A Novel Approach for Dynamic Service Integration using Business Logic Property Evaluation System

Thirumaran.M
Pondicherry Engg. College
Pondicherry

Dhavachelvan.P
Pondicherry University
Pondicherry

Aranganayagi.G
Pondicherry Engg. College
Pondicherry

ABSTRACT

The rapid progression of network technology and vibrant market demand constrains enterprises to collaborate and interact with their business partners in a coordinated and well-organized fashion. The demand also extends to have a secure atmosphere to share and integrate information such as data, service logics, etc. with their partners to achieve their target goals. Integrating service logics from diverse enterprises is not an easy task. It requires developers to understand the business logic entirely and in more important, there should be a system to confine the enterprises to access the shared resources as stated in the Service Level Agreement. There is no standard model to carry out these necessities. The model proposed in this paper integrates the service logics dynamically without breaching SLA. Here Enterprise Service Bus (ESB) is employed to share the resources between the organizations. Source control management system in the model facilitates enterprises to carve up the resources as stated by the SLA. Integration layer integrates the service logics dynamically though the dedicated matching module, Functionality Analyzer and BPEL engine. This allows enterprises to expose and integrate the service logics flexibly and reliably.

General Terms

Data sharing, Web based service, B2B collaboration, B2B integration, Security

Keywords

Service integration, B2B integration, B2B collaboration, Web service, Business Logic Model, Enterprise Service Bus(ESB).

1. INTRODUCTION

In the present-day globalised, networked economy, efficient information sharing and tight collaboration with suppliers and partners across the supply chain are critical to the success of a company’s business. The automated business interactions between suppliers and trading partners allows companies to better share information and optimize processes across the value chain from better demand forecasting to streamlined manufacturing. Web service enables companies to collaborate closely with their business partners and share needed resources such as information and business logic through technology called service integration. Dynamic service integration requires secure and reliable infrastructure for resource sharing in such a way interoperability between the services should be managed while integrating service logic at runtime. The purposed system integrates the services dynamically to share the Business logic securely and reliably as stated by the SLA. It requires checking authorization, granting to access the resource dynamically as per the contract, monitoring the resources while sharing and delivering the necessary information to the possessor of the service to cope up the changes made in the service. This is consummated by adding Source Control Management System and Service Integration to the ESB layer. ESB provides an enterprise platform that implements standardized interfaces for communication, connectivity, transformation and security. But, in order to protect each partner’s own interests, security strategies must be developed and integrated in the service environment. Unfortunately, traditional security approaches deal with security concerns from a technical perspective (i.e. data transmission or authentication, etc.) and do not support end-to-end security in a distributed environment of business services and collaborative processes. In this paper, we attempt to improve end-to-end security by annotating service descriptions with security objectives used to generate convenient Quality of Protection Agreements between partners. Conversely, agreements are processed by a dedicated matching module called Functional Analyzer in Integration layer with respect to security requirements stated by the SLA. In addition to this, we need a mechanism to monitor the resource while sharing to adapt the modifications done by the developers.

Source Control Management tracks the modification and facilitates impact analysis between the existing and modified services that ensures computability criteria. The source control management system allows us to see the historical background behind the changes made to the business logic of the web services. This helps the developers to see where the changes have been progressively made and include or remove the change as per the need. In addition to these all, prior to the integration functionality level assessment is necessary to achieve interoperability. For this, property cube is developed which contains necessary properties to be evaluated to suggest diverse ways to integrate and to list out various actions can be carried out with the shared services. After evaluating the service for each property, BPLS schema is generated with necessary information for each property and service logic. Theorem for each property in the property cube is described briefly. While integration, BPLS schema is evaluated and directs the system to integrate as interoperability is managed. We presented implementation for Service Integration by evaluating properties using GlassfishESB based on Netbeans with J2EE technology.

The rest of the paper is organized as follows. Section 2 describes the works in research field related to our work. Section 3 details the design and operation of the proposed model. Section 4 describes the property evaluation methodology. Section 5 presents concluding remark.

2. RELATED WORKS

There are plenty of publications in the area of using Web Services to support B2B, service and application integration since this area has attracted researches from various research institutes.

