Business Web Service Development with ContextMaps

Qizhong Wen

A Thesis

in

The Department

of

Computer Science and Software Engineering

Presented in Partial Fulfillment of the Requirements

for the Degree of Master of Computer Science at

Concordia University

Montreal, Quebec, Canada

March 2005

Your file Votre référence
ISBN: 0-494-04458-6
Our file Notre référence
ISBN: 0-494-04458-6

Abstract

Business Web Service Development with ContextMaps

Qizhong Wen

Business processes can be specified in two ways: executable business processes models and business protocols. Although there are a number of factors that guide the development of models in Web Services, the goal of this thesis is to provide the notation to model both the internal business processes defined as executable business processes and the external business interactions defined in the business protocols. Moreover, this form of modeling is made to be easily understandable by both business participants and software developer participants so that it can be smoothly transferred to Business Process Executable Language (BPEL) to implement in a Web Service environment. By studying the current methods such as Business Process Modeling Notation (BPMN) and Unified Modeling Language (UML) for Web service modeling, I have developed a method to model the Web Services with ContextMaps which bridge the gap between technical users and business users, between the abstract external design and the detail internal design and also exhibits the relationship between Web Services Description Language (WSDL) and BPEL. Furthermore, the 4P (pattern-able, plug-able, process-able and perform-able) abilities of ContextMap notation can show the optimal solutions for Web Service. Finally, a model is given to ensure the consistency of system modeling between the main process and its sub processes, and also between different system domains.

Acknowledgements

I would like to express my special gratitude to all the people who gave me the great help during my master program study.

I would like to express my extreme gratitude to Prof. Wojciech M. Jaworski, Prof. Peter Grogono, and Prof. Gary Boyd. I have benefited greatly from their expertise vision and clarity of thought in my study. They gave me a lot of helpful suggestion and encouraged my interest in the information representation and Web Service fields. Without their guidance and advice, the present work would have not been possible. My sincere thanks are also extended to Halina Monkiewicz, the Graduate Program Advisor, for her collaboration support.

Last but not the least; I express deep veneration to my dearest wife, Xiaomin Dai and lovely daughter, Ya Wen for their love and support.

Table of Contents

List of Figures

List of Tables

Chapter 1 Introduction

1.1 Overview

Today, one of the toughest jobs for managers is catching up with the rapid changes in technology. Web services and service-oriented-architecture (SOA) are one of these major changes. There is an abundance of information contained not only within an organization data store, but also in the schemas and blueprints of the systems themselves. Analysts who must learn how a system operates may face an information overload problem. There are some useful products for mapping, modeling, designing and implementing Web Services. However, these products still lack the overall capability to meet such needs for complex Web Services included the service-oriented and object oriented architectures in one domain.

This thesis addresses open problems in Web Services modeling and introduces the Web Service architecture in order to show a powerful methodology, ContextMaps, which can answer the modeling questions raised from the other products such as UML, and BPMN and provides a flexibility developing environment. Through the customized ContextMap schema, we can model either services-driven flow or object-oriented flow without losing any information. Furthermore, it provides for the design of patterns, analysis, mining and executable activities of Web service. Through the functions of ContextMaps, analysts can integrate different systems into a single domain which shows how high-granularity details are related to their contexts by combining them within the same diagram—ContextMaps. Thus we can calculate the complexity of each node of flows, trace the business flow and data flow, provide the pattern of Web Service to guide their designs, use the mining function that lets analysts explore the different levels of contexts by their combinations of "degree of interest", run the executing function to retrieve the executable source codes and pass the source codes through the XML syntax check. By comparison with the most popular tools such as UML and BPMN, the advantage of ContextMaps is demonstrates here to be very great for Web Service development.

The Problem

Modeling Web Service requirements motivate this thesis: abstracting the internal business domain and external Web Service architectures. Therefore, a new modeling method is expected to map the language features of BPEL and WSDL and represent the relationship between them in the same domain "Web Service Application," which means that we need to provide an environment to model service oriented design, object oriented design and their relationships. Actually, the modeling tool BPMN is good at service-oriented modeling but weak in object-oriented modeling. Its target audience is business users. On the contrary, another well-used modeling tool UML is strong in object-oriented modeling, but powerless in service-oriented modeling because UML is directed to technical users. In fact, Web Service covers both service-oriented and object-oriented design. Therefore, none of existing tools easily supports Web Service design. The idea tool should be empowered with the capability to analysis the complexity of a design and to zoom the relationship of contexts by different views, which show the overview of business requirement, a detail action and data definition. Finally, some functions are required, especially for validating our modeling by ContextMaps.

These requirements could be divided into the following sub-functions.

Define ContextMap Schema: This schema should be pattern-able for Web Service application. Through this patternable ContextMap schema, a complex Web Service can be easily modeled, developed and implemented. Specifically, users' requirements can be represented in the notation graph. The whole workflow and detail sub flow become more easily traceable. And the relationship between WSDL and BPEL is easily presented in the same domain.

Query the ContextMaps: By querying the selected sets, the ContextMaps can explore the general and detail business process by view's "degree of interest", analyze the relationship between each system, calculate the complexity of nodes in the design, extract the constraints of business, provide business design pattern and implement process.

Validate the modeling by ContextMaps: Once the web applications are modeled with our patternable schema, the BPEL source code can be extracted. The WSDL and BPEL codes need to be checked for error free by XML syntax. Finally, a Web Service environment is set up to test the real case.

1.2 Contributions

This thesis presents a solution for modeling, designing, implementing and analyzing Web Services using ContextMaps. ContextMaps are capable of handling the

"4P's" which are essential to business Web Service development. The 4P's are process ability, plug ability, pattern ability, and performance. We will define a Web Service schema in ContextMaps to present the relationship between external business processes and internal processes, between service oriented models and object-oriented models, and between graph notation and text notation. At the same time, a solution is provided to keep the consistency of different domains and processes. For example, the relationship between WSDL and BPEL is that WSDL defines the network services, and then BPEL refers to the variable types, XML schema, and partnerlink types which are defined in WSDL. In fact, there are 15 activities defined in the BPEL, and each activity can include sub activities. My schema can represent this relationship. Next, following the above idea; an enhanced tool of ContextMaps is generated for extracting the BPEL code to show the ContextMaps ability of process and performance. Finally, to demonstrate the 4P abilities of ContextMaps, new functions are developed and a Web Service application is implemented. A real application is executed for testing. All the problems addressed in the section 1.2 are solved.

Chapter 2 State of the Art

2.1 Service-Oriented Architecture (SOA) and Web Service

A Web Service is a software application or component registered over the Internet using a vendor/platform/language-neutral data interchange format XML to invoke the service and provide the response applying a rigorously defined message exchange pattern, and producing an answer that is sufficiently well-defined to be processed by a software application. Service-Oriented architectures are all about connections among the service providers and service consumers. In order to avoid ambiguities at the interface between the internal and external service, XML is the core of basic technology in the service-oriented architecture. Service-oriented architectures have been used for long time, but they use an alternative model to the more traditionally tightly coupled object-oriented models that have emerged in the past decades. While SOA-based systems involve both tightly coupled and loosely coupled modeling, they do not exclude Object Oriented Models, individual services can themselves be built with object-oriented designs, and the overall design is service-oriented. Since it allows for objects within the system, SOA is object-based but it is not as a whole object-oriented. The difference lies in the boundaries themselves.

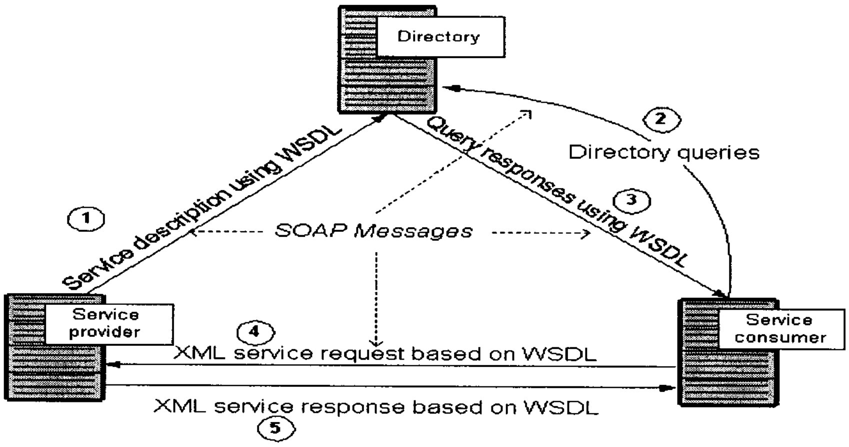

Directory
Service provider
Service consumer
1
Service description using WSDL
2
Directory queries
3
Query responses using WSDL
SOAP Messages
4
XML service request based on WSDL
XML service response based on WSDL
5

Figure 1-1 Web Services Basics [5]

1. A service provider is defined in the WSDL as a port. Its operation services are registered in the directory of services. The directory Universal Description, Discovery, and Integration (UDDI) directory can publish the provider's response over the web services.
2. A service consumer, which is defined in the WSDL as another port, sends one or more queries to the UDDI to locate a particular needed service which is provided by a service provider and determine how to communicate with that service provider.
3. Both a service provider and a service consumer know each other by the definitions of their locations and operations in the WSDL. Through the UDDI, part of the WSDL provided by the service provider is passed to the service consumer. This tells the service consumer what the requests and responses are needed for the service provider.
4. The service consumer sends a request to the service provider by using the WSDL.

5. The service provider provides the service consumer with the expected response by using the WSDL.

2.2 Introduction to Terminologies in Web Services

XML Schema provide shared vocabularies and allow machines to control the structure of XML documents. It expresses shared vocabularies to define the data type, structure content and semantics of XML documents.

Standard Business Process Modeling Notation (BPMN) provides a graphical notation that is easy to model the business process by all business users. At the same time, technical developers can easily transfer the processes to executable code according to these users' initial drafts. BPMN is similar to UML2 Activity Diagrams. However, BPMN and UML have very different approaches to business process modeling, which will be supported with an internal model using graphical representation that will enable the generation of executable BPEL. Thus, BPMN creates a standardized bridge for the gap between the business process design and process implementation in detail.

Business Process Execution Language (BPEL) defines specification of business process behavior based on Web Services. It's created on XML 1.1 specification. Business process can be described in two ways [2]:

a) By executable business processes which model the actual behavior of a participant in a internal business interaction.

b) By business protocols, in contrast, these use process descriptions that specify the mutually visible message exchange behavior of each of the parties involved in the protocol, without revealing their internal behavior. Hence, the process descriptions for business protocols are called abstract processes. BPEL is used to model the behavior of both executable and external abstract process.

Web Services Description Language (WSDL) use XML to describe the services of a network. WSDL define how to access Web services and what a service of operations will be performed. It treats Web services as messages which are transmitted between collections of network endpoints, or ports. The messages include the information of documents and processes. WSDL is one of the three foundation standards of web services.. In WSDL, a port is defined by linking to a network address with a reusable binding, and a collection of ports defines a service of operations. The port type which is an abstract collection of operations. The detailed protocol and data format defined for a particular port type constitutes a reusable binding [4]. the abstract definition of ports and messages is unattached with their network environments or data format bindings to support dynamic invocation. This allows the reuse of abstract common definitions:

Universal Description, Discovery, and Integration (UDDI) provides a Web-based distributed directory which defines a set of services to support the description and discovery of those Web Services providers, Web Services consumers, and the technical interfaces how to access those Web services. The idea is to "discover" organizations and the services that organizations offer, much like using a phone book or dialing information. The Web services are registered in the UDDI.

These services include elements of the description of data and Web services. These elements are called binding Templates which bind the information of called services called tModel which includes the service provider's name, detail information and pointer of their URLs. UDDI is based on the SOAP 1.1 protocols and it also supports HTTP, XML, XML Schema protocols. It's about the publishing and discovering a Web service. It defines an infrastructure for a Web Service-based software environment for both publicly available services and services only exposed internally within an SOA environment.

Simple Object Access Protocol (SOAP) is a lightweight protocol. It defines the structured information to allow the message communicating among different platforms. It is based on XML. The purpose of SOAP is that to unified such techniques as Component Object Model (COM), Common Object Request Broker Architecture (CORBA), Distributed COM (DCOM), and Remote Method Invocation (RMI) which are prone to weak to communicate in a Web environment. There are three components in SOAP which are SOAP envelope, SOAP header and SOAP body. SOAP use the remote procedure call (RPC) to communicate over the different operation systems and languages. WSDL as the endpoints is enveloped in SOAP. So the documents and services in WSDL are bound to the Web Services through SOAP. SOAP can bind the underlying communication protocols such as HTTP and SMTP.

Unified Modeling Language (UML) is a graphic notation which is used to model the concepts and widely used in object-oriented modeling. It helps the user to specify, visualize, and document models of software systems, including their structure and design, in a way that meets all of these requirements, UML2 is a new specification which enhanced to model active classes with complex behavior using state machines. There are two classifications of UML2 diagrams which are: Structural Modeling Diagrams (package, class or structure, object, composite, component, and deployment diagram) and Behavioral Modeling Diagrams (use case, activity, state machine, communication, sequence, timing, interaction overview diagram). Behavior Diagrams are often used for modeling business processes.

The Relationship between WSDL and BPEL, the WSDL file is not part of the BPEL specification but improves understanding of the language constructs. The WSDL file defines the content and structure of communication. The business partners and the flow of interaction are defined in the BPEL file. Messages, port types and service link types of the WSDL definitions are referenced.

2.3 Popular Tools for Modeling the Web Services

2.3.1 BPMN

BPMN stands for Business Process Modeling Notation. It is the new standard for modeling business processes and Web Service processes, as put forth by the Business Process Management Initiative (BPMI – www.BPMI.org). BPMN is a core enabler of a new initiative in the Enterprise Architecture world called Business Process Management (BPM). Business Process Management is concerned with managing change to improve business processes.

BPMN consists of one diagram – called the Business Process Diagram (BPD). The BPMN Business Process Diagram has been designed to be easy to use and understand by people, but also provides the ability to model complex business processes. It has also been designed specifically with Web Services in mind. BPMN is only one of three specifications that the BPMI has developed – the other two are a Business Process Modeling Language (BPML) and a Business Process Query Language (BPQL). All have been developed using a solid mathematical foundation, which enables a BPMN Business Process Diagram to map directly to BPML, in the same way that a physical data model maps directly to Data Definition Language (DDL).

There are competing standards for BPML, chief among them is the Business Process Execution Language for Web Services (BPEL4WS) created in a joint venture by BEA, IBM, Microsoft, and others. However, BPMI has created BPMN so that it maps readily to any business process execution language including BPEL.

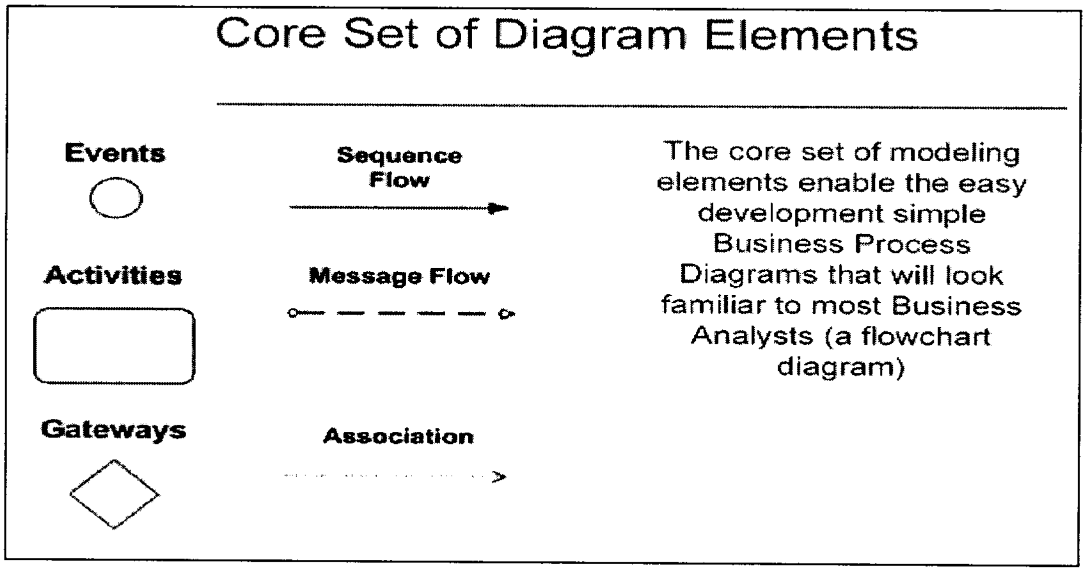
BPMN provides a number of advantages to modeling business processes over the Unified Modeling Language (UML). First, it offers a process flow modeling technique that is more conducive to the way business analysts model. Second, its solid mathematical foundation is expressly designed to map to business execution languages, whereas UML is not. BPMN can map to UML, and provide a solid business modeling front end to systems design with UML.

2.3.2 Model of BPMN

Inter-operation of business processes at the human level, rather than the software engineering level, can be solved with standardization of the Business Process

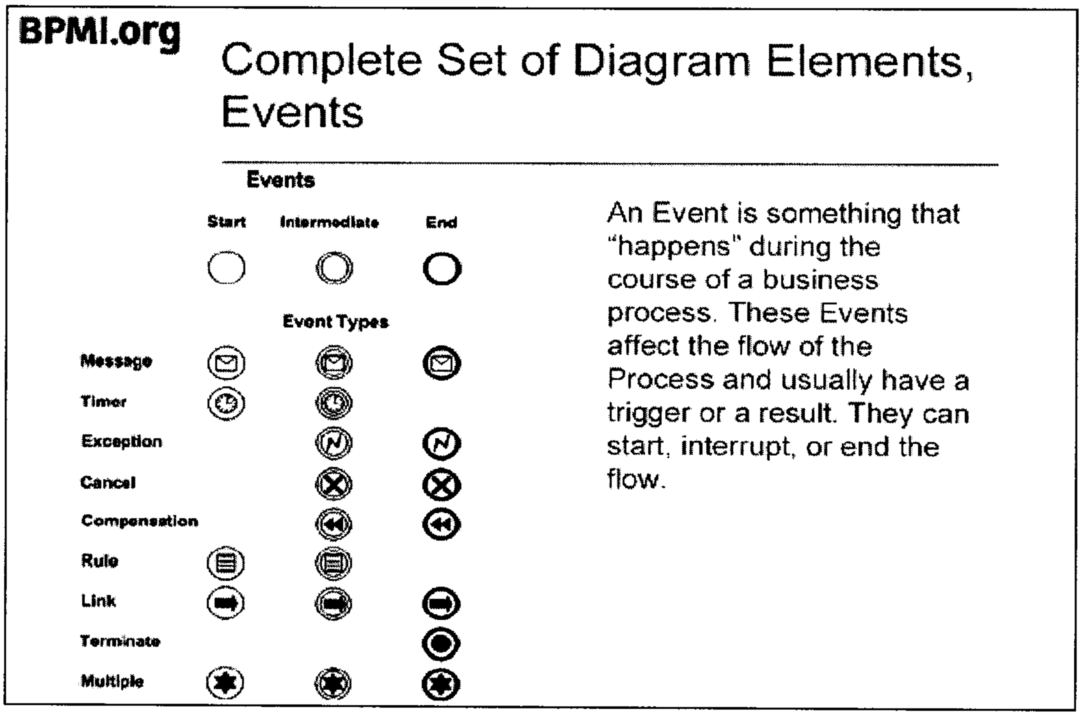
Modeling Notation (BPMN). BPMN provides a Business Process Diagram (BPD), which is a diagram designed for use by the people who design and manage business processes. BPMN also provides a formal mapping to an execution language of BPM Systems (BPEL4WS). Thus, BPMN would provide a standard visualization mechanism for business processes defined in an execution optimized business process language.

BPMN Notation



The core set of modeling elements enable the easy development simple Business Process Diagrams that will look familiar to most Business Analysts (a flowchart diagram)

Figure 2-1 Core Set of Diagram Elements [1]

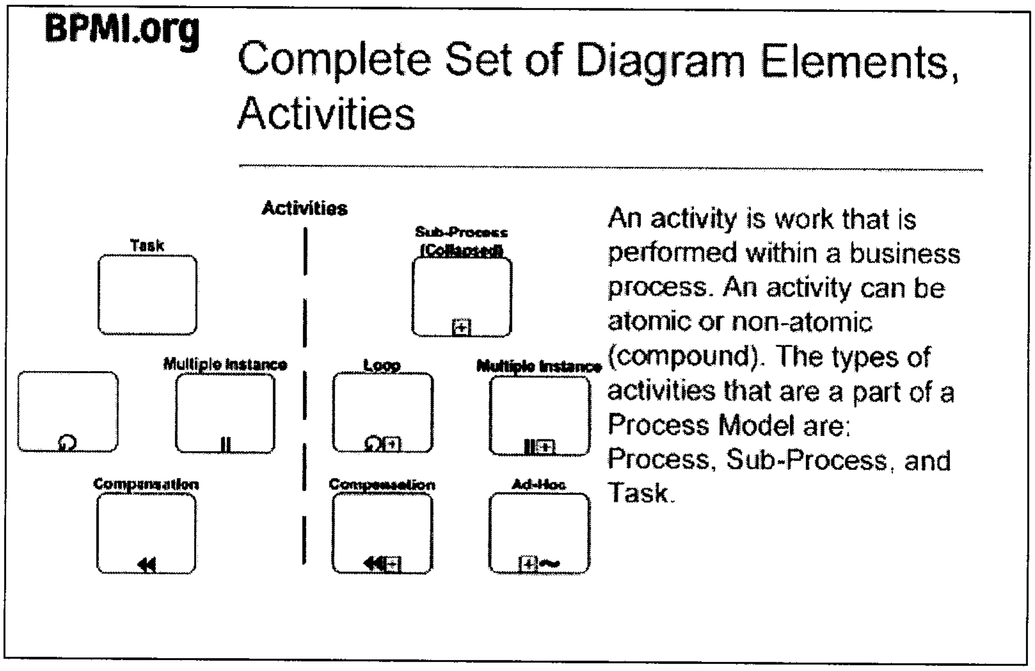


BPMI.org

Complete Set of Diagram Elements, Events

An Event is something that "happens" during the course of a business process. These Events affect the flow of the Process and usually have a trigger or a result. They can start, interrupt, or end the flow.

Figure 2-2 Complete Set of Diagram Elements, Events [1]

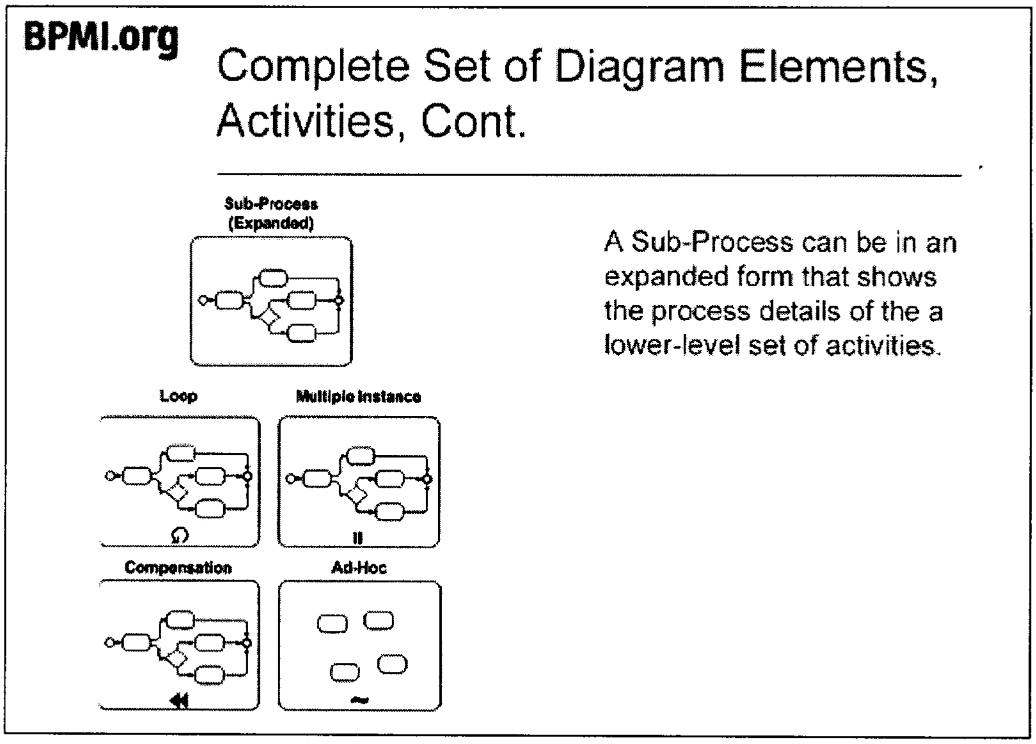


BPMI.org

Complete Set of Diagram Elements, Activities

An activity is work that is performed within a business process. An activity can be atomic or non-atomic (compound). The types of activities that are a part of a Process Model are: Process, Sub-Process, and Task.

