ISSN-2320-7884 (ONLINE)
ISSN-2321-0257 (PRINT)

Testing of web services by using ontology

Ch.Sharath[1], D.Basawaraj [2]

[1]Student, Dept of CSE, CMR Technology,Hyderabad, AP, INDIA

[2]Associate Professor & HOD, Dept of CSE, CMR Technology,Hyderabad, AP, INDIA

Abstract: Web services square measure rising technologies that may be thought-about because the results of the continual improvement of net services as a result of the tremendous increase indemand that is being placed on them. they square measure they are quickly evolving and are expected to alter the paradigms of each software package development and use, by promoting software package reusability over the web, by facilitating the wrapping of underlying computing models with HTML, and by providing various and complicated practicality fastand flexibly within the style of composite service offerings. during this paper, produce one internet application and Framework for internet services(WS). internet application facet any link fault means that internet services to show fault link in background facet. however project work every and extremely communication link to be checking method mistreatment J unit tool. J unit tool is networking tool that method is testing for Framework internet Services(WS).

Index Terms—software testing, web services, J unit testing tool

. 1. INTRODUCTION

Web Services (W3C, 2004b) area unit thought of a brand new paradigm in building softwareapplications; this paradigm relies on open standards and therefore the web. net Servicesfacilitate the interconnection between heterogeneous applications since it isbased on XML open standards that will be accustomed decision remote services or exchangedata. net Services area unit thought of AN implementation or realization of the Service-Oriented design (SOA) (Singh & Huhns, 2005), that consists ofthree roles: Service Requester (Consumer), Service supplier, and repair Publisher(Broker). To implement SOA, net Services rely upon a gaggle of XML-based standards like easy Object Access Protocol (SOAP), net Service DescriptionLanguage (WSDL) and Universal Description, Discovery and Integration(UDDI). a tangle that limits the expansion of net Services is that the lack of trustworthinessby the requesters of net Services as a result of they will solely see the WSDLdocument of an internet Service, however not however this service was enforced by theprovider.An example of employing a net Service is once building AN application that needsto get data a few book (e.g., worth and author) given the book's ISBN.Amazon offer an internet Service (see Cornelius, 2003) to satisfy this requirementand exploitation the approach during this chapter it will assess however strong the service is beforeusing it.Software Testing is principally accustomed assess the standard attributes and detectfaults during a software and demonstrate that the particular program behavior willconform to the expected behavior. Testing techniques are often divided into blackbox and white box counting on the supply of the ASCII text file; if take a look at knowledge aregenerated counting on the supply code, then a

ISSN-2320-7884 (ONLINE)
ISSN-2321-0257 (PRINT)

testing technique belongs to whitebox, whereas if the ASCII text file is inaccessible, then a testing technique belongs toblack box.

This chapter's approach of net Services take a look ating assumes that the tester onlyhave the WSDL document of the net Service underneath test and not the ASCII text file,for this reason recording machine testing techniques are used. Testing are often accustomed solve a part of the issues of net Services trustworthiness; by assessing the standard attributes of an internet Service underneath take a look at, the boldness of the requesters of thisWeb Service can increase or decrease per the take a look at results. it'll alsohelp the requesters to decide on between net Services doing an equivalent task. However,Web Services testing still face several issues like inaccessibility of thesource code to the requesters which the normal testing techniques do notcope with the new characteristics introduced by net Services standards (Zhang & Zhang, 2005). This chapter introduces AN approach to unravel a part of these issues,which relies on analyzing WSDL documents so as to get testcases to check the strength quality attribute of net Services.

1.1 Generation of testbed.

A service usually depends on different services to perform it's operate. However, inservice unit testing and conjointly in progressive serviceintegration testing, the service underneath take a look at must beseparated from different services that it depends on.Techniques are developed to get servicestubs or mock services to switch the otherservices for testing.

1.2 Checking the correctness of take a look at outputs.

Researchwork has been reportable within the literature to ascertain thecorrectness of service output against formal specifications,such as exploitation metamorphic relations , ora ballot mechanism to match the output from multiple equivalent services [39], [40], etc.

Testing tools. variety of prototypes and commercialtools are developed to support various activities in testing in WS.

2. WEB APPLICATION TESTING TECHNIQUES

In today's changing and competitive web-based business situation, organizations forever ought to take a look at their net based mostly applications before the launch of their web site. By testing, any organization is certain that the net application can work absolutely and can be simply accepted by the end-users. The testing techniques conjointly check the net application's browser compatibility; load testing, quantifiability testing, stress testing and determination testing. made net Applications (RIAs) are dynamical the manner we have a tendency to deliver and use web-based applications.

Here are few of the basic testing techniques for web applications:

1. **Functional Testing:** This testing is employed for checking all the links of the online pages; kind testing, cookie testing and information affiliation.

