FAULT PREDICTION IN WEB SERVICES

Arjun Muthusami, MCA – II year,
SRM University, Department of Computer Applications,
Kattankulathur, Chennai, India. PIN: 603203.
Mobile: 9940661436

Abstract— The reliance on Web Services is increasing and there is growing need for effective verification that the services offered is without any fault. Itis important to find out which Web services are responsible for a failed business process because we could penalize these Web services and exclude them from the business process in the future. Model based diagnosis (MBD) has been a well established practice for its several positive aspects including cognitively being better understood by development and testing teams. Automata is a simple and formally well defined model being used for monitoring and diagnosis of system faults. Here we have reviewed works on automata for web service fault management and also propose a model of stochastic automata for the purpose.

Text words: **Fault Management, Quality of Service (QoS), Service Level Agreements (SLAs), Web Services.**

I. INTRODUCTION

Web service compositions are based on a set of services working together to achieve an objective and are normally defined at programming time as a "business process" that describes the sequencing and coordination of calls to the component web services. In a web service composition, when one of the component web service fails the entire composition is affected. For a Web service process, the symptom of a failure is that exceptions are thrown and the process halts. As the process is composed of multiple Web services, it is important to find out which Web services are responsible for the failure. The current throw-and-catch mechanism is very preliminary for diagnosing faults. It relies on the developer associating the faults with exceptions at design time. When an exception is thrown, we say certain faults occur. But this mechanism does not guarantee the soundness and the completeness of diagnosis.

Because classical approaches of fault management do not give a deeper insight into the faults and usually do not allow a fault diagnosis, model- based methods of fault detection were developed. "Model based Diagnosis" (MBD) refers to use of models of the observed system as a basis for fault detection and diagnosis. Among many classical models that can be used to formalize business processes , we concentrate on automata models of business process since automata are a natural way to model system behaviour, especially dynamic behaviour.

The rest of the paper is organized as follows. Section – II briefs about Web Services and Fault Management, Section – III give details about Automata in Web Services and Fault Management, Section – IV discusses about the Proposed approach in the study, Section – V indicates the Formalisation of the study. We conclude the paper in Section – VI.

II. WEB SERVICES AND FAULT MANAGEMENT

With ever growing use of Internet, Web services become increasingly popular and their growth rate surpasses even the most optimistic predictions. A web service can fail due to software bugs, unstable communication over the Internet, and overloaded or complete crash of service servers.

A service that is frequently failing can tarnish the provider's reputation and business. One of the most important challenges with the deployment of Web Services is ensuring that services are correct and available despite faults.

Fault Classification: Web services execution faults can be classified into three main categories based on the cause of occurrence:

1.Violations of agreed upon Service Level Agreements (SLAs) and policies with regards to functional (e.g., price limits or delivery deadlines) and non-functional requirements (e.g. service response time, service availability and security). In this case the service execution might be completed but the results are not conforming to the negotiated SLAs and collaboration policies.

2.Functional and behavioural faults refer to the scenario where a constituent service cannot complete a task execution or the service delivers incorrect results due to computational/logic errors, erroneous data flow or semantic incompatibility of the exchange messages. Additionally, behavioural failures can be caused by conversation exceptions such as improper invocation order of service operations, lost messages when processing fails and interrelated messages processed individually.

3.Operational faults refer to communication infrastructure exceptions and middleware failures of the hosting servers and the database servers. Examples of such faults could be network unavailability causing disconnections, network congestion causing message loss, and overloaded application server causing excessive delays and timeouts.

These faults can be categorised into three system levels. Fault management systems involve a combination of multiple steps – Monitoring / Detection, Diagnosis, Recovery, and Restart / Repair, that are typically independently developed and optimized.

1.Monitoring / Fault detection recognizes that something unexpected has occurred. The execution of Web service process is monitored to find the unobserved behaviors of the system given the normal system behavior model and record necessary and sufficient information for online/offline diagnosis Techniques fall here into two classes: off-line and on-line. Verification is an off-line technique, done to guarantee that the deployed services satisfy a set of requirements and temporal properties. On-line techniques provide a real-time detection capability that is performed concurrently with service execution.

2. Diagnosis - Fault diagnosis pinpoints one or more root causes of the problem, to the point where corrective action can be taken. The unobserved behaviours found while monitoring are further analysed (online) to determine the causes of exceptions (failures). Typically, fault diagnosis encompasses both fault detection and fault location.

3. Fault confinement limits the fault impact by attempting to contain the spread of fault effects in one area of the Web service, thus preventing contamination of other areas.

TABLE I. - Fault Types and Examples

Violations of agreed upon Service Level Agreements (SLAs) and policies

QoS violation faults	QoS value beyond threshold.

Functional and behavioural faults

Web –Application Level Faults

Internal data faults	Data quality faults (value mismatch; missing data: null values).