Figure 2-3 Complete Set of Diagram Elements, Activities [1]

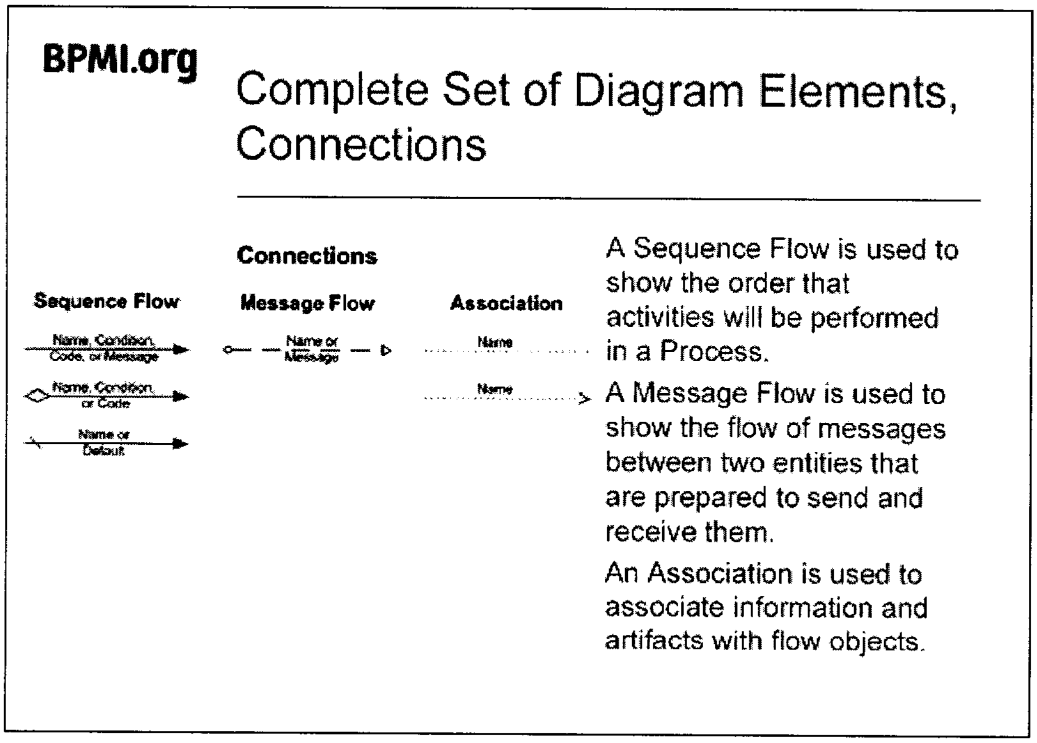


BPMI.org

Complete Set of Diagram Elements, Activities, Cont.

A Sub-Process can be in an expanded form that shows the process details of the a lower-level set of activities.

Figure 2-4 Complete Set of Diagram Elements, Activities, Cont. [1]



BPMI.org

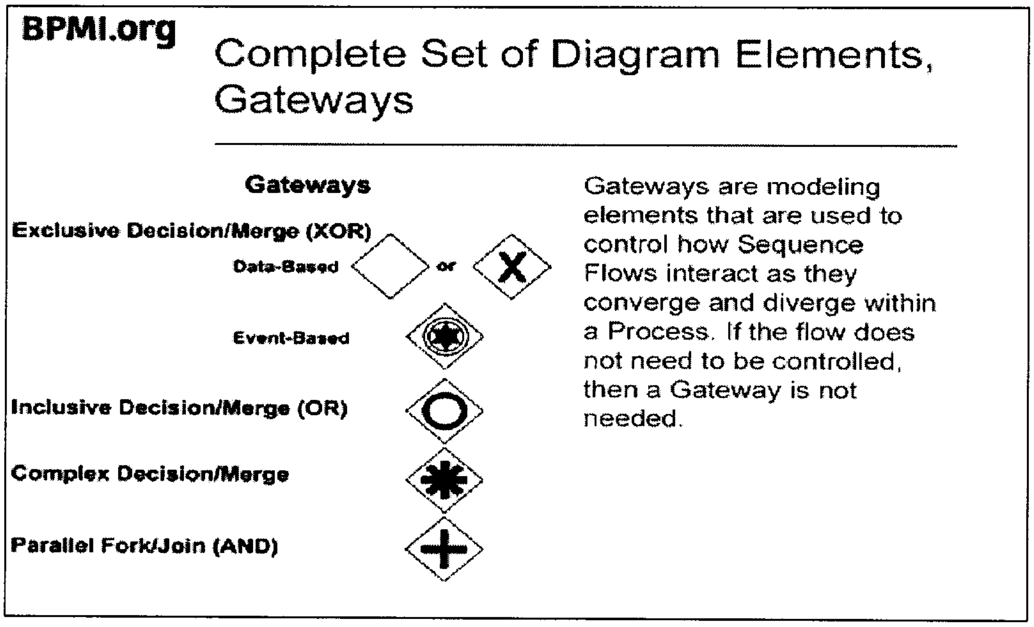
Complete Set of Diagram Elements, Connections

A Sequence Flow is used to show the order that activities will be performed in a Process.

A Message Flow is used to show the flow of messages between two entities that are prepared to send and receive them.

An Association is used to associate information and artifacts with flow objects.

Figure 2-5 Complete Set of Diagram Elements, Connections [1]

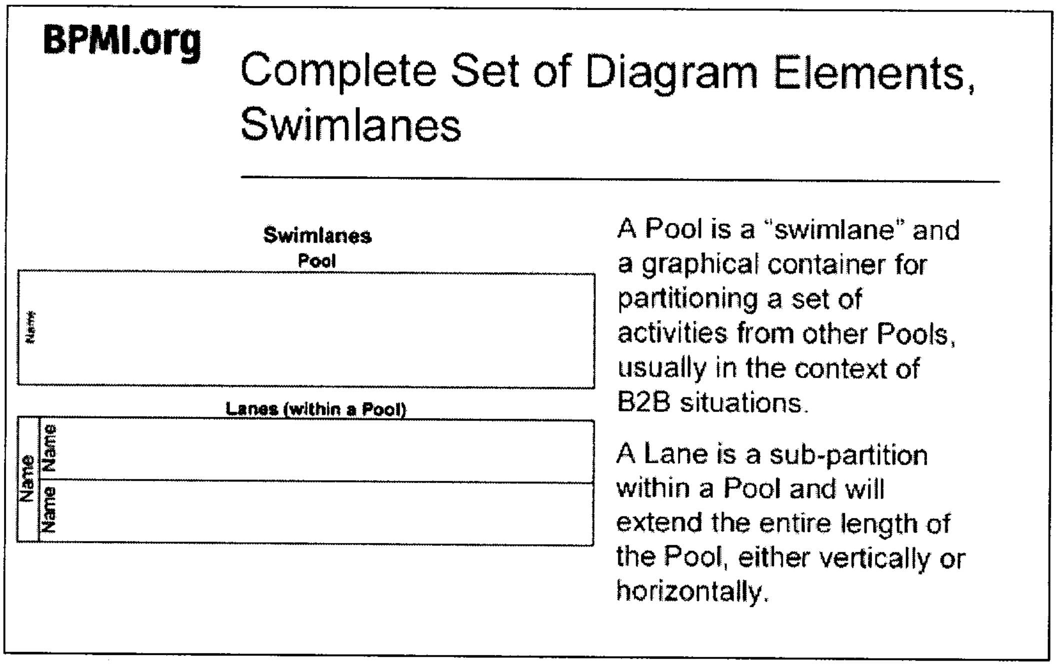


BPMI.org

Complete Set of Diagram Elements, Gateways

Gateways are modeling elements that are used to control how Sequence Flows interact as they converge and diverge within a Process. If the flow does not need to be controlled, then a Gateway is not needed.

Figure 2-6 Complete Set of Diagram Elements, Gateways [1]



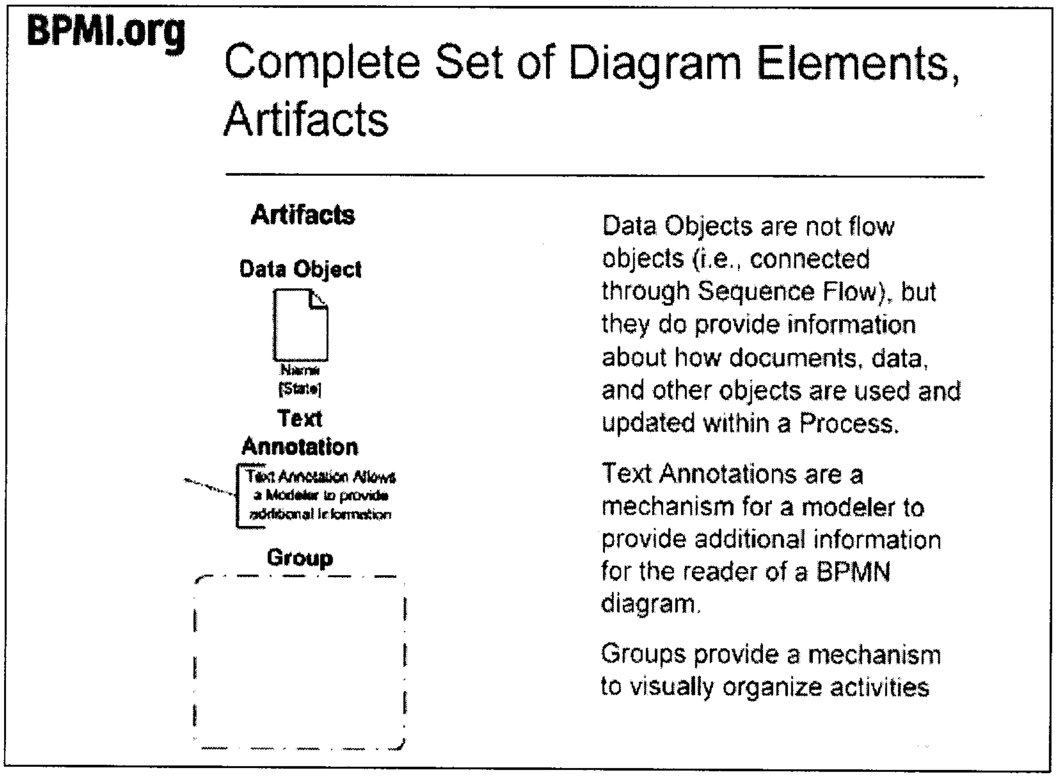
BPMI.org

Complete Set of Diagram Elements, Swimlanes

A Pool is a "swimlane" and a graphical container for partitioning a set of activities from other Pools, usually in the context of B2B situations.

A Lane is a sub-partition within a Pool and will extend the entire length of the Pool, either vertically or horizontally.

Figure 2-7 Complete Set of Diagram Elements, Swimlanes [1]



BPMI.org

Complete Set of Diagram Elements, Artifacts

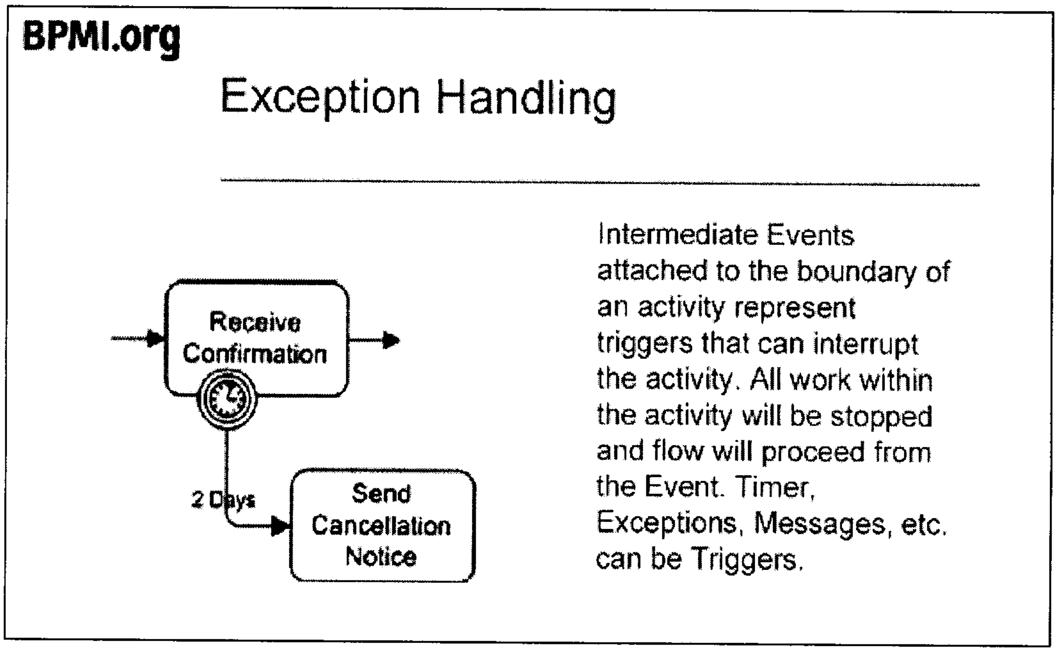
Artifacts
Data Object
Name
[State]
Text Annotation
Text Annotation Allows a Modeler to provide additional Information
Group

Data Objects are not flow objects (i.e., connected through Sequence Flow), but they do provide information about how documents, data, and other objects are used and updated within a Process.

Text Annotations are a mechanism for a modeler to provide additional information for the reader of a BPMN diagram.

Groups provide a mechanism to visually organize activities

Figure 2-8 Complete Set of Diagram Elements, Artifacts [1]

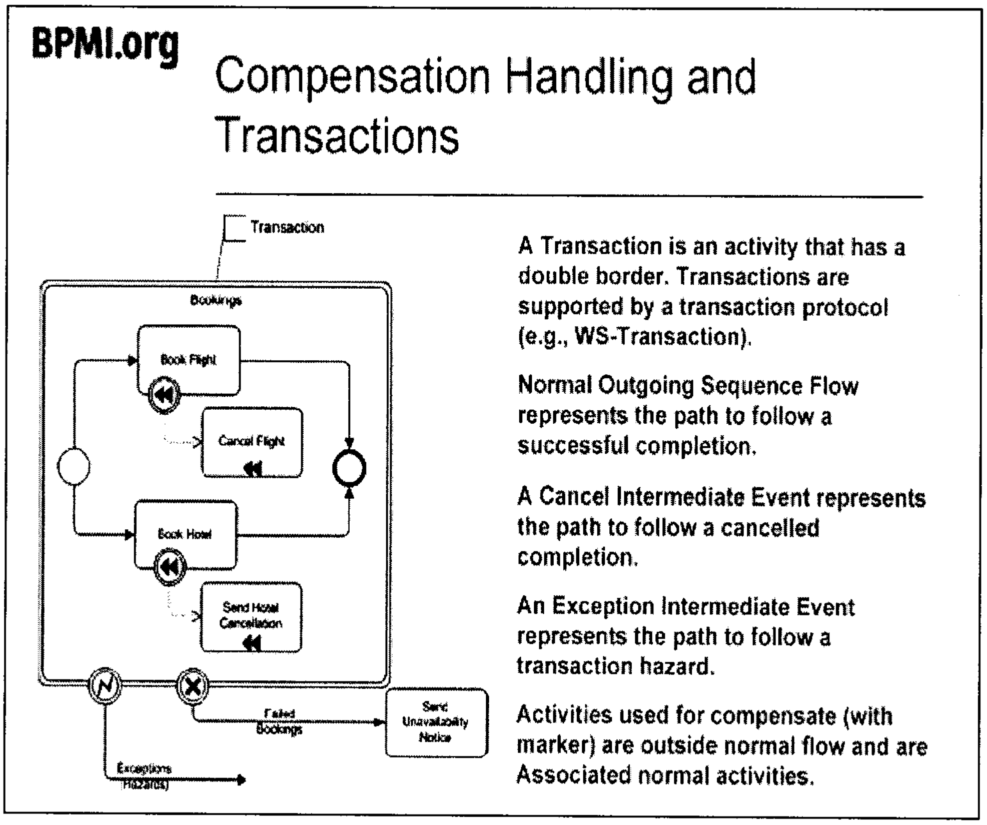


BPMI.org

Exception Handling

Intermediate Events attached to the boundary of an activity represent triggers that can interrupt the activity. All work within the activity will be stopped and flow will proceed from the Event. Timer, Exceptions, Messages, etc. can be Triggers.

Figure 2-9 Exception Handling [1]



BPMI.org

Compensation Handling and Transactions

A Transaction is an activity that has a double border. Transactions are supported by a transaction protocol (e.g., WS-Transaction).

Normal Outgoing Sequence Flow represents the path to follow a successful completion.

A Cancel Intermediate Event represents the path to follow a cancelled completion.

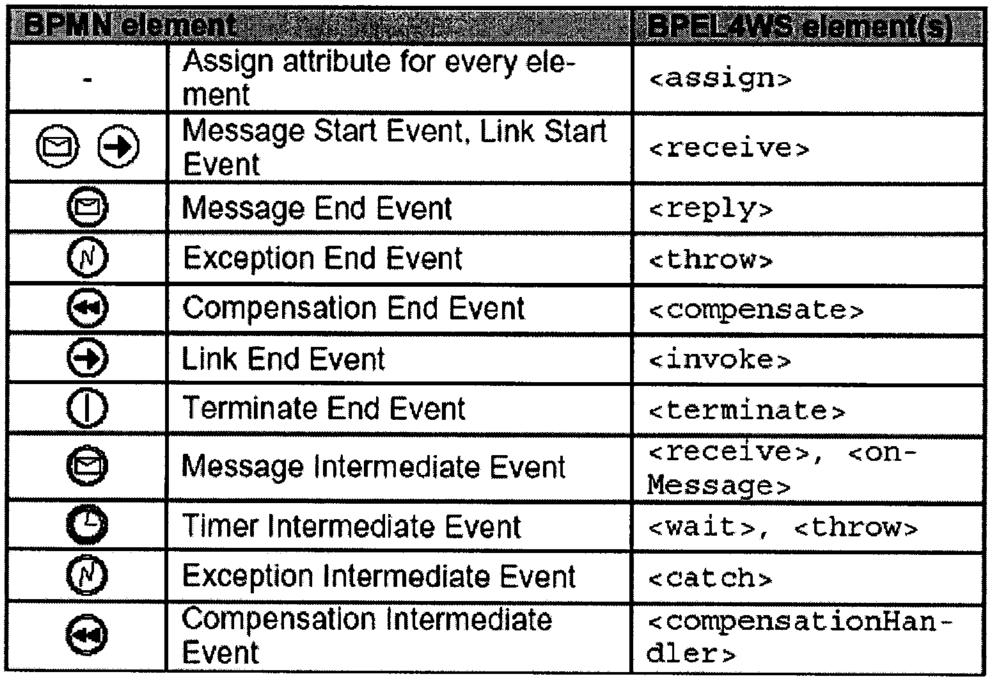
An Exception Intermediate Event represents the path to follow a transaction hazard.

Activities used for compensate (with marker) are outside normal flow and are Associated normal activities.

Figure 2-10 Compensation Handling and Transactions [1]

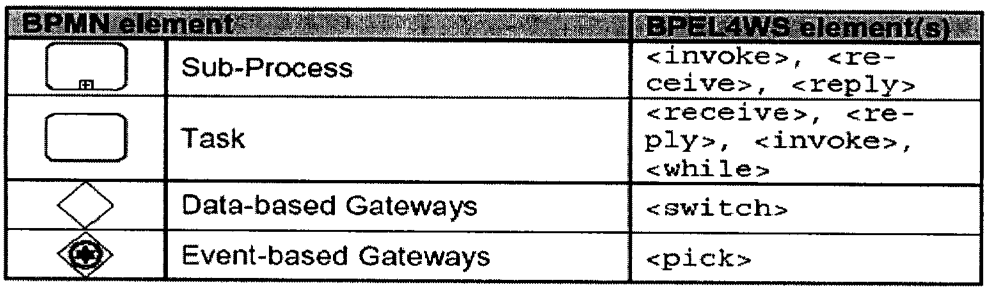
2.3.3 From BPMN to BPEL

According to the comments given by the BPMN.org, the following diagrams are supposed to be mapped to BPEL. The BPMN specification mentions the concept of "Token Analysis", using a token to traverse all possible sequence flows of the process. The details are yet an open issue.



BPMN element		BPEL4WS element(s)
-	Assign attribute for every element	<assign>
✉ ➔	Message Start Event, Link Start Event	<receive>
✉	Message End Event	<reply>
Ⓝ	Exception End Event	<throw>
⏪	Compensation End Event	<compensate>
➔	Link End Event	<invoke>
⦶	Terminate End Event	<terminate>
✉	Message Intermediate Event	<receive>, <onMessage>
🕒	Timer Intermediate Event	<wait>, <throw>
Ⓝ	Exception Intermediate Event	<catch>
⏪	Compensation Intermediate Event	<compensationHandler>

Table 2-1 BPMN to BPEL Mapping (1) [1]



BPMN element		BPEL4WS element(s)
⊞	Sub-Process	<invoke>, <receive>, <reply>
▭	Task	<receive>, <reply>, <invoke>, <while>
◇	Data-based Gateways	<switch>
◈	Event-based Gateways	<pick>

Table 2-2 BPMN to BPEL Mapping (2) [1]

2.3.4 A Sample of BPMN

This small sample is a part of the "Place Order Server Side" sub-process, containing activities, gateways and events. This sample is referred form [6].