2. Usability Testing: This testing checks the navigation and user friendliness of the online pages. Through this testing it's ensured whether or not the content is correctly checked and is well comprehendible to the users. It conjointly checks

ISSN-2320-7884 (ONLINE)
ISSN-2321-0257 (PRINT)

whether or not the anchor text links area unit operating properly, whether or not sitemaps and facilitate files area unit having correct info and every one the links area unit operating.

3. Interface Testing: This checks if the online server and application server interface, application server and information server interface have correct interaction or not. This take a look at ensures that the users don't see any error messages.

4. CompatibilityTesting: Compatibility Testing is incredibly necessary because it checks browser compatibility, OS compatibility, mobile browsing and printing choices.

5. Performance Testing: Performance testing includes internet load testing and internet stress testing. internet load testing technique checks if several users will access a similar page at a similar time and whether or not an online page will handle serious load on any specific page. internet stress testing is completed on the location to visualize that however can the location react and recover throughout the strain time.

6. Security Testing: This checks the safety of the online applications. For security functions, internal pages shouldn't open if you're not logged into the web site. alternative statistics shouldn't be seen although the user is logged in. The files ought to solely lean the choice for downloading and it shouldn't be accessed while not downloading. CAPTCHA for automates scripts logins ought to be tested. SSL ought to be tested for security measures.

After finishing all the testing, a live testing is critical for internet primarily based applications and internet sites. Then transfer the location and complete testing ought to be done. These days, internet applications area unit accessed from totally different varieties of devices like desktops, PDAs, iPhones, etc. it's vital to see whether or not the online application is compatible to those devices.

Web applications is provided to an outsized and numerous audience however there's a risk of being exposed to an outsized set of probable loopholes as so much as productive package testing results is concerned:

1. varied Application Usage (Entry – Exit) ways area unit potential

2. folks with variable backgrounds & technical skills might use the appliance. Also, variations might rise from cross-platform problems to distinction in browsers, network sorts or network speeds, computer network and net application variations, etc. – leading to problems regarding the package.

3. Even on a similar browser, applications is also dead otherwise supported native problems like screen resolution/ hardware/ package configuration of the system

4. The applications might need testing for incapacity compliance and usefulness

5. Firewalls or allied security threats

To conclude, the whole method of internet Application testing includes thereforeme very necessary and important steps so on make sure that finish users area unit happy with the applications.

2.1 WebServices (WS):

A Web service may be a software designed to support practical machine-to-machine interaction over a network. it's AN interface represented in an

ISSN-2320-7884 (ONLINE)
ISSN-2321-0257 (PRINT)

exceedingly machine-processable format (specifically WSDL). different systems act with the net service in an exceedingly manner prescribed by its description victimisation SOAP messages, generally sent victimisation protocol with AN XML publishing in conjunction with different Web-related standards.

2.2 Agents and Services:

A Web service is AN abstract notion that has to be enforced by a concrete agent. (See Figure 1-1) The agent is that the concrete piece of software system or hardware that sends and receives messages, whereas the service is that the resource characterised by the abstract set of practicality that's provided. as an instance this distinction, you would possibly implement a specific internet service victimisation one agent someday (perhaps written in one programming language), and completely different|a special|a unique|a distinct} agent succeeding day (perhaps written in an exceedingly different programming language) with an equivalent practicality. though the agent could have modified, the net service remains an equivalent.

2.2 Requesters and Providers:

The purpose of an internet service is to produce some practicality on behalf of its owner -- someone or organization, like a business or a personal. The supplier entity is that the person or organization that has AN acceptable agent to implement a specific service. (See Figure 1-1: Basic study Roles.)

A requester entity may be a person or organization that desires to create use of a supplier entity's internet service. it'll use a requester agent to exchange messages with the supplier entity's supplier agent.

In most cases, the requester agent is that the one to initiate this message exchange, although not continuously. notwithstanding, for consistency we have a tendency to still use the term "requester agent" for the agent that interacts with the supplier agent, even in cases once the supplier agent really initiates the exchange.

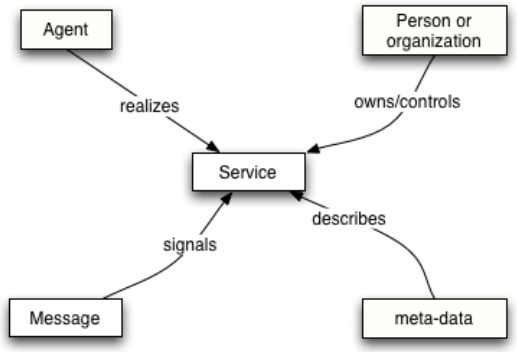


Figure 1-1. The General Process of Engaging a Web Service

2.3 Service Description:

The mechanics of the message exchange area unit documented during a net service description (WSD). (See Figure 1-1) The WSD could be a machine-processable specification of the net service's interface, written in WSDL. It defines the message formats, datatypes, transport protocols, and transport publication formats that ought to be used between the requester agent and therefore the supplier agent. It additionally specifies one or a lot of network locations at that a supplier agent is invoked, and should give some data concerning the message exchange pattern that's expected. In essence, the service description represents associate degree agreement governing the mechanics of interacting thereupon service.