Nassim LAGA, Emmanuel BERTIN in a paper, defines and implements new mechanisms that enable a seamless integration of a service in the end-user pervasive environment. The peculiarity of their approach is that they have defined, and implemented at the end-user device, a distributed mechanism that automatically detects compatible functionalities and link them each other's even if they are running on different devices. Consequently, end-users can for instance read emails on the mobile and play attached movies on the Laptop, or they can search addresses on the a directory on a laptop and display the locations of the results on a Map service on the mobile. [1]. Lu Liu et al present an innovative architectural approach which is able to proactively self diagnose and self-adapt evolution occurring in provision of search and rescue capabilities in service-oriented P2P (ServP2P) networks. This architecture provides flexibility and agility in handling dynamic changes and evolution encountered in the delivery of rescue capability. The experimental results indicate that the proposed approach provides a high-level of reliability and sustainability to achieve dependable service integration for capability provision [2]. YAO Qiong, SUN Peng in their paper, driven by high-quality services integration, the networking technologies are studied to identify a networking method to support the integration of high-quality services including high-definition video applications. Above on this, an overall architecture of home network is built. The novel architecture can support multiple high-quality services integration and guarantee the end-to-end QoS between devices in home network. Finally, the open characteristic of the architecture is discussed [3].

Marko Banek proposes a strategy for enabling a broader use of e-business in Croatian tourism. The core of the strategy is integration of tourist services and products enabled by a national tourism portal. The portal will be used by service and product providers, which will register their services/products and offer them publicly to other providers and customers. In this way, complex integrated services of a high quality can be developed for the customers, to the mutual benefit of the customers and all providers. The portal is based on the SOA architecture and the Web service technology and hence platform independent. Two particular aspects of their portal are: using the available data for business analysis, and semantic-aware service browsing [4]. Junichi Toyouchi, Motohisa Funabashi for realizing the expectations from the view of middleware technology, a service integration platform with dynamic brokering and flow control mechanisms is presented with a prototype system that shows satisfying primal requirements. The major issue for applying the platform to real applications is to build the application development environment, especially a profile template definition tools, matching policy definer, and etc.. By analyzing requirements of the possible applications, they continue to improve the development environment for OSCAR platform [5].Ka Cheuk WU presented an integrated and personalized tour planning portal based on Semantic Web service technologies for knowledge management which employs ontology to classify and manage service [6]. Camlon H. Asuncion presented an innovative approach to increase the flexibility of integration solutions in the context of service mediation by separating the more dynamic aspects of the requirements as business rules while keeping the more stable parts in the business process. In his approach, initially the requirements are specified as goal models, over time it is represented in terms of business rules. These business rules are then made executable by exposing them as Web services and incorporating them into the design of the business process[7]. Olga Levina[8], suggests an extraction process for business rules identification from business process models. Applying this process introduces a structured approach and management aspects within rules discovery by focusing on rule sources that are important for the process goal and providing a rule structure. Mohammed Alawairdhi[9], suggests a business-logic-based framework for evolving software systems is proposed. The goal of the framework is evolving software in a higher abstract layer. Olegas Vasilecas[10], suggests The rules are expressed in several models, there is a risk that overall specification is inconsistent. Unambiguous models are crucial for the successful implementation of IS models transformation and finally code generation tasks. Therefore it is necessary to check consistency among related rules models. The problem of models inconsistency can be solved by using of formal or partially formal models with constraints. According to Anis Charfi[11], the Business Rules Group , a business rule is a statement that defines or constrains some aspect of the business. It is intended to assert business structure or to control the behavior of the business. Business rules are usually expressed either as constraints or in the form if conditions then action. The conditions are also called rule premises. The business rule approach encompasses a collection of terms (definitions), facts (connection between terms) and rules (computation, constraints and conditional logic). Terms and Facts are statements that contain sensible business relevant observations, whereas rules are statements used to discover new information or guide decision making. Michael zur Muehlen[12], suggests a business rule is a statement that aims to influence or guide behavior and information in an organization. Business rules can be categorized in accordance to their source or structure: Mandates, Policies, and Guidelines. Jose F. Mejia Bernal [13], proposed a way for representing Dynamic Business Process in terms of Rules based on patterns identification. With this approach it is easy to apply on a business process instance both user-based personalization rules and automatic rules inferred by an underlying context-aware system.

3. PROPOSED MODEL FOR WEB SERVICE INTEGRATION

Fig 1 demonstrates proposed model for service integration. The proposed architecture in figure 1 exemplifies how the system automatically integrates the services across the enterprises securely. ESB as a mediator, associates different Service Provider and integrates the services dynamically through the components such as Message Broker, Intelligent Router, security and communication handler.

Security Component in ESB provides message integrity, confidentiality, and authentication while sharing the services. Message integrity is afforded by leveraging XML Signature and security tokens to ensure that messages have originated from the appropriate sender and were not modified in transit.

