Testing Strategies for Evaluation of User Interfaces in SOA-based Systems

Piotr Chynał
Wrocław University of Technology
Wybrzeże Wyspiańskiego 27
50-370 Wrocław, Poland
piotr.chynal@pwr.wroc.pl

Paper presented at MIDI 2013 Conference, 24-25.06.2013, Warsaw, Poland.

Abstract

This paper presents a research that was conducted to determine the best techniques and testing strategies for user interface evaluation of SOA-based systems. Although we can find many articles on, for example, the design of web services, few guidelines exist on user interfaces and their usability for SOA-based systems. To achieve our goal we have performed various tests to determine which evaluation techniques are most suitable for such systems. The tests were conducted on a system which was created at our university – Platel. As a result we came up with a set of most effective techniques for user interface evaluation, and some guidelines for testing strategies, for SOA-based systems.

Author Keywords

HCI; SOA; Usability; User Interface

ACM Classification Keywords

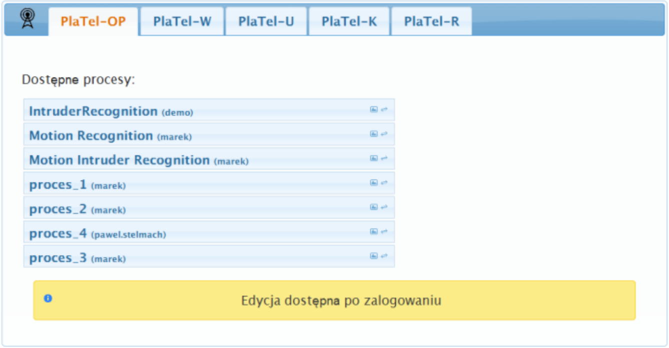
HCI

Introduction

SOA (Service Oriented Architecture) paradigm delivers general methodology for creating information systems in form of interoperable services [9]. Systems designed using this paradigm are composed of separate services, which are executed on demand. The problem of design and evaluation of user interface (UI) for services in SOA-based systems is quite new. In such systems, user interface is usually adapted to the personal users' needs and/or environment settings [7]. Although the functionality of a SOA system is implemented in a scattered manner, users who operate the system also need a consistent look and intuitive interaction with the interface. To ensure that we can perform various types of quality and usability tests.

Usability testing is one of the methods of software testing. We can evaluate usability of web systems and desktop and mobile applications. According to the norm ISO 9241 we can define usability as "The extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency, and satisfaction in a specified context of use" [6]. Usability testing allows us to increase lernability, satisfaction and efficiency for the given system and also thanks to such tests we can find and eliminate errors.

To validate the quality of user interface we can also use other tests and methods, for example, white-box and black-box testing [3]. To validate which of the known methods of user interface testing best apply to the SOA-based systems, we decided to evaluate the UI of a system that was created at our university - PlaTel, which is an application of SOA paradigm in service composition and execution [12]. Platel system was implemented in jQuery and HTML 5 with CSS (Fig. 1).



Dostępne procesy:

IntruderRecognition (demo)
Motion Recognition (marek)
Motion Intruder Recognition (marek)
proces_1 (marek)
proces_2 (marek)
proces_4 (pawel.stelmach)
proces_3 (marek)

Edycja dostępna po zalogowaniu

Fig. 1. Evaluated SOA-based system – Platel

Testing Methods

Little is known about how to maximize usability and quality of SOA-based systems. Developers of such systems face multiple challenges, such as [2]:

- the difficulty of web service configuration,
- fragmentation of the discovery user experience across multiple interfaces,
- confusing hierarchies of service navigation,
- business modeling support,
- hidden relationships between services,
- inconsistent design.

To ensure that we can fully evaluate the quality of user interfaces in SOA-based system, we decided to use various testing techniques on our system and determine which are the most effective. We took under consideration many popular methods and come up with the list of techniques that would be most effective for SOA-based systems. We eliminated for example techniques like eyetracking [5] and clicktracking [11], which would not provide valuable results, because the interfaces in SOA application are often reloaded dynamically with output from web services, and those methods are more effective for static interfaces, like in typical HTML websites. In the end we have chosen the following methods:

Functional tests

Functional test (black-box testing) is a method of software testing that evaluates the functionality of an application. For functional testing we have decided to conduct a test with users that would perform specified tasks in the system and test its particular functionalities. Focus group testing is a popular method, involving users, for evaluation of designs and interfaces [8]. The participants are encouraged to freely give their honest opinions about the product, including suggestions on how to make it better. We wanted to perform such test especially focusing participants' attention on system functionalities.

Structural tests

Structural tests (white box testing) are a method of testing software that evaluates internal structures of an application. In this method we wanted to design test cases to test the system by entering on input such data, which would go through each implemented path, using a mouse movement and mouse clicks recording and playback software.

