Evaluating Service Business Logic using Finite State Machine for Dynamic Service Integration

Thirumaran.M
Pondicherry Engg. college
Pondicherry

Dhavachelvan.P
Pondicherry University
Pondicherry

Aranganayagi.G
Pondicherry Engg. College
Pondicherry

ABSTRACT
Dynamic business environment drives enterprises to work more closely, flexibly and carve up resources with their business partners to provide comprehensive, efficient and customized web services. This demand a mechanism to integrate the service logics from diverse system by scrutinizing the dependency exist on the service logics. To ascertain the dependency between the service logics, developers need to comprehend the whole service logics and must identify correct way to integrate them. It puts developers in bottleneck. The framework proposed in this paper discovers required service logics, ascertains the dependency between the service logics and integrates them dynamically. It employs FSM to recognize the dependency relation subsists on located logics. The system studies the logic flow through FSM and determines dependency relation exist on business rules, functions and parameters. From the resolved dependency relation, it decides proper way for integration. Integration adapter in the framework integrates the service logics in run time through the revealed style. FSM is also exploited to measure the quality parameters of the integrated service through the property evaluator. Thus this ascent to integrate the service logics robotically without developer's intercession at any stage.

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

Keywords
Service integration, B2B integration, B2B collaboration, Web service, Finite State Machine (FSM).

1. INTRODUCTION
In the current networked business environment, most information systems need to interoperate with other internal and external information systems to carry out the necessities. In this web service has a great demand to correlate and work with other services. Enterprise Application integration (EAI) is the process of linking application in one organization with other in order to simplify and automate business processes to the greatest extent, while at the same time avoiding having to make sweeping changes to the existing applications or data structures. EAI is the "unrestricted sharing of data and business processes among any connected application or data sources in the enterprise". But the current demand is not delighted with application integration, it insists to integrate the web services in rule, functional or logic level. Integrating web services in this level leads to exponential problem in syntactic and semantic level. To tackle this, developers must figure out the whole service and must ascertain proper mode to integration which is complex and time consuming task. So the present focus is to have a computerized system to integrate the service logics automatically for the given requirement. The proposed framework in this paper faces these challenges and integrates the service logic robotically. The proposed framework aims to extract the logics automatically and integrates them dynamically for the given requisite. The framework spots exact structure for integrating the service logics by analyzing dependency between the rules, functions, and parameters and vice versa. It scrutinizes the dependency between them through transitions of Finite State Machine (FSM). Finite State Machine, a framework of a computational system, consisting of a set of states (including a start state), an alphabet of symbols that serves as a set of possible inputs to the machine, and a transition function that maps each state to another state (or to itself) for any given input symbol. The machine operates by being fed a string of symbols, and moves through a series of states. The computational core of a Turing machine is a finite state machine. also called finite state automaton. Here the state transitions of FSM depict the swing of the business logic. Through this, it ascertains the dependency relation reside on rules, functions and parameters. From the disclosed dependency relation, the system integrates the service logics automatically. Also the framework evaluates quality attributes of the developed service using FSM through component called Property evaluator. The proposed work focuses on two properties such as computability and traceability which measures computational time, utilized memory space, etc. Traceability verifies the flow, assesses the risk, checks completeness and helps to improve the quality by tracing each and every step of the service. The property evaluation techniques are explained briefly in this paper.

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 framework. 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. Many common services (such as electronic authentication, Authorization Management and some other services will be reused in modem service industry) were duplication of development and the information in various fields cannot be shared. As a result, it leads to not only a mass of waste of resources but also various "isolated islands of information". These have seriously affected the development of the modern service industry.