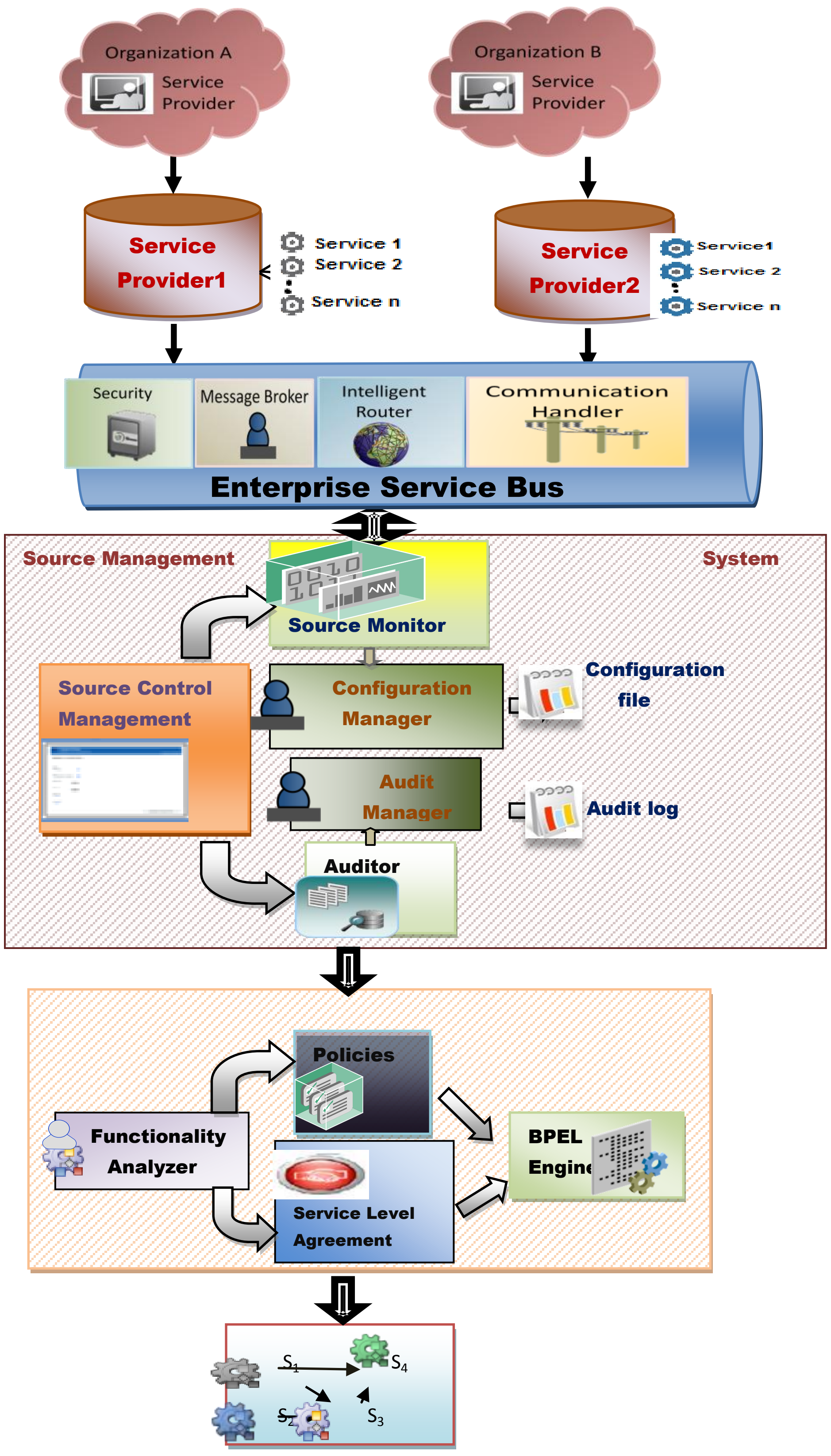


Fig 1. Proposed Model For Dynamic Service Integration

Similarly, message confidentiality leverages XML Encryption and security tokens to share the services confidentially. It delivers the authenticated request to the Message Broker. Message Broker mediates communication amongst applications, allows business information to flow between disparate applications across multiple hardware and software platforms which validates the received request and identifies sender and receiver, transforms the request in to standard format and routes the formatted request to Intelligent Router. Business rules can be applied to the data flowing through the message broker to route and transform the information.

Intelligent Router seeks to route messages, not by a specified destination, but by the actual content of the message itself. In a typical application, a message is routed by opening it up and applying a set of rules to its content to determine the parties interested in its content. This is the main component adds dynamicity to ESB. It identifies the required service derived from the request and passes this information to Communication Handler.