ISSN-2320-7884 (ONLINE)
ISSN-2321-0257 (PRINT)

2.4 Semantics

The semantics of an online service is that the shared expectation concerning the behavior of the service, especially in response to messages that area unit sent thereto. In effect, this can be the "contract" between the requester entity and therefore the supplier entity relating to the aim and consequences of the interaction. though this contract represents the general agreement between the requester entity and therefore the supplier entity on however and why their individual agents can act, it's not essentially written or expressly negotiated.

Overview of participating an online Service

There area unit many ways that a requester entity may interact and use an online service. In general, the subsequent broad steps area unit needed, as illustrated in Figure 1-1 the requester and supplier entities become far-famed to every different (or a minimum of one becomes understand to the other); the requester and supplier entities somehow agree on the service description and linguistics which will govern the interaction between the requester and supplier agents the service description and linguistics area unit realised by the requester and supplier agents and the requester and supplier agents exchange messages, so acting some task on behalf of the requester and supplier entities. (I.e., the exchange of messages with the supplier agent represents the concrete manifestation of interacting with the supplier entity's net service.)

3. WEB SERVICES ARCHITECTURE

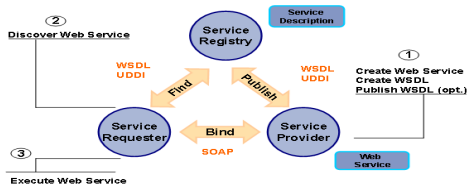


Fig-3: Web services architecture

Fig- 3Although it's necessary, the centralized service written account isn't the sole model for internet service discovery. the only sort of service discovery is to request a replica of the service description from the service supplier. when receiving the request, the service supplier will merely e-mail the service description as Associate in Nursing attachment or offer it to the service requestor on a transferable media, like a disc. though this sort of service discovery is straightforward, it's not terribly economical since it needs previous data of the online service, likewise because the contact data for the service supplier.

Between these 2 extremes, there's a necessity for a distributed service discovery methodology that has references to service descriptions at the service provider's point-of-offering. Fig3.1 the online Services review Language provides this sort of distributed discovery methodology, by specifying the way to examine an internet website for accessible internet services. The WS-Inspection specification defines the locations on an internet website wherever you may seek for internet service descriptions.

Since the online Services review Language focuses on distributed service discovery, the WS-Inspection specification enhances UDDI by facilitating the invention of services accessible on websites, however which can not be listed however in a very UDDI written account. extra data on the link between the online Services review Language and UDDI will be found within the WS-Inspection and UDDI Relationship .

ISSN-2320-7884 (ONLINE)
ISSN-2321-0257 (PRINT)

3.1 WS-Inspection summary:

The WS-Inspection specification doesn't outline a service description language. WS-Inspection documents offer a way for aggregating differing kinds of service descriptions. among a WS-Inspection document, one service will have quite one relation to a service description. for instance, one internet service could be delineate exploitation each a WSDL file and among a UDDI written account. References to those 2 service descriptions ought to be place into a WS-Inspection document. If multiple references square measure accessible, it's useful to place all of them within the WS-Inspection document so the document shopper will choose the kind of service description that they're capable of understanding and wish to use. Fig-3.2 provides an outline of however WS-Inspection documents square measure used.

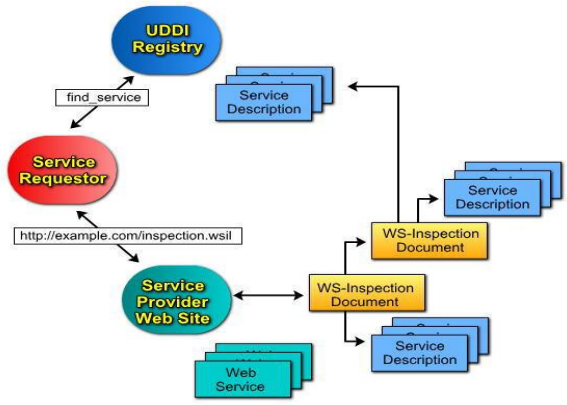


Figure 3.2: WS-Inspection overview

The WS-Inspection specification contains two primary functions, which are discussed in more detail in the next two sections.

- It defines an XML format for listing references to existing service descriptions.
- It defines a set of conventions so that it is easy to locate WS-Inspection documents.

3.2 The Architectural Models: The Service Oriented Model focuses on aspects of service, action and so on. Fig3.3 While clearly, in any distributed system, services cannot be adequately realized without some means of messaging, the converse is not the case: messages do not need to relate to services.