To deal with these problems, Xu Huiyang proposed a novel modern service industry service integration system based

on SOA to integrate and compose common services. The system establishes an agent layer between service users and platform identifies actual service components required for the request, orchestrates and composes the identified service components in right way [1]. The system can only integrate the services in component level but in many case it requires to integrate at various level. Deng Hui-fang presented an approach to SOA based service integration which makes the service granularity flexible to change and makes the service components reusable at different grade of granularities. As a result, a great number of systems can maintain the existing good framework and mutually provide all necessary and easy-to-use services when they need to share information and interact with each other through integration[2]. Hui Zhang proposed an agent-based Web services integration framework which integrates the service ensuring the QOS of integration system. To realize the QOS, he constructed QoS-based integration path selection algorithm which improves service integration efficiency and reduces the integration cost through this selection mechanism[3]. Wen Ouyang proposed Web service integration algorithm to integrate the existing Web services in order to satisfy a pool of tasks and, at the same time, to minimize the Web service hop count. The proposed algorithm optimally assigns the Web services to tasks waiting to be executed and finds suitable way to integrate Web services so that the hop count between Web services is minimized. This way, we can speed up the processing time of these tasks. This problem plays important roles when the response time is an important factor in evaluating the performance of the application [4]. Liu Yong discussed research progress of geography information service integration, problem in of geography information service integration and proposed service integration framework based on business template which contains a set of business components, logic relation of the components and allows Business user to find suitable business template to build application [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 frameworks, 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]. Above works paves way to semi-automatically integrates the services but there is no standard approach to dynamically integrate the services.

ZhuorenJiang proposed an innovative framework, 'Multi-layer Structure for Dynamic Service Integration (MSFDSI)' in SOA which adds authorized institution and a service integration & analysis adapter to achieve the service authorization, service analysis and dynamic service integration. The service integration & analysis adapter judges the requested service is already existed or need to be integrated. If it's the service need to be dynamically integrated, searches and chooses the suitable services in service registry, uses the interfaces defined in service contract and integrates the services [8]. Lu Liu presented an innovative approach of dependable dynamic service integration for the delivery of rescue capability in service-oriented peer-to-peer (ServP2P) networks. In this approach for delivery of rescue capabilities, resources for information provision and decision support are wrapped into services with standard interfaces to enable better interoperability. The set of services can be dynamically integrated using workflows to form higher levels of functionality to fulfill or support the achievement of mission objectives [9].

W.J.Yan proposed B2B integration approach for SME by leveraging the characteristics of Web Services which utilizes pull and push mechanisms for effective information exchange and sharing between trading partners. This approach has been incorporated in a B2Bi Gateway which enables SMEs to participate in business-to-business collaboration by making use of Web Services [10]. Liyi Zhang proposed a framework called WSMX (Web Service Frameworking execution), a software system that enables the creation and execution of Semantic Web Services based on the Web Service Frameworking Ontology (WSMO) for enterprise application integration. It improves Service discovery, simplifies change management and supports semi-automatic service composition and enhanced interoperability between services [11]. Thomas Haselwanter presented a framework based on the WSMX was build to tackle heterogeneities in RosettaNet messages by using the axiomatised knowledge and rules. It supports communication between partners, data and process mediation using WSMX integration middleware [12].

3. PROPOSED FRAMEWORK FOR WEB SERVICE INTEGRATION

Fig 1 demonstrates proposed framework for service integration. Requirement Analyzer analyzes the request, slices them into number of parts and strips it into standard format. It searches required services in the service repository to process the formatted request and extracts the required service logics. From the given request, it identifies the way in which the service logic can be integrated. For instance, if the request is to develop a advance search service which searches by file type and content type. Here the two service logics must be merged together to satisfy the request, so union prototype can be used to integrate this two. In this way, Requirement Analyzer analyzes the sliced request and identifies the way to integrate the service. Dependency Analyzer analyzes the located logic and breaks up into business rules, functions and parameters. It constructs Finite State Machine (FSM) as the state transition depicts the dependency between rules, functions, and parameters and vice versa. FSM is a behavior framework composed of a finite number of states, transitions between those states, and actions, similar to a flow graph in which one can inspect the way logic runs when certain conditions are met. It has finite internal memory, an input feature that reads symbols in a sequence, one at a time without going backward; and an output feature, which may be in the form of a user interface, once the framework is implemented. The operation of an FSM begins from one of the states (called a *start state*), goes through transitions depending on input to different states and can end in any of those available, however only a certain set of states mark a successful flow of operation (called *accept states*). Here FSM begins with the beginning of the logic transits with the logic flow and ends at the end of the logic.