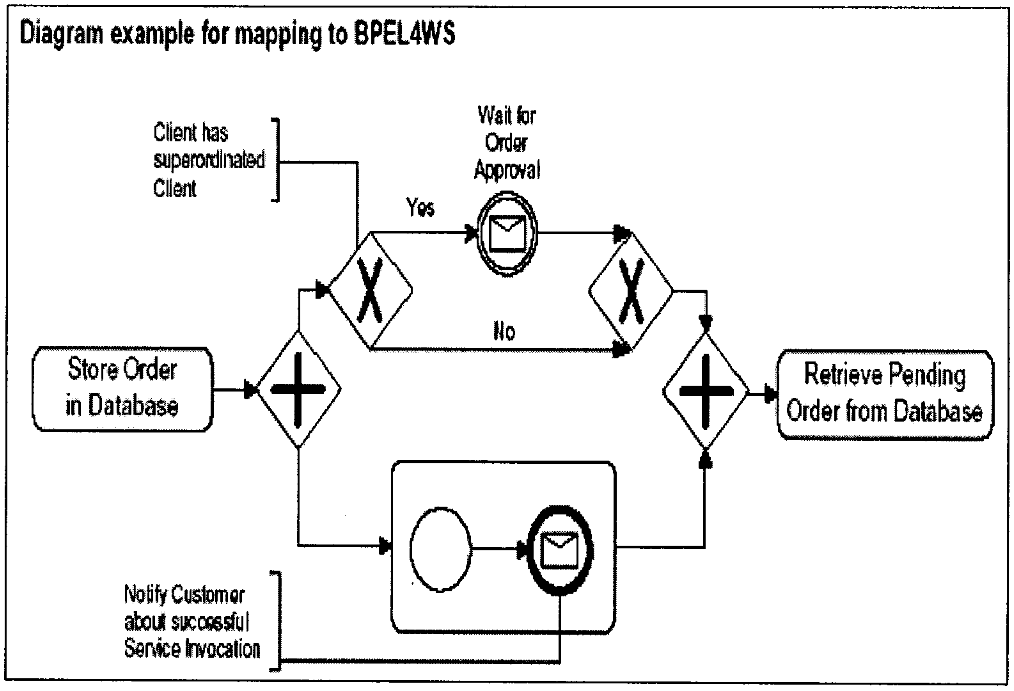
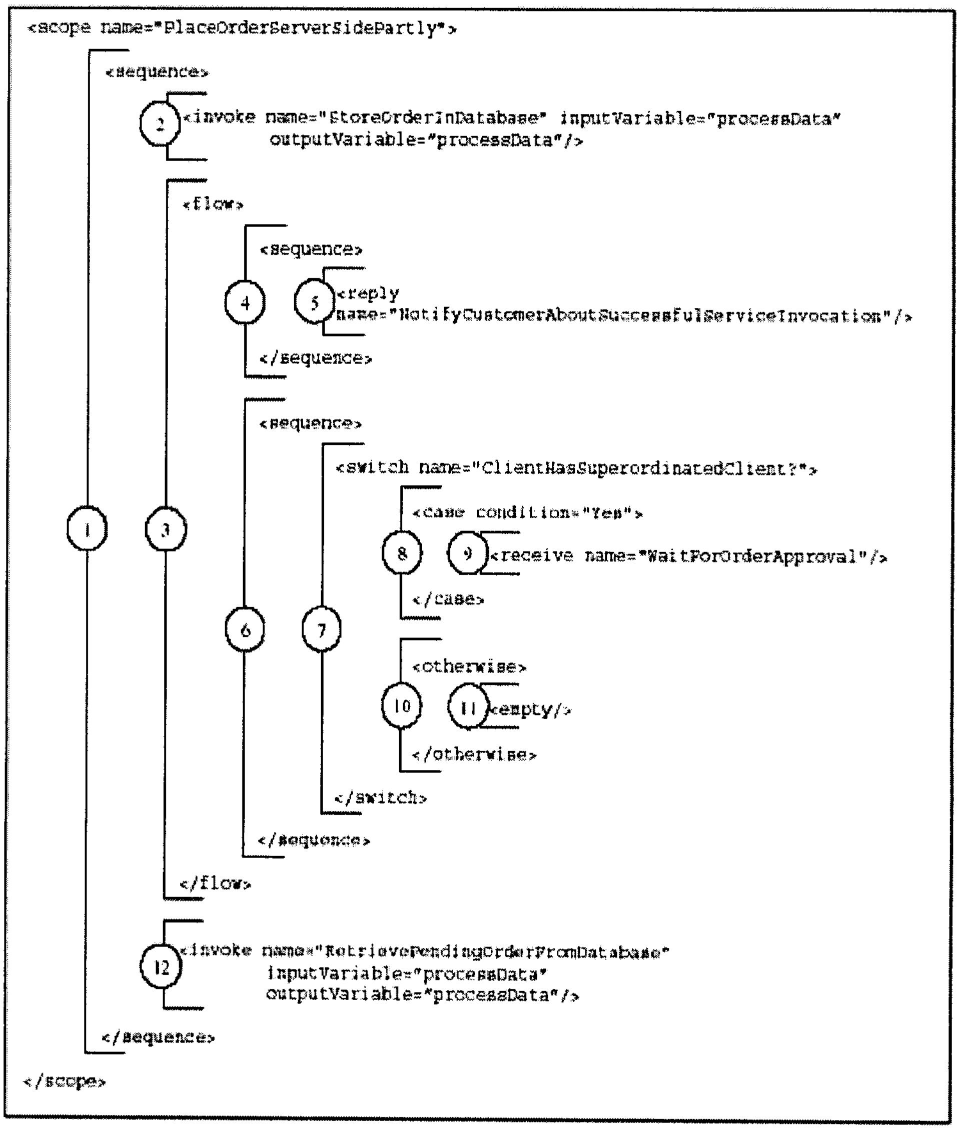


Diagram example for mapping to BPEL4WS
Client has superordinated Client
Wait for Order Approval
Yes
No
Store Order in Database
Retrieve Pending Order from Database
Notify Customer about successful Service Invocation

Figure 2-11 Example for Mapping to BPEL

Derived BPEL4WS source code



```
<scope name="PlaceOrderServerSidePartly">
    <sequence>
        <invoke name="StoreOrderInDatabase" inputVariable="processData"
                outputVariable="processData"/>
        <flow>
            <sequence>
                <reply
                name="NotifyCustomerAboutSuccessfulServiceInvocation"/>
            </sequence>
            <sequence>
                <switch name="ClientHasSuperordinatedClient?">
                    <case condition="Yes">
                        <receive name="WaitForOrderApproval"/>
                    </case>
                    <otherwise>
                        <empty/>
                    </otherwise>
                </switch>
            </sequence>
        </flow>
        <invoke name="RetrievePendingOrderFromDatabase"
                inputVariable="processData"
                outputVariable="processData"/>
    </sequence>
</scope>
```

Figure 2-12 BPEL Source Code Mapping [7]

2.3.5 BPMN Problem Domain Analysis

From the above BPMN to BPEL mapping, we can see that BPMN offers a process-centric approach which is more natural and intuitive for the business analyst to use.

With BPMN, control and message flows of processes are modeled first.

However, with the current BPMN notation, BPEL codes are not fully mapped because BPMN is an abstract business level mapping. Their activities are not been mapped including the detailed inner activities, object declarations, porttype, partnerlink, and partners. Second, without digging tools, the relationships between upper level and lower level processes are missing, so the consistency of system is hard to prove. Third, there is no mapping to the WSDL, so it can't represent the relationships between BPEL and WSDL.

2.4.1 UML Modeling

According to the current UML2 notations, The UML2 mapping for the Web Services can be decoupled into WSDL and BPEL mapping;

UML to WSDL Mapping Profile

<<Web Service>>	Service	The service is represented as a UML component that realizes one or more interfaces and resides at a particular location. The <<reside>>relationship will capture the actual URL location information.
Interface	PortType	Each portType is represented as a UML interface realized by one or service. The realization relationship will capture the binding information.
<<WebDocument>>	Message	Each message is represented as a UML class. A mapping from XML Schema to and from UML is required to model the message and part structure.
Attribute or Association End	Part	Each part of the message can be represented either as a association to another <<WebDocument>>.
Node	Address location	The node represents the server on which the service resides. The node may identify a set of resident services and a service resides on the more than one node.

Table 2-3 UML to WSDL Mapping Profile [2]

2.4.2 UML to BPEL Mapping Profile

<<process>> class	BPEL process definition
Activity graph on a <<Process>> class	BPEL activity hierarchy
<<port>> associations	BPEL partner declarations
<<process>> class attributes	BPEL variables
Hierarchical structure and Control flow	BPEL sequence and flow activities
Decision nodes	BPEL Switch activities and transition conditions
<<receive>>,<<reply>>, <<invoke>> activities	BPEL receive, reply, invoke activities
<<protocol>> package with <<role>> classes	BPEL service links types <<role>> classes

Table 2-4 UML to BPEL Mapping Profile [2]

2.4.3 Analysis of UML Problem Domains

The requirements of model driven Web Services development need the UML can model business processes, and detail internal design. Besides, as the business processes are over the Web environment, the relationships between internal and external processes need to be represented. Current UML2 still can't fulfill the needs for business Web services design because:

- **Most Business Analysts Are Unfamiliar to UML**

UML defines a number of diagrams that fall into one of three categories that describe:

1. Static application structure
2. Dynamic behavior
3. Management and organization of software solutions

Of these categories it is the dynamic behavior diagrams that are often used for modeling business processes, such as the UML Activity diagram and Use Case diagram. UML offers an object-oriented approach to the modeling of applications. Most UML methods ask you to find the objects first using static structure diagrams, and then ask the user to build dynamic behavior diagrams to show how objects interact. As a way to model, this method is alien to most business analysts.

- **There Isn't an Implementation View of Business Models in UML**

UML is an assemblage of diagrams that are the results of the collective best practices of the various founding practitioners. Unfortunately, what this means is that the diagrams are an aggregation that have not been specifically designed to work with each other. As a consequence, developers can only model part of their applications with UML; the detailed implementation level is not covered.

- **There Isn't the Mathematical Foundation to Map to BPEL's in UML**

UML does not define any execution meta-model for business processes modeled with it. Instead, any execution meta-model must be defined using the Model Driven Architecture (MDA) standard.

- **The Different Relationships of Domains Can't Be Expressed in UML**

Finally, the business Web services need to present the relationship between internal and external processes over different domains.

Chapter 3 A Solution using ContextMap Techniques

3.1 Introduction to ContextMap Techniques

A methodology to represent Web Services, which presents the business requirements and can guide the designer of detailed designs is urgently needed. ContextMaps provide such a method. Through the customized notations of ContextMaps, This technique can efficiently recover and model generic schemata for processes, objects and views in these systems.

ContextMaps were first introduced by Wojciech M. Jaworski, more useful information about ContextMaps including the evolution of ContextMap technique is given on the www.gen-strategies.com website. Historically, this technology was initially developed as a means of recovering and refining knowledge from legacy system. During the late 1970s and early 1980s, it was named as Array Based Language and was based on the conceptual graphs introduced by J. F. Sowa. In the late 1980s, it was renamed as ABL/W4 (W4 indicates the capability to show: “What”, “When”, “Where” and “Which”). In the early 1990s, by considering existing notations and methodologies, Professor Jaworski first introduced the concept of Context Maps and named this technique as ContextMaps, or Jointed Map. ContextMaps can represent knowledge assets in 4P-able format (process-able, plug-able, pattern-able, and performance-able). ContextMaps techniques usually use Microsoft Excel spreadsheets as an environment to present the modeled information. Through the “CONTEXT+” tool of ContextMaps, the mapped information can be retrieved, analyzed, processed and executed.

3.2 ContextMap Technique

On the Website http://www.gen-strategies.com/, built by Professor Wojciech M. Jaworski, ContextMaps are defined by an aggregation of context tuples. A Context tuple is a generic association of set members cast in roles. In the extended spreadsheet display a column of roles and the related set members define context tuple. In the graphical view, context tuple is represented by a compound edge and the connected compound nodes. A directed edge object consists of tail object, middle object and head object. While context tuples represent system behaviors, processes, tasks, procedures and programs, the aggregated context tuples form ContextMaps. Development of consistent structures and system views is facilitated by the existence of a library of generic and coherent templates schemata. The development of new templates allows for the building of multiple-models and interfaces to translate existing methodologies, notations and legacy systems. ContextMap representation is based on the fundamental notions of set-concept and schema-relation concepts. This provides ContextMap notation with multiple-modeling capability and power. ContextMaps introduce the concept of creating style sheets to control knowledge based information access and navigation. ContextMaps represents the relationship between different information nodes in a spreadsheet display by vertical columns. The tool of ContextMaps is called "CONTEXT+" which is based on the logical query of spreadsheet structure. The "CONTEXT+" queries the specific information that viewers expect to get.

3.3 ContextMap Syntax and Process

The syntax of Context Maps is based on the Relationship-Oriented paradigm, defined by relating Sets (concepts) and Set Members. In Context Maps, the relationships are represented by kTuples, which are exhibited regarded as vertical columns in the spreadsheet display. The kTuple consists of: Set, Set member and Role Tuples. The relevant mechanism is that each role is logically positioned and linked to a certain set member defined in context Tuples. From this, each set member can link its set, Concept. This construct is the fundamental structure defined by the concepts and instances related by roles.

Second, information is processed using the syntax of ContextMaps in an electronic spreadsheet. The original form of its information is modeled without any distortions. Moreover, process-ability is added to the information. Thus, after this process, a view of its information is easier to understand and edit. Compared to conventional diagrams, a view of the map can present more context in a limited computer screen.

In order to explain more clearly and in detail ContextMap notations, an example is used to demonstrate how to represent the "Web Service basics" (figure 1-1) and ContextMaps (figure 3-1). .

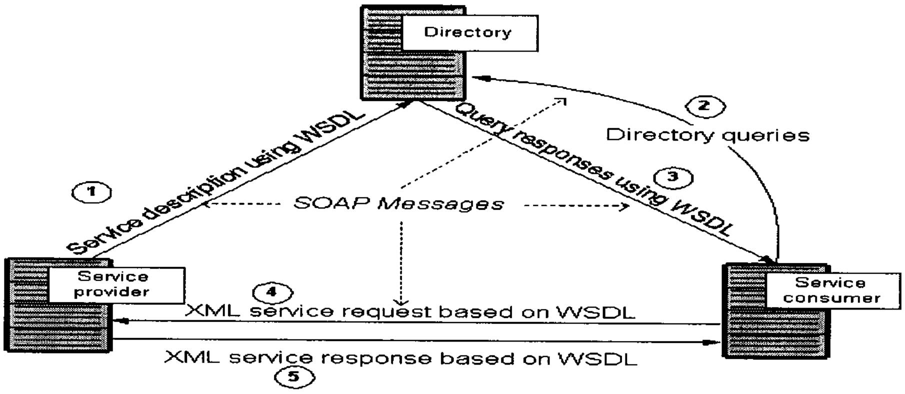

Directory
Service description using WSDL
Query responses using WSDL
Directory queries
SOAP Messages
Service provider
Service consumer
XML service request based on WSDL
XML service response based on WSDL
1
2
3
4
5

Figure 1-1 Web Services Basics

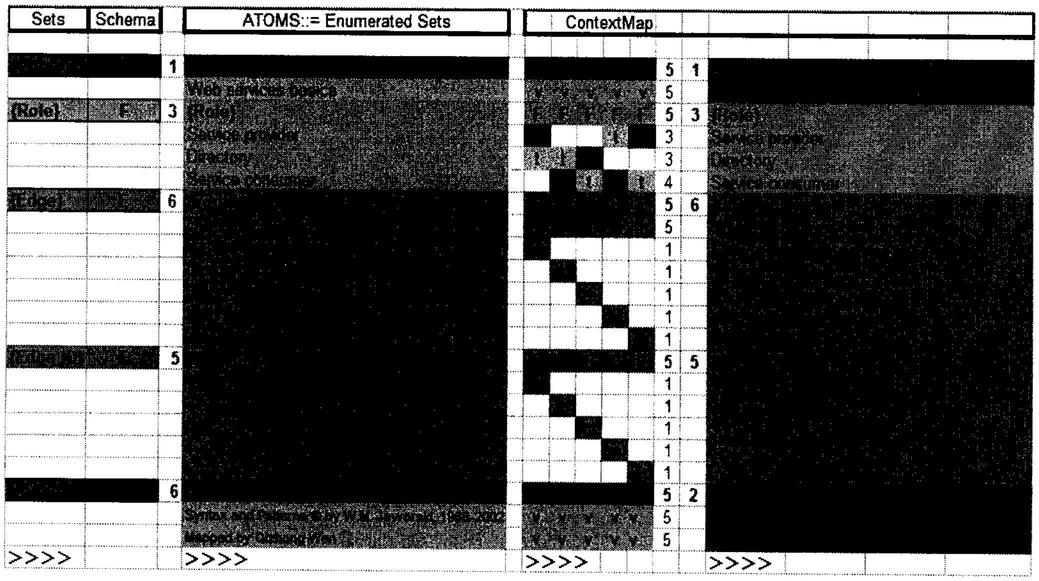


Figure 3-1 Web Services Basics Mapped in ContextMaps

1. We define these sets {Context, Role, Edge, Edge ID, Author} for the "Web Services basics", see the column "sets" in the figure 3-1.
2. We pick the correct notation based on the basic elements of ContextMap natations such as nodes, edges and flows. In this example, these schemata (N, E, F) are used. See the column "Schema" in the figure 3-1.
3. We expand each set by filling in its members. See the column "ATOMS:= Enumerated Sets" in figure 3-1.
4. We collect the above three steps and flows into ContextMaps. See the column

"Context" in figure 3-1.

In the above ContextMaps, figure 3-1, nodes can be represented as "Sets" or "Components". The following describes the syntax of the ContextMaps:

- The bold { } is called set, such as {Context} and {Role}
- The elements under the bold { } is called set member, such as "Service provider" and "Directory".
- The contents in columns under ContextMaps are called Context Tuples
- The single capital letters in column in ContextMaps are set roles, such as the letter N, F and E.
- The lower case letters or digits in column are member roles, such as f, t, m and v.
- The left column with numbers is the count of the member roles.
- The right column with numbers is the count of the set members.

All information is kept in the ContextMap repository. With the CONTEXT+ tool, each ContextMap can be driven into a specific view and be traced back to the general view.

3.4 ContextMap Notation

ContextMap notation can be applied for many fields such as:

1. Information system architecture
2. Automation of system design
3. Evolving information systems
4. Software evaluation and renewal

5. Systems workstations

6. Modeling of web sites and knowledge hubs

7. Recovery and reuse of system patterns

8. ContextMap notations are explained as follows:

As ContextMaps provides the flexibility to configure the syntax, the syntax of ContextMaps is expandable. Wojciech M. Jaworski defined the syntax.

1) The symbols of sets:

A-(A)ggregation of columns - context tuples
E - (E)dge properties
F - (F)low graph nodes
L - (L)flow graph with cycles
N - (N)ode properties
V - (V)alue
S - (S)equence
G - (G)uard
R - (R)esource
O - (O)bject
I - (I)dentifier
X - Cartesian Product
? – unknown

Table 3-1 ContextMap Notation (1)

2) The Symbols of set members:

v – marker
? – unknown
M - (m)iddle of 'arrow'
f – tail of 'arrow'

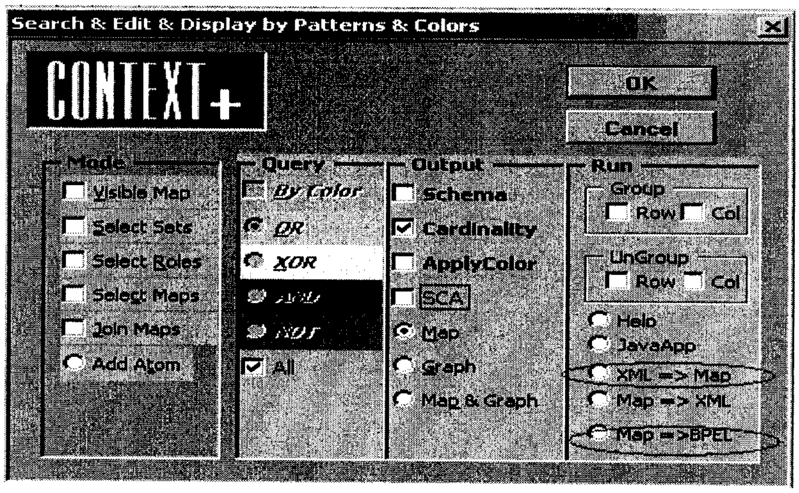
t - head of 'arrow'
b = both f/t
f - (f)rom node
t - (t)o node
l - (l)oop
b = f/t - both nodes component
f - (f)rom node component
t - (t)o node component
l - (l)oop node component
numerical value
integer value
y – yes
o – otherwise
r - (r)ead
u - (u)pdate
d - (d)elete
x - component of Cartesian Product
c – concurrence
j – join from fork

Table 3-2 ContextMap Notation (2)

3) Each notation has its own color property, so we can apply different colors to combine more meaningful syntax.

3.5 "CONTEXT+" Tools of ContextMaps

The "CONTEXT+" tool provides the capability to process and perform the ContextMaps. In order to retrieve the BPEL source from ContextMaps, we enhanced some functions to do that such as "Map=>BPEL", "XML<=>Map". Now, users can apply those functions showed in the following figure 3-2. It helps designers analyze, design a complex system. In the following charts, I will use most functions of this tool to show the modeling of Web Services.



Mode: □ Visible Map □ Select Sets □ Select Roles □ Select Maps □ Join Maps ○ Add Atom

Query: □ By Color ● OR ○ XOR ○ AND ○ NOT ☑ All

Output: □ Schema ☑ Cardinality □ ApplyColor □ SCA ● Map ○ Graph ○ Map & Graph

Run: Group □ Row □ Col; UnGroup □ Row □ Col; ○ Help ○ JavaApp ○ XML => Map ○ Map => XML ○ Map =>BPEL

Figure 3-2 The Schemata View of Map with Pattern

3.6 Main Functions of CONTEXT+ Tool

The powerful CONTEXT+ tool provide four divisions of functions (Mode, Query, Output, and Run). We can use the tool to:

View different mappings by focusing on "degree of interest" which allows users write a expression for a combined query.

Merger more mapping together for analyzing.

Apply specific colors for noticeable display.

Calculate the cardinality for analyzing the complexity of each set and its member.

Extract the schema to understand the basic design elements.

Generate the BPEL code.

View the XML file and to check the XML syntax.

Chapter 4 Implementation of a Web Service Application using ContextMaps

4.1 Design Rationale

The purpose of the Web Services is to achieve universal interoperability between applications by using web standards. Web services can integrate different domains together. On the other hand, Web Services specifications define such spaces as BPEL, WSDL, UDDI, XML schema and SOAP. These specifications allow applications to find each other and interact following a loosely coupled, platform-independent model. Our ContextMaps become a useful tool for Model-Driven Architecture (MDA) design.

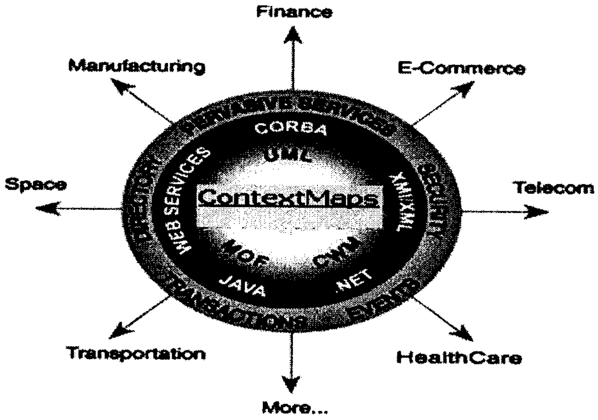


Figure 4-1 MDA Core with ContextMaps

ContextMaps can model those areas such as Finance, E-Commerce, Telecom, etc. and the schema of ContextMaps can map different languages and software platforms like JAVA, XML/XMI, and .NET. The notation of ContextMaps also can represent the UML and Meta-Object Facility (MOF). ContextMaps repository uses Microsoft Excel acting as a Common Warehouse Meta-model (CWM). The

CONTEXT+ tool transfers the MDA from the Platform Independent Models (PIM) to the Platform Specific Models (PSM).

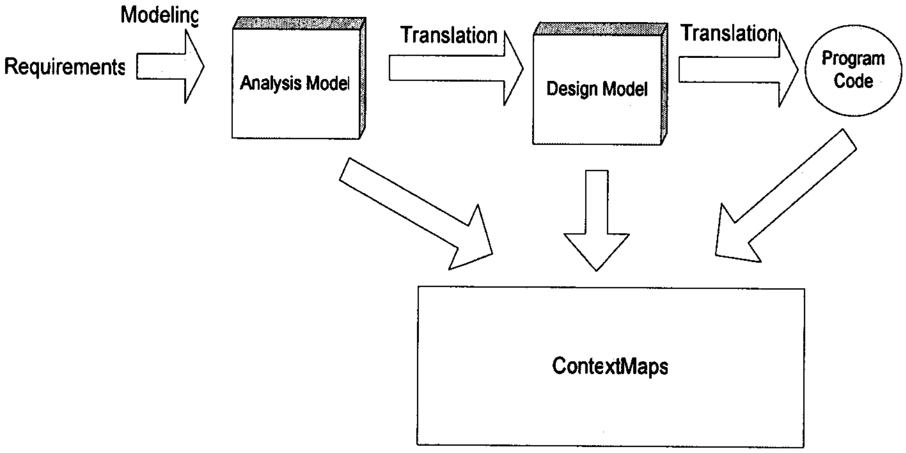


Figure 4-2 MDA with ContextMaps

In order to integrate the PIM and PSM, first, ContextMaps are able to map a logical model like PIM's modeling. Second, this PIM's modeling is then transformed into the ultimate PSM's modeling where the related codes are mapped in ContextMaps which are based on a set of rules defined as templates.

4.2 Features of MDA with ContextMap Technology

Here is a list of my contributions:

1. Consistency – use a ContextMaps as a repository to merge different domains or different Contexts into one domain which is platform independent.