Communication Handler facilitates sharing the services between organizations as stated by the agreement. It provides platform for Source Control Management System and Service Integration layer to monitor the resource while sharing.

Source Control Management system allows changes in service during runtime, enhances productivity and boosts application quality as it automatically versions files, labels and organizes them as they change during the development process. Also it enables parallel development and allows us to see the historical background behind the changes made to the Business logic services. It accomplishes these services through Source Monitor, Configuration Manager, Auditor and Audit Manager.

Source Monitor monitors the service while sharing and allows the developers to make changes, which does not affect the coherence between the existing and the modified services. This retains the crux of the web service and does not change the functionality of the service.

Configuration manager manages the changes made in the services, defines mechanisms for managing different versions, focuses on establishing and maintaining consistency of a system's performance and stores the needed information in the Configuration file. This helps the possessor of the service to see where the changes have been progressively made and include or remove the changes.

Auditor examines the modified service, identifies modifications done in the service, analyzes the risk and sends the information to the Audit Manager. Audit manager reviews, assesses and prepares report in audit log holds information for the Service Provider to track and manage the changes.

Functional Analyzer provides an infrastructure for SLA automation system which offers services such as automatic SLA creation, SLA monitoring and control. In runtime, it monitors the resource while sharing and gives accession as stated.

SLA (Service Level Agreement) is an xml document that defines mutual understandings and expectations between the service providers to share the services. It contains information such as purpose of the agreement, parties involved in SLA and their respective roles, validity period, scope, restrictions, service level objectives, penalties, etc..,

Policy is also an xml document that defines various constraints for merging, composing, substituting the service logic to achieve interoperability goals.

BPEL Engine integrates the services as stated by the SLA in such a way interoperability between the services is managed and it offers a comprehensive and easy-to-use infrastructure for creating, deploying and managing business processes. It is XML-based language and has a very restrictive framework employing a handful of operations that guide state transitions among a Collection of loosely-coupled services.

4. PROPERTY EVALUATION SYSTEM

Functionality analyzer in the model analyzes the service logics with various QOS attributes and ascertains various actions can be carried out with the logic Fig 3 depicts the properties which are evaluated by the functionality analyzer. In this section, evaluation technique of each property is discussed briefly.

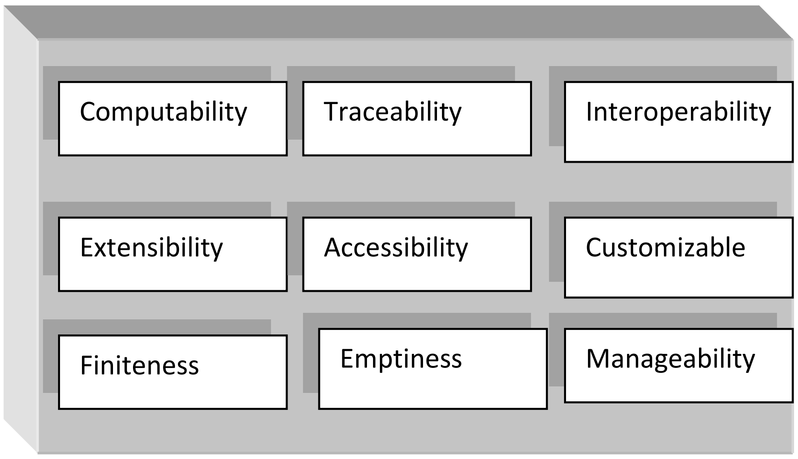


Fig 2. Property Cube for Service Integration

4.1. Computability

Theorem 1: Let Business logic BL is a collection of Business Rules (BR), Business function (BF) and parameters (P). If elements in the set are computable, then BL is computable.
Proof:

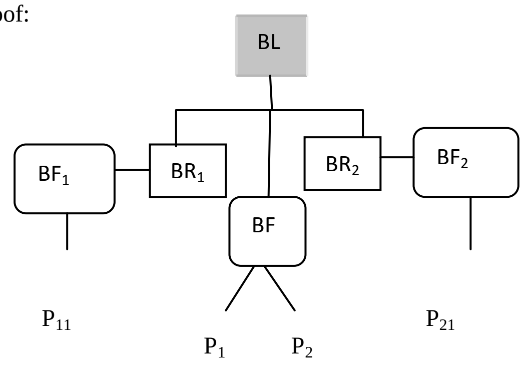


Fig 3. Computability

BL=> { [BR_1,BR_2], [BF]}, BR_1=>{BF_1}, BR_2=>{BF_2}, BF=>{P_1,P_2}, BF_{R1}=>{P_{11}}, BF_2={P_{21}}

It shows parameters P_1, P_2, P_{11} and P_{21} are computable for all values . So parameters returned from the functions BF, BF_1 and BF_2 are also computable. As functions BF_1 and BF_2 are computable, Business rules BR_1 and BR_2 are computable. As all parameters, functions and rules are computable, BL is computable.