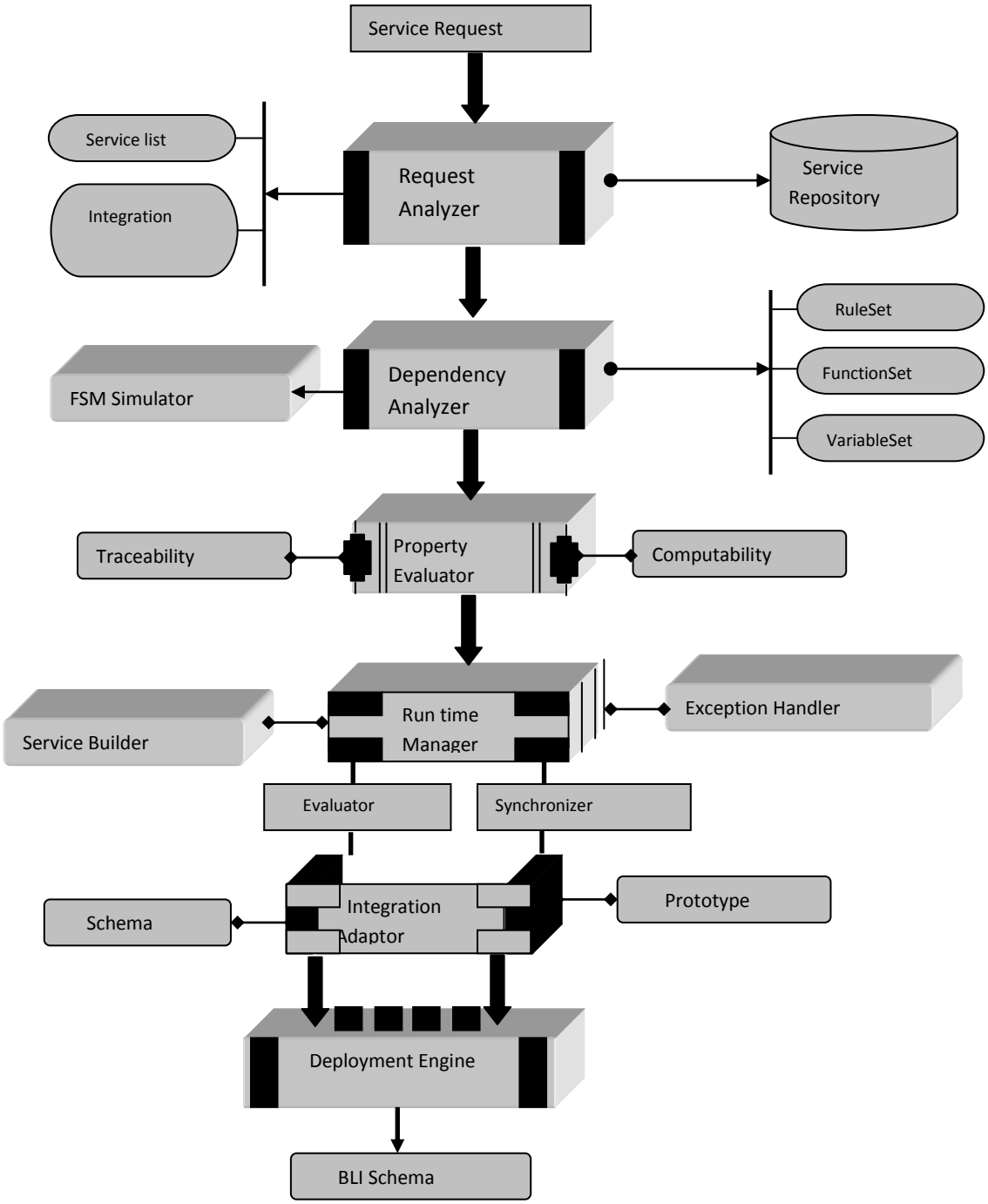

Fig.1.Proposed Framework for Service Integration

Property Evaluator evaluates the constructed FSM and analyzes the properties such as computability and traceability. Dependency analyzer employs Rule bound analysis, Function bound analysis and domain variable bound analysis using FSM and verifies the logic is extracted completely. Rule bound analysis analyzes the located rule is dependent with any other in the logic. If so, it must be extorted with the dependent parts. It detects this through transition states of FSM. Similarly, it analyzes for function and domain variables using Function bound and domain variable bound analysis. Computability examines the ability to solve a particular problem within time limit and in limited memory space. If a function has arguments, $i_1, i_2, \ldots, i_k$, then these integers are initially placed on the tape separated by 1's. On the input , if the TM halts with a tape consisting of 0^m for some m, after the computation on this input, then we say that $f(i_1,i_2, \ldots, i_k) = m$, where f is the function of k arguments computed by this Turing machine. A function or a (decision type) problem is said to be

computable, if there exist a Turing machine which computes the function or answer the problem. [i.e., the TM will halt on all the inputs and gives the correct output for all the input]. Otherwise, the function is said to be non-computable. Traceability in general is 'ability to chronologically interrelate the uniquely identifiable entities in a way that matters'. Informally a function f from N→ N is traceable if we can obtain, in some simple way, a finite collection of possible values for each f(x). This collection of possibilities is known as a trace for f, and the most common requirement on the trace is that it should be computable. A set A is jump traceable, if the function JA(x) is traceable with respect to some computable order function. Here JA(x) is the universal partial A-computable function, JA(x) = Ax (x). We note that JA(x) denotes the output of the universal function, rather than the value of A0(x). For this reason, jump traceability is akin to, but not the same as being super flow. A set A is strongly jump traceable, if for every computable order function h, A is jump traceable with respect to h. A strongly jump traceable set is very close to being computable, because we can predict properties of A to any degree of accuracy we require. There is an obvious connection between a c.e. trace, and a system of descriptions (i.e. a pre_x-free machine). The former can be viewed as a discrete" version of the latter c.e. trace places discrete bounds on cardinality, while a machine talks about measure.

