies and towards the end we discuss some of our current work in building an automation tool (named SQUID) for service identification.

1. Introduction

With the increased focus of the enterprises towards web based solutions and services, there has been an increasing demand for business services and componentization of business functionalities. Even in scenarios involving cross-organizational processes, the participating organizations most often require to expose their processes as a set of business functionalities which could be provided as a service (which may not necessarily be a web-service) and extended by the service provider as and when requested by the service consumer.

Some of the typical assumptions which are often misleading in a business process automation scenario are to consider business processes as a simple orchestration of web-services or even a collection of system activities comprising of workflows. Most of the times business processes involve human activities, utility functions and even totally manual scenarios such as 'collect/process documents' etc. A bottom up approach aggregating fine grained WSDL interfaces is hence not always a feasible scenario to provide a business service. A detailed example scenario later in Section 4.1 justifies our assumptions.

While realizing SOA through web-services, the biggest problem often faced is to consolidate the numerous existing web-services which may not be originally designed for use under the SOA architecture. The current trends within enterprises are to follow a bottom up approach for service identification and definition for SOA based service centric systems (SCSS). The raison d'être for these bottom up approaches is that most of the enterprise systems are still dependent on legacy systems for some or the other part of their functionality. Even if the systems are not legacy oriented, it is highly probable that the fine grained services which they expose might not have been designed for SOA architecture or consumable in process centric systems.

Handling such fine-grained services is not only difficult because of the level of complexity involved in integration but also prone to changes due to change in requirements. Even with various approaches involving loosely coupled service integration, handling fine grained services is a humongous task owing to the sheer complexity involved in integration and update for a large business scenario. Business services alleviate the problem to a certain extent where the integrators can deal with services of low granularity, providing business functions. However, the bigger problem of identifying and composing such business services remains. In this work we try to address this problem through our approaches in implementing SQUID system using hierarchical process maps represented in UML which we convert to MOF meta-models for identification algorithms.

2. Previous Work

There has been quite a lot of work on service identification, with approaches ranging from top-down (business-process or architecture driven) to bottom-up via exposing current assets and aggregating them. While [1] and [2] describe some top down approaches, there has been some work towards bottom up [3] and meet in the middle approaches [4] [5]. A good comparison of currently available service identification approaches could

978-0-7695-3313-1/08 $25.00 © 2008 IEEE
DOI 10.1109/SERVICES-2.2008.16

be found in [6] where Karsten et al provides a stakeholder based approach to SOA development.

Inaganti et al in their article [7] detail out some of these service identification approaches. One of the biggest issues faced by most of these is their dependency on functional domains, and hence most of the current work involves human involvement. There has been some work by Zisman et al [8] [4] [9] towards a UML based service discovery framework. Their UML based service discovery tool is an effective mechanism to iteratively identify services using behavioral models. While UML profile for web-services [10] have been extensively used for service ontology modeling, and has been pushed around by IBM, the current approaches are far from providing an automated service discovery framework. There even exists some work towards ontology mapping for MDA conversions [11] for achieving PIM (platform independent) models for web-services from processes but most of these works are theoretical in nature.

The current state of practice of service identification and service definition within organizations adopting SOA and SCSS is more of a bottom up approach. One of the biggest reasons which could be attributed to this is that most organizations tend to expose their current system functionality within existing applications and legacy systems as services. Not only this requires understanding of the functional domain, but also sound classification criteria to classify services which most often are fine grained performing a certain task. Having process-maps (as shown in Figure2) could hence be a good starting point for effective automation of the service identification exercise.

Although a lot of literature exists in this area, there still lacks a unified methodological approach to service identification and the general practice of the industry has been manual categorization, analysis and preparation based steps as mentioned in [6]. Our current work tries to address some of those issues and the SQUID tool semi-automates some of the heuristics used currently for such manual exercises.