2. Notation – customized our ContextMap notations to represent the features of WSDL, and BPEL.

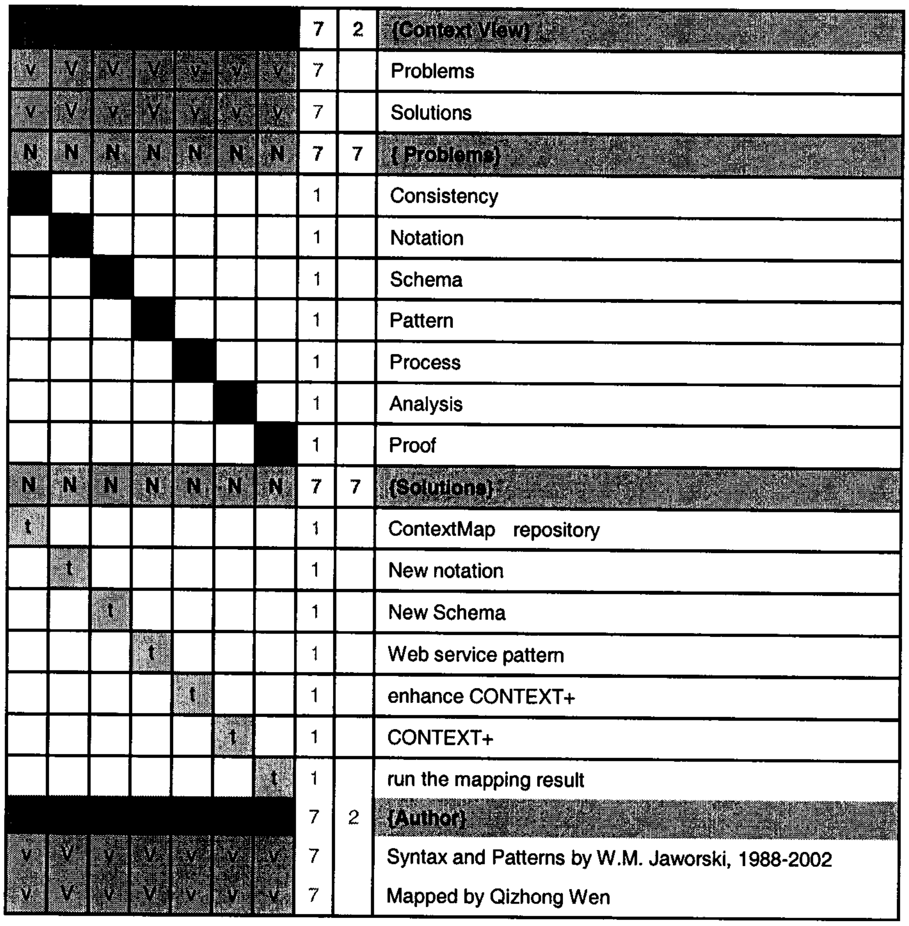
3. Schema – created new ContextMap schema to represent the relationship between WSDL and BPEL.

4. Pattern – provided a Web Service pattern to guild the design.

5. Process – programmed the functions: to generate the BPEL code, and check the syntax of XML in the CONTEXT+ tool.

6. Analysis – can analyze the complexity of a Web Service by calculating the number of the branches, zoom the "degree of interest" and trace the workflow and relations.

7. Proof – created a web environment to run the generated code.



■	■	■	■	■	■	■	7	2	{Context View}
V	V	V	V	V	V	V	7		Problems
V	V	V	V	V	V	V	7		Solutions
N	N	N	N	N	N	N	7	7	{ Problems}
■							1		Consistency
	■						1		Notation
		■					1		Schema
			■				1		Pattern
				■			1		Process
					■		1		Analysis
						■	1		Proof
N	N	N	N	N	N	N	7	7	{Solutions}
t							1		ContextMap repository
	t						1		New notation
		t					1		New Schema
			t				1		Web service pattern
				t			1		enhance CONTEXT+
					t		1		CONTEXT+
						t	1		run the mapping result
■	■	■	■	■	■	■	7	2	{Author}
V	V	V	V	V	V	V	7		Syntax and Patterns by W.M. Jaworski, 1988-2002
V	V	V	V	V	V	V	7		Mapped by Qizhong Wen

Table 4-1 Show the Problem Domain and Its Solutions

4.3 Web Services Example "Loan Flow Plus" Process with ContextMaps

In order to fully exhibit our ContextMap methodology for Web Service, we implement a real case of Web Services. The ContextMap technology is embedded in each step of Web Services implementations which experiences: requirement, analyze, design, and test phases. As each BPEL process defines a WSDL process, The ContextMaps will also map the relationship between BPEL and WSDL.

The requirement of "Loan Flow Plus" is described as following:

1. The Loan Flow Plus is initiated from a user Portal page (JSP) and orchestrates several Web Services into a business flow.
2. Integrated the JAVA/J2EE by EJB technology to check the authorized access and credit rating and with theses error handler.
3. The orchestration autumnally distributes the user's loan application to multiple trading partners. We use two banks, Bank of Montreal (BMO) and Royal Bank of Canada (RBC) as trading partners. The distribution process is concurrent process.
4. Initiate a manual task so that a customer can review the best loan offer and decide whether to accept the offer or not.
5. Provide customer interfaces to the flow via Portal integration - in this case implemented with JSPs.

A visual illustrating the flow requirements of the "LoanFlowPlus" process is shown in following figure 4-3.

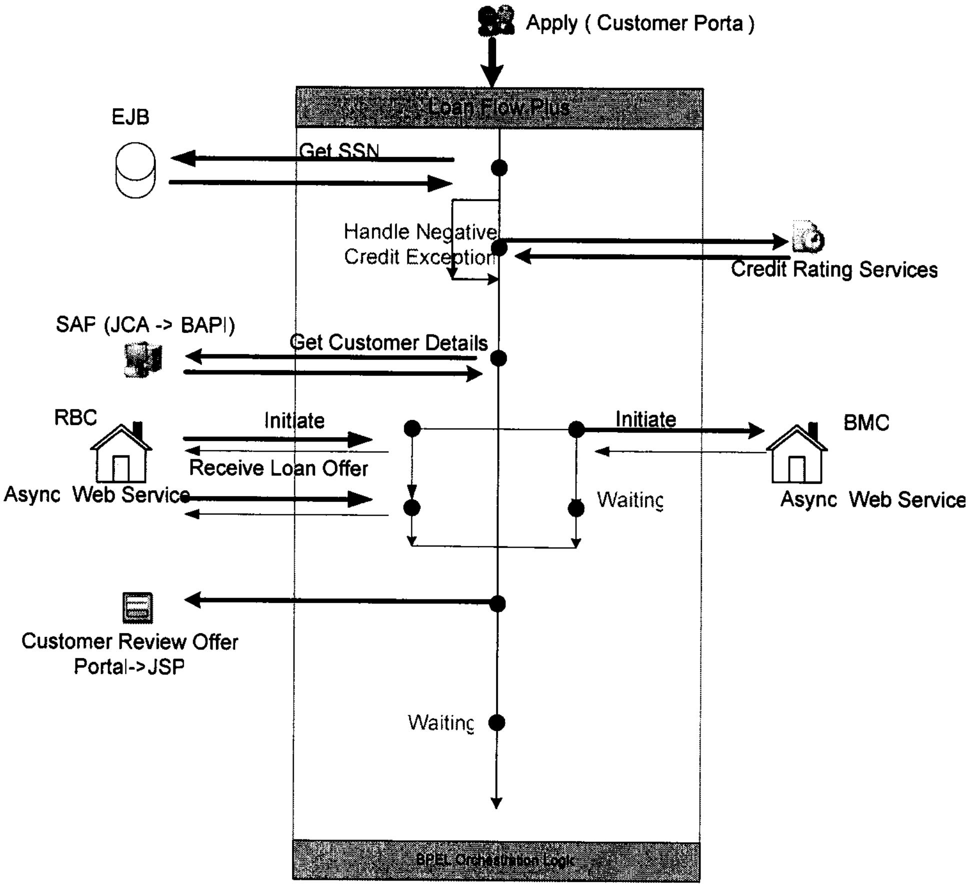


Figure 4-3 LoanFlowPlus Example Requirements

4.4 BPEL Specification and WSDL Specification

Because a development of BPEL process experiences the following seven steps as listed below, our BPEL design pattern refers these steps. However, BPEL defines 15 activities that are those <Receive>, <Reply>, <Invoke>, <Assign>, <Throw>, <Terminate>, <Wait>, <Empty>, <Sequence>, <Switch>, <While>, <Pick>, <Flow>, <Scope>, and <Compensate>. Each activity can include its sub activity so we have to consider the most important steps and activities which become parts of ContextMap sets and the template of Web Service design pattern.

1. Create a dictionary of messages

To use XML schema to define a message type in WSDL, BPEL can refer the defined massage type and then exchange the message. It's one of standard relationships between BPEL and WSDL.

2. Create empty skeletons

BPEL flow will at least have:

- An initial <process> tag
- A <partner> for the client who initiates the flow
- A <variable> for the input for the flow and usually a <sequence> for the main body of the flow

3. Define BPEL <Sync.> and <Async.> Process

<Receive> or <Invoke> activity can trigger a sync. process.

A asynchronous process needs <Receive> or <Invoke> activity to trigger and a <Reply> activity to finish this process.

4. Define a BPEL sequence and concurrent process

<Sequence>" activity indicates to start a sequence process.

<Flow> activity indicates to start a concurrent process.

5. Identify Partners

Each service participating in the flow needs to be declared as a partner. To do so we must define PartnerLinkType in WSDL. Then BPEL can refer this partnerLinkType to create PartnerLinks according to certain PartnerLinkType. It's one of relationships between BPEL and WSDL.

6. Create the PartnerLink in BPEL

   Define "my role" and "partner role" under the PartnerLink.

7. Declare variables

   Declare XML variables for messages that will be exchanged with the partnerLinks and messages that will be used to store temporary variables. The variables are defined in WSDL. It's one of relationships between BPEL and WSDL.

Based on the above our analysis of seven steps and characters of 15 activities, we generate following sets as BPEL design pattern as the following table 4-2.

BPEL Sets	Set Roles	BPEL Activities
{BPEL Process}	A	
{BPEL Activity Hierarchy}	E	
{Sequence and Flow Activities}	L	Activites{Sequence, Flow}
{Pre-Conditions: Wait, Switch, While, Pick, Scope }	G	Activites{Wait, Switch, While, Pick, Scope}
{BPEL Receive, Reply,Invoke}	N	Activites{Receive,Reply,Invoke}
{Post-condition: Throw,Switch,Terminate,Compensate}	P	Activities{Throw,Empty,Terminate,Compensate}
{BPEL Variable: Assign}	O	Activities {Assign}
{MyRoles}	N	
{PartnerRoles}	N	
{BPEL PartnerLinks }	F	
{BPEL Source}	S	

Table 4-2 BPEL Sets/Design Pattern

Next, we define the WSDL sets which are related with BPEL. Form the above seven steps of analysis, those sets are {WSDL service}, {WSDL message}, {PartnerLinkType}, and {WSDL port type}. See the following table 4-3.

WSDL Sets	Set Roles
{WSDL service}	A
{WSDL message}	O
{PartnerLinkType}	F
{WSDL port type}	O

Table 4-3 WSDL Sets

4.5 ContextMap Notation and Schemata for BPEL and WSDL

According to the above analysis, we merge the table 4-2 and table 4-3 to generate the design pattern for Web Service design in ContextMaps. The schemata of BPEL process by ContextMaps are illustrated as the following table 4-4.

1	2	3	4	5	6
■	■	■	21	4	**{Context}**
		■	11	1	**{BPEL Process}**
■			4	1	**{WSDL Service}**
		X	11	1	**{Edge Type}**
		E	11	1	**{Edge Component}**
		P	11	12	**{BPEL Activity Hierarchy}**
		■	11	14	**{Sequence and Flow Activities}**
		■	11	0	**{Pre-Conditions: Wait, Switch, While, Pick, Scope }**
		N	11	11	**{BPEL Receive, Reply, Invoke Activities}**
		P	11	2	**{Post-condition: Throw,Empty,Terminate,Compensate}**
		O	11	2	**{WSDL Message}**
		O	11	15	**{BPEL Variable: Assign}**
F	O		4	4	**{PartnerLinkType}**
	N	N	17	3	**{MyRoles}**
	N	N	17	4	**{PartnerRoles}**
N	F	F	21	6	**{BPEL PartnerLinks }**
O			4	2	**{WSDL Port type}**
		S	11	356	**{BPEL Source}**
■	■	■	21	3	**{Content Source}**
■	■	■	21	2	**{Author}**
V	v	V			Syntax and Pattern by W.M. Jaworski, 1988-2002
V	v	V			Mapped by Qizhong Wen

Table 4-4 The Schemata of BPEL Process and Relationship with WSDL

The notations of sets and set members used here are described in chapter 3.4.

4.6 Demonstration of "LoanFlowPlus" Modeling in ContextMaps

We follow the table 4-2 and table 4-3 to fill the each BPEL member and WSDL member individually. See the following tables 4-5 and 4-6. After that, we use table 4-4 as a pattern to collect these fillings in our ContextMaps(See the tables 4-7,8,9,10). Because the mapping of "LoanFlowPlus" is much larger than to be fitted in this paper size, we can use the "query" of CONTEXT+ to trace the relationship in our mapping.

BPEL Sets	ContextMap Notation	ATOMS::=Enumerated Sets
{BPEL Process}	A	{BPEL Process}
		Loan Flow Plus
{BPEL Activity Hierarchy}	E	{BPEL Activity Hierarchy}
		Receive the input from requestor
		GetSSN
		GetCreditRating
		...
{Sequence and Flow Activities}	L	{Sequence and Flow Activities}
		Receive input from requestor/S1
		EJB to retrive SSN
		GetCreditRatingService
		...
{Pre-Conditions}	G	{Pre-Conditions}
{BPEL Receive, Reply,Invoke}	N	{BPEL Receive, Reply,Invoke}
		receive name="receiveInput"
		invoke name="initiateTask"
		receive name="receiveTaskResult"
		...
{Post-condition}	P	{Post-condition}
		Valid Client check failure
		Credit check failure
{BPEL Variable}	O	{BPEL Variable}
		input/Global
		loanApplication/Global
		...
{MyRoles}	N	{MyRoles}
		LoanFlowPlusRequestor
		LoanServiceRequester
		TaskManagerRequester
{PartnerRoles}	N	{PartnerRoles}
		LoanFlowPlusProvider
		CreditRatingServiceProvider
		...
{BPEL PartnerLinks }	F	{BPEL PartnerLinks }
		Client
		CreditRatingService
		...
{BPEL Source}	S	{BPEL Source}
		<process name="LoanFlowPlus" targetNamespace="http://samples.cxdn.com" ...

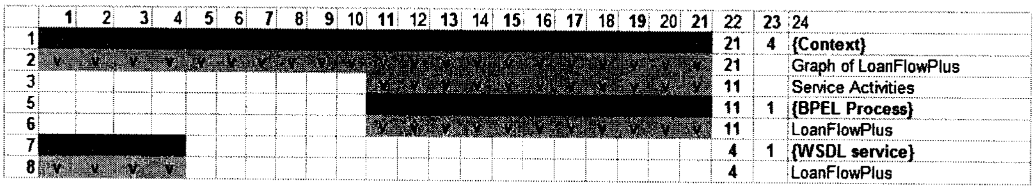
Table 4-5 BPEL Sets and Members

WSDL Sets	ContextMap Notation	ATOMS::=Enumerated Sets
{WSDL service}	A	{WSDL service}
		LoanFlowPlus
{WSDL message}	O	{WSDL message}
		LoanFlowPlusRequesterMessage
		LoanFlowPlusResultMessage
{PartnerLinkType}	F	{PartnerLinkType}
		LoanFlowPlus
		CreditRatingService
		LoanService
		TaskManager
{WSDL port type}	O	{WSDL port type}
		LoanFlowPlus
		LoanFlowPlusCallBack

Table 4-6 WSDL Sets and Members

4.7 Explanations of the Result in ContextMaps

The example are mapped in ContextMaps are shown in the following table 4-7,8,9.



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	21	4	**{Context}**
2	v	v	v	v	v	v	v	v	v	v	v	v	v	v	v	v	v	v	v	v	v	21		Graph of LoanFlowPlus
3											v	v	v	v	v	v	v	v	v	v	v	11		Service Activities
5											■	■	■	■	■	■	■	■	■	■	■	11	1	**{BPEL Process}**
6											v	v	v	v	v	v	v	v	v	v	v	11		LoanFlowPlus
7	■	■	■	■																		4	1	**{WSDL service}**
8	v	v	v	v																		4		LoanFlowPlus

Table 4-7 BPEL and WSDL Mapping in ContextMaps (1)

Sets, {Context}, are the top-level structure of the "LoanFlowPlus". We use notation "A" to indicate this set is a template aggregation. There are two set members "Graph of LoanFlowPlus" and "Service Activities" under the {Context}. We put the notation "v" to indicate these two sets are values of the {Context}. The member "Graph of LoanFlowPlus" contains another two sets {WSDL service} and {BPEL process}, The member "Service Activities" defines the BPEL processes. Each set {WSDL service} and {BPEL process} has a member "LoanFlowPlus". Therefore, from the top level mapping, two domains WSDL and BPEL are mapped in one repository, ContextMaps.

According to the business flow, see the figure 4-3, we map the BPEL flow in the following table 4-8.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	■	■	■	■	■	■	■	■	■	■	■	11	12	**{BPEL activity hierarchy}**
2	■											1		Receive the input from requestor
3		■										1		GetSSN
4			■									1		GetCreditRating
5				■								1		Handle negative
6					■							1		GetCustomerDetail
7						■						1		Bank1_Initiate
8							■					1		Bank2_Initiate
9								■				1		ReceiveLoanOffer1(Bank1)
10								■				1		ReceiveLoanOffer2(Bank2)
11									■			1		Initiate Task (Best Offer)
12										■		1		WaitCustomerReply
13											■	1		Receive the confirmation from the customer
14	■	■	■	■	■	■	■	■	■	■	■	11	14	**{Sequence and flow activities}**
15	■	e	e									3		Receive input from requestor/S1
16	t	■										2		EJB to retrieve SSN
17		t	■									2		GetCreditRatingService
18			t	■								2		Handle negative

19				t	■							2		SAP GetCustomerDetails
20					t	■	■					3		Initiate-Flow
21						■						1		Call Bank1
22							■					1		Call Bank2
23						t		j				2		Receive Bank1 Offer
24							t	j				2		Receive Bank2 Offer
25								t	■			2		GetBestOffer(End of Concurrency)
26									t	■		2		Initiate Send the Doc. to the Customer
27										t	■	2		Receive the confirmation from the customer
28											t	1		Send the Confirmation to Customer
29	■	■	■	■	■	■	■	■	■	■	■	11	0	**{Pre-Conditions}**
30	N	N	N	N	N	N	N	N	N	N	N	11	11	**{BPEL receive,reply, invoke activities}**
31	1											1		receive name="receiveInput"
32			1									1		Invoke name="initiateTask"
33			2									1		receive name="receiveTaskResult"
34					1							1		Invoke name="invokeCR"
35						1						1		Invoke name="invokeUnitedLoan"
36						2						1		receive name ="Bank1LoanService"
37							1					1		Invoke name="invokeBank2"
38							2					1		receive name ="StarLoanService"

39									1			1		Invoke name="initiateTask"
40										1		1		receive name="receiveTaskResult"
41											1	1		Invoke name="replyoutput"
42												11	2	**{Post-condition}**
43		y										1		Valid Client check failure
44			y									1		Credit check failure

Table 4-8 BPEL Mapping in ContextMaps (2)

Table 4-8 is to display the module of mapping of BPEL process, which follows our design pattern (table 4-4). Here, we will introduce the technology for mapping the flow with the sets: the loop, pre-condition, post-condition, synchronies, asynchronies, parallel, and concurrency.

Role "m" means that the activity is on the middle of the edge in the table 4-8. Specifically, we pick column 2 as an example. The activity in row 3"GetSSN" is between the starting node"EJB to retrieve SSN" in row 16 and ending node "GetCreditRatingService" in row 17.

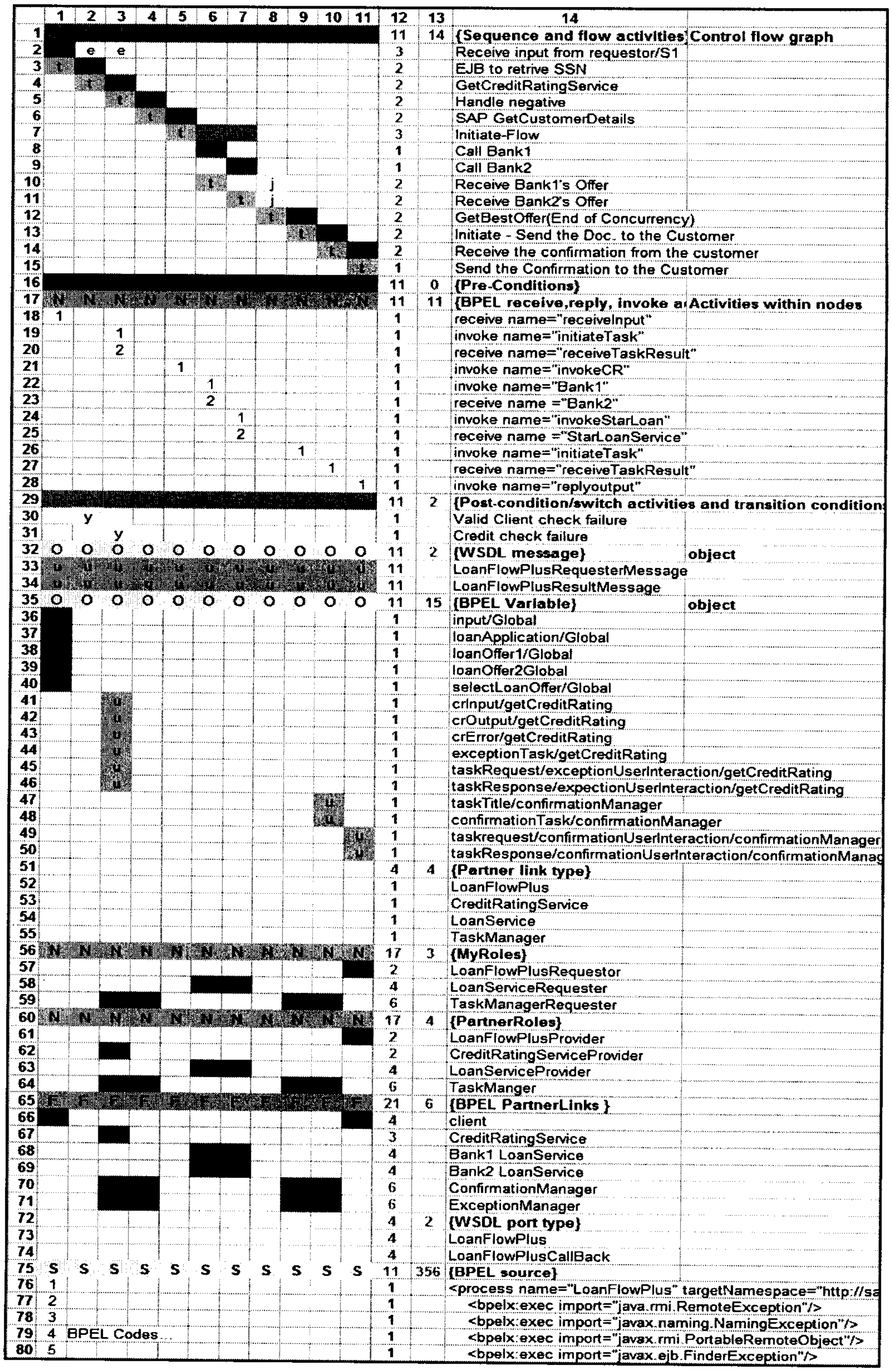
The role "L" in row 14 means that the under the sets "Sequence and flow activities" in row 14, the operation "GetSSN" can have loops. The terminated condition of the loop dependents on "{pre-conditions}" in row 29, we use notation "G" or "{Post-condition/switch activities and transition conditions}" in row 42, we use notation "P". In this "GetSSN" process, the post-condition is triggered. After the validity of client check is failure, we look the same column, and there is "e" which means controlled by the error handler and forced the process to be backed to the node “Receive input from requestor/S1” in row 15.