Runtime Manager renovates the extracted logic as a complete service and builds it. While building the service, if bugs or exception thrown, it handles and resolves it. Then it carries the WSDL of the developed logic into common space to integrate them. Integration adapter basically facilitates business integration to integrate various applications developed in various environments. Integration adapter evaluates the schema and identified pattern for integration. If it is feasible to integrate with observed pattern, it uses the prototype available in the system for the explored pattern and integrates it using this prototype. Finally deploys the integrated service in deployment engine and builds Business Logic Integration (BLI) schema which holds detailed information of this integration process such as pattern employed, results of computability, traceability, accessibility level, etc.

4. PROPERTY EVALUATION SYSTEM

4.1. Computability

Computability is an essential criterion in web service which determines whether the modified service is computable with in time limit.

Theorem-The business logic problem is *computable* if it solves the problem exactly in such a way it always terminates and produces the correct output for *all* possible inputs to P. If not, the BL is *non computable*. Proof- We considers programs that take some arbitrary input (say from stdin). We denote the result of a program P run with input x by P(x). We use the mathematical technique of proof by contradiction, or *reduction ad absurdum*. Suppose, for the sake of contradiction, that there exists such a halting programHalt(P, x). (We will show that this leads to an obvious contradiction, and therefore, we must conclude that no such program exists.) It takes two inputs: a program P and its input x. Program Halt(P, x) outputs yes if P(x) halts, and no otherwise. Note that by our hypothesis, Halt(P, x) itself always halts for any pair of inputs. If (P,x) halts, it is computable, otherwise it is non computable.

Here computability is illustrated with an example. First Business logic is represented as logic flow diagram, which signifies the flow of business logic, transition states are identified and indicated below. FSM is constructed whose states transits as the described states. From FSM it evaluates computability property as described above.