Application portability testing

Nowadays, when users often use mobile devices and various browsers, adaptation is one of the key challenges for web application developers, so we wanted to check the adaptation of the user interface by running the application on different platforms and in various environments.

Quality tests

For the quality tests we wanted to perform standard usability research using techniques such as expert evaluation [1], heuristic evaluation [10] and a test with users [11]. Moreover we decided to test the quality of application using various validators, for example CSS and Readability validators.

Case Study

The evaluation of Platel system consisted of the following tests:

Functional tests

The purpose for focus group research of the Platel system was to obtain remarks concerning participants experience with the system, its functionalities and usability. We have prepared tasks for users, which allowed testing all the above factors. The participants' tasks were for example:

- executing application forms
- sending input date to the particular web service

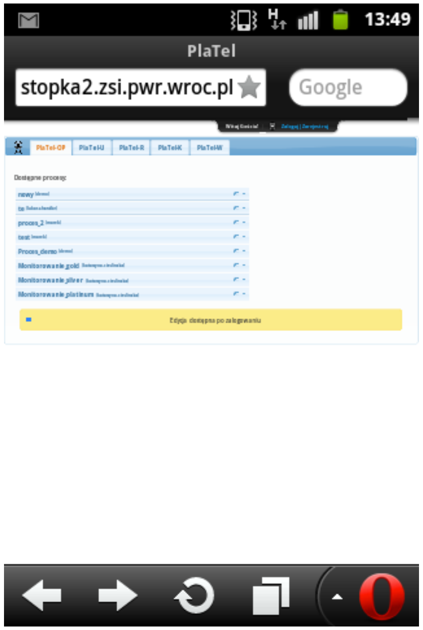


Fig. 2. Platel on Android system

- executing web services

The evaluation has been divided into two sessions. In the first one we have performed a test with system developers (6 people – 5 male and 1 female) that were already familiar with the system. In the second session we have evaluated 8 (7 male and 1 female) students that never used this system before.

During the discussion, participants mentioned various problems and qualities of the Platel system. Moreover we prepared a survey for the participants, in which they wrote their feedback regarding their experience with the system

Structural tests

For structural tests we have used Selenium IDE1 (Integrated Development Environment) – a tool that allows to create test scripts prototypes. It can be run as a Firefox browser plugin for automated testing. It works in two steps – recording and playback and can be run with various configuration options and parameters (Fig. 2).

Application portability testing

For the portability testing, Platel application was launched on various environments and operating systems and we tested how each functionality works in them. First we checked different Internet browsers in the Windows OS:

- Opera 11.61
- Firefox 8
- Chrome 17
- Internet Explorer 9

Then we run the application on Safari 5.1.3 browser on Mac OS X in version 10.7.3, on iMac computer. We also checked mobile devices with various Android versions, Android SDK emulator and Apple's devices - iPad 2 16GB 3G and iPhone 4 16GB with iOS 5.0.1.

Quality tests

For the quality evaluation we have conducted expert evaluation for guest account and for logged users. We have also conducted heuristic evaluation of the Platel system – we have examined the interface and evaluated its compliance with recognized usability principles, based on 10 Nielsen Heuristics [10]. For the test with users we used the focus group evaluation that was conducted during functional tests. Moreover we have checked the application using various validators, such as W3C HTML 52 syntax and CSS validator3, JuicyStudio Readability test4 - test that determines the readability of the application, Vischeck5 - colorblind

[1] http://seleniumhq.org/

[2] http://validator.w3.org/

[3] http://jigsaw.w3.org/css-validator/

[4] http://juicystudio.com/services/readability.php

[5] http://www.vischeck.com/

vision simulator, W3C Link Checker6 - checks links and anchors and W3C Unicorn7 - unified W3C validator.

Results of Case Study

After performing all the tests we have gained a lot of valuable data regarding Platel usability:

Functional tests

During the focus group evaluation, participants identified 11 critical, 7 medium and 7 minor problems. Both groups of users (developers and students) have identified similar problems with the system. We have also obtained opinions about the product, including suggestions, how to make it better. The feedback from the survey was also valuable, because not every participant was active during the discussion, and this way we could get detailed information from everybody.

Participants assessed that getting around the system was intuitive (average was 4.2 in rating scale from 1 to 5) and the system is easy to use (4.1 rating). However they had a lot of reservations about the functionality of the system by highlighting in particular a large number of inconsistent solutions in the system and assessed system functionality very low (2.0 rating).

Structural tests

While running Selenium application with test scripts set on fast execution we have encountered errors. Some web services have longer loading times then other, which resulted in errors, because particular elements were missing and the application could not get to them. When Selenium was set to execute the scripts slower, it did not encountered any problems, and we were able to check each implemented path in the interface.