From row 31 to row 41, we can see which BPEL activities are involved. We mark the number with “1” or “2”. From the BPEL specification, we look at each column under the sets “{BPEL receive, reply, invoke activities}. If it has only a “1” in that column, we know that it’s “synchronies” process. If there are “1” and “2” in that column, we know that it’s an “asynchronies” process.

We use the role “c” in row 20. It means that process “Bank1_Initiate” in row 7 and process “Bank2_Initiate” in row 8 are parallel running process.

In row 23, 24 and 25, there are two “j”s and one “t” in column 8. That means “ReceiveLoanOffer1(Bank1)” process in row 9 and “ReceiveLoanOffer2(Bank2)” process in row 10 are merged and trigger the process “Initiate Task (Best Offer)” in row 11.

We use CONTEXT+ to hide some information, which is discussed above and to focus the contents that we are going to present below. The following table 4-9 shows the mapping with the rest of BPEL features.



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	
1	■	■	■	■	■	■	■	■	■	■	■	11	14	**{Sequence and flow activities**	**Control flow graph**
2	■	e	e									3		Receive input from requestor/S1	
3	t	■										2		EJB to retrive SSN	
4		t	■									2		GetCreditRatingService	
5			t	■								2		Handle negative	
6				t	■							2		SAP GetCustomerDetails	
7					t	■	■					3		Initiate-Flow	
8						■						1		Call Bank1	
9							■					1		Call Bank2	
10						t		j				2		Receive Bank1's Offer	
11							t	j				2		Receive Bank2's Offer	
12								t	■			2		GetBestOffer(End of Concurrency)	
13									t	■		2		Initiate - Send the Doc. to the Customer	
14										t	■	2		Receive the confirmation from the customer	
15											t	1		Send the Confirmation to the Customer	
16	■	■	■	■	■	■	■	■	■	■	■	11	0	**{Pre-Conditions}**	
17	N	N	N	N	N	N	N	N	N	N	N	11	11	**{BPEL receive,reply, invoke a**	**Activities within nodes**
18	1											1		receive name="receiveInput"	
19			1									1		invoke name="initiateTask"	
20			2									1		receive name="receiveTaskResult"	
21					1							1		invoke name="invokeCR"	
22						1						1		invoke name="Bank1"	
23						2						1		receive name ="Bank2"	
24							1					1		invoke name="invokeStarLoan"	
25							2					1		receive name ="StarLoanService"	
26									1			1		invoke name="initiateTask"	
27										1		1		receive name="receiveTaskResult"	
28											1	1		invoke name="replyoutput"	
29	■	■	■	■	■	■	■	■	■	■	■	11	2	**{Post-condition/switch activities and transition condition**	
30		y										1		Valid Client check failure	
31			y									1		Credit check failure	
32	O	O	O	O	O	O	O	O	O	O	O	11	2	**{WSDL message}**	**object**
33	u	u	u	u	u	u	u	u	u	u	u	11		LoanFlowPlusRequesterMessage	
34	u	u	u	u	u	u	u	u	u	u	u	11		LoanFlowPlusResultMessage	
35	O	O	O	O	O	O	O	O	O	O	O	11	15	**{BPEL Variable}**	**object**
36	■											1		input/Global	
37	■											1		loanApplication/Global	
38	■											1		loanOffer1/Global	
39	■											1		loanOffer2Global	
40	■											1		selectLoanOffer/Global	
41			u									1		crInput/getCreditRating	
42			u									1		crOutput/getCreditRating	
43			u									1		crError/getCreditRating	
44			u									1		exceptionTask/getCreditRating	
45			u									1		taskRequest/exceptionUserInteraction/getCreditRating	
46			u									1		taskResponse/expectionUserInteraction/getCreditRating	
47										u		1		taskTitle/confirmationManager	
48										u		1		confirmationTask/confirmationManager	
49											u	1		taskrequest/confirmationUserInteraction/confirmationManager	
50											u	1		taskResponse/confirmationUserInteraction/confirmationManag	
51												4	4	**{Partner link type}**	
52												1		LoanFlowPlus	
53												1		CreditRatingService	
54												1		LoanService	
55												1		TaskManager	
56	N	N	N	N	N	N	N	N	N	N	N	17	3	**{MyRoles}**	
57											■	2		LoanFlowPlusRequestor	
58						■	■					4		LoanServiceRequester	
59			■	■					■	■		6		TaskManagerRequester	
60	N	N	N	N	N	N	N	N	N	N	N	17	4	**{PartnerRoles}**	
61											■	2		LoanFlowPlusProvider	
62			■									2		CreditRatingServiceProvider	
63						■	■					4		LoanServiceProvider	
64			■	■					■	■		6		TaskManger	
65	F	F	F	F	F	F	F	F	F	F	F	21	6	**{BPEL PartnerLinks }**	
66	■										■	4		client	
67			■									3		CreditRatingService	
68						■	■					4		Bank1 LoanService	
69						■	■					4		Bank2 LoanService	
70			■	■					■	■		6		ConfirmationManager	
71			■	■					■	■		6		ExceptionManager	
72												4	2	**{WSDL port type}**	
73												4		LoanFlowPlus	
74												4		LoanFlowPlusCallBack	
75	S	S	S	S	S	S	S	S	S	S	S	11	356	**{BPEL source}**	
76	1											1		<process name="LoanFlowPlus" targetNamespace="http://sa	
77	2											1		<bpelx:exec import="java.rmi.RemoteException"/>	
78	3											1		<bpelx:exec import="javax.naming.NamingException"/>	
79	4	BPEL Codes...										1		<bpelx:exec import="javax.rmi.PortableRemoteObject"/>	
80	5											1		<bpelx:exec import="javax.ejb.FinderException"/>	

Table 4-9 BPEL Mapping in ContextMaps (3)

Sets, {BPEL variable}, list all variables used in "LoanFlowPlus". The value of 15 in the cell (35,13) means that there are 15 variables, which are listed from row 36 to row 50. Some set members end with "Global" that mean they are global variables. Others variables with path afterwards, so we can see these local variables used in which activities. Form row 36 to row 74, the role "c" means the object is created, and the "u" means the object is updated.

From Sets, {MyRoles} on row 56 to {BPEL PartnerLinks} in row 65, we follow the BPEL specification that each partner link has to some roles involved what are my roles and partner roles. So we use ContextMap notation "F" in row 65 to indicate that the {BPEL PartnerLinks} is a flow node with two sets {MyRoles} and {PartnerRole} by vertical view, For example, the partner link "Bank1 Loan Service" in row 68 and "Bank2 Loan Service" in row 69 have the same my role "Loan Service Requestor" in row 58 and partner role "Loan Service Provider" in row 63.

Sets, {BPEL source} in row 75 contain all BPEL source for our application "LoanFlowPlus". There are the sequenced numbers listed after row 75. Those sequenced sentences will be executed one by one according to the flow listed from row 1 to row 15, and will be switched by the conditions listed in row 16 and row 29 form the vertical view.

WSDL and BPEL Relationship in ContextMaps

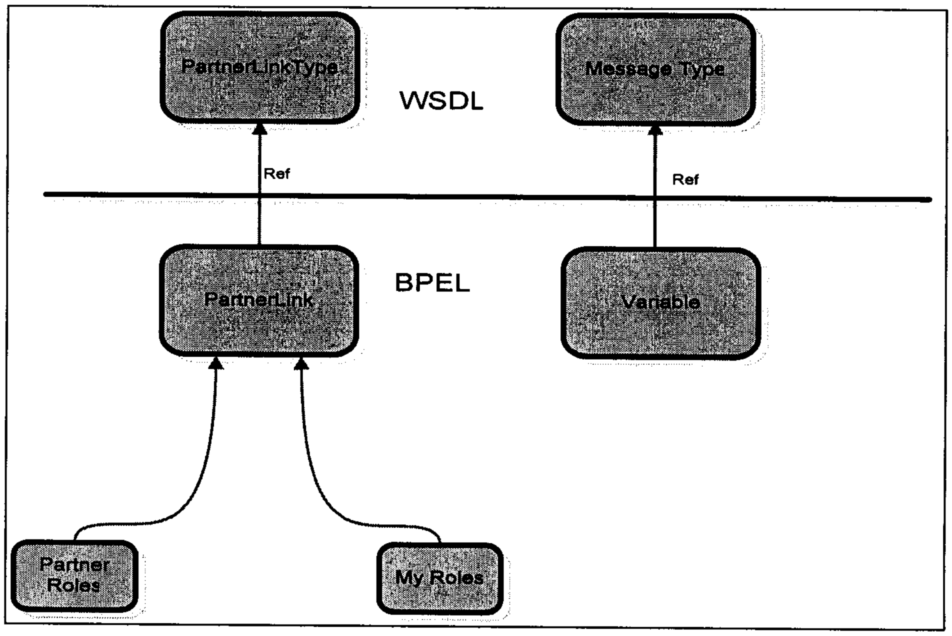
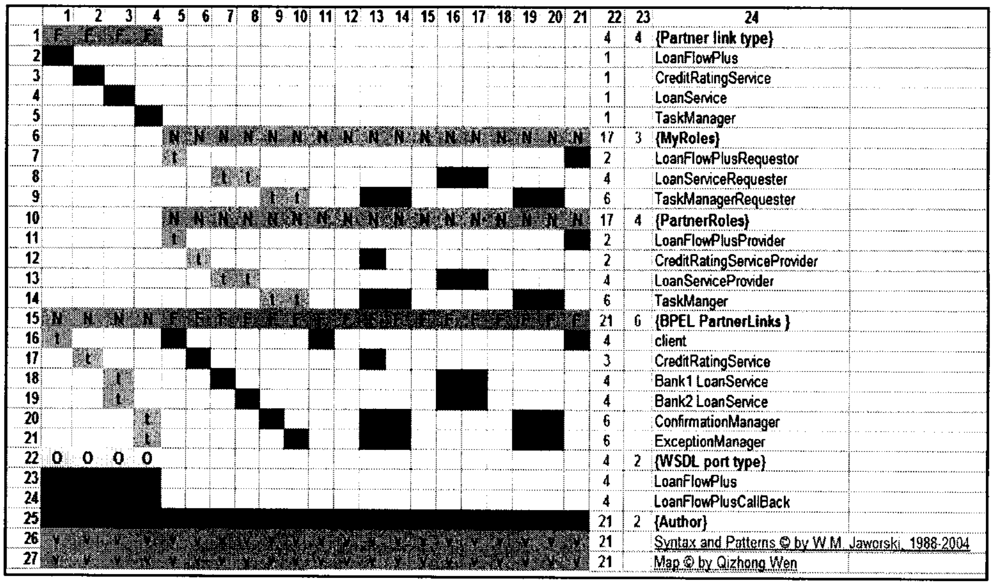


Figure 4-4 Relationship Display

From the above relationships, we can find that the defination of "PartnerLink" and "Variable" in BPEL needs to refer the "PartnerLinkType" and "Message Type" which are defined in WSDL. the following mapping, table 4-10 mapping the relationship between WSDL and BPEL in ContextMaps.



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	
1	F	F	F	F																		4	4	{Partner link type}	
2	■																					1		LoanFlowPlus	
3		■																				1		CreditRatingService	
4			■																			1		LoanService	
5				■																		1		TaskManager	
6					N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	17	3	{MyRoles}	
7					t																■	2		LoanFlowPlusRequestor	
8							t	t								■	■					4		LoanServiceRequester	
9									t	t			■	■					■	■		6		TaskManagerRequester	
10					N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	17	4	{PartnerRoles}	
11					t																■	2		LoanFlowPlusProvider	
12						t							■									2		CreditRatingServiceProvider	
13							t	t								■	■					4		LoanServiceProvider	
14									t	t			■	■					■	■		6		TaskManger	
15	N	N	N	N	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	21	6	{BPEL PartnerLinks }	
16	t				■						■										■	4		client	
17		t				■							■									3		CreditRatingService	
18			t				■									■	■					4		Bank1 LoanService	
19			t					■								■	■					4		Bank2 LoanService	
20				t					■				■	■					■	■		6		ConfirmationManager	
21				t						■			■	■					■	■		6		ExceptionManager	
22	O	O	O	O																		4	2	{WSDL port type}	
23	■	■	■	■																		4		LoanFlowPlus	
24	■	■	■	■																		4		LoanFlowPlusCallBack	
25	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	21	2	{Author}	
26	V	V	V	V	V	V	V	V	V	V	V	V	V	V	V	V	V	V	V	V	V	21		Syntax and Patterns © by W.M. Jaworski, 1988-2004	
27	V	V	V	V	V	V	V	V	V	V	V	V	V	V	V	V	V	V	V	V	V	21		Map © by Qizhong Wen	

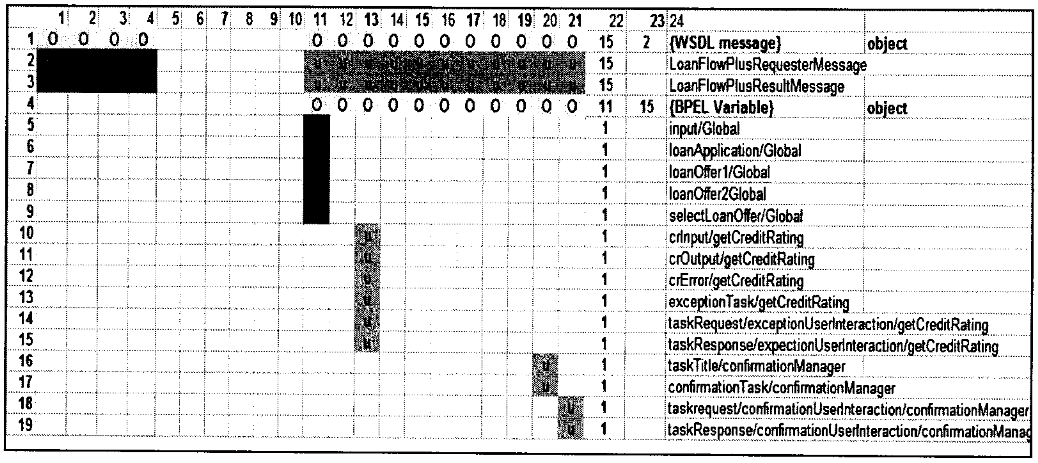
Table 4-10 The Schemata of Relationship between BPEL and WSDL

Sets, {Partner link type}, in row 1, we use notation"F" that means this sets are flow nodes. From row 2 to row 5, there are 4 members under the sets {Partner link type}. These partner link types are the basic elements in WSDL, and each "partnerlink" in BPEL will refer to its related "partnerlinktype" in WSDL. In the row15 and from column 1 to column 4, we use the notation "N" that means these sets come form those sets with notation "F". In a vertical view, we follow the flow form the role "f" to the role "t".This is one relations that we need to map.

Sets, {BPEL PartnerLinks}, in row 15, and from column 5 to column 21, we use the notation "F" that means the sets {BPEL PartLinks} include another two sets {MyRoles} and {PartnerRoles}. In the area form cell (16, 5) to cell (21,10), the role with "f" vertically map to those "t"s in each role of {MyRoles} and {PartnerRoles}. This means the sets { MyRoles} and {PartnerRoles} are under the sets {BPEL PartnerLinks}. This is the second relationship we map.

Mapping Message Types (WSDL) and Variables (BPEL) in ContextMaps

(See the following table 4-11)



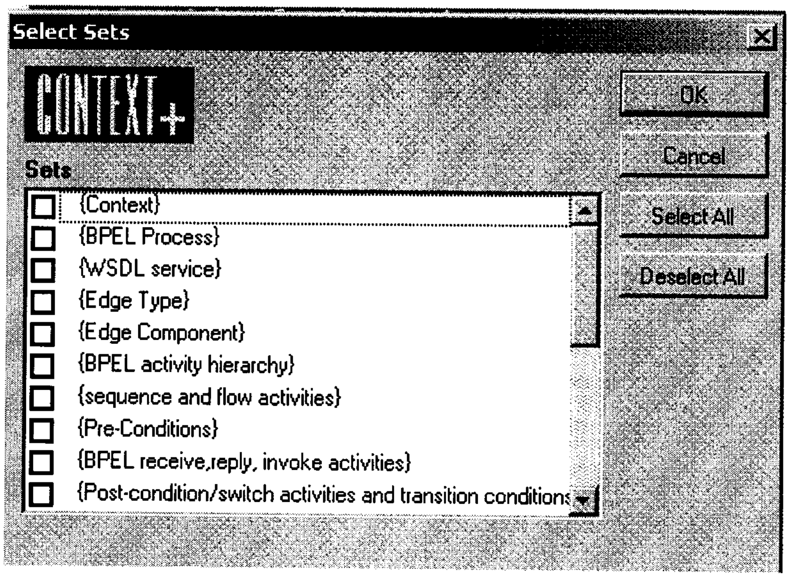
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	
1	0	0	0	0							0	0	0	0	0	0	0	0	0	0	0	15	2	{WSDL message}	object
2	■	■	■	■							u	u	u	u	u	u	u	u	u	u	u	15		LoanFlowPlusRequesterMessage	
3	■	■	■	■							u	u	u	u	u	u	u	u	u	u	u	15		LoanFlowPlusResultMessage	
4											0	0	0	0	0	0	0	0	0	0	0	11	15	{BPEL Variable}	object
5											■											1		input/Global	
6											■											1		loanApplication/Global	
7											■											1		loanOffer1/Global	
8											■											1		loanOffer2Global	
9											■											1		selectLoanOffer/Global	
10													u									1		crInput/getCreditRating	
11													u									1		crOutput/getCreditRating	
12													u									1		crError/getCreditRating	
13													u									1		exceptionTask/getCreditRating	
14													u									1		taskRequest/exceptionUserInteraction/getCreditRating	
15													u									1		taskResponse/expectionUserInteraction/getCreditRating	
16																				u		1		taskTitle/confirmationManager	
17																				u		1		confirmationTask/confirmationManager	
18																					u	1		taskrequest/confirmationUserInteraction/confirmationManager	
19																					u	1		taskResponse/confirmationUserInteraction/confirmationManag	

Table 4-11 Mapping Message Types (WSDL) and Variables (BPEL)

From Column 1-4 is WSDL domain and these massage types are created here. The column 11-21 is BPEL domain and BPEL variables from row 5-19 refer to these message types.

Query the Content and Flow by CONTEXT+ Tools

When we finish the above development of Web Services following our pattern able ContaxtMap, we can analyses our service by CONTEXT+ which is introduced in the chapter 3.6. We can query the related sets or members by single selection or combination. See the following figure 4-5.

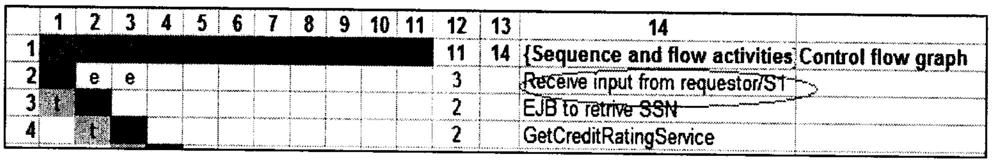


Sets

- □ {Context}
- □ {BPEL Process}
- □ {WSDL service}
- □ {Edge Type}
- □ {Edge Component}
- □ {BPEL activity hierarchy}
- □ {sequence and flow activities}
- □ {Pre-Conditions}
- □ {BPEL receive,reply, invoke activities}
- □ {Post-condition/switch activities and transition condition:

Figure 4-5 Select Set Screens

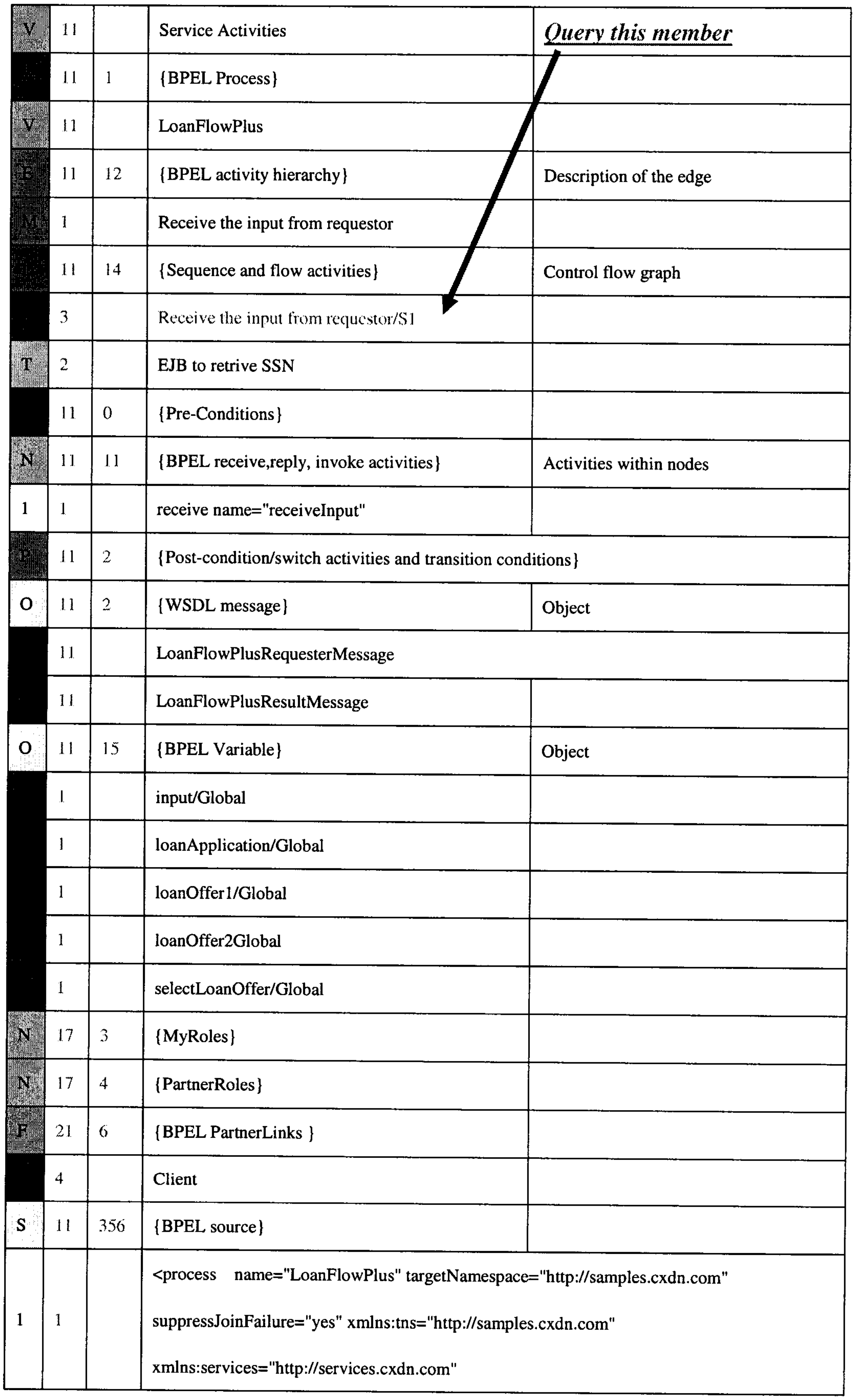
For example, let's query the activity "Receive the input from requestor" which is a member of the set "{BPEL activity hierarchy}". See the follow figure 4-6.