Example: The requirement is to create a service, e-payment to calculate total price for the list of purchased items and to transact the calculated amount. In the existing shopping application, we have billing service which computes total cost for the purchased items and transaction service in banking application transacts the amount. By integrating these two services, required new service e-payment can be developed. Here integration should be done in such a way that the processing time of the integrated service bounded within a time limit.

logic1

BL_1: public string billing(){
$BF_{1:}$ String username=username.get();
String password=password.get();
$DR_{f1:}$ String sql="select * from shopping where username="+username+" and password="+password;
ResultSet rs=st.executeQuery(sql);
$CR_{r1:}$ if(rs.next()){
$BF_{r1:}$ double amount=calculateamount();
String accno=accountno.get();
BF_{r2} String accno1=123456;
$BF_{f1:}$ String result="Amount to be paid="+amount;
$P_{1:}$ return result;
}}

logic 2

BL_2: public string transact(){
BF_{21} String accno=accno.get();
String accno1=accno1.get();
String amount=amount.get();
BF_{22} String transid1=transid.set();
DR_{f1} Statement st=con.createStatement();
ResultSet rs=st.executeQuery("select Balance from bank where Accountno='"+ accno+"'");
DR_{r1} double balance=rs.getDouble("Balance");
CR_{r1} if((balance-amount)>1000){
DR_{rr1} st.executeUpdate("update bank set balance= balance-"+amount+" where Accountno="+accno+"";);
DR_{rr2} st.executeUpdate(update bank set balance= balance+"+amount+" where Accountno="+accno1+"");
BF_{f1} String transid=" Amount"+amount+"transferred from"+accno+" to "+accno1;
BF_{r2} String result= "Ur transaction id is "+transid1+" Ur transaction completed successfully";
P_2 return result;}

Solution : Integrated logic

BL_1 public string ebilling(){
BF_{l1} String username=username.get();
String password=password.get();
$DR_{fl1:}$ String sql="select * from shopping where username="+username+" and password="+password;
ResultSet rs=st.executeQuery(sql);
CR_{lfr1} if(rs.next()){
BF_{lfrr1} double amount=calculateamount();
String accno=accountno.get();

String accno1=123456;
BF_{lfrr1} transact(accno,amt,accno1);
} BL_2 public String transact(String accno, double amt, String accno1){

BF_{l1} String transid1=transid.get();
DR_{lf1}ResultSet rs=st.executeQuery("select Balance from bank where Accountno='"+ accno+''');
DR_{lfr1} double balance=rs.getDouble("Balance");
CR_{lfrr1} if((balance-amount)>1000){
DR_{lfrrr1}String sql="update bank set balance= balance-"+amount+" where Accountno="'+accno+'"";
st.executeUpdate(sql);
$DR_{lfrrrr1}$ sql="update bank set balance= balance+"+amount+" where Accountno="'+accno1+'"";
st.executeUpdate(sql);
$P_{lfrrrr1}$String transid=" Amount"+amount+"transferred from"+accno+" to "+accno1;
P_{lfrrr2}String result= "Ur transaction id is "+transid1+" Ur transaction completed successfully";
}