Web service Level Faults

Web service execution faults	Missing parts in input message, wrong order of operation invocations (internal to a service)
Web service coordination faults	Component service unavailable, process failure (time out).

Operational Faults - *Infrastructure & Middleware level faults*

Node faults	Node (application server or client device) has failed.
Network faults	missing connection, low bandwidth
Generic faults	Denial of service, wrong authentication

Web –Application Level Faults

Application coordination faults	Application Failure due to reply timeout, resources not available at right time
Actor faults	Customer is not connected when a synchronous communication is needed.

4. Recovery utilizes techniques to eliminate the effects of faults. Three basic recovery approaches are available: fault masking, retry and rollback. Fault masking techniques hide the effects of failures by allowing alternative information to outweigh the incorrect information. Retry undertakes one more attempt Automata for Web Services Fault Monitoring and Diagnosis at an operation and is based on the premise that many faults are transient in nature. Rollback makes use of the fact that the Web service operation is backed up (check pointed) at some point in its processing prior to fault detection and operation recommences from that point.

5.Restart occurs after the recovery of undamaged information.

a. Hot restart: resumption of all operations from the point of fault detection and impossible only if no damage has occurred.

b. Warm restart: only some of the processes can be resumed without loss.

c. Cold restart: complete reload of the system with no processes surviving. The Web services can be restarted by rebooting the server.

6.Repair - A failed component is replaced. Repair can be offline or on-line.

III. AUTOMATA IN WEB SERVICES FAULT MANAGEMENT: A SURVEY

Currently, fault management in business process is similar to exception handling provision programming languages have. The method mainly resorts to default action instead of probing into causes of error and providing solutions. Of late, researchers have proposed Model Based Diagnosis (MBD) for fault management. They have considered automata as a natural choice for clear picturization of state changes and unambiguous interpretation to model dynamic behaviour of web services and have supported their usages for the purposes. Here a brief review is presented. Many automata models have been proposed for web service process verification [3,4,6,9], monitoring [12,9,5] and diagnosis [13]. Table 2 summarises these models and their limitations.

TABLE II (a) Models for Verification

Model	Features	Remarks
Formal Verification of BPEL4WS Business Collaborations - VERBUS [3]	A modular and extensible framework for the verification of business processes in which several process description languages and verification tools can be integrated. The prototype receives as input a BPEL4WSprocess specification and a set of properties, automatically	Model does not handle concurrency and link for control flow.

	translates the specification to a formal specification language based finite state machines and verifies it using a model–checker.	
Model-based Verification of Web Service Compositions - Foster, Uchitel, Magee,& Kramer[4]	The model describes a formal approach to modeling and verifying the compositions of web services workflows using the Finite State Processes (FSP) notation. Verification is done prior to deployment, during the design phase.	Model does not map correlation, data and *link*.
Analysis of Interacting BPEL Web Services -Fu, Bultan, & Su[6]	The interactions of composite web services are modeled as *a* guarded automaton. BPEL specifications of web services are translated to guarded automata, followed by the translation of the guarded automata to a verification language.	Model does not map correlation, and *link*.
Modeling and Verifying Web Service Applications with Time Constraints - Jia et al [8]	A formalism called WS Timed automata is introduced to capture the timed behavior of the web service. The BPEL4WS specification of business process is translated to timed Automata and then Uppaal tool issued to simulate and verify the correctness of the system.	

TABLE II (b) Models for Monitoring and Diagnosis

Model	Features	Remarks
Model based approach for web process monitoring. Yan et al.[12]	Map BPEL into automata .The control flows are mapped to different structures of automata. Concurrent branches in flow are modeled as pieces of synchronizing automata. To represent data flow, state variables are defined and mapped to variables in BPEL. In addition, transition rules containing state variables are defined to model the triggering conditions in control flow.	Model does not map link.
Runtime Monitoring of Web Service Conversations - J Simmonds et al[9]	Concentrates on the dynamic analysis via runtime monitoring, which tries to ensure the quality of an application through the analysis of runtime events. A subset of UML Sequence Diagrams of the business process is identified as a property specification language and these diagrams are transformed to automata. This automata is later used to perform conformance checking of execution traces against the given specification.	Model does not map concurrence.
A Methodology for Online Monitoring of Non-Functional Specifications of Web-Services Raimondi [5]	Models web services as timed -automata for monitoring non-functional specifications of web services (such as latency and reliability).	
A Model-based Approach for Diagnosing Faults in Web Service Processes Yan et al[13]	Automata are used to give a formal modeling of Web service processes described in BPEL. For diagnosis, execution trajectories of the business process are constructed based on the model of the process and the observations from the execution. The variable dependency relations are utilized to diagnose the Web services within the trajectory responsible for the thrown exceptions.	A deterministic model.