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	
1	■	■	■	■	■	■	■	■	■	■	■	11	14	{Sequence and flow activities}	Control flow graph
2	■	e	e									3		Receive input from requestor/S1	
3	t	■										2		EJB to retrive SSN	
4		t	■									2		GetCreditRatingService	

Figure 4-6 Select a Member for Query

■	21	3	{Context}	
V	21		Graph of LoanFlowPlus	



V	11		Service Activities	***Query this member***
	11	1	{BPEL Process}	
V	11		LoanFlowPlus	
E	11	12	{BPEL activity hierarchy}	Description of the edge
M	1		Receive the input from requestor	
	11	14	{Sequence and flow activities}	Control flow graph
	3		Receive the input from requestor/S1	
T	2		EJB to retrive SSN	
	11	0	{Pre-Conditions}	
N	11	11	{BPEL receive,reply, invoke activities}	Activities within nodes
1	1		receive name="receiveInput"	
P	11	2	{Post-condition/switch activities and transition conditions}	
O	11	2	{WSDL message}	Object
	11		LoanFlowPlusRequesterMessage	
	11		LoanFlowPlusResultMessage	
O	11	15	{BPEL Variable}	Object
	1		input/Global	
	1		loanApplication/Global	
	1		loanOffer1/Global	
	1		loanOffer2Global	
	1		selectLoanOffer/Global	
N	17	3	{MyRoles}	
N	17	4	{PartnerRoles}	
F	21	6	{BPEL PartnerLinks }	
	4		Client	
S	11	356	{BPEL source}	
1	1		<process name="LoanFlowPlus" targetNamespace="http://samples.cxdn.com" suppressJoinFailure="yes" xmlns:tns="http://samples.cxdn.com" xmlns:services="http://services.cxdn.com"	

			xmlns:bpelx="http://schemas.collaxa.com/bpel/extension" xmlns:task="http://services.collaxa.com/task" xmlns:xsd="http://www.w3.org/2001/xmlschema" xmlns="http://schema.xmlsoap.org/ws/2003/03/bussiness-process/">
2	1		<bpelx:exec import="java.rmi.RemoteException"/>
3	1		<bpelx:exec import="javax.naming.NamingException"/>
4	1		...

Table 4-12 Query Result

From this query (table 4-12), we know:

(1). this activity is the detail process to finish the activity which is described in the edge {BPEL activity hierarchy}.

(1) this activity is used in BPEL process.

(2) this activity goes to activity "EJB to ret rive SSN"

(3) this activity needs to use which BPEL "receive" activity, refer to which WSDL message, declare which variables, and BPEL PartnerLinks.

(4) which BPEL codes are to be executed.

4.8 Validate the Mapping

After modeling the Web Services, we have to validate our modeling. Our solution is: (A) to extract the BPEL codes. (B) to verify these BPEL codes by XML syntax for grammar checking because BPEL must be in accord with the XML 1.1 specifications and, (C) to run the BPEL codes in a BPEL engine to show how the real work flows go. Those VBA programming codes are archived in the "ContextMaps" under the Yahoo group.

Under the tool CONTEXT+, the following figure 4-7 is the user interface, which is to extract the BPEL source codes.

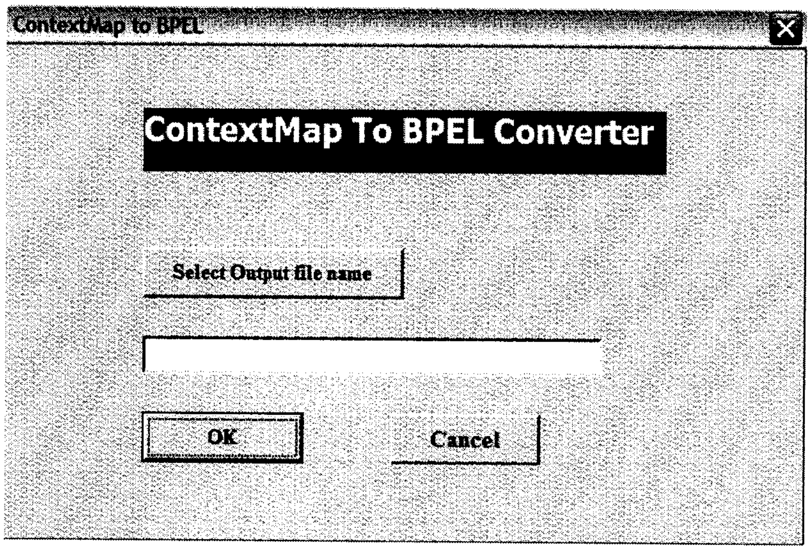


Figure 4-7 ContextMaps to BPEL Convert

(1) Get the BPEL code

Run function "Map=>BPEL", to retrieve the BPEL source code. See the figure 4-7.

(2) Check the BPEL syntax

Run function “XML=>Map” to check the BPEL code by XML syntax such as the match of the XML makeup tags, and the relationship between the parent and the child nodes.

(3) Check the BPEL flow

Set up the Web application, and run the BPEL with the related application directly to show the results.

4.8.1 CONTEXT+ Tools for Retrieving BPEL Source Code

In the table 4-9, specifically; the range from row 2 to row 15 is the flow of our traverse. See the following codes. The direction is that from the upper row to lower row, and from the left column to the right column.

```
With mysheet
            'find "f", "c" or "j" in the    {sequence and flow activities}
      For j = col_start_activities To col_end_activities
        For i = row_start_activities To row_end_activities
            s_content = .Item(i, j)
        If s_content = "f" Or s_content = "c" Or s_content = "j" Then
            Call OutputComment(i)
             'the row item is out put as a comment of activity
             ' traverse the tree, at the same column to search the "t", if there is
             a "t" then check the condition and prepare to output the source
             codes
             'search the next "f" in the same column
            If s_content = "f" Then
               Call SearchTree(i, j)
               Exit For
            End If
        End If
        Next i
      Next j
 End With
```

Figure 4-8 Pseudo Code

(A.) In each column, find the first “f, and locate the “t” in the same column, and then along the same column going down to find whether or not the pre-condition or post-condition are applied. Next, to print out the context of the cell as a comment which its column number is 24 and its row number is the same as that of related role such as “f”, “t”, “y”, “e”, “c” and “j”. Finally, to output those codes in the same column and under the sets {BPEL source} marked with the sequential numbers till the first column is processed.

(B.) We look back the “t” in this column. From this “t”, we search the next “f” in the same row and locate the column.

(C.) Normally, the process is from “f” to “t” in a vertical view. If the post-condition meets, the flow is from “f” to “e”.

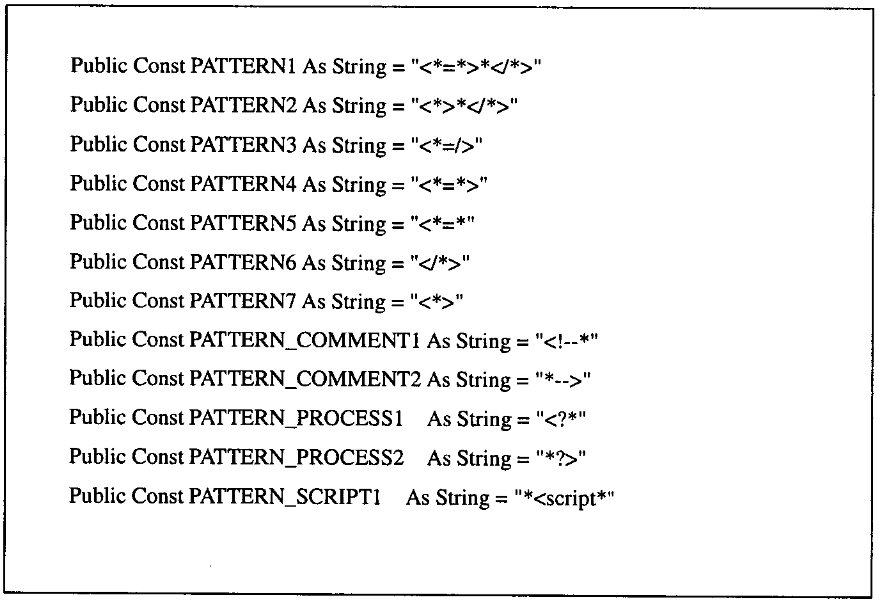
(D.) We treat the role “c”, as a mark of concurrency which follows the “t” in the nearest left side column. The output of the comment is in the column 24 and the same row as that of “c”.

(E.) We treat the role “j”, as a mark of joining which follows the “t” at the same row of “j”. Then follow the “j” to look at the “t” on the same column.

(F.) Repeat the process on the above list from (A) to (E) till the row is at the end of row in the set {Sequence and flow activities}.

4.8.2 Validation of BPEL Code by XML Syntax

When the BPEL code is downloaded, we use the function “XML to ContextMaps” to check whether there is any syntax error of the BPEL codes such as unmatched markup and incorrect relationship between the parent node and the child node. Because the BPEL and WSDL are layered on the XML specification, we follow the XML syntax to generate a XML tree, which can be viewed in the ContextMaps and be calculated the numbers of the XML elements, properties, and values of properties. We classified following patterns as our tokens for parsing XML files.



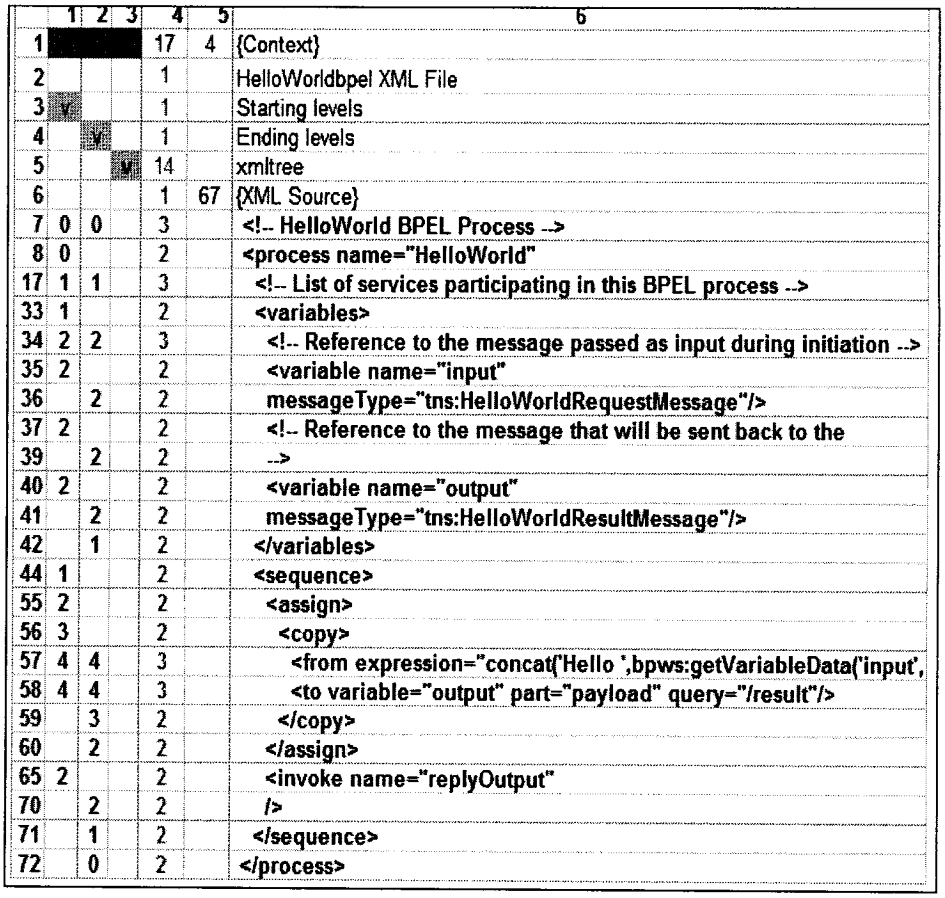
```
Public Const PATTERN1 As String = "<*=*>*</*>"
Public Const PATTERN2 As String = "<*>*</*>"
Public Const PATTERN3 As String = "<*=/>"
Public Const PATTERN4 As String = "<*=*>"
Public Const PATTERN5 As String = "<*=*"
Public Const PATTERN6 As String = "</*>"
Public Const PATTERN7 As String = "<*>"
Public Const PATTERN_COMMENT1 As String = "<!--*"
Public Const PATTERN_COMMENT2 As String = "*-->"
Public Const PATTERN_PROCESS1    As String = "<?*"
Public Const PATTERN_PROCESS2    As String = "*?>"
Public Const PATTERN_SCRIPT1    As String = "*<script*"
```

Figure 4-9 Pseudo Code (Token Patterns)

Therefore, we can verify the BPEL by our XML parser. See the result from the following table 4-13,14,15. This helps check complete matched pairs of markups.

1	■	■	■	■	4	4	{Context}
2	V				1		sampleXML Filesample
3		V			1		Starting levels
4			V		1		Ending levels
5				V	1		Xmltree
6	S				1	1	{XML Source}
7				F	1	2	{XML Nodes}
8							[End of Nodes]
9				N	1	2	{XML Node Markups}
10							[End of Node Markups]
11				N	1	2	{XML Node Properties}
12							[End of Nodes Properties]
13				N	1	2	{XML Node Property Values}
14							[End of Node Property Values]
15				N	1	2	{XML Node Comments}
16							[End of XML Comments]
17				N	1	2	{XML Node Processing Instructions}
18							[End of XML Processing Instructions]
19	■	■	■	■	4	2	{Author}
20	v	v	v	v			Syntax and Patterns by W.M. Jaworski, 1988-2002
21	v	v	v	v			Mapped by Qizhong Wen

Table 4-13 The XML Schema by ContextMaps



	1	2	3	4	5	6
1	■	■	■	17	4	{Context}
2				1		HelloWorldbpel XML File
3	v			1		Starting levels
4		v		1		Ending levels
5			v	14		xmltree
6				1	67	{XML Source}
7	0	0		3		**<!-- HelloWorld BPEL Process -->**
8	0			2		**<process name="HelloWorld"**
17	1	1		3		**<!-- List of services participating in this BPEL process -->**
33	1			2		**<variables>**
34	2	2		3		**<!-- Reference to the message passed as input during initiation -->**
35	2			2		**<variable name="input"**
36		2		2		**messageType="tns:HelloWorldRequestMessage"/>**
37	2			2		**<!-- Reference to the message that will be sent back to the**
39		2		2		**-->**
40	2			2		**<variable name="output"**
41		2		2		**messageType="tns:HelloWorldResultMessage"/>**
42		1		2		**</variables>**
44	1			2		**<sequence>**
55	2			2		**<assign>**
56	3			2		**<copy>**
57	4	4		3		**<from expression="concat('Hello ',bpws:getVariableData('input',**
58	4	4		3		**<to variable="output" part="payload" query="/result"/>**
59		3		2		**</copy>**
60		2		2		**</assign>**
65	2			2		**<invoke name="replyOutput"**
70		2		2		**/>**
71		1		2		**</sequence>**
72		0		2		**</process>**

Table 4-14 The Hierarchy of XML Nodes with ContextMaps

(* Syntax and Patterns by W.M. Jaworski, 1988-2002)

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
1	F	F	F	F	F	F	F	F	F	F	F	F	F	F	14	14	**{XML Nodes}**	
2	■														1		n1	Process
3	t	■													2		n2	PartnerLinks
4		t	■												2		n3	PartnerLink
5	t			■											2		n4	Variables
6				t	■										2		n5	Variable
7				t		■									2		n6	Variable
8	t						■								2		n7	Sequence
9							t	■							2		n8	Receive

10							t		■						2		n9	Assign
11									t	■					2		n10	Copy
12										t	■				2		n11	from
13										t		■			2		n12	to
14							t						■		2		n13	invoke
15																	[End of Nodes]	
16	N	N	N	N	N	N	N	N	N	N	N	N	N	N	14	13	{XML Node Markups}	
17	■														1		process	
18		■													1		partnerLinks	
19			■												1		partnerLink	
20				■											1		Variables	
21					■	■									2		variable	
22							■								1		Sequence	
23								■							1		receive	
24									■						1		Assign	
25										■					1		Copy	
26											■		...		1		From ...	....

Table 4-15 The BPEL Elements in The XML Tree by ContextMaps

As the limited size to display, the table 4-16 continues the table 4-15 which is linked by each column.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
1	N	N	N	N	N	N	N	N	N	N	N	N	N	N	14	19	{XML Node Properties}
2	1		1		1	1		1					1		6		Name
3	2														1		targetNamespace
4	3														1		suppressJoinFailure
5	4														1		xmlns:tns

6	5														1		Xmlns
7			2												1		partnerLinkType
8			3												1		myRole
9			4												1		partnerRole
10					2	2									2		messageType
11								2					2		2		partnerLink
12								3					3		2		portType
13								4					4		2		Operation
14								5				1			2		Variable
15								6							1		createInstance
16											1				1		Expression
17												2			1		Part
18												3			1		Query
19													5		1		inputVariable
20																	[End of Nodes Properties]
21	N	N	N	N	N	N	N	N	N	N	N	N	N	N	14	21	{XML Node Property Values}
22	1														1		"HelloWorld"
23	2,4														1		"http://samples.cxdn.com"
24	3							6							2		"yes"
25	5														1		"http://schemas.xmlsoap.org/ws/2003/03/business-process/"
26			1					2					2		3		"client"
27			2					3							2		"tns:HelloWorld"
28			3												1		"HelloWorldProvider"
29			4												1		"HelloWorldRequester"
30	N	N	N	N	N	N	N	N	N	N	N	N	N	N	14	26	{XML Node Comments}
31	t														1		<!-- HelloWorld BPEL Process -->

32		t													1		**<!-- List of services participating in this BPEL process -->**
33			t												1		**<!--**
34			t												1		**The 'client' role represents the requester of this service. It is**

Table 4-16 BPEL Values and Comments in XML Tree by ContextMaps

(*Syntax and Patterns by W.M. Jaworski, 1988-2002*)

The logic to check BPEL codes by XML syntax is to have a ContextMap schema for showing BPEL in XML format first. See the following table 4-13.

In table 4-13, in row 6, under the set {XML Source}, we put the downloaded BPEL source inside.

In row 7, under the set {XML Nodes}, we mark the hierarchic number for each XML element, which actually is as the BPEL element.

In row 9, under the set {XML Node Markups}, we keep the name of each node, which is vertically linked the node in the set {XML Nodes}.

In row 11, under the set {XML Node Properties}, we keep the property name, which is vertically linked the each XML node.

In row 13, under the set {XML Node Property Values}, we keep the value of each property, which is vertically linked the each XML property.

In row 15, under the set {XML Node Comments}, we keep the XML comments, which actually are the BPEL comments, and are vertically linked the each XML node.

In row 14, under the set {XML Node Processing Instructions}, we keep some instructions.

Next, we run "XML to ContextMaps". If the BPEL is correct as measured by XML syntax, the result is automatically shown in ContextMaps to be the same as the table 4-14, 14,16.

In the following table 4-14, column 1 represents each start level of the BPEL element and column 2 indicates its end level. The CONTEXT+ scans whole BPEL source code and records each hierarchy of BPEL elements. If the number of the starting level is the same as the number of the ending level that means this node is without its child like row 17.

The top level is from "0", which starts at cell (row 8, column1) and ends at cell (row 72, column2). Under the top level, there are total four 4 levels. Also we can see the most inner levels are at row 57 and row 58.

The table 4-15 and table 4-16, we display four parts of "HelloWorld'. At row 1 in the table 4-11, we use "F" to say that {XML Node} is the beginning node which includes the node {XML Node Markups}, {XML Node Properties}, {XML Node Property Values} and {XML Node Comments}.

In the table 4-15, the XML tree starts from "f" at (row2, col1), and there are three "t"s at row 3, row 5, row 8 in the column 1. These three "t"s is the children of parent "f" at cell (row2, col1). There is an "f" at cell (row 17, col1), means the element "process' comes from the node "n1". Follow the matched "f' in the same column, we named the each BPEL element with a sequenced node.

In the table 4-15, follow the vertical view, the numbers in column 1 in the set {XML Node Markups} such as n1 to n5 from row 2 to row 6. That means there are five properties (Name, targetNamespace, suppressJoinFailure, xmlns:tns and Xmlns) under the element process from cell(row1 to 6, col1) in the table 4-16.

In the table 4-16, each property has a value embedded. A “1” at cell (row22, col1) means the value “HelloWorld” is under the property “Name” at cell (row2, col1). We can also find that property “targetNamespace” and “xmlns:tns” have the same property value “http://samples.cxdn.com”.

In the table 4-16, the “t“ in the set {XML Node Comments} is linked to the “f” of the set {XML Nodes}, along with the same column. The comment at cell (row30, col17) is comment the node n1 at cell (row2, col17) in the table 4-16.

4.8.3 Validate the Flow

Our test procedure follows the graph in our mapping, from row 2 to row 15 in the table 4-9. In order to fully test the flow, we need to add some EJB functions and a database to make the whole process run able. Here, we skip those tests, which retrieve more detailed student information from SAP system.

Step 1. The database “Loan_customer” is to keep the user information and implementation in “Pointbase” database.

Field	**Type**	**Text**
SSN	char(11)	Student No.
Name	char(50)	Student Name
Email	char(30) , Primary key	Student's Email
Provider	char(20)	Bank Name

Status	char(1)	Student Status
Password	char(10)	Student's Password

Table 4-17 Structure of Table "Loan_customer"

Step 2. The EJB "mybean.java" is to get user information and check user authority.

We use ContextMaps to model the "Mybean' instead of UML class diagram. The "Mybean" class model is shown in ContextMaps, (Table 18). By the Cardinality in CONTEXT+, we can see that there are 6 external packages used. The class "mybean" include 35 functions and 10 properties.

	1	2	3	4	5
1	■	■	**2**	**1**	**{Context}**
2	V	V	2		Mybean.java
3	F		**1**	**6**	**{Package}**
4	■		1		import java.sql.*;
5	■		1		import javax.sql.*;
6	■		1		import java.util.*;
7	■		1		import javax.ejb.*;
8	■		1		import javax.naming.*;
9	■		1		import java.beans.*;
10	N	F	**2**	**1**	**{Class}**
11	t	■	2		public mybean
12		N	**1**	**35**	**{Functions}**
13		t	1		mybean()
14		t	1		void getSampleProperty()
15		t	1		void setSampleProperty(String value)
16		t	1		void addPropertyChangeListener (PropertyChangeListener listener)
17		t	1		void removePropertyChangeListener(PropertyChangeListener listener)
18		t	1		void setCustomerName(String name)
19		t	1		String getSSN()
20		t	1		void setSSN(String ssn)

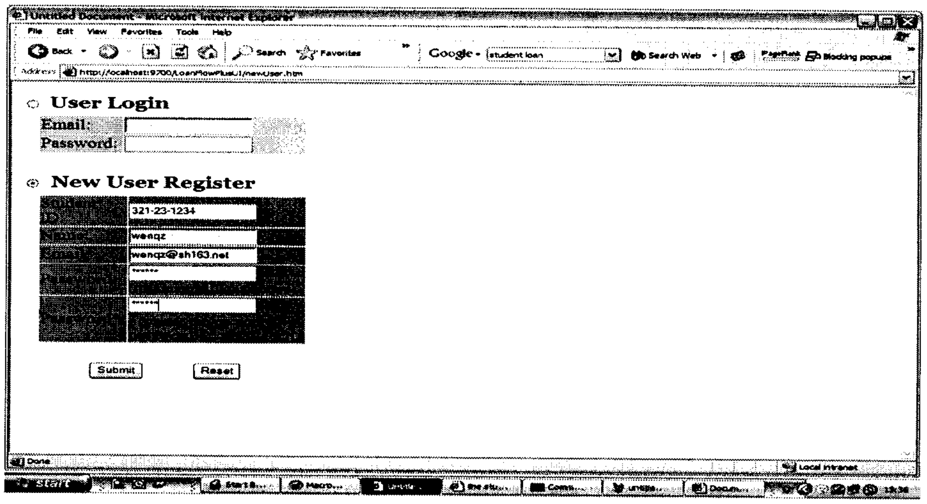
21		t	1		boolean getLogin()
22		t	1		void setLogin()
23		t	1		String getPassword()
24		t	1		void setPassword(String password)
25		t	1		String getEmail()
26		t	1		void setEmail(String email)
27		t	1		void setProvider(String provider)
28		t	1		String getProvider()
29		t	1		void setApproved(boolean approved)
30		t	1		boolean selectByEmail(String primaryKey) throws NamingException,SQLException
31		t	1		boolean ejbCreate(String ssn, String name, String email,String password)
32		t	1		String ejbFindByPrimaryKey(String primaryKey) throws FinderException
33		t	1		Collection ejbFindByCustomerName(String name) throws FinderException
34		t	1		void ejbRemove()
35		t	1		void setEntityContext(EntityContext context)
36		t	1		void ejbActivate()
37		t	1		void ejbPassivate()
38		t	1		void ejbLoad()
39		t	1		void ejbStore()
40		t	1		Connection makeConnection()
41		t	1		void insertRow
42		t	1		void deleteRow(String id)
43		t	1		boolean selectByPrimaryKey(String primaryKey)
44		t	1		boolean selectByEmailPassword(String primaryKey,String password)
45		t	1		Collection selectByCustomerName(String name)
46		t	1		void loadRow() throws NamingException,SQLException
47		t	1		void storeRow() throws NamingException,SQLException
48		N	1	10	**{Properties}**
49		t	1		private String id;
50		t	1		private String ssn;
51		t	1		private String name;

52		t	1		private String email;
53		t	1		private String password = "";
54		t	1		private String provider = null;
55		t	1		private String status = null;
56		t	1		private boolean login = true;
57		t	1		private EntityContext context;
58		t	1		private String dbName = "java:/BPELSamplesDataSource";
59			2	2	**{Author}**
60	v	v	2		Syntax and Patterns by W.M. Jaworski, 1988-2002
61	v	v	2		Mapped by Qizhong Wen

Table 4-18 Class Diagram of EJB "Mybean" with ContextMaps

Step 3.List of some user interface and JSP codes

There are more than 23 JSP screens or source codes involved in this Web Service "LoanFlowPlus". The source codes and demo results are archived in Yahoo group "ContextMaps". Here, we pick some screens for demo purposes.