Logic Flow Diagram

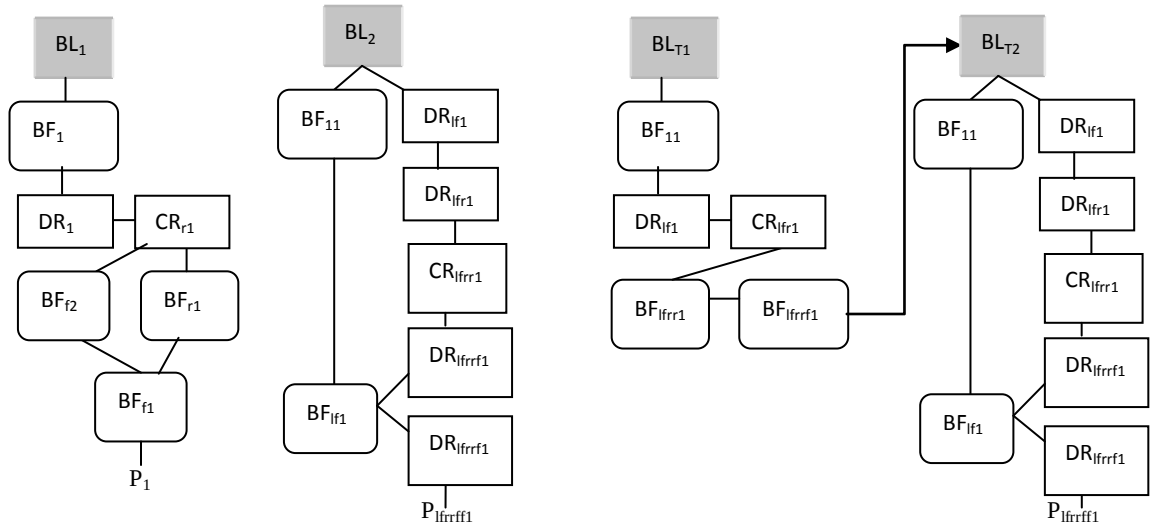

Fig 2. Flow Diagrams for Computability Evaluation

Transition States

$BL_1 \rightarrow \{ BF_1\}$
$BF_1 \rightarrow \{DR_{f1}\}$
$DR_{f1} \rightarrow \{ CR_{r1}\}$
$CR_{r1} \rightarrow \{ BF_{r1}\}$
$BF_{r1} \rightarrow \{ BF_{f1}\}$
$BF_{f1} \rightarrow \{P_1\}$

$BL_2 \rightarrow \{[BF_{lf1}, BF_{11}]\}$
$BF_{lf1} \rightarrow \{ DR_{lfr1}\} BF_{11} \rightarrow \{BF_{lf1}\}$
$DR_{lfr1} \rightarrow \{CR_{lfrr1}\}$
$CR_{lfrr1} \rightarrow \{[DR_{lfrrf1}], [BF_{lf1}]\}$
$DR_{lfrrf1} \rightarrow \{DR_{lfrrf1}\}$
$DR_{lfrrf1} \rightarrow \{ P_{lfrrf1}\}$

$BL_{T1} \rightarrow \{ BF_{11}\}$
$BF_{11} \rightarrow \{DR_{lf1}\}$
$DR_{lf1} \rightarrow \{ CR_{lfr1}\}$
$CR_{lfr1} \rightarrow \{ BF_{lfrr1}\}$
$BF_{lfrr1} \rightarrow \{BF_{lfrrf1}\}$
$BF_{lfrr1} \rightarrow \{ BL_{T2}\}$
$BL_{T2} \rightarrow \{[BF_{lf1}, BF_{11}]\}, BF_{11} = BF_{lfrr1}$
$BF_{lf1} \rightarrow \{ DR_{lfr1}\} BF_{11} \rightarrow \{BF_{lf1}\}$

4.2. Traceability

Traceability in general is 'ability to chronologically interrelate the uniquely identifiable entities in a way that matters'. It verifies the flow, assesses the risk, checks completeness and helps to improve the quality by tracing each and every step of the service.

Let h : N ! N be computable.

Theorem 1- A computable trace with bound h is a sequence (Tn)n2N of non-empty sets such that |Tn| ∈ h(n) for each n . From n, one can compute the finite set Tn. (Tn)n2N is a trace for the function f : N ! N if f (n) 2 Tn for each n. We say that A is computably traceable if there is a fixed h such that each function f _T A has a computable trace with bound h. [13].

Here traceability with illustrated with example. Logic flow diagram is presented with transition representation for the business logic. Also traceability path is implied.

Example: The requirement is to develop a service for telecom department to calculate monthly bill for every customer, e-mail the bill automatically, facility to pay the bill online and email the transaction status. Our integrated E-billing service calculates monthly bill for every customer and has facility to pay the bill

online. Additional functions in the requirement can be added to this service.

Logic 1.