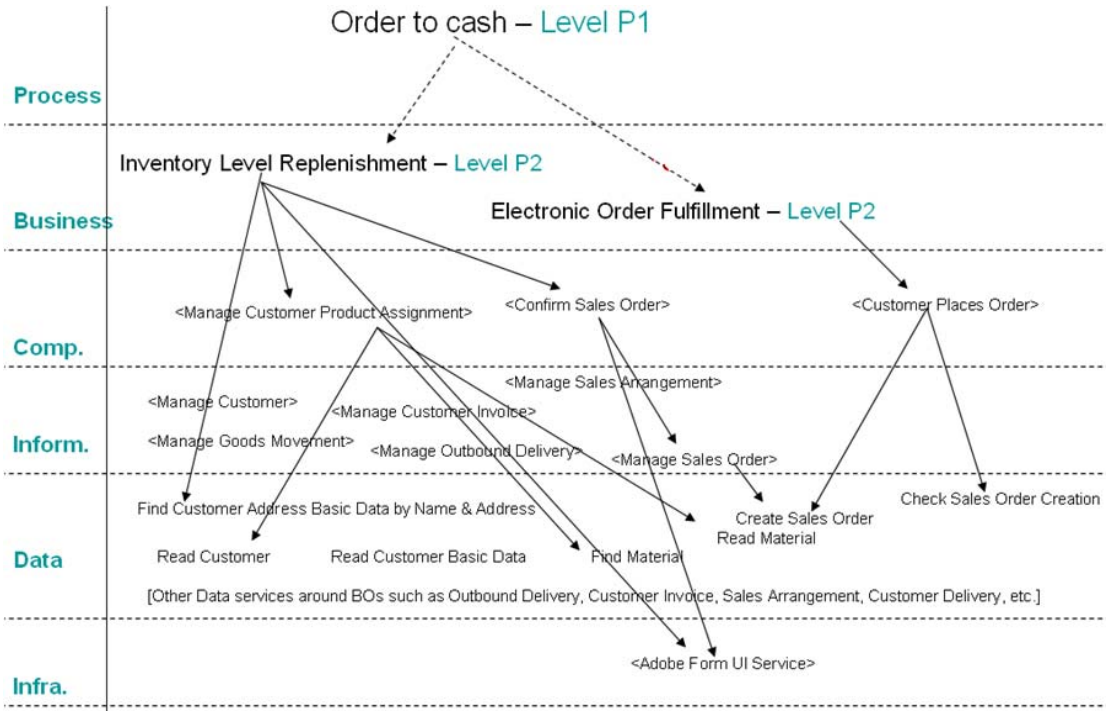
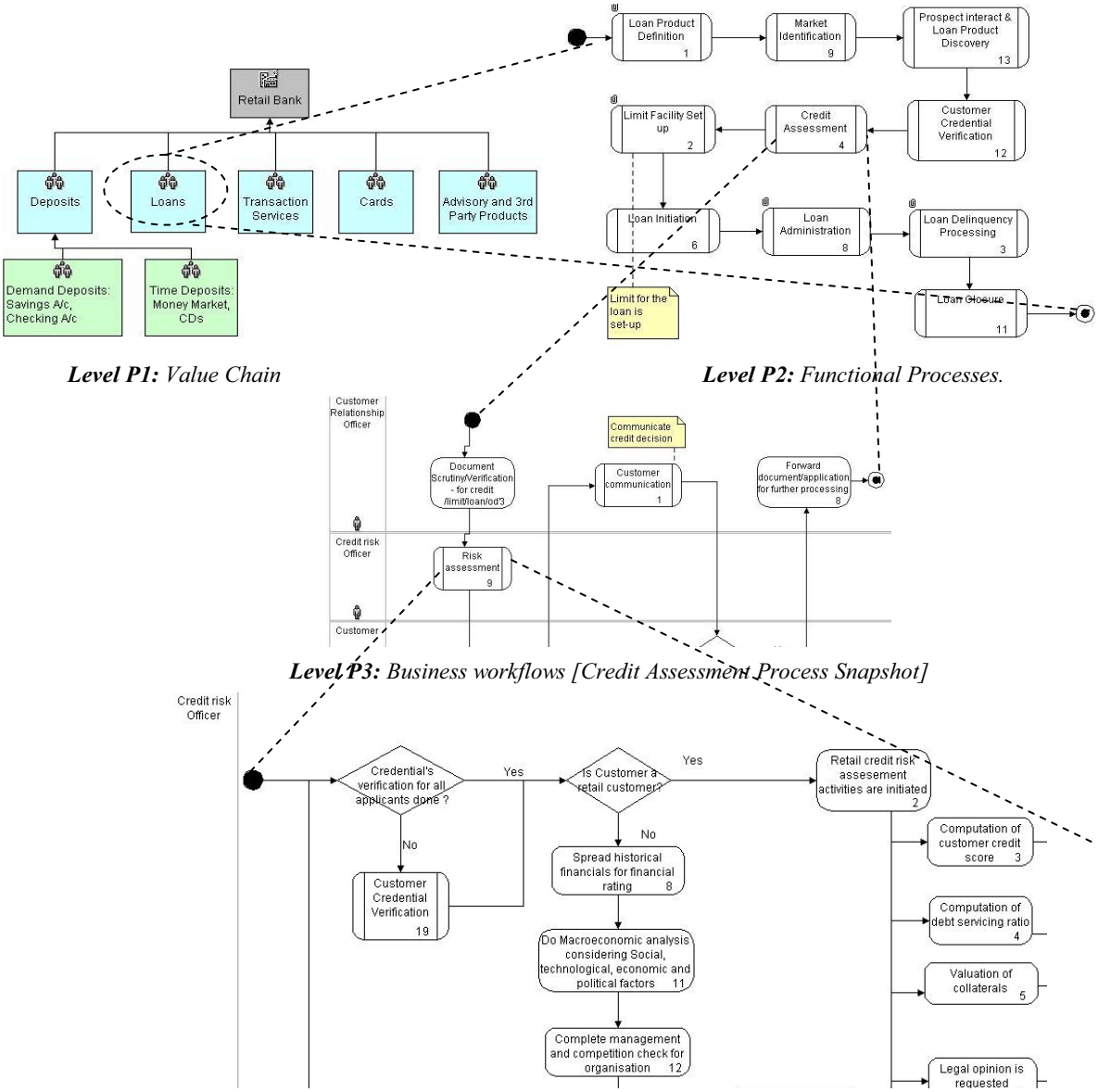