4.2. Traceability

Theorem 2: BL is said to be traceable if every element of BL is traceable.

Proof:

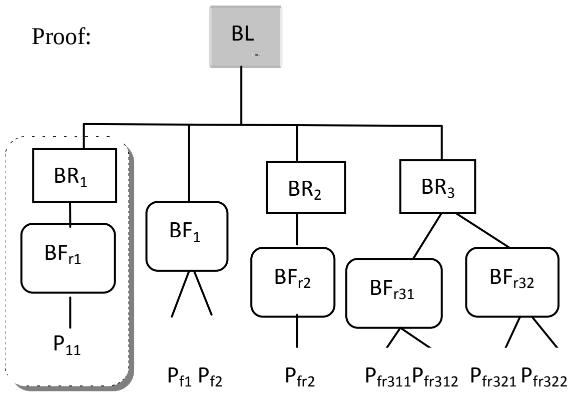


Fig 4. Traceability

BL=> { [BR_1,BR_2, BR_3] [BF_1]}
BR_1=>{BF_{r1}}, BR_2=>{BF_{r2}}, BR_3=>{BF_{r31}, BF_{r32}}
BF_1=>{P_{f1},P_{f2}},
BF_{r1}={P_{rf11}}, BF_{r2}={P_{f2}} BF_{r31}=>{ P_{311} ,P_{312}} BF_{r32}=>{ P_{321}, P_{322}}
Here Parameter P_{rf11} is traced out by tracing business rule BR_1 and $BF_{r1.}$ Similarly all rules, functions and parameters can be traced out. Thus Business logic BL is traceable.

4.3. Interoperability

Theorem 3: Let BL=> { [BR_1,BR_2] [BF_1]}, BL is interoperable if a new rule or function can be added to the set without altering existing set.

Proof:

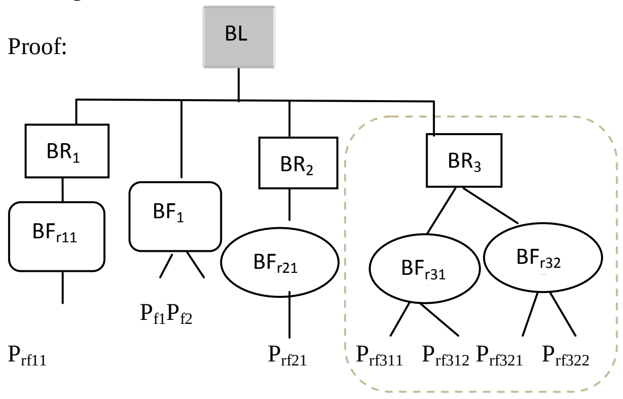


Fig 5. Interoperability

BL=> { [BR_1,BR_2] [BF_1]}BL=> { [BR_1,BR_2, BR_3] [BF_1]}
BR_1=>{BF_{r1}}, BR_2=>{BF_{r2}},BF_1=>{P_{f1},P_{f2}},
BR_1=>{BF_{r1}}, BR_2=>{BF_{r2}}, BR_3=>{BF_{r31}, BF_{r32}}
BF_1=>{P_{f1},P_{f2}}BF_{r11}={P_{rf11}}, BF_{r2}={P_{rf21}} BF_{r11}={P_{rf11}},
BF_{r2}={P_{rf21}}BF_{r31}=>{ P_{rf311} ,P_{rf312}} BF_{r32}=>{ P_{rf321}, P_{rf322}}
Here new rule BR_3 with function BF_{r31}, BF_{r32} is added to BL without disturbing exiting set BL=> { [BR_1,BR_2] [BF_1]}.

4.4. Extensibility

Theorem 4: If BF ∈ BL and BF is extensible, then BL is also extensible.