IV. PROPOSED APPROACH

Any system involving uncertainties, unpredictable human actions or system failures requires a non deterministic treatment. So far, the web services have been modeled using deterministic approaches only, which cannot distinguish between states that are highly probable and those that are less probable. As an example let us understand the execution of the Loan Approval Process, shown in fig 1.

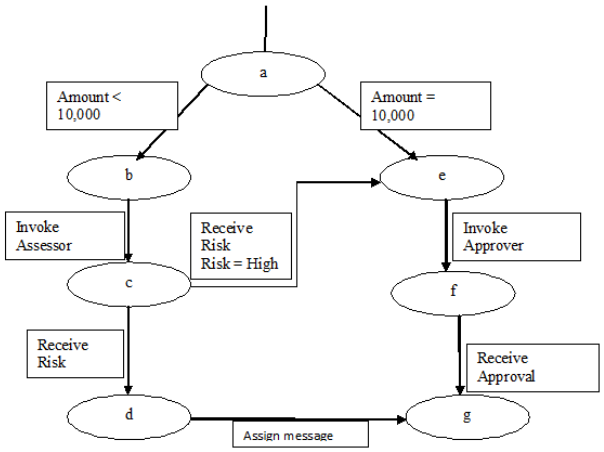

Fig 1. Loan approval process

The loan approval process is the same as throne described in the BPEL Specification 1.1 [12] and the model is self-describing. Based on the loan amount received, the process invokes a Loan Assessor web service (when the credit required is<10000) or a Loan Approver web service (when the credit required is ≥ 10000) whose jobs are to approve or reject the loan. The Loan Assessor web service calculates the risk in approving the loan. Risk assessment practically changes in times for its dependence to current situation that is naturally dynamic like share values and personal choices.

This stochastic nature of control variables like here 'risk' leads to different behavioural traces of system execution, for e.g.{a,b,c,d,g} when the risk is Low and{a,b,c,e,f,g} when the risk is High. Based on risk assessment and domain specific rules, one may assign probabilities to traces. Based on this idea we propose a model based fault management system that follows the steps given below:

1. Model web services using stochastic automata.
2. Develop programming primitives to implement the model.
3. Study the application of the model for
   - Monitoring of Web Services.
   - Fault Diagnosis of Web Service.

V. FORMALISATION

Stochastic automaton model: The stochastic model of a business process can be expressed an automaton (X, Σ, T, P) where

- X is a finite set of states,
- Σ is a finite set of events,
- T☐X×Σ×X is a finite set of transitions,
- p (x′, e|x) is a state transition probability defined for all x, x′☐X ,e☐Σ.

We associate with each nondeterministic transitional probability value, which specifies the probability with which this transition may occur. Table 3 gives examples for states, events, transitions and Table 4gives assumed state transition probabilities with reference to the loan approval process discussed in the previous section. Execution of an instantiation of the model loan approval process is controlled by current risk assessment that is predicted from observable facts like share values and other dependant variables.

TABLE III. Example for states, events, transitions

Notation	Example
X , finite set of states	a, b, c, d, e, f, g
Σ , finite set of events	Receive, Invoke, Assign, Reply
T, finite set of transitions	(a, Receive amount<10000,b),(a, Receive amount≥10000,e), (b, Invoke Assessor, c), (c, Receive Risk =Low, d), (c, Receive Risk = High ,e),(d, Assign Message, g), (e, Invoke Approver, f), (f, Receive Approval, g)

TABLE IV. Assumed state transition probabilities

T (finite set of transitions)	p(state transition probability)
(a, Receive amount<10000,b)	0.3
(a, Receive amount≥10000, e)	0.7
(c, Receive Risk = Low, d)	0.6
(c, Receive Risk = High, e)	0.4

This gives rise to a predicted trace of the model (say rpd, the predicted run). And the execution of model instantiation gives a trace of states (say rob, the observed run). The difference between two traces beyond an agreeable limit (say a threshold value δ) gives an indication of error. With reference to the example given in previous section:

Possible predicted traces (rpd) are

- {a,b,c,d,g} for risk = Low
- {a,b,c,e,f,g} for risk=High

If the observed trace (rob) of states for the process instantiation is {a, b, c, e, f, g} when the predicted trace of states is {a,b,c,d,g}, it is a clear indication of error due to wrong calculation of risk. We can hence use the model for estimating the likelihood of possible state transitions and predict the possible execution trace of the process. Such an execution trace can be utilized to monitor fault occurrences in the business process as :| rob –rpd | ≥ δ

We plan to initialize the probability values based on the execution history of a web service. We propose to monitor service level faults among the faults listed in Table1. The

ultimate goal is to give a formal stochastic model of a business process that would help in analysis, monitoring and fault diagnosis of the process.