BL_1 public string ebilling(){
BF_{l1} String username=username.get();
String password=password.get();
String email-id=email-id.get();
$DR_{fl1:}$ String sql="select* from shopping where username="+username+" and password="+password;
ResultSet rs=st.executeQuery(sql);
CR_{lfr1} if(rs.next()){
BF_{lfrr1} double amount=calculateamount();
String accno=accountno.get();String accno1=123456;
BF_{lfrr1} send(email-id,amount);
BF_{lfrr2} transact(accno,amt,accno1);}
BL_2public String transact(String accno, double amt, String accno1){
BF_{l1} String transid1=transid.get();
DR_{lf1}ResultSet rs=st.executeQuery("select Balance from bank where Accountno='"+ accno+"'");
DR_{lfr1} double balance=rs.getDouble("Balance");
CR_{lfrr1} if((balance-amount)>1000){
DR_{lfrrr1}String sql="update bank set balance= balance-"+amount+" where Accountno="+accno+"";
st.executeUpdate(sql);
$DR_{lfrrrr1}$ sql="update bank set balance= balance+"+amount+" where Accountno="+accno1+"";
st.executeUpdate(sql);
P_{lfr} String transid=" Amount"+amount+"transferred from"+accno+" to "+accno1;
BF_{lf1}String result= "Ur transaction id is "+transid1+" Ur transaction completed successfully";
BF_{lff1} send(email-id,result);
P_{lfrrr1} returnresult;}}

Logic Flow Diagram

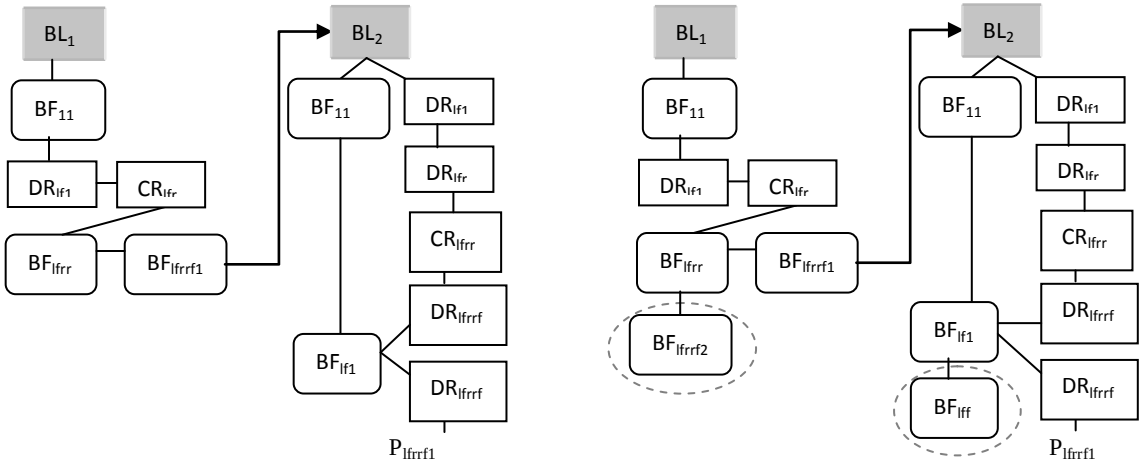

Fig 3. Flow Diagrams for Traceability Evaluation

State Transition

BL_1→{ BF_{11}}
BF_{11}→{DR_{lf1}}
DR_{lf1}→{ CR_{lfr1}}
CR_{lfr1}→{ BF_{lfrr1}}
BF_{lfrr1}→{BF_{lfrrf1}}
BF_{lfrrf1}→ {BL_2 }
BL_2→{[BF_{lf1},BF_{11}]}
BF_{lf1} →{ DR_{lfr1}}BF_{11}→{BF_{lf1}}
DR_{lfr1}→ {CR_{lfrr1}}
CR_{lfrr1}→ {[DR_{lfrrf1}],[BF_{lf1}]}
DR_{lfrrf1}→ {DR_{lfrrf1}}
DR_{lfrrf1}→ { P_{lfrrf1}}

BL_1→{ BF_{11}}
BF_{11}→{DR_{lf1}}
DR_{lf1}→{ CR_{lfr1}}
CR_{lfr1}→{ BF_{lfrr1}}
BF_{lfrr1}→{BF_{lfrrf1}}
BF_{lfrr1}→{BF_{lfrrf2}}
BF_{lfrrf1}→ {BL_2}
BL_2→{[BF_{lf1},BF_{11}]}
BF_{lf1} →{ DR_{lfr1}}BF_{11}→{BF_{lf1}}
DR_{lfr1}→ {CR_{lfrr1}}
CR_{lfrr1}→ {[DR_{lfrrf1}],[BF_{lf1}]}
DR_{lfrrf1}→ {DR_{lfrrf1}}
BF_{lf1} →{ BF_{lff1}}