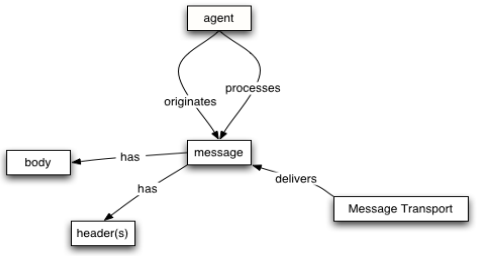


Figure 3.3. Simplified Service Oriented Model

The software system infrastructure consists of AN Enterprise Service Bus (Apache ServiceMix), a BPEL progress Execution Engine (a custom-made version of Apache ODE), a Resource written account (based on the WSO2 Governance written account and accessed via a written account adapter) and a collection of custom engineered core elements. These are:

- The progress musician selects the concrete services provided by the Resource/Service written account at runtime. particularly, it implements a ranking mechanism to support this native choice step.

- It is that the Router's responsibility to forward messages to the suitable endpoints. For outgoing messages, that address AN end point simply by linguistics info, the router handles the service binding and ensures the supply of appropriate, dynamically created endpoints. For inward messages it will be organized to route them via a policy social control purpose. It may also apply fault handling ways within the case of communication faults.

ISSN-2320-7884 (ONLINE)
ISSN-2321-0257 (PRINT)

• The Notification Agent collects notifications and events from numerous elements and produces higher level notifications in step with a collection of rules. the upper level notifications square measure then evaluated by call support and system governance.

• The Administration element provides interfaces for administering the orchestration elements and therefore the set of deployed progress templates.

4. JUNIT TESTING TOOL

JUnit may be a unit testing framework for the Java artificial language. it's necessary within the check driven development, and is one in all a family of unit testing frameworks put together called JUnit.
JUnit promotes the concept of "first testing then coding", that stress on putting in the check knowledge for a chunk of code which may be tested 1st and so are often enforced . productivity and stability of program code that reduces programmer stress and therefore the time spent on debugging.

JUnit check Case:
A Unit test suit may be a a part of code that ensures that the another part of code (method) works obviously. to attain those desired results quickly, check framework is needed .JUnit is ideal unit check framework for java artificial language.

A formal written unit test suit is characterised by a glorious input Associate in Nursing by an expected output, that is discovered before the check is dead. The glorious input ought to check a precondition and therefore the expected output ought to check a postcondition. There should be a minimum of 2 unit check cases for every requirement: one positive check and one negative check. If a demand has sub-requirements, every sub-requirement should have a minimum of 2 check cases as positive and negative.

Find defects. this is often the classic objective of testing. A check is run so as to triggerfailures that expose defects. Generally, we glance for defects altogether fascinating elements of theproduct.

Maximize bug count. the excellence between this and "find defects" is that total numberof bugs is additional necessary than coverage. were focus narrowly, on solely a number of highriskfeatures, if this is often the thanks to realize the foremost bugs within the time accessible.

Block premature product releases. This tester stops premature cargo by finding bugs so serious that nobody would ship the merchandise till they're mounted. for each releasedecision meeting, the tester's goal is to possess new showstopper bugs.

facilitate managers build ship no-ship choices. Managers area unit generally involved withrisk within the field. they require to grasp regarding coverage (maybe not the oversimplified codecoverage statistics, however some indicators of what proportion of the merchandise has been addressedand what proportion is left), and the way necessary the glorious issues area unit. issues that appearsignificant on paper however won't result in client discontentment area unit in all probability not relevant to the ship call.
Minimize technical support prices. operating in conjunction with a technical support orhelp table cluster, the check team identifies the problems that result in involves support. These area unit typically peripherally associated with the merchandise beneath test--for example, obtaining the merchandise towork with a selected printer or to

ISSN-2320-7884 (ONLINE)
ISSN-2321-0257 (PRINT)

import knowledge with success from a 3rd party databasemight stop additional calls than a low-frequency, data-corrupting crash.

Assess correspondence to specification. Any claim created within the specification is checked.Program characteristics not self-addressed within the specification don't seem to be (as a part of thisobjective) checked.

Conform to laws. If a regulation specifies a precise variety of coverage (such as, atleast one check for each claim created regarding the product), the check cluster creates theappropriate tests. If the regulation specifies a mode for the specifications or different documentation, the check cluster in all probability checks the design. In general, the check cluster isfocusing on something coated by regulation and (in the context of this objective) nothingthat isn't coated by regulation.

Minimize protective causa risk. Any error that would result in Associate in Nursing accident or injury isof primary interest. Errors that result in loss of your time or knowledge or corrupt knowledge, however that don’tcarry a risk of injury or harm to physical things area unit out of scope.Find safe situations to be used of the merchandise (find ways in which to induce it to figure, in spite of thebugs). Sometimes, all that you’re longing for is a technique to try to to a task that may consistentlywork--one set of directions that somebody else will follow that may dependably deliver thebenefit they're speculated to result in. during this case, the tester isn't longing for bugs. He istrying out, by trial and error purification and documenting, how to try to to a task.