Figure1: Service Types in a multi-tiered process map (Illustrative)

3. Problem Statement

In this work we try to identify services with granularity varying from business service (high granularity) to data services (low granularity) from UML based hierarchical business process maps represented in XMI. Figure1 above illustrates the different types of services that could be envisioned in an order-to-cash scenario for the financial domain we use for our service identification exercises. We assume a multi-layered business process scenario (As shown in Figure2) as an input for our service identification. These UML models are then converted to MOF models which are utilized for our heuristic based service identification approaches.



Level P4: Task-flow: [*Risk Assessment Process Snapshot*]

Figure 2: An example cross section of a process map for Banking

4. Preliminaries

4.1. Defining the service hierarchy

The different levels of service granularities in realizing an SCSS are typically process service, business service, Composite Service, Informational service, Data Service, Utility Service, Infrastructure Service and Partner Service. We define these services in Table1. While a business service realizes the requirements of the business process where it participates, and is hence IT independent, a data service is the finest grained service which provides operation at implementation level. For an effective SOA realization it is important that the service portfolio takes into account the correct service granularity. Not only it helps in easier maintainability, but it also helps in effective governance of the service portfolios. Figure3 presents a service hierarchy meta-model which is used in some of the heuristic based conversions later in Section 6.

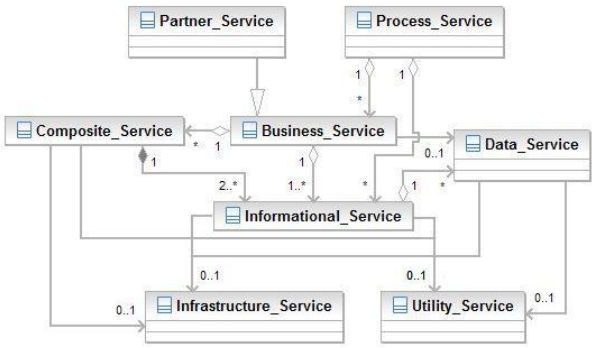


Figure3: Service Hierarchy meta-model

Service Type	Details
Process Service	A service whose operations are guided by the process definition. These are reactive services which need business events that would trigger various activities that would be using business and informational services.
Business Service	A service encapsulating transactional nature of functionalities that would build business context over other informational and data service.
Composite Service	A service with either composition or aggregation of multiple other services. The internal invocations are abstracted from the consumer providing a unified view. An orchestration would help composite service to be synchronous in nature and choreography would help composite service to be asynchronous.
Informational service	Services that focusing on providing processed data and whose operations are atomic, executed and realized by one provider on a particular type of runtime environment/platform.
Data Service	Services that provide normalized and aggregated view of critical data entities (or master data) such as Customer, Order, Claim and so on. These services are often realized along with Master Data Management strategies.
Utility Service	A service, whose operations are, shared among various services due to the commonly accepted practices or standardization such as payments, credit card transactions etc. Due to the utility or commodity nature of these services, business might often like to use the best possible provider may be from external sources too.
Infrastructure Service	A specialized technical automation service that provide essential infrastructural capabilities to other services
Partner Service	A manifestation of Business, Informational or Data Service offered to external business partners based on agreed terms.

Table1: Service hierarchy

4.2. Defining the process hierarchy

The business process architecture or process repository which stores the business process model is organized as a process hierarchy. We use the guidelines provided in Table 2. The workflow modeling guidelines used in the illustration are elaborated in [12] and [13].

Granularity level	Organizational level	Details
P 1	Value chain	• P1 is the highest level of the process architecture and usually corresponds to a function or Line of Business.
P2	Functional processes	• Each value chain at P1 is typically composed of many business processes. • Level P 2 may need to be broken into a series of processes and sub processes.