Untitled Document - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Google - student loan
Search Web
PageRank Blocking popups
Address http://localhost:9700/LoanFlowPlusUI/newUser.htm

○ User Login

Email:

Password:

● New User Register

321-23-1234

wenqz

wenqz@sh163.net

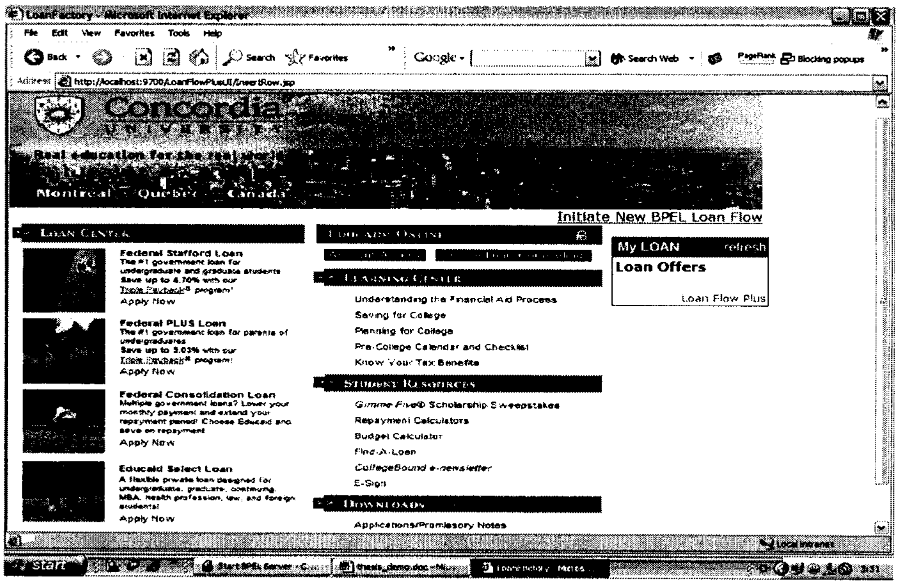
******

******

Submit
Reset
Done
Local intranet
start

Figure 4-10 New User Sign in Screen

The user input his information as the figure 4-10 prompted, and the system call EJB "mybean" to check the valid user. If it is a valid user login in, the system calls the screen, figure 4-11. Otherwise, the system sends the error message to the user.


LoanFactory - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://localhost:9700/LoanFlowPlusUI/insertRow.jsp
Concordia UNIVERSITY
Real education for the real world
Montreal · Quebec · Canada
start
Start BPEL Server - C...
thesis_demo.doc - Mi...
LoanFactory - Micros...

Initiate New BPEL Loan Flow

LOAN CENTER

Federal Stafford Loan
The #1 government loan for undergraduate and graduate students
Save up to 4.70% with our Triple Payback® program¹
Apply Now

Federal PLUS Loan
The #1 government loan for parents of undergraduates
Save up to 3.03% with our Triple Payback® program!
Apply Now

Federal Consolidation Loan
Multiple government loans? Lower your monthly payment and extend your repayment period! Choose Educaid and save on repayment
Apply Now

Educaid Select Loan
A flexible private loan designed for undergraduate, graduate, continuing, MBA, health profession, law, and foreign students!
Apply Now

EDUCAID ONLINE

LEARNING CENTER

Understanding the Financial Aid Process
Saving for College
Planning for College
Pre-College Calendar and Checklist
Know Your Tax Benefits

STUDENT RESOURCES

Gimme Five® Scholarship Sweepstakes
Repayment Calculators
Budget Calculator
Find-A-Loan
CollegeBound e-newsletter
E-Sign

DOWNLOADS

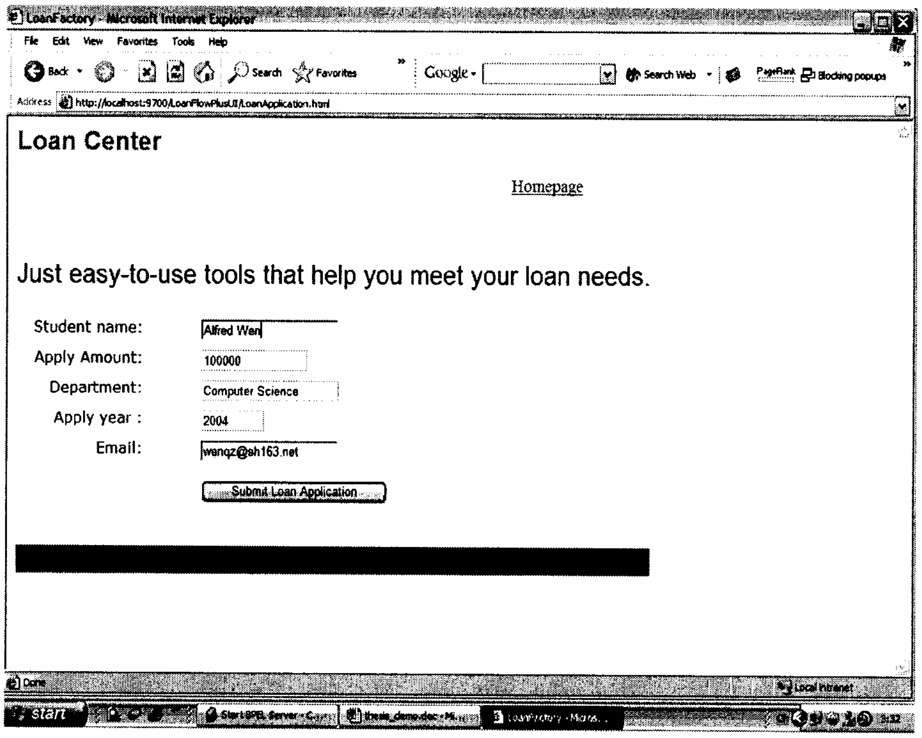
Applications/Promissory Notes

My LOAN refresh

Loan Offers

Loan Flow Plus

Figure 4-11 Student Loan Application


LoanFactory - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://localhost:9700/LoanFlowPlusUI/LoanApplication.html
Done
Local intranet
start
Start BPEL Server - C...
thesis_demo.doc - Mi...
LoanFactory - Micros...

Loan Center

Homepage

Just easy-to-use tools that help you meet your loan needs.

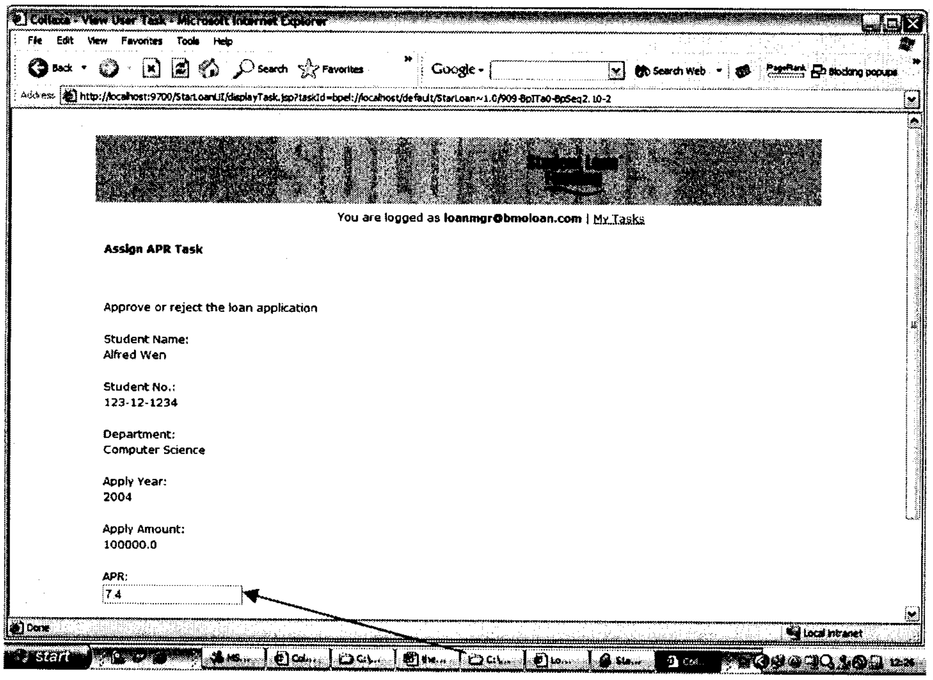
Student name:	Alfred Wen
Apply Amount:	100000
Department:	Computer Science
Apply year :	2004
Email:	wenqz@sh163.net

Submit Loan Application

Figure 4-12 Loan Application Form

The figure 4-12 is the user portal. When the user fills his loan information in this screen, the system will call EJB "CreditRatingServiceEJB". If the user's credit is good, the system sends user's application to the Royal Bank and Bank of Montreal to wait a APR rate from each bank. Otherwise, the system sends the error message to the user.

The figure 4-13 shows the Bank of Montreal receives the loan application and assigns the APR rate to the user.


Collaxa - View User Task - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back
Search
Favorites
Google
Search Web
PageRank
Blocking popups
Address http://localhost:9700/StarLoanUI/displayTask.jsp?taskId=bpel://localhost/default/StarLoan~1.0/909-BpITa0-BpSeq2.10-2
Done
Local intranet
start

You are logged as **loanmgr@bmoloan.com** | My Tasks

Assign APR Task

Approve or reject the loan application

Student Name:
Alfred Wen

Student No.:
123-12-1234

Department:
Computer Science

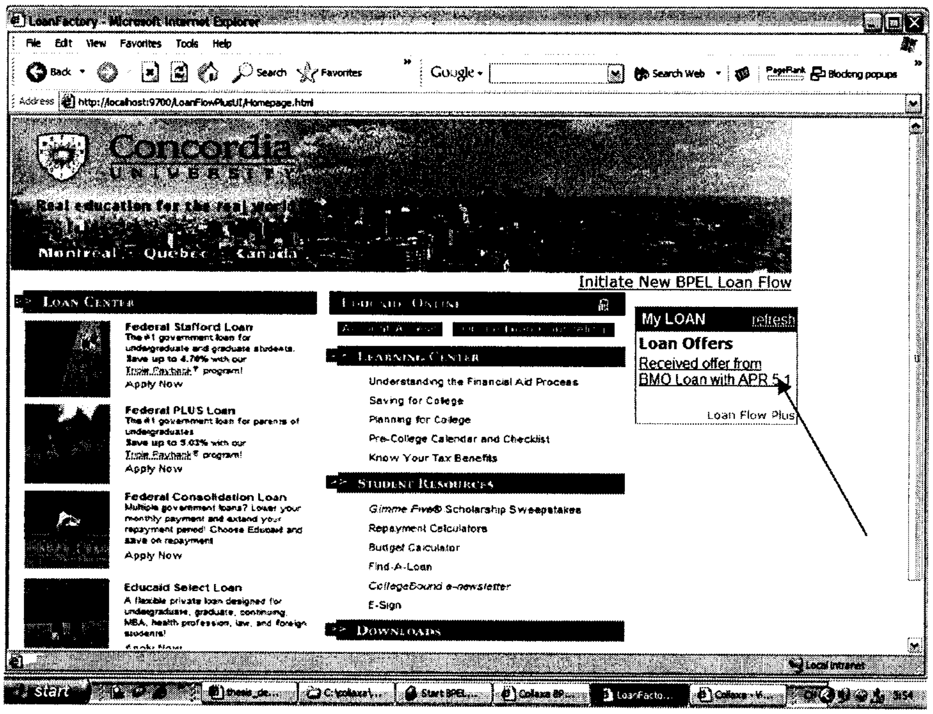
Apply Year:
2004

Apply Amount:
100000.0

APR:
7.4

Figure 4-13 Bank Side Screen

The following figure 4-14 shows the result in the loan applicant's screen.



LoanFactory - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Google Search Web PageRank Blocking popups
Address http://localhost:9700/LoanFlowPlusUI/Homepage.html
Concordia UNIVERSITY
Real education for the real world
Montreal · Quebec · Canada
Local intranet
start thesis_de... C:\collaxa\... Start BPEL... Collaxa BP... LoanFacto... Collaxa - V... 3:54

Initiate New BPEL Loan Flow

LOAN CENTER

Federal Stafford Loan
The #1 government loan for undergraduate and graduate students.
Save up to 4.70% with our Triple Payback® program!
Apply Now

Federal PLUS Loan
The #1 government loan for parents of undergraduates
Save up to 5.03% with our Triple Payback® program!
Apply Now

Federal Consolidation Loan
Multiple government loans? Lower your monthly payment and extend your repayment period! Choose Educaid and save on repayment
Apply Now

Educaid Select Loan
A flexible private loan designed for undergraduate, graduate, continuing, MBA, health profession, law, and foreign students!
Apply Now

EDUCAID ONLINE

LEARNING CENTER

Understanding the Financial Aid Process
Saving for College
Planning for College
Pre-College Calendar and Checklist
Know Your Tax Benefits

STUDENT RESOURCES

Gimme Five® Scholarship Sweepstakes
Repayment Calculators
Budget Calculator
Find-A-Loan
CollegeBound e-newsletter
E-Sign

DOWNLOADS

My LOAN refresh

Loan Offers

Received offer from BMO Loan with APR 5.1

Loan Flow Plus

Figure 4-14 User Side Screen, Receive the Offer from The Bank

Step 4. To test the Web Service, we have to set the following environment. We choose Collaxa BPEL server, version 2.0, and release 6.0 as our BPEL engine.

The BPEL server architecture

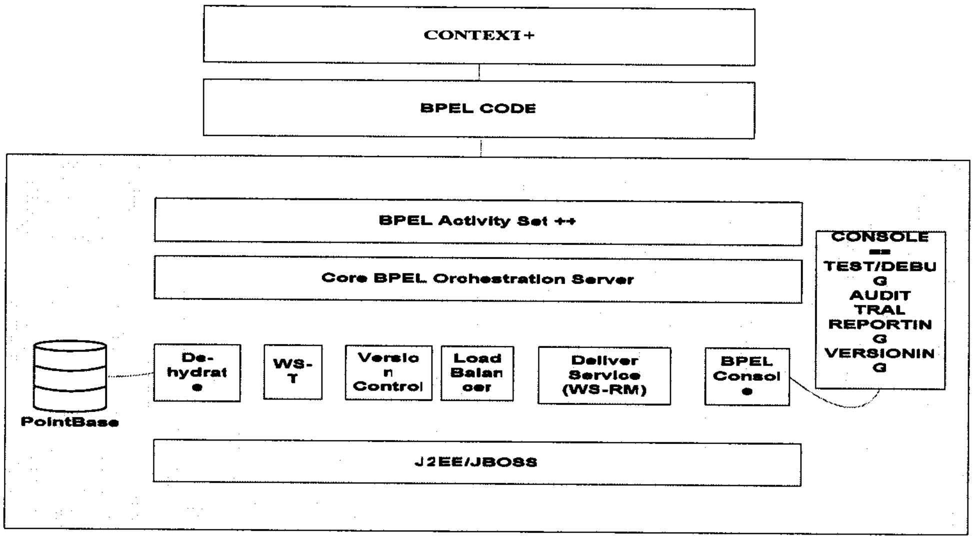


Figure 4-15 BPEL Server Architecture

Here is brief introduction to the components in BPEL server, more information can be obtained from the Collaxa Co.

The **BPEL server** is the core of the solution. The BPEL processes are executed, synchronous and asynchronous interactions are collaborated here.

The **dehydration module** keeps BPEL instances in a database and it will be triggered and then reply the services when receiving the request from remote partnerLinks and other asynchronous or long-running activities.

The **WS-T module** coordinates BPEL compensating processes.

The **version control module** offers a software maintenance function to trace each versioning of your BPEL processes.

The **delivery service** provides massage transfers by binding communication protocols such as SOAP and JMS. It also provides the function for authentication, and encryption.

The **BPEL console** monitors execution events and provide a visual view for visual flow trails, textual audit trails and debugging.

System Components:

Operation system: Microsoft Windows/2000/XP with IE6.0

JDK version: J2SDK 1.4.2.03

Java/J2EE: BEA WebLogic application server or a bundled JBoss version.

Datebase: Pointbase version 2.0.23

BPEL server: Collaxa BPEL server, version 2.0, release 6.0

4.9 Summary

Through the implementation of the above Web Services, we can see that our ContextMaps are a very powerful tool which reflects the 4P (pattern, plug, process and perform-able) abilities of ContextMaps. Especially, The ContextMaps bridge the model between the service driven and the object driven models, between BPEL and WSDL. What's more, it solves the consistency problems between different systems and between processes and their sub processes. Through its CONTEXT+ tool, it helps us to understand the business process and to analyze the complexity of each process.

Chapter 5 Experimental Results

From the above chapter 4 by the sample"LoanFlowPlus", we can see that the ContextMap technology can map the BPEL process and its relationship with WSDL. Because of its 4P's abilities, ContextMaps can play a role between BPMN and UML.

5.1 Testing Approaches, Methods and Results

Here, we use three approaches (BPMN, UML and ContextMaps) to map a simple Web Service "HelloWorld". The purpose is that to show the difference among these three popular model methodologies. We pick the source code from the Collaxa tutorials.

The "HelloWorld" is an asynchronous process, which accept a client's character input and return the client's with the initial word "Hello" and client's input.

BPEL "Hello World" code:

```
<!-- HelloWorld BPEL Process -->
<process name="HelloWorld"
         targetNamespace="http://samples.cxdn.com"
         suppressJoinFailure="yes"
        ....
         >
      <partnerLinks>
               <partnerLink name="client"
                      ... />
       </partnerLinks>
    <variables>
              <variable name="input"
                  messageType="tns:HelloWorldRequestMessage"/>
              <variable name="output"
                  messageType="tns:HelloWorldResultMessage"/>
    </variables>
     <sequence>
         <receive name="receiveInput" partnerLink="client"
               ...
         <assign>
```

```
            <copy> ...</copy>
         </assign>
        <invoke name="replyOutput"
                partnerLink="client" .... />
      </sequence>
</process>
```

WSDL "Hello World" code

```
<?xml version="1.0"?>
<definitions name="HelloWorld"
      ... >
   <types>
        <schema attributeFormDefault="qualified"
                ...          >
         <element name="name" type="string" />
         <element name="result" type="string"/>
        </schema>
    </types>
    <message name="HelloWorldRequestMessage">
        <part name="payload" element="tns:name"/>
    </message>
    <message name="HelloWorldResultMessage">
        <part name="payload" element="tns:result"/>
    </message>
    <portType name="HelloWorld">
        <operation name="initiate">
            <input message="tns:HelloWorldRequestMessage"/>
        </operation>
    </portType>
    <portType name="HelloWorldCallback">
        <operation name="onResult">
            <input message="tns:HelloWorldResultMessage"/>
        </operation>
    </portType>
<plnk:partnerLinkType name="HelloWorld">
        <plnk:role name="HelloWorldProvider">
            <plnk:portType name="tns:HelloWorld"/>
        </plnk:role>
        <plnk:role name="HelloWorldRequester">
            <plnk:portType name="tns:HelloWorldCallback"/>
        </plnk:role>
    </plnk:partnerLinkType>
```

```
</definitions>
```

5.1.1 The Difference among BPMN, UML and ContextMap Modeling

We put three methods of modeling diagrams first for easy comparison, and the explanations of each modeling are in the following section.

BPMN Modeling

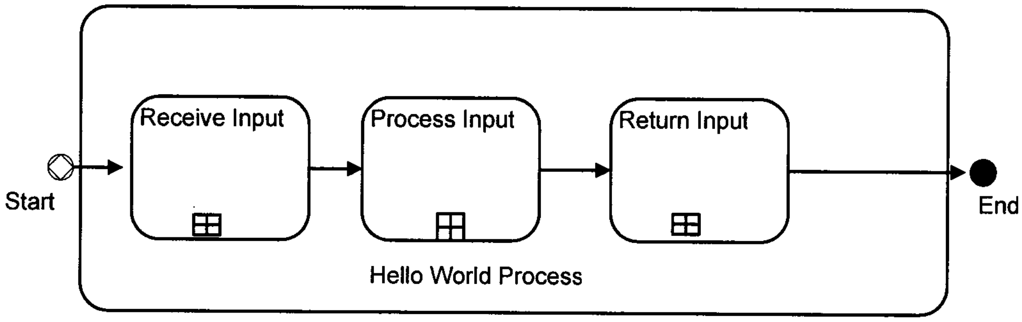


Figure 5-1 BPMN Hello_World Modeling

UML modeling - Mapping Asynchronous Sequence

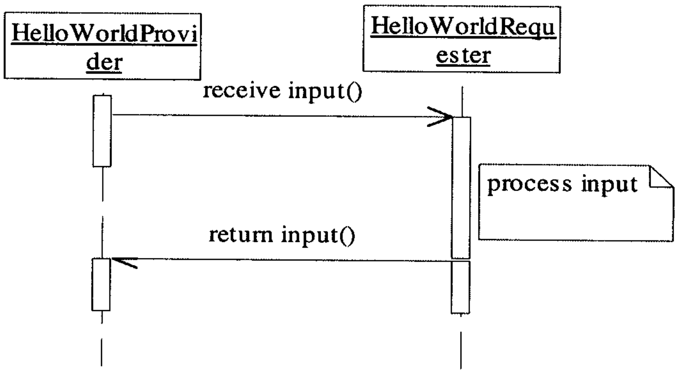


Figure 5-2 UML Hello_World Sequence Diagram

UML Modeling - Mapping Process

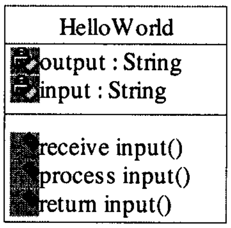


Figure 5-3 UML Hello_World Calss Diagram

UML Modeling - Mapping Partners

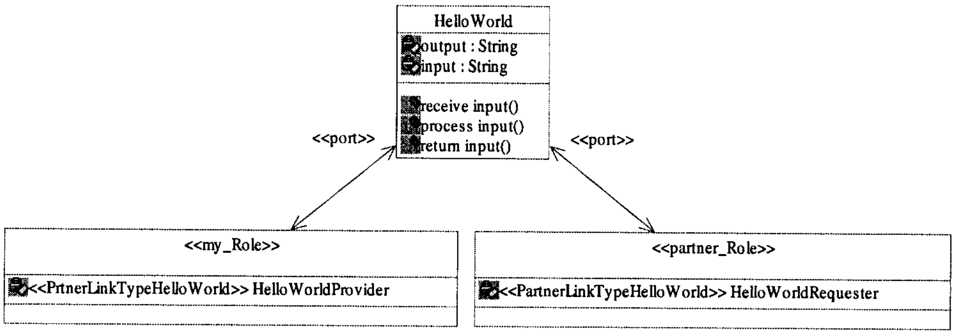


Figure 5-4 Partner Mapping Diagram

ContextMap Modeling

	1	2	3	4	5	6	7	8
1	■	■	■	■	■	5	4	{Context}
2	V	V	V	V	V	5		Graph of Hello World
3			V	V	V	3		Service Activity
4	V					1		PartnerLinkTypes
5		V				1		Partners
6			■	■	■	3	1	{BPEL Process}
7			V	V	V	3		Hello World
8	■					1	1	{WSDL service}
9	V					1		Hello World
10			X	X	X	3	1	{Edge Type}
11			X	X	X	3		BPEL Process
12			■	■	■	3	1	{Edge Component}

13			v	v	v	3		WSDL definiation
14			▨	▨	▨	3	3	{BPEL activity hierarchy}
15			▨			1		Receive input
16				▨		1		Process Input
17					▨	1		Return Input
18			■	■	■	3	3	{sequence and flow activities}
19			■		t	2		Receive input
20			t	■		2		Process Input
21				t	■	2		Return Input
22			■	■	■	3	0	{Pre-Conditions}
23			N	N	N	3	2	{BPEL receive,reply, invoke activities}
24			1			1		receive name="receiveInput"
25					1	1		invoke name="replyOutput"
26			▨	▨	▨	3	0	{Post-condition/switch activities and transition conditions}
27			O	O	O	3	2	{WSDL message}
28			■			1		HelloWorldRequesterMessage
29					■	1		HelloWorldResultMessage
30			O	O	O	3	2	{BPEL Variable}
31			u	u	u	3		input/Globe
32				u	u	2		output/Globe
33	P					1	1	{Partner link type}
34	■					1		HelloWorld
35		N	N	N	N	4	1	{myRoles}
36		t			■	2		HelloWorldProvider
37		N	N	N	N	4	1	{PartnerRoles}
38		t	■			2		HelloWorldRequester
39	N	P	F	F	F	5	1	{BPEL partners }
40	t	■	u		u	4		Client

41	O					1	1	{WSDL port type}
42	■					1		HelloWorld
43	O					1	1	{WSDL PartnerLinkType}
44	■					1		HelloWorld
45			S	S	S	3	59	{BPEL source}
46			1			1		<!-- List of services participating in this BPEL process -->
47			2			1		<partnerLinks>
48	■	■	■	■	■	5	2	{Author}
49	V	V	V	V	V	5		Syntax and Patterns © by W.M. Jaworski, 1988-2004

Table 5-1 Hello_World Mapped by ContextMaps.