Assess quality. this is often a tough objective as a result of quality is multi-dimensional. The natureof quality depends on the character of the merchandise. as an example, a video game that isrock solid however not fun may be a lousy game. To assess quality -- to live and reportback on the extent of quality -- you almost certainly want a transparent definition of the foremost importantquality criteria for this product, and so you wish a theory that relates check results to the definition. as an example, responsibility isn't on the subject of the quantity of bugs within the product. Itis (or is usually outlined as being) regarding the quantity of reliability-related failures which will be expected during a amount of your time or a amount of use. (Reliability-related? In measuringreliability, a company may not care, as an example, regarding misspellings in errormessages.) to form this prediction, you wish a mathematically and by trial and error soundmodel that links check results to responsibility. Testing involves gathering the information required bythe model. This may involve intensive add areas of the merchandise believed to bestable still as some add weaker areas. Imagine a responsibility model supported counting bugs found (perhaps weighted by some variety of severity) per N lines of code orper K hours of testing. Finding the bugs is vital. Eliminating duplicates is vital.

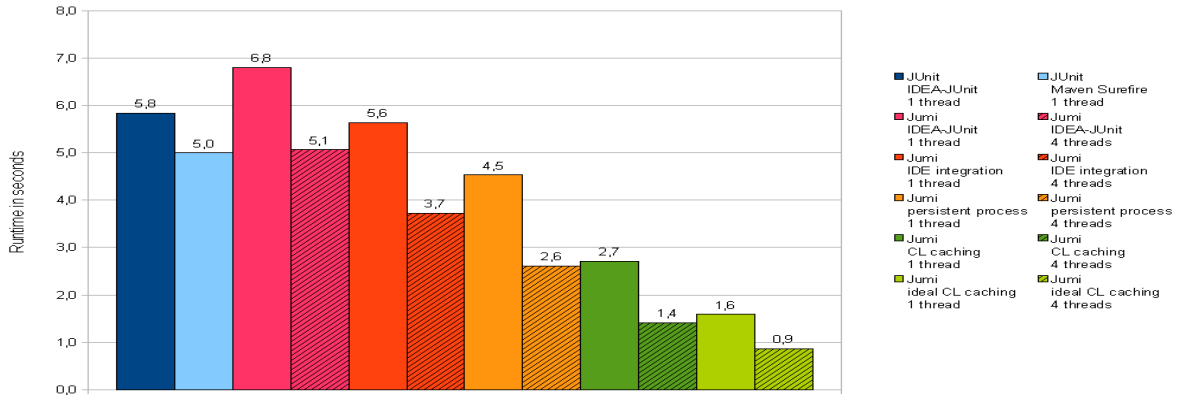
Troubleshooting to form the bug report easier to grasp and additional doubtless to repair is (inthe context of assessment) out of scope.

Verify correctness of the product:

It's not possible to try and do this by testing. will|you'll|you'll be able to} provethat the merchandise isn't correct otherwise you can demonstrate that you simply didn’t realize any errors in agiven amount of your time employing a given testing strategy. However, you can’t check thoroughly,and the product may fail below

ISSN-2320-7884 (ONLINE)
ISSN-2321-0257 (PRINT)

conditions that you simply didn't check. the most effective you'll be able to do (if you've got a solid, credible model) is assessment--test-based estimation of the likelihood of errors. (See the discussion of dependableness, above).



Assure quality. Despite the common title, quality assurance, you can't assure quality bytesting. You can't assure quality by gathering metrics. You can't assure quality by settingstandards. Quality assurance involves building a prime quality product and for that, youneed accomplished folks throughout development UN agency have time associate degreed motivation and anappropriate balance of direction and inventive freedom. this can be out of scope for a checkorganization. it's inside scope for the project manager and associated executives. Thetest organization will actually facilitate during this method by performing a good vary oftechnical investigations, however those investigations aren't quality assurance.We area unit at the tip of our cook's tour through JUnit. the subsequent figure shows the planning of JUnit at a look explained with patterns.