P3	Business workflows	• Each P2 level organizational process will have one or more P3 workflows to implement the process/sub process. • The workflows consist of a series of activities linked to each other in a sequential fashion. • P3 level follow "swim lane" method for process mapping.
P4	Tasks	• Each activity in P3 level will have a series of tasks in P4 level in it. • Tasks are the final level of granularity and correspond to step- by-step flow of events.

Table2: Process hierarchy

While most academic works on SOA stress on UDDI based registries, in practice the large scale industrial projects hardly use UDDI's; it has been partly due to their bulk and also the cost of their management. It has been an industrial practice to use governance frameworks which at minimum could comprise of portals for service management. In some of our work we came across scenarios with thousands of services. Such a service hierarchy is useful for these cases. Figure1 provides a good snapshot of such a scenario where in a financial domain the different activities correspond to services of different granularity. Zhuopeng Zhang et al in [14] provide an overview of service granularity optimization particularly in scenarios dealing with legacy migration. Most of services are typically derived from process models as they represent the requirements, business functions and workflows in a comprehensible format.

5. Identifying services from process repository

In this section we take a business process scenario from retail banking domain to elaborate the approach of

identifying services and their requirements. Figure2 depicts a simplified view of the P1 level containing the various lines of business within a retail banking organization. This level is not relevant to identification of services. The P2 level for the Loans function gives the functional flows for the Loan business function. Using the definition of business service given in Table1, we find that each block in the model qualifies for a business service.

Taking the above example we find that this business functionality comes from the direct need of the business and is not derived based on any software application. It is also reusable across other lines of business in the organization like checking account, loans, credit card etc, which can be identified by performing a similar analysis on those processes. Any of the function here can provide an autonomous business function and can be executed any number of times for different application forms taken as input. It is repeatable as it does not depend on the current or previous applications and is stateless as it does not have to retain the knowledge of the output of the previous instance of service execution. Going a layer lower, we look into the Layer P3 of business processes which implements the workflows and the task-flows. This provides the candidate services of finer granularity which are mostly system oriented.

6. The Squid Approach overview

The SQUID system utilizes XMI [15] based business-process meta-data which are rendered into persistent classes by using Novosoft UML XMI 1.0 Reader [16]. The central idea behind using business processes represented in XMI is to utilize the functional flow along with architectural elements which otherwise require manual intervention in other service identification approaches. In Figure4 we describe the SQUID approach for handling service identification using process maps.

There exists some work [11] towards transformation of UML process models into WSCI based model depicting choreography of exchanged messages. One of our experiment included using the BPDM as the meta-model for our BPMN based process models and then using the WSCI meta-model for model transformation. However, the theoretical model conversion approach has many practical limitations. Firstly, mapping the elements of the target and the source meta-model through QVT or OCL based transformation rules is a humongous task. Unlike the mapping between simple processes to its WSDL definition, mapping entire layered process architecture to service choreography is hence neither feasible; nor is it a very simple task. Our approach hence uses the statistics based approaches for service identification.