5. EVALUATION RESULTS

We evaluated the proposed work by developing number of services and examined integration capability for various requisite. Accordingly measured time taken for functioning each and every component in the framework. The total service integration time is calculated by summing up the time taken to execute each and every service of the component in the framework. S.I.T= R.A.T + D.A.T + P.E.T + S.B.T + S.D.T. R.A.T (Request Analysis Time) is time taken to execute the request analyzer component, D.A.T (Dependency Analysis Time) be time taken to analyze the dependency through FSM using Rule bound, function bound and domain variable bound analysis, P.E.T (Property Evaluation Time) is time taken to evaluate the properties such as computability and traceability through FSM, S.B.T (Service Build Time) is time taken to build

the extracted logic and handle the bugs and exception and S.D.T (Service Deployment time) is time taken to integrate the built services and to deploy it in deployment engine. Performance evaluation of each service is compared and it is depicted graphically in fig 2.

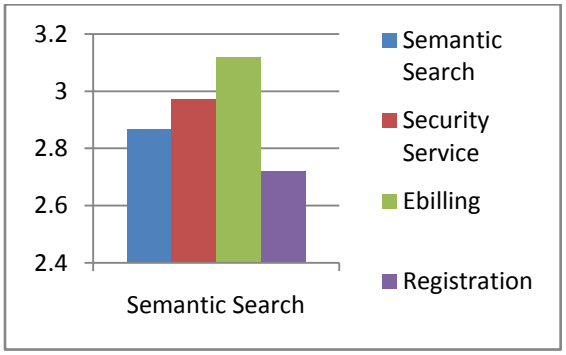

Fig 4. Evaluation Result

6. CONCLUSION

This paper introduces an innovative framework for service integration. It proposes a novel concept to ascertain the dependency between the service logics through Finite State Machine (FSM). Also the paper evaluates various quality criteria through FSM using Property evaluator and it is explained briefly. The proposed framework is evaluated and the results are tabulated and pictured graphically. As the proposed framework has the characteristics of flexibility and platform independence, we can reduce the development difficulty while saving development time and development investment.

7. REFERENCES

[1] Xu Huiyang, Song Meina and Song Junde, "A New Service Integration System for Modern Service Industry Based on SOA", IEEE Conference.

[2] Deng Hui-fang and Xu Guang-feng, "A Study and Design of SOA-based Service Integration for Logistics Customs-clearance", International Symposium on Parallel and Distributed Processing with Applications,2010.

[3] Hui Zhang and Kerong Ben, "Agent-based Web Services Integration Framework",1st International Conference on Information Science and Engineering ,2009.

[4] Wen Ouyang and Min-Lang Chen, "An Optimal Web Services Integration Using Greedy Strategy", IEEE Asia-Pacific Services Computing Conference, 2008.

[5] Liu Yong, "Study on geography information service semantic integration method based on business template", Interational Conference on Computer and Communication Techologies 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] Lu Liu, , Jie Xu, Duncan Russell, KP Lam, Zongyang Luo, Kaigui Wu and Dave Collins, "Dependable Dynamic Service Integration on Service-Oriented Peer-to-Peer Networks", First International Conference on Advances in P2P Systems, 2009.

[10] W.J. Yan, P.S. Tan and E.W. Lee," A Web Services-enabled B2B Integration Approach for SMEs", IEEE international Conference on Industrial Informatics, July 13-16, 2008.

[11] Liyi Zhang and Si Zhou, "A Semantic Service Oriented Architecture for Enterprise Application Integration", Second International Symposium on Electronic Commerce and Security, 2009.

[12] Thomas Haselwanter, Paavo Kotinurmi, Matthew Moran, Tomas Vitvar, and Maciej Zaremba, "WSMX: A Semantic Service Oriented Middleware for B2B Integration", available at http://www.vitvar.com/tomas/!publications/icsoc2006-WSMX.pdf.

[13] André Nies , "Superhighness and strong jump traceability", The University of Auckland ICALP 2009, available at http://www.cs.auckland.ac.nz/~nies/talklinks/ICALP09Web.pdf