4.1 Junit Summary:

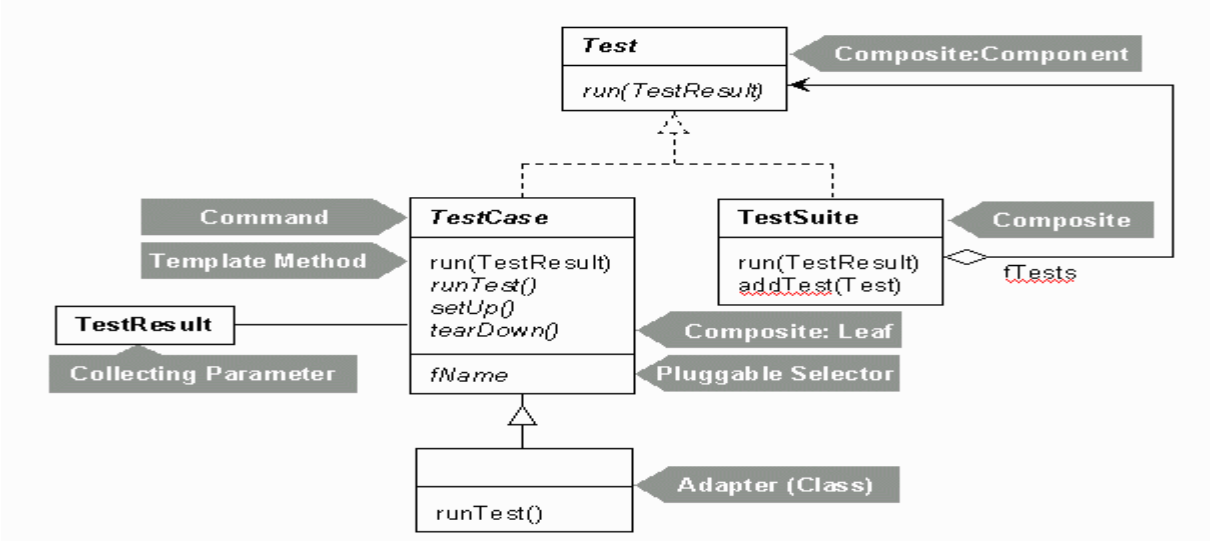


Figure 4.1: JUnit Patterns Summary

ISSN-2320-7884 (ONLINE)
ISSN-2321-0257 (PRINT)

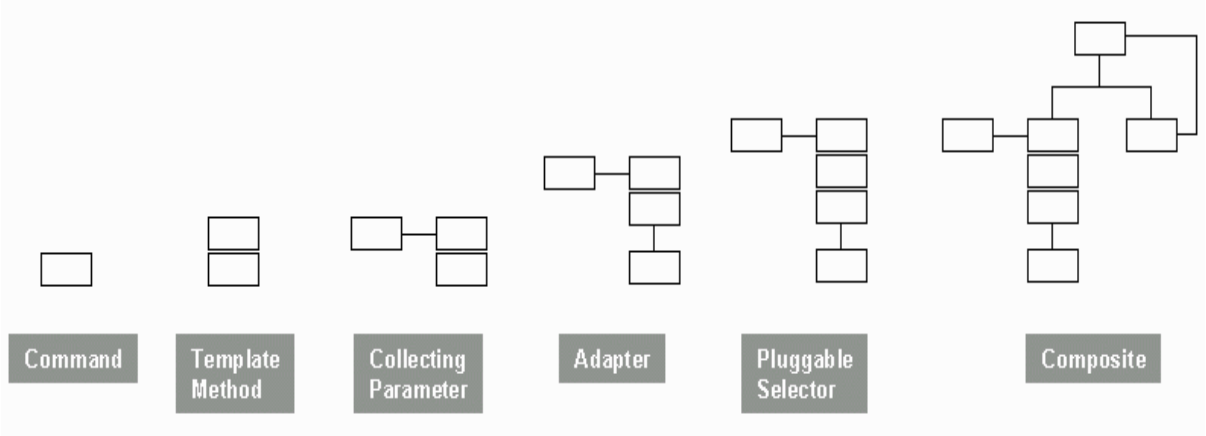


Figure 4.2: JUnit Pattern Storyboard

Notice how TestCase, the central abstraction in the framework, is involved in four patterns. Pictures of mature object designs show this same "pattern density". The star of the design has a rich set of relationships with the supporting players. Here is another way of looking at all of the patterns in JUnit. In this storyboard you see an abstract representation of the effect of each of the patterns in turn. So, the Command pattern creates the TestCase class, the Template Method pattern creates the run method, and so on. (The notation of the storyboard is the notation of figure 6 with all the text deleted).One point to notice about the storyboard is how the complexity of the picture jumps when we apply Composite.

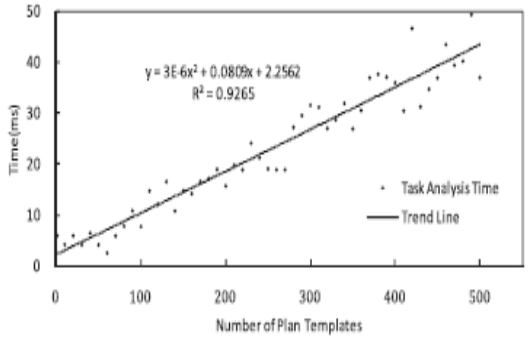


Fig-4.3 Time spent by task analyzer.

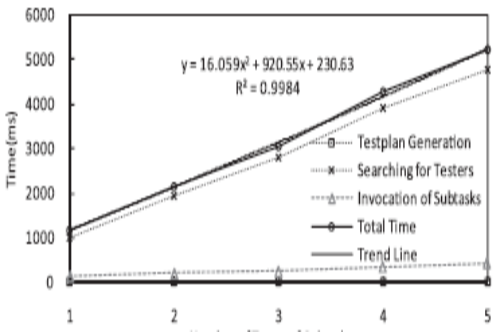


Fig-4.4 Time dependence on the number of subtasks.