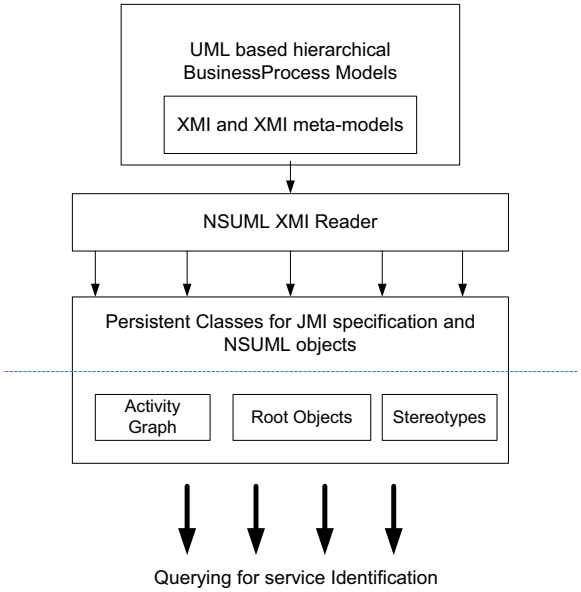


Figure 4: SQUID system approach

6.1. Handling UML based process maps

In **Figure2** we present the four layered cross section of the business processes for the retail banking. **Table2** explains the details of our multi layered process modeling. These UML based process maps are then converted to XMI. The use of XMI which is now an ISO standard (ISO/IEC 19503:2005) enables the representation of process meta-model in an interoperable manner.

6.1. 1. Role of XMI

XMI is the XML representational format for interoperability of the designed and developed UML models. It is built from a meta-model storage facility called MOF® (Meta Object Facility). The UML based business process models as in **Figure2** are converted to XMI and the resultant XMI 1.0 which is compliant with RTP UML 1.4 is fed into the XMI reader called NSUML XMI reader which is present as a plug-in in the code generator. This generates the MOF (Meta Object Facility) back and is used for meta-modeling and mapping new elements. **Figure5** details out the entire process of using XMIs and MOF Meta-model generation. One of the biggest reasons for using XMI was its cross-platform compatibility as most of the UML tools existing today are either based on XMI or atleast they can import and export XMI data.

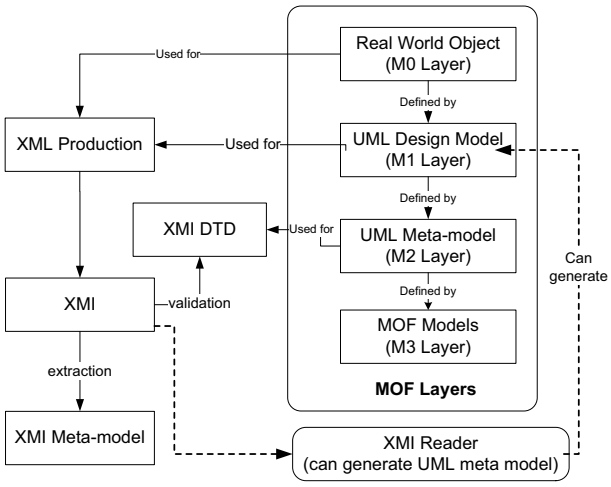


Figure5: The role of XMI

The usage of MOF provides an interoperable four-layered architecture and provides meta-models which can be exported from one platform to another. These serve as an intermediary between the design models for the process designers and the object models for the programmers. We use Novasoft's NSUML for the meta-data generation. A local main memory implementation through NSMDF provides the persistent classes which form the base for service identification algorithms and heuristics.

6.2. Heuristics for identification

Our heuristics for service identification are aided by the business process architecture which we use to a considerable extent. The structured modeling of business processes ensures a layered modeling, starting from functional processes at the top, to task-flows at the bottom. It helps us in identifying services ranging from low granularity (business services) to highest granularity (data services).

Some of the basic principles for service identification have been the following: i) Reusability ii) Lose coupling iii) Composability iv) Autonomous Nature etc.

We further make the following assumptions from the business process perspectives for service identification:

- **Layer2:** P2 Level provides only business services as it is of low granularity and contains business functions.
- **Layer3:** P3 Level and below provide potential candidate services which could be application, data or utility services.
- One of the key requirements for candidate service sharing amongst different process nodes is their having common work-items which is normally reflected by their technical properties in the XMI.