Proof:

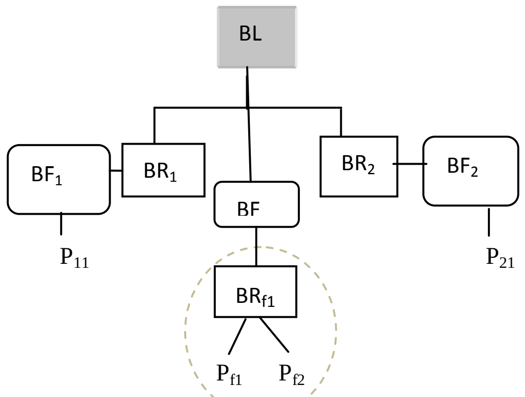


Fig 6. Extensibility

BL=> { [BR_1,BR_2] , [BF]}
BR_1=>{BF_1}, BR_2=>{BF_2}, BF=>{ BR_{f1}}
BR_{f1}=>{P_{f1},P_{f2}}, BF_1={P_{rf1}}, BF_2={P_{rf2}}
Business function BF is extended by adding new rule BR_{f1} in the fig . As it doesn't change the existing structure of BL, BL can be extensible.

4.5. Accessibility

Theorem 5: BL is accessible, if it is authorized to access the logic according to the agreement.

Proof:

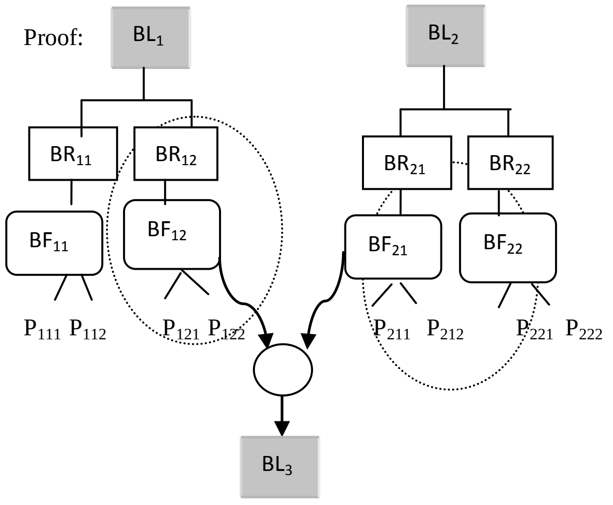


Fig 7. Accessibility

BL_1=> { BR_{11},BR_{12}}, BL_2=> {BR_{21},BR_{22}}
BR_{11}=>{BF_{11}}, BR_{12}=>{BF_{12}}, BR_{21}=>{BF_{21}},
BR_{22}=>{BF_{22}}BF_{11} =>{ P_{111} , P_{112}}, BF_{12}=>{ P_{121} , P_{122}},
BF_{21}=>{ P_{211} , P_{212}}, BF_{r4} =>{ P_{221} , P_{222}} BL_3=> {
BR_{12},BR_{21}} BR_{12}=>{BF_{12}}, BR_{21}=>{BF_{21}}
BF_{12}=>{ P_{121} , P_{122}}, BF_{21}=>{ P_{211} , P_{212}}

BL_1 and BL_2 contain two branches, but BL_3 is authorized to access only one branch of each according to the contract. So BL_3 can access as shown in fig.

4.6. Customizable

Theorem 6: BL is customizable if modification can be done according to client's request.

Proof:

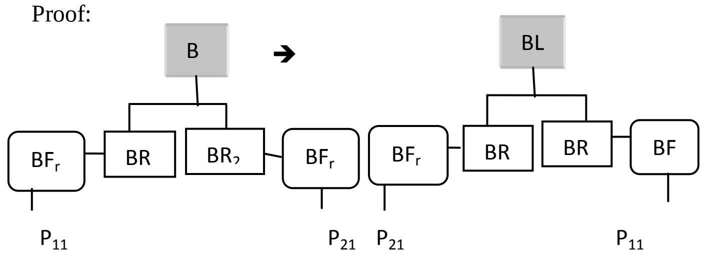


Fig 8. Customizable

Original logic	**Modified Logic:**
BL=> { BR_1,BR_2}	BL=> { BR_2,BR_1}
BR_1=>{ BF_{r1}}	BR_2=>{ BF_{r2}}
BR_2=>{ BF_{r2}}	BR_1=>{ BF_{r1}}
BF_{r1}=> P_{fr1} ,	BF_{r2}=> P_{fr2}.
BF_{r2}=> P_{fr2} ,	BF_{r1}=> P_{fr1}.

Thus the business logic can be modifiable according to client's request.