ISSN-2320-7884 (ONLINE)
ISSN-2321-0257 (PRINT)

The time spent on 3.b only depends on the performanceof the tester (s). It is irrelevant to the efficiency of the broker.Therefore, it is omitted in our experiment. shows the average lengths of execution times ondifferent tasks with the number of different types ofsubtasks ranging from 1 to 5. A quadratic polynomialfigure fits the curve very well with R2 ¼ 0:9984.In summary, the experiments show that the broker iscapable of dealing with test problems of practical sizes with respect to the number of testers registered, the size of the knowledge base, and the complexity of test tasks.

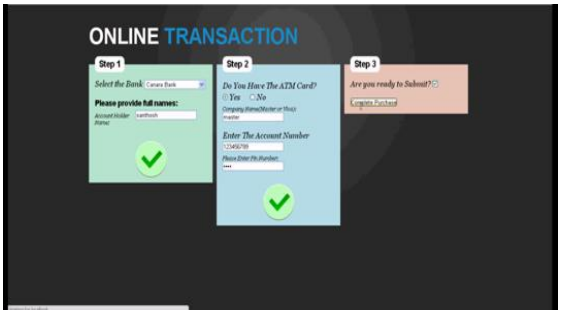
5. TESTING MODULUS

5.1Test/Functional Service Generation:

F-service should be accompanied with a special T-service so that test executions of the F-service can be performed by the corresponding T-service. Thus, the normal operation of the original F-service is not disturbed by test requests and the cost of testing are not charged asreal invocations of the F-service. The F-service provider can distinguish real requests from the test requests so that no real world effect is caused by test requests. F-service should also provide further support to other test activities. For example, the formal specification of the semantics of the service, the internal design, such as UML diagrams.

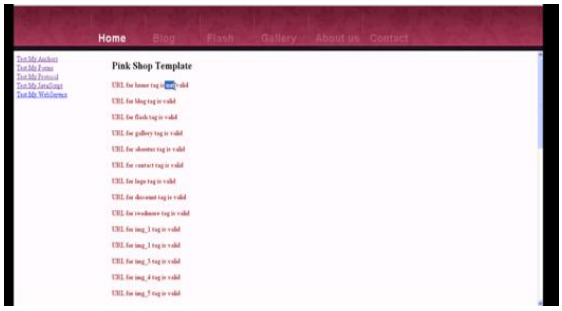
5.2 Test case generation:

Besides the service specific T-service that accompanies an F-service, a test service can also be a general purpose test tool that performs various test activities, such as test planning, test case generation, and test result checking, etc. A general purpose T-service can be specialized in certain testing techniques or methods such as the generation of test cases from WSDL. The test broker TB decomposes the test task into a sequence of subtasks and searches for appropriate testers for each subtask by submitting search requests to the registry. It then selects one tester for each subtask. In this example, we assume two testers TG and TE are selected.



5.3 Result Checking:

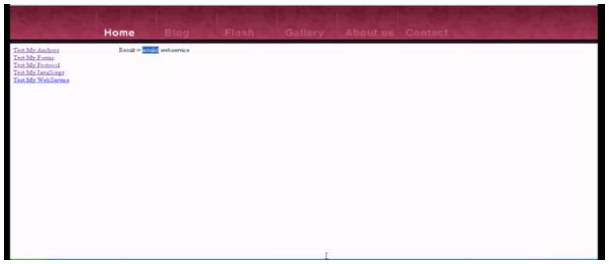
ISSN-2320-7884 (ONLINE)
ISSN-2321-0257 (PRINT)



After checking the trustworthy of tester TG, the insurer A's T-service releases its design model to TG. After successfully obtained the design model, TG produces a set of test cases and returns a test suite to the test broker TB. The test broker then passes the test cases to TE, requests for the test invocation of the insurer A's services using the test cases and requests it to check the output correctness and to measure the test coverage. TE performs these tasks by collaboration with the insurer A's T-services. The test results are then returned to the test broker TB.

5.4 Report Preparation:

Finally, TB assembles a test report containing information about test output correctness and test adequacy. The test report is sent to CIB, which is used to determine whether the dynamic link will take place.



6. CONCLUSION

In our proposed system, we presented service oriented architecture for testing Web Services. In this architecture, various T-services collaborate with each other to complete the test tasks. We employ the ontology of software testing STOWS to describe the capabilities of T-services and test tasks for the discovery, registration and invocation of T-services. The knowledge of intensive composition of T-services has realized by the development and employment of the test brokers, which are also called T-services. We have implemented the architecture in Semantic WS technology. Case studies have been demonstrated the feasibility of the architecture and illustrated how to wrap up general purpose testing tools and turn them into T-services and how to develop service specific T-services to support the testing of a WS. Experimental evaluation and also shows the scalability of the approach.