The other heuristics include:

- All system activities in the process model can be modeled as candidate services
- All interactive activities partly provide candidate services
- A process that has all system activities, then such process can be qualified as process service
- Activities related to the role outside the organization will be considered as candidate services regarded as 'Partner Service'
- All the consecutive activities (automated or interactive) within on swim lane can be merged to be a candidate service
- If the activities are common across different work products consider all the activities to be part of a candidate service
- Multiple consumers for the same service are realized by multiple interfaces.
- Partner services are realized by interfaces to an external role in the process swim lane. For convenience purposes most partner-link services are kept as a process service (unless they are pure data services with fixed data attributes)

Our service identification algorithms comprise of iterations over XMI meta-models to built statistical information regarding business process activities. The availability of persistent classes through NSUML helps us in providing APIs and constructs for effective querying based on the assumed heuristics. These heuristics provide us with possible candidate services whose technical properties are extracted from the XMI and used for service interface generation.

In Figure6 we present a heuristic based selection algorithm. This is based on the iterative querying of the process MOF model based on specific XMI attributes. We currently use an in-house tool called *Influx*[∇] which uses standard process modeling guidelines for generating process XMI models. SQUID utilizes the properties of the XMI attributes for service identification.

Most of the XMI attributes used for processing are derived from process modeling and are sensitive to case information. As of now our approach stands as a semi-automated approach for otherwise manual service identification exercise, providing a list of candidate services. Figure6 provides a representative algorithm which implements some of the heuristics.

[∇] An internal requirements and process management tool

```
Input: Activity Graph Database for individual Process Modules D={G1,
       G2, G3...Gn}, where Gi is the the Activity graph for each process
       module
Output: Set of activities comprising of candidate services CS={S1, S2,
        S3....Sm} with type as T1, T2, T3, T4
Algorithm:
MF = φ;
Current_Traversal_Level = 1;
while Current_Traversal_Level ≤ 4 do
    foreach Gi ∈ D do
        Traverse until level Current_Traversal_Level;
        Build Frequent ItemSet FI of process activities;
    end
    if (Current_Traversal_Level = 1) and (FI ≥ Layer2 − threshold)
    then
        Mark FI element as a business service;
        // It is P2 level activity which is repeated Add FI elment to SET
        CS;
        Mark the service type as Business-service with type = T1 ;
    end
    if (Current_Traversal_Level = 2) and (FI ≥ Layer3 − threshold)
    then
        foreach Activity Ai ∈ FI do
            if Ai satisfies reuse threshold condition then
                Add Ai to SET CS;
                Mark the service type as Application-service with type =
                T2 ;
            end
        end
        if (Current_Traversal_Level = 3) and (FI ≥
        Layer4 − threshold) then
            foreach Activity Ai ∈ FI do
                if Ai satisfies reuse threshold condition then
                    Add Ai to SET CS;
                    Mark the service type as Application-service with type
                    = T3 ;
                end
            end
        end
        // Checking for distance threshold
        foreach Activity Ai ∈ CS do
            Identify Activity set Sj and (Aj∈Sj) is
            (DistanceThreshold(Ai,Aj)≤User_Provided_Threshold) ;
            Mark the composite service Aij as a business service ;
            Add Aij to SET CS;
            Mark the service type as T1 ;
            Repeat Until Sj equals φ ;
        end
    end
    Increment Current_Traversal_Level by 1;
end
```

Figure6: The heuristic based selection algorithm

Here, ***Reuse Threshold*** defines the minimum number of times an activity is repeated so that it should be a service (or a part of service).

Distance Threshold defines the minimum distance between two activities satisfying reuse threshold condition so that they can participate in a business service

An additional criterion of work-item matching is used by looking at the XMI properties while matching two activities.

6.3. Deriving Hierarchical candidate services

Some of the services which we derived from processes described in **Figure2** are as follows:

Service	Type of service
Credit Assessment	*Business Service*
Customer Communication	*Business Service*
Customer Information Recovery Service	*Business Service*
Credit Risk Evaluation	*Composite Service*
Collateral Evaluation	*Composite Service*
Product Listing	*Composite Service*
Customer Verification	*Informational Service*
Retail Customer Assessment	*Informational Service*
Non Retail Customer Assessment	*Informational Service*
Product Description	*Data Service*
Restricted List Search	*Data Service*
Credit Score Computation	*Utility Service*
Dept Servicing Ration Computation	*Utility Service*

Although at this point of time the services identified are classified on the basis of their granularity, but it is still at an abstract level providing user a choice for selection. Service realization requires detailed technical properties about process activities which though can be added through Influx, it makes the XMI's too complex to handle. The identification process assumes the process modeling as per specifications in [13] and for now we use Influx for process modeling and XMI generation. Although the use of a standard meta-modeling language like XMI allows that the same can be used with other modeling tools like Rational etc.