4.7. Finiteness

Theorem 7: BL is finite if number of components in BL such as rules and functions are finite.
Proof:

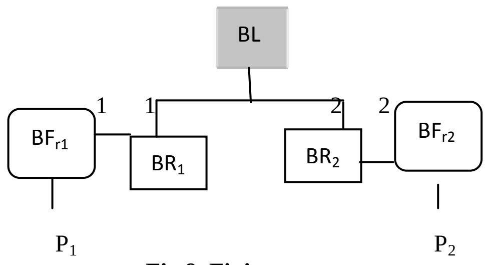


Fig 9. Finiteness

Number of rules and functions in the logic is finite, and then BL is finite.

4.8. Emptiness

Theorem 8: BL sets empty property if requested rule or function is not in the existing logic BL.
Proof:

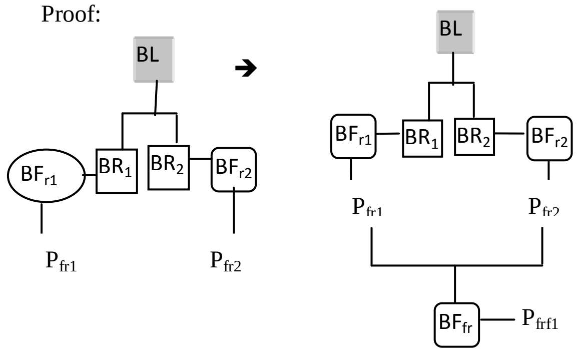


Fig 10.Emptiness

BL=> { BR_1,BR_2}	BL=> { BR_1,BR_2}
BR_1=>{ BF_{r1}} BR_2=>{ BF_{r2}}	BR_1=>{ BF_{r1}} BR_2=>{ BF_{r2}}
BF_{r1}=> P_{fr1} , BF_{r2}=> P_{fr2}.	BF_{r1}=> P_{fr1} , BF_{r2}=> P_{fr2}.
	{P_{fr1}, P_{fr2}}=>{ BF_{rf1}}
	BF_{r1}=> P_{frf1}.

Function BF_{fr1} is needed to compute the result returned by BF_{r1} & BF_{r2}. As it is not in BL, it needs to be added newly.

4.9. Manageability

Theorem 9: Business logic is manageable if it produces the final result with in a time Limit.

Proof:

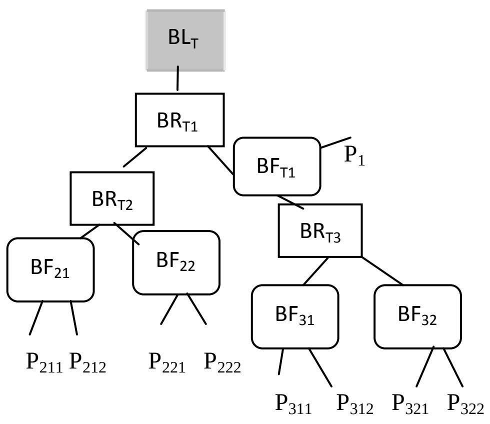


Fig 11. Manageability

BL_{T1}=> {BR_{T1}}
BR_{T1} => {[BR_{T2}],[BF_1]}
BR_{T2} =>{ BF_{21}, BF_{22}} BF_{T1}=> {[BR_{T3}], [P_1]}
BF_{21}=> {P_{211},P_{212}} BF_{22}=> {P_{221}, P_{222}} BR_{T3}=>{ BF_{31}, BF_{32}}
BF_{31}=> {P_{311}, P_{312}} BF_{32}=> {P_{321} , P_{322}}
Processing time of whole logic can be managed by managing time for each component of the logic.

5. IMPLEMENTATION METHODOLOGY & EVALUATION RESULTS

Here we developed tour fare enquiry service by integrating hotel fare enquiry service and travel fare enquiry service. The fare enquiry service logic of hotel reservation and travel reservation are composed together after evaluating various properties discussed above. BPEL diagram of composed service is depicted in Fig 2. The integrated service is evaluated with various inputs, response time and execution time are measured and plotted graphically in fig.3.