7.REFERENCES

1. F. McCabe, E. Newcomer, M. Champion, C. Ferris, and D.Orchard, Web Services Architecture, W3C Working Group Note, http://www.w3.org/TR/ws-arch,2004.

2. M. Stal, "Web Services: Beyond Component-Based Computing,"Comm. ACM, vol. 45, no. 10, pp. 71-76, Oct. 2002.

3. G. Canfora and M. Penta, "Service-Oriented Architectures Testing: A Survey," Software Eng.: Int'l Summer Schools (ISSSE 2006-2008),Revised Tutorial Lectures, A. Lucia and F. Ferrucci, eds., pp. 78-105, Springer-Verlag, 2009.

4. M. Bozkurt, M. Harman, and Y. Hassoun, "Testing Web Services:A Survey," Technical Report TR-10-01, Dept. of Computer Science,King's College London, Jan. 2010.

5. X. Bai, W. Dong, W. Tsai, and Y. Chen, "WSDL-Based AutomaticTest Case Generation for Web Services Testing," Proc. IEEE Int'l Workshop Service Oriented System Eng. (SOSE '05), pp. 215-220, Oct.2005.

6. W. Tsai, R. Paul, W. Song, and Z. Cao, "Coyote: An XML-Based Framework for Web Services

ISSN-2320-7884 (ONLINE)
ISSN-2321-0257 (PRINT)

Testing," Proc. IEEE Int'l Symp. High Assurance Systems Eng. (HASE '02), pp. 173-174, Oct. 2002.

7. N. Looker, M. Munro, and J. Xu, "WS-FIT: A Tool for Dependability Analysis of Web Services," Proc. 28th Ann. Int'l Computer Software and Applications Conf. (COMPSAC '04), pp. 120-123, Sept.2004.

8. J. Offutt and W. Xu, "Generating Test Cases for Web Services Using Data Perturbation," SIGSOFT Software Eng. Notes, vol. 29,no. 5, pp. 1-10, Sept. 2004.

9. S.C. Lee and J. Offutt, "Generating Test Cases for XML-Based Web Component Interactions Using Mutation Analysis," Proc. 12th Int'l Symp. Software Reliability Eng. (ISSRE '01), pp. 200-209, Nov. 2001.

10.W. Xu, J. Offutt, and J. Luo, "Testing Web Services by XMLPerturbation," Proc. IEEE 16th Int'l Symp. Software Reliability Eng.(ISSRE '05), pp. 257-266, Nov. 2005.

11.M.P. Emer, S.R. Vergilio, and M. Jino, "A Testing Approach for XML Schemas," Proc. 29th Ann. Int'l Conf. Computer Software and Applications Conf. (COMPSAC '05), pp. 57-62, July 2005.

12. A. Bertolino, J. Gao, and E. Marchetti, "XML Every-FlavorTesting," Proc. Second Int'l Conf. Web Information Systems andTechnologies (WEBIST '06), pp. 268-273, Apr. 2006.

13.A. Bertolino, J. Gao, E. Marchetti, and A. Polini, "Systematic Generation of XML Instances to Test Complex Software Applications,"Rapid Integration of Software Engineering Techniques,N. Guelfi, et al., eds., pp. 114-129, Springer, 2007.

14. A. Bertolino, J. Gao, E. Marchetti, and A. Polini, "Automatic Test Data Generation for XML Schema-Based Partition Testing," Proc. Second Int'l Workshop Automation of Software Test (AST '07), p. 4,May 2007.

15. J.B. Li and J. Miller, "Testing the Semantics of W3C XML Schema,"Proc. 29th Ann. Int'l Computer Software and Applications Conf. (COMPSAC '05), pp. 443-448, July 2005.

16. L.F. de Almeida and S.R. Vergilio, "Exploring Perturbation Based Testing for Web Services," Proc. IEEE Int'l Conf. Web Services (ICWS '06), pp. 717-726, Sept. 2006.

17. S. Hanna and M. Munro, "An Approach for WSDL-Based Automated Robustness Testing of Web Services," Information Systems Development: Challenges in Practice, Theory, and Education,C. Barry, et al., eds., vol. 2, pp. 493-504, Springer, 2009.

18. J. Garcı´a-Fanjul, J. Tuya, and C. de la Riva, "Generating Test Cases Specifications for BPEL Compositions of Web Services Using SPIN," Proc. Int'l Workshop Web Services Modeling and Testing (WS-MaTe), 2006.

19. C. Bartolini, A. Bertolino, E. Marchetti, and I. Parissis, "Data Flow-Based Validation of Web Services Compositions: Perspectives and Examples," Architecting Dependable Systems, R.V. Lemos, et al.,eds., pp. 298-325, Springer-Verlag, 2008.20.C. Bartolini, A. Bertolino, and E. Marchetti, "Introducing Service-Oriented Coverage Testing," Proc. IEEE/ACM 23rd Int'l Conf.Automated Software Eng. (ASE '08), pp. 57-64, 2008.