7. Conclusions and Future Work

In this work we propose we detail out some of our initial work in the development of SQUID, an automated tool for service identification. The aim of the work is to support service discovery through UML based process modeling. We address two main challenges, namely a) the need to incorporate service discovery as part of process design b) the provision of an approach for service discovery based on various heuristics.

We consider some of the practical aspects of process modeling which involve the creation of UML based process maps. Our approach utilizes these process maps which capture the domain knowledge in the form of functional and organizational flows, and uses them as an input for SQUID. The use of XMI enables the representation of process meta-model in an interoperable

manner. One of the problems which we have been facing are the issues with XMI version compatibility (as from the initial version 1.0 to 2.1 there has been a drastic change in the XMI specs), but we hope to resolve them by utilizing XSLT transformations for XMI interchange (and currently there do exist some implementations for the same). Service realization and contract management is one more future area but it requires process models with detailed technical parameters which still remains one of the challenging tasks to implement.

8. References

1. Hausmann, J. H., Heckel, R. and Lohmann, M., "*Model-based Discovery of Web Services*", IEEE International Conference on Web Services (ICWS'04), USA, 2004.
2. Klein M. and Bernstein A. "*Toward High-Precision Service Retrieval*". IEEE Internet Computing, 30-36, January 2004.
3. Hoschek W. "*The Web Service Discovery Architecture*", IEEE/ACM Supercomputing Conf., Baltimore, USA, 2002.
4. Andrea Zisman, George Spanoudakis: *"UML-Based Service Discovery Framework"*. ICSOC 2006: 402-414
5. Ali Arsanjani, Abdul Allam: "*Service-Oriented Modeling and Architecture for Realization of an SOA*". IEEE SCC 2006: 521
6. Klose K, Knackstedt R, Beverungen D (2007) "*Identification of Services - A Stakeholder-Based Approach to SOA Development and its Application in the Area of Production Planning"*, In Proceedings of the Fifteenth European Conference on Information Systems (Österle H, Schelp J, Winter R eds.), 1802-1814
7. S. Inaganti and G. K. Behara, "*Service Identification: BPM and SOA Handshake*", BPTrends, March 2007, http://www.bptrends.com/
8. George Spanoudakis, Andrea Zisman, Alexander Kozlenkov: *"A Service Discovery Framework for Service Centric Systems"*. IEEE SCC 2005: 251-259
9. Kozlenkov A., Spanoudakis G., Zisman A., Fasoulas V., Sanchez Cid F.: *"Architecture-driven Service Discovery for Service Centric Systems"*, International Journal of Web Services Research, 4(2), pp. 81-112, 2007
10. UML 2.0 Profile for Software Services http://www.ibm.com/developerworks/rational/library/05/419_soa/
11. Gorka Benguria, Xabier Larrucea, Brian Elvesæter, Tor Neple, Anthony Beardsmore and Michael Friess *"A Platform Independent Model for Service Oriented Architectures"* Book on Enterprise Interoperability, Springer Pg 23-32
12. R. Sindhgatta1, S. Thonse, "Functional and Non-functional Requirements Specification for Enterprise Applications" 6th International Conference on Product Focused Software Process Improvement, 2005.
13. J. M. Bhat and N Deshmukh, "Methods for Modeling Flexibility in Business Process", BPMDS 05 Proceedings, June 2005
14. Zhuopeng Zhang, Ruimin Liu, Hongji Yang:, "Service Identification and Packaging in Service Oriented Reengineering" SEKE 2005: 620-625
15. OMG XMI 1.0 Formal Specification http://www.omg.org/cgi-bin/apps/doc?formal/00-06-01.pdf
16. Novosoft UML XMI 1.0 Reader http://NSUML.sourceforge.net

Appendix: A section of our Process meta-model