VI. CONCLUSION

A web service is an emerging technology for business process integration. One of the most important challenges with the deployment of Web Services is ensuring that services are correct and available despite faults. In this paper, we discussed how automata are used for formal modeling of web services and also for verification, monitoring and diagnosis. However, the use of these models to cope with stochastic nature of web services is not explored. In this context, we propose a model of web services using stochastic automata. Further, we intend to study the stochastic nature of environment in which the Web services are deployed and work on the feasibility of modeling Web Services using stochastic automata as discussed in the previous section. Further, the state e.g. for risk prediction can be computed with neural network. We would like to work on a hybridized model with neural network and automata for monitoring and fault diagnosis of web services in future.

ACKNOWLEDGMENT

My whole hearted gratitude to Dr S. Subbarayan M.Sc., PhD, Professor & HOD / Deptt of Computer Application, SRM University, Kattankulathur and all my Department colleague for their constant support, valuable suggestion and for providing the computing facilities to carry out this work.

I extend my gratitude to Mrs. K. SUBHASHINI, M.C.A, M.Phil., (M.Tech), Asst. Professor in Department of Computer Applications, SRM University for her moral support, guidance and encouragement to carry on the work.

I feel privileged to extend my deep sense of gratitude to Mrs. F.Ezhil Mary Arasi MCA, M.Phil, M.S, (Ph.D) Asst. Professor in Department of Computer Applications, SRM University for her continuous support and timely guidance to carry on the work with the moral support and encouragement.

REFERENCES

[1] Abdelkarim Erradi, Piyush Maheshwari, School of Computer Science and Engineering University of New South Wales, Sydney, Australia, IBM India Research Lab New Delhi, India, "Recovery Policies for Enhancing Web Services Reliability".

[2] D Thorsley and D. Teneketzis, "Diagnosability of stochastic automata," in Proc. 42nd IEEE Conf. Decision and Control, Dec. 2003, pp.6289–6294.

[3] Fisteus, J., Fern´andez, L., &Kloos, C. (2004)."Formal verification of bpel4ws businesscollaborations". In K. Bauknecht, M. Bichler, & B.Prll (Eds.), Proc. of 5th international conference ecommerce and web technologies (ec-web) (p. 76-85).Springer.

[4] Foster.H., Uchitel.S.,Magee.J., & Kramer.J. (2003)."Model-based verification of web service compositions". In Proc. of eighteenth IEEE international conference on automated software engineering (ase03) (p. 152-161).

[5] F. Raimondi, J. Skene, W. Emmerich, and B.Wo´zna."A methodology for online monitoring on-functional specification of web-services". In D.K. C. Attiogb´e, editor, Proceedings of the First International Workshop on Property Verification for Software Components and Services (PROVECS'07), number 567 in ETH Technical Report, pages 50–59.

[6] Fu, X., Bultan, T., & Su, J. (2004). "Analysis of interacting bpel web services". In Proc. of the 13thinternational World Wide Web conference (www'04).ACM Press.

[7] Hamscher, W., Console, L., & de Kleer, J. (Eds.).(1992). "Readings in model-based diagnosis. Morgan Kaufmann".

[8] Jia Mei, Huaikou Miao, Qingguo Xu, Pan Liu"Modeling and Verifying Web Service Applications with Time Constraints", Computer and Information Science, ACIS International Conference Issue, August 2010 pp. 791-795.

[9] Jocelyn Simmonds, Yuan Gan, Marsha Chechik, Shiva Nejati, Bill O'Farrell, Elena Litani, "Runtime Monitoring of Web Service Conversations", IEEE Transactions on Services Computing June 2009 pp.223-244.

[10] M.G. Fugini, E. Mussi, Politecnico di Milano, Dipartimento di Elettronicaed Informazione, "Recovery of Faulty Web Applications through Service Discovery", Via Ponzio 34/5 - I-20133.

[11] Pat. P.W. Chan1, Michael R. Lyu1, and Miroslaw Malek Department of Computer Science and Engineering The Chinese University of Hong Kong Hong Kong, China, "Making Services Fault Tolerant".

[12] Yan, Y., Pencol´e, Y., Cordier, M.-O., &Grastien,A. (2005). "Monitoring web service networks in a model-based approach". In 3rd IEEE European conference on web services (ecows05). Sweden: IEEE Computer Society.

[13] Y. Yan, P. Dague, Y. Pencolé and M.-O.Cordier."A Model-based Approach for Diagnosing Faults in Web Service Processes". International Journal of Web Services Research JWSR, 5(4), Oct.-Dec.2008.

AUTHOR PROFILE

Arjun Muthusami (M. Arjun) is studying 2nd year in Master of Computer Applications (MCA) at SRM University, Kattankulathur, Chennai, India, PIN 603203. He passed Bachelor Degree in Computer Science, B.Sc (C.Sc) in First class, from D.G.Vaishanva College, University of Madras, Chennai, Mobile No: 9940661436.