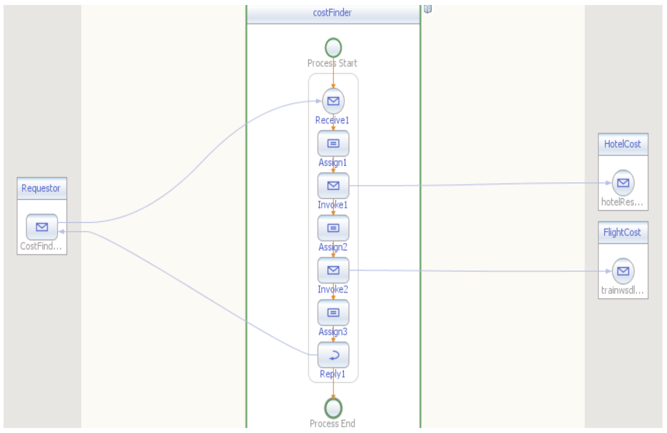
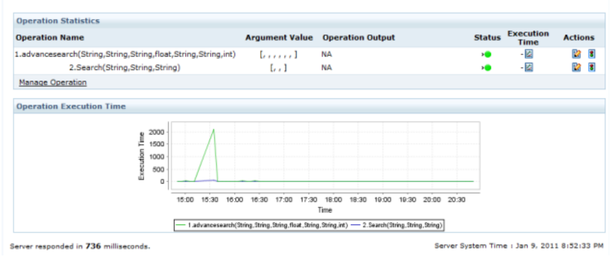


Fig 12. Tour reservation service: Composition of hotel & travel reservation



Operation Statistics

Operation Name	Argument Value	Operation Output	Status	Execution Time	Actions
1.advancesearch(String,String,String,float,String,String,int)	[, , , , , ,]	NA	●	-	
2.Search(String,String,String)	[, ,]	NA	●	-	

Manage Operation

Operation Execution Time

Server responded in 736 milliseconds.

Server System Time : Jan 9, 2011 8:52:33 PM

Fig 13. Evaluation result of integrated service

6. CONCLUSION

The paper presents a novel approach to share and integrate the service logic securely and flexibly. ESB facilitates enterprises to share the service logics securely and Source Control Management system assists to access the service logic and restrains as per SLA. Functionality analyzer in integration layer evaluates the logic with various properties and helps to integrate the service logic as interoperability between the service logic is managed. This helps enterprises to integrate their service logics with their business partners securely and reliably.

7. REFERENCES

[1] Lu Liu, Jie Xu, Duncan Russell, KP Lam, Zongyang Luo, Kaigui Wu, Dave Collins "Dependable Dynamic Service Integration on Service-Oriented Peer-to-Peer Networks", 2009 First International Conference on Advances in P2P Systems.

[2] Network YAO Qiong, SUN Peng and NI Hong, "Research on High-quality Service Integration Oriented Network Technology and Architecture in Home", International conference on Advanced computer theory and Engineering, 2010.

[3] Marko Banek, Damir Juri, Damir Pintar, Zoran Skoir, Mihaela Vrani and Boris Vrdoljak, "E-business infrastructure for supporting the integration of tourist services", 50th International Symposium ELMAR, 2008.

[4] Junichi Toyouchi and Motohisa Funabashi, "Development of Service Integration Platform for One-stop Service Applications", IEEE conference, 2001.

[5] Liu Yong, "Study on geography information service semantic integration method based on business template", International Conference on Computer and Communication Technologies in Agriculture Engineering, 2010.

[6] Ka Cheuk WU and Dickson K.W. Chiu, "Toward Tourist Service Integration and Personalization with Semantic Web Services: A Case Study in Hong Kong", IEEE International Conference on e-Business Engineering, 2008.

[7] Camlon H. Asuncion, Maria-Eugenia Iacob and Marten J. van Sinderen, "Towards a flexible service integration through separation of business rules", 14th IEEE International Enterprise Distributed Object Computing Conference, 2010.

[8] Zhuoren Jiang, Yan Chen and Ming Yang, "A research on multi-layer structure for dynamic service integration", IEEE international conference, 2010.

[9] Mohammed Alawairdhi, "*A Business-Logic Based Framework for Evolving Software Systems*", 2009 33rd Annual IEEE International Computer Software and Applications Conference.

[10] Olegas Vasilecas, "*Ensuring Consistency of Information Systems Rules Models",* the International Multi conference on Computer Science and Information Technology, 2008.

[11] Anis Charfi, *"Hybrid Web Service Composition: Business Processes Meet Business Rules", ICSOC'04*, November 15–19, 2004, New York, New York, USA.

[12] Michael zur Muehlen, *"Business Process and Business Rule Modeling Languages for Compliance Management: A Representational Analysis",* Twenty-Sixth International Conference on Conceptual Modeling - ER 2007 - Tutorials, Posters, Panels and Industrial Contributions, Auckland, New Zealand.

[13] Jose F. Mejia Bernal, *"Dynamic Context-Aware Business Process: A Rule-Based Approach Supported by Pattern Identification",* SAC'10, March 22-26, 2010, Sierre, Switzerland.