5.1.2 Analyzing the Problem of BPMN Modeling

From the same description of BPMN notation such as receive input, process input and return input, these internal processes could be mapped as one of the following activities <receive>, <reply>, and <invoke>. So BPMN still needs to be specified as a XML language layer above business process execution languages.

The current BPMN system still lacks enough notations. For example, we can't see whether these processes belong to synchronized or asynchronized processes. Therefore, the current BPMN notations don't have enough notations to model Web Services.

1. BPMN notation stays on a higher level modeling. As a result, it can not model object flow.
2. To model detail lower level process such as those processes under the "process input" needs more graph models. If the application is large enough, more graph models will cause the inconsistency.

3. The BPMN modeling notation lacks the process ability to analysis and traces for each BPMN model by merging or demeshing different BPMN modeling.
4. The current BPMN can not map the relationships between WSDL and BPEL.
5. The current BPMN modeling notation can not provide a design pattern to guide a design.

5.1.3 Analyzing the Problem of UML Modeling

1. To model a Web Service needs to use several different UML diagrams to present degree of intrest. If the application is large and complex, more UML diagrams need to model a part of system views, the whole modeling will become information overload and inconsistency prone.
2. UML lacks enough notations to model the relations between WSDL and BPEL.
3. UML lacks of process-ability to analysis and trace each UML modeling diagrams by merging or demeshing different BPMN modeling.
4. UML requires different design patterns for Web Service design and many of them are information overloaded.

5.2 Interpretation of Results

We compare three modeling tools (BPMN, UML and ContextMaps) to see their individual features of mapping Web Services.

BPMN Modeling: As the BPMN.Org said the purpose of BPMN is to provide a notation that is readily understandable by all business users and technical users. Therefore, it maps an abstract level flow. According to the current specification 1.0, we believe that there are ambiguities when the BPMN process maps to detail BPEL activities, because current BPMN didn't provide a notation to model detailed BPEL activities and some mapped processes in BPMN can associate with more than one BPEL activity. If Web Services are very complex, more individual maps are needed, and the validation of consistency is a problem.

UML Modeling: First, there are not enough notations to map Web Services such as to map BPEL activities and their nest activities, the relationship between WSDL and BPEL. Second, different views use different mapping, and therefore raise questions of consistency and information overload. Third, the mapping can't be further processed such as to couple or decouple different maps, to analyze the map.

ContextMaps: See the previous "ContextMap schemata for BPEL" (Table 4-4). We find the weakness of ContextMaps appear only for mapping the BPEL activities, and their inner activities. Actually, this is not a real problem. The densities of integrated activities are adjustable. In our schema of Web Services, we just divide all 15 activities into four sets;

Sets {sequence, flow} include activities {sequence, flow}

Sets {pre-condition} include activities {wait, switch, while, pick, scope}.

Sets {BPEL Activities} include activities {receive, reply, invoke, assign}.

Sets {Post-condition} include activities {throw, compensate, terminate, empty}.

Our ContextMaps are pattern-able. By following our schema, a Web Service is easy to develop. Our schema can represent each relationship in Web Services. Our ContextMaps are pluggable. We can de-merge and merge the mapping. Our ContextMaps are process able. We can query and analyze different sets and keep system consistency. Our ContextMaps are executable. We can extract the related BPEL code and then check the mapping by XML syntaxes or run the BPEL code in BPEL engine to show the results.

Chapter 6 Conclusions and Future Work

6.1 Conclusions

The MDA standard requires a model that can model the abstract and detail information down to code levels. When this standard applies to Web Service design, the model should represent the user requirement, the language specifications of WSDL, BPEL, the relationships between WSDL and BPEL, and detail BPEL codes. Therefore, since they do so ContextMaps can model both service oriented and object oriented design and process modeling information.

We apply ContextMaps technology to SOA development which can benefited from its 4P (pattern-able, plug-able, process-able and perform-able) ability.

(1) Users can customize the ContextMaps nations to meet their particular requirements. For example, we can define "c" for a concurrent process and "j" for a join function.

(2) Designers can use the schema of ContextMaps to accelerate their Web Service Development and guarantee the quality of software design.

(3) Designers can use the function of process tool in the ContextMaps to ensure the consistency of Web Service design and to trace the process from the top general to very detailed by selecting the "degree of interest" view and to avoid information overloaded.

(4) Designers can use the executable function in the ContextMaps to generate the programming codes while providing full traceability.

The following table is our comparison among BPMN, UML and Contextmaps which is based on our above analyses.

No.	Features	BPMN	UML	ContextMaps	Comment
1	Service Oriented Mapping	Yes	part	yes	Abstract level design
2	Object Oriented Mapping	No	yes	yes	Detail process design
3	Customized Schema	No	No	yes	Notation and schema
4	Different Domain Mapping	Part	No	yes	Every domain in Web service (Different Actors, Systems...)
5	Consistency	Part	No	yes	Trace each relationship among the different domains, or each process and its sub-process
6	Plug-able	No	No	yes	Merger related mapping
7	Process-able	No	No	yes	Different views from Querying and calculating the complexity of each node and merger sub modules together
8	Pattern-able	No	No	yes	Provide pattern such as Web service pattern
9	Execute-able	No	No	yes	Run the mapping
10	Mapping Relationship (BPEL, WSDL and ...)	No	No	yes	Definition in WSDL, use in BPEL
11	Mapping levels of BPEL activities	No	little	almost	All top level activities and their nest activities

Table 6-1 Comparison of Three Tools of Modeling

6.2 Future Work

Future work for the ContextMaps might lie in the following main directions:

1) Model More Business Areas

As ContextMaps regards as a core of MDA, we need to enrich our notations and schema to model driven business integration such as finance, manufacturing, transportation and so forth.

2) Enhance XMI Function in ContextMaps

We need to enhance CONTEXT+ tools to transfer models between ContextMap models and other models such as UML2, EMF (Eclipse Modeling Framework) through XMI (XML Metadata Interchange). Thus, we can view and modify UML and EMF model in ContextMaps. Furthermore, we can benefit form the open source Integrated Development Environment (IDE) like Eclipse to manage the model form ContextMaps. Then, ContextMaps can share the user friendly interface of Eclipse.

3) Change the ContextMap Repository to Databases

As ContextMaps are affected by Microsoft Excel's limitations, we need to move the ContextMap repositories to the major database such as Oracle, SQL, and DB2 to enhance the performance. Besides, these databases already have the ability to transfer the XML file to the database and vice versa. Therefore, ContextMaps will be automatically XML based.

References

Web based sources and references

1. Business Process Modeling Notation (BPMN)

   http://www.bpmn.org/

2. Business Process Execution Language (BPEL)

   http://www-106.ibm.com/developerworks/library/ws-bpel/

3. XML Schema

   http://www.w3.org/XML/Schema

4. Web Services Description Language (WSDL) 1.1

   http://www.w3.org/TR/wsdl

5. Web Services and Service-Oriented Architectures

   http://www.service-architecture.com/

6. Model Driven Web Services Development using UML,XML,WSDL & BPEL4WS, SI-SE 2004, Sirdhar Iyengar

   http://www.ifi.unizh.ch/si-se/sise2004/Tutorial_Iyengar.pdf

7. Process Modeling Techniques

   http://pesoa.org/pages/Publications/Fachberichte022004/PESOA_TR_1-2004.pdf

8. Object Management Group / MDA

   http://www.omg.org/mda/

9. Collaxa BPEL4WS Tutorial June, 2004
10. Collaxa BPEL4WS Development Guide June, 2004
11. General Strategies Inc

    http://www.gen-strategies.com

12. Bean implementation

    http://www.jdon.com/idea/ejb/jboss.htm

13. J2EE JavaServer Pages Technology

    http://java.sun.com/products/jsp/

14. Eclipse UML2 Project

    http://www.eclipse.org/uml2/

15. OWL Web Ontology Language Overview

http://www.w3.org/TR/owl-features/

16. Resource Description Framework (RDF)

http://www.w3.org/RDF/

Printed Materials

a) Mark Birbeck, et al: Professional XML, 2nd Edition. ISBN 1861005059

b) Brogden, William B, "Java developer's guide to servlets and JSP", San Francisco, Calif. ; London : SYBEX, c2001

c) Xu Guo Zhu, "Due Diligence Strategy with Context Maps", Major Report, Concordia University 2002

d) Grady Booch, James Rumbaugh, Ivar Jacobson,"The UML User's Guide", Addision Wesley, 1998.

e) Wojciech M. Jaworski, "Context Maps: Conceptual Spreadsheets for Data and Knowledge", Warehousing 1995.

f) Wojciech M. Jaworski, "System Analysis and Design in the Classroom: InfoMAPs Teaching Factory", Modeling and Simulation Conference, Pittsburgh, Pa., May 3-4, 1990

g) Wojciech M. Jaworski, et al. (1987). The ABL/W4 methodology for system modeling. *System Research Journal* 4(1), 23-37

h) Wojciech M. Jaworski, et al. "Representing processes, schemata and templates with jMaps", Semiotica 125(1/3), 229-47, 1999.

i) Wojciech M. Jaworski, Michailidis A.A., "Recovery and Enhancement of System Patterns: InfoSochemata and InfoMaps", ATW '94, University of Massachusetts –Lowell, Lowell, Massachusetts, June 1994.

Appendices

User Manual/Tutorial: "Modeling of Web Services with Context+"

(A) Mode Section Functionality

2. Open the Microsoft Excel file "loan_flow_plus_new", click the icon "Context+" in the Excel tool bar. We can see the screen as the figure 3-2.

3. **"Visible Map"** to display the map without hidden parts

>Select the hidden map in Excel -> click "Visible Map" in the above figure 3-2-> display all the hidden rows and columns.

4. **"Select Sets"** to Select the specific sets for viewing

>Click "Select Sets" in the figure 3-2-> view the pop up the screen for set selections -> click the sets which you want to view -> display the map only with selected sets.

5. **"Select Maps"** to display all the available worksheets in the current workbook and some of these worksheets to perform "By Color" operation simultaneously. The result of selected maps will be opened in the new sheet with the sheet name "original sheet name + Result"

>Click "Select Maps" -> click the "by color" -> click "OK"-> view the result in the new sheet with the sheet name "original sheet name + Result"

6. **"Join Maps"** to merge one or more selected maps together in one sheet. The criteria: comparing the values of concept column within each selected worksheet, join the columns with the same values of concept. Or add additional rows with different values of concept. Then save the result into

separate sheet named “Merge Result”. This function can be implemented independently.

>Click “Join Maps”-> Select the maps for joining-> show the result in the new sheet with the name “Merge Result”

7. **“Add Atom”** to insert Row to all spreadsheet of the Active Workbook.

>Select the member-> click “Add Atom”->click “OK”

(B) Query Section Functionality

8. **“By Color”** - This button will be implemented with one of ‘AND’, ‘XOR’, ‘OR’, ‘NOT’ operations based on selected predefined - query. The predefined - query should be marked in three places in the Context Maps.

9. Concept Value cells under Concept Set column, as showing Predefined Query (1) in the following figure A-1.

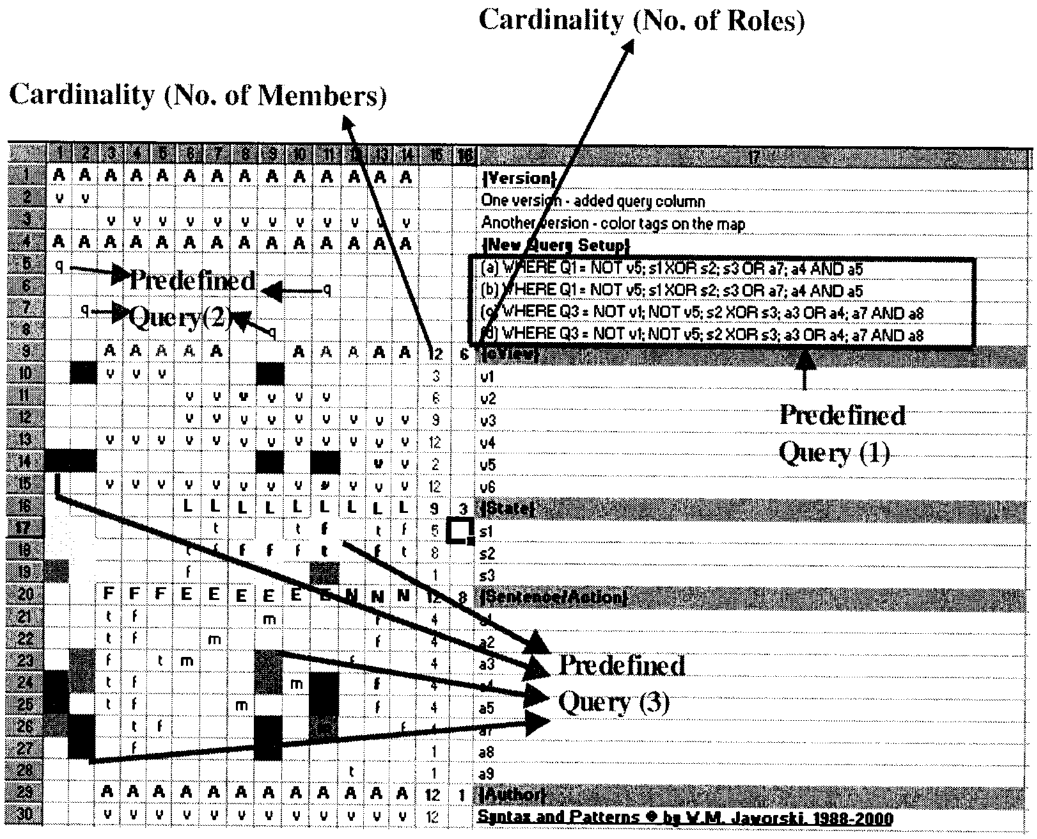
10. Cells with “q” value relevant to the Concept Value columns with predefined queries, as showing Predefined Query (2) in the following figure A-1.

11. Tagged cells with color background, Red stands for AND operation, Yellow stands for XOR operation, Bright Green stands for OR operation and Light Blue stands for NOT operation, as showing Predefined Query (3) in the following figure A-1. And all these colors are optional.

12. Select at least one predefined query as described above. If only one predefined query is selected, it will ignore the selected ‘AND’ or ‘XOR’ or ‘OR’ or ‘NOT’ operations from the QUERY Form, and only implement the

predefined query. If multiple predefined queries are selected, it will implement each predefined query first, then it will merge the result based on the selected 'AND' or 'XOR' or 'OR' or 'NOT' operations. Detailed implementation is as following:

>Predefined queries in the Context Map-> having tagged the cells for the queries to implement->click one of the 'AND', 'XOR', 'OR', 'NOT' operations->click "By Color" button->click 'OK'.



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
1	A	A	A	A	A	A	A	A	A	A	A	A	A	A			{Version}
2	v	v															One version - added query column
3			v	v	v	v	v	v	v	v	v	v	v	v			Another version - color tags on the map
4	A	A	A	A	A	A	A	A	A	A	A	A	A	A			{New Query Setup}
5	q																(a) WHERE Q1 = NOT v5; s1 XOR s2; s3 OR a7; a4 AND a5
6											q						(b) WHERE Q1 = NOT v5; s1 XOR s2; s3 OR a7; a4 AND a5
7		q															(c) WHERE Q3 = NOT v1; NOT v5; s2 XOR s3; a3 OR a4; a7 AND a8
8									q								(d) WHERE Q3 = NOT v1; NOT v5; s2 XOR s3; a3 OR a4; a7 AND a8
9			A	A	A	A	A			A	A	A	A	A	12	6	{eView}
10		■	v	v	v				■						3		v1
11						v	v	v	v	v	v				6		v2
12						v	v	v	v	v	v	v	v	v	9		v3
13			v	v	v	v	v	v	v	v	v	v	v	v	12		v4
14	■	■							■		■		v	v	2		v5
15			v	v	v	v	v	v	v	v	v	v	v	v	12		v6
16						L	L	L	L	L	L	L	L	L	9	3	{State}
17							t			t	f		t	f	5	□	s1
18						t	f	f	f	f	t		f	t	8		s2
19	▨					f					▨				1		s3
20			F	F	F	E	E	E	E	E	E	N	N	N	12	8	{Sentence/Action}
21			t	f					m				f		4		a1
22			t	f			m						f		4		a2
23		▨	f		t	m			▨			f			4		a3
24	■	▨	t	f					▨	m	■		f		4		a4
25	■		t	f				m			■		f		4		a5
26	▨	■		t	f				■		▨			f	4		a7
27		■		f					■						1		a8
28												t			1		a9
29			A	A	A	A	A	A	A	A	A	A	A	A	12	1	{Author}
30			v	v	v	v	v	v	v	v	v	v	v	v	12		Syntax and Patterns ◆ by V.M. Javorski, 1988-2000

Figure A-1 Sample for Query by Color

13. **"AND"** –select at least two cells from different rows to generate corresponding map which remains all the not empty columns within these rows.

> Tag the 2-n selected cells by holding CTRL key-> click "AND"->click "OK".

"XOR" - select at least two or multiple cells from different rows to generate corresponding map, which remains the exclusive not empty columns within these rows.

> Tag the 2-n selected cells by holding CTRL key-> click "XOR"->click "OK".

14. **"OR"**- select at least one or multiple cells from different rows to generate corresponding map which retains the each of the not empty columns from within these rows.

> Tag the 1-n selected cells by holding CTRL key-> click "OR"->click "OK".

15. **"NOT"**- select only one cell to retrieve negation columns and generate corresponding map.

> Tag the only single cell-> click "NOT"->click "OK"

(C) Output Section Functionality

16. **"Schema"**- Extract the schema from the map. The schema of Web service is our patterns of Web service. See the table 4-2.

>Click any member->click "OR" in "Query" section->click "schema"->click "OK".

17. **"Cardinality"** – Display the number of not empty roles of each row and the number of sub-concept value of each Concept set. It helps us to see the

complexity of each set.

>Click any member->click "All" in the "Query" section -> click "cardinality" ->click "OK".

18. **"ApplyColor"-** apply different color in each cell for the spreadsheet based on the following criteria. First, the font of the whole row is bolded and the light gray color is used as background if there is a concept value. Second, the values of cardinality, two columns located before the column of concept. Third, the different roles apply different colors. For example, "A"—Dark Gray, "v"—Light Gray, and "N"—Rose.

>Select the map-> click "ApplyColor"->click "OK"

19. **"Map", "Graph",and "Map & Graph"** – at present, all outputs are in Context Map formats. But graph format is recommended for future work.

(D) Output Section Functionality

20. **"Group (Row and Col)"**- Save the display space, each range of group is based on each range of set.

21. Select any member in the active map-> click "Row" or "Col" -> click "OK".

22. **"Ungroup (Row and Col)"**- View the whole map in ungroup way.

23. Select any member in the active map-> click "Row" or "Col" -> click "OK"

24. **"Help"-** Display (this) Help spreadsheet.

25. Click "Help"

26. **"XML->Map"**- transfer any XML what is syntax error free to ContextMaps.

>Select the sheet named "Schema" -> open the "CONTEXT+" and click "XML->Map" -> Select the XML file -> click "go" -> the XML file be shown I the "Result" sheet.

27. **"Map-> XML"-** Transfer the XML of ContextMaps to XML file

>Select the sheet with XML mapping -> open the "CONTEXT+" and click "Map-> XML" -> select the directory and XML file name-> click "OK".

28. **"Map-> BPEL"** –Transfer the BPEL mapping to BPEL file.

> Select the sheet with BPEL mapping -> open the "CONTEXT+" and click "Map-> BPEL" -> select the directory and BPEL file name-> click "OK".

Glossary

BPEL Business Process Execution Language (BPEL) that is describe the Business process behavior.

BPMN Business Process Modeling Notation (BPMN) that creates a standardized bridge for the gap between the business process design and process implementation.

ContextMaps An array based language and based on the conceptual graphs that are defined by an aggregation of context tuples. A Context tuple is a generic association of set members cast in roles. ContextMaps represent knowledge assets with 4P (process-able, plug-able, pattern-able and perform-able).

CONTEXT+ A tool in the ContextMaps to extract the information from ContextMaps.

CWM The purpose of OMG's Common Warehouse Metadata Initiative (CWMI) is to enable easy interchange of metadata between data warehousing tools and metadata repositories in distributed heterogeneous environments.

EMF Eclipse Modeling Framework (EMF) is a modeling framework and code generation facility for building tools and other applications based on a structured data model. From a model specification described in XMI, EMF provides tools and runtime support to produce a set of Java classes for the model, a set of adapter classes that enable viewing and command-based editing of the model, and a basic editor. Models can be specified using annotated Java, XML documents, or modeling tools like Rational Rose, then imported into EMF.

MDA The Model-Driven Architecture (MDA) starts with the well-known and long established idea of separating the specification of the operation of a system from the details of the way that system uses the capabilities of its platform.

MOF The Meta-Object Facility (MOF) Specification defines an abstract language and a framework for specifying, constructing, and managing technology neutral metamodels. A metamodel is in effect an abstract language for some kind of metadata.

PIM Platform Independent Models.

PSM Platform Specific Models.

SOA Service Oriented Architecture (SOA) is a world wide mesh of collaborating services, which are published and available for invocation on the Service Bus.

SOAP Simple Object Access Protocol (SOAP) that is used for binding communication protocols and carrying the message to transmit in the Web environment.

UDDI Universal Description, Discovery, and Integration (UDDI) that is used for registering Web Services and their locations.

UML The Unified Modeling Language (UML), a language for visualizing, specifying, constructing, and documenting the artifacts of a software-intensive system.

UML2 is the latest UML version in the process being released.

WSDL Web Services Description Language (WSDL) that is a interface between external and internal operation.

XML The Extensible Markup Language that describes a class of data objects called XML documents and partially describes the behavior of computer programs which process them.

XML Schema XML Schema makes it possible to describe the type and structure of XML documents.

XMI the XML Metadata Interchange Format (XMI) specifies an open information interchange model that is intended to give developers working with object technology the ability to exchange programming data over the Internet in a standardized way, thus bringing consistency and compatibility to applications created in collaborative environments.