Application portability testing

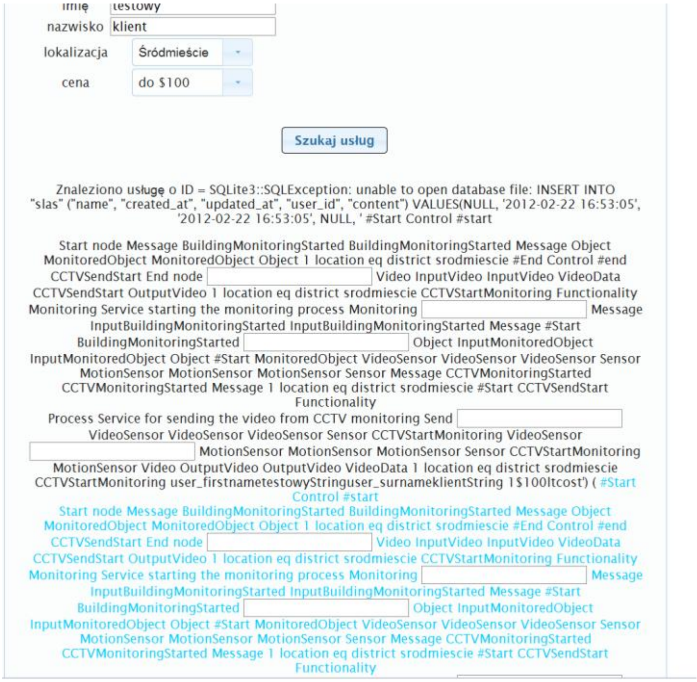
During evaluation of Platel system on different browsers on Windows OS, only in the IE browser few elements of interface were not shown properly. On Mac OS X every element of the system was working correctly, but on mobile Apple devices we have encountered many problems with visualizations of web services – there were problems with panel resolution and with interaction using gestures. On Android system it was even worse, the jQuery scripts were not loading at all, and there were also problems with resolution, Platel was displayed only on the part of the screen. However while running the application in the Opera browser instead of default Android browser, only visualization of web services was not working correctly (same as on Apple mobile devices).

Quality tests

First quality test was the expert evaluation. Our three experts have found 8 critical, 7 medium and 4 minor usability problems. Same as in the focus group test, most of the problems were related to application functionality (Fig. 3).

[6] http://validator.w3.org/checklink

[7] http://validator.w3.org/unicorn/



imię testowy
nazwisko klient
lokalizacja Śródmieście
cena do $100

```
Znaleziono usługę o ID = SQLite3::SQLException: unable to open database file: INSERT INTO
"slas" ("name", "created_at", "updated_at", "user_id", "content") VALUES(NULL, '2012-02-22 16:53:05',
'2012-02-22 16:53:05', NULL, ' #Start Control #start

Start node Message BuildingMonitoringStarted BuildingMonitoringStarted Message Object
MonitoredObject MonitoredObject Object 1 location eq district srodmiescie #End Control #end
CCTVSendStart End node Video InputVideo InputVideo VideoData
CCTVSendStart OutputVideo 1 location eq district srodmiescie CCTVStartMonitoring Functionality
Monitoring Service starting the monitoring process Monitoring Message
InputBuildingMonitoringStarted InputBuildingMonitoringStarted Message #Start
BuildingMonitoringStarted Object InputMonitoredObject
InputMonitoredObject Object #Start MonitoredObject VideoSensor VideoSensor VideoSensor Sensor
MotionSensor MotionSensor MotionSensor Sensor Message CCTVMonitoringStarted
CCTVMonitoringStarted Message 1 location eq district srodmiescie #Start CCTVSendStart
Functionality
Process Service for sending the video from CCTV monitoring Send
VideoSensor VideoSensor VideoSensor Sensor CCTVStartMonitoring VideoSensor
MotionSensor MotionSensor MotionSensor Sensor CCTVStartMonitoring
MotionSensor Video OutputVideo OutputVideo VideoData 1 location eq district srodmiescie
CCTVStartMonitoring user_firstnametestowyStringuser_surnameklientString 1$100ltcost') ( #Start
Control #start
Start node Message BuildingMonitoringStarted BuildingMonitoringStarted Message Object
MonitoredObject MonitoredObject Object 1 location eq district srodmiescie #End Control #end
CCTVSendStart End node Video InputVideo InputVideo VideoData
CCTVSendStart OutputVideo 1 location eq district srodmiescie CCTVStartMonitoring Functionality
Monitoring Service starting the monitoring process Monitoring Message
InputBuildingMonitoringStarted InputBuildingMonitoringStarted Message #Start
BuildingMonitoringStarted Object InputMonitoredObject
InputMonitoredObject Object #Start MonitoredObject VideoSensor VideoSensor VideoSensor Sensor
MotionSensor MotionSensor MotionSensor Sensor Message CCTVMonitoringStarted
CCTVMonitoringStarted Message 1 location eq district srodmiescie #Start CCTVSendStart
Functionality
```

Fig. 3. Functional errors in the Platel application

For the heuristic evaluation almost every heuristic was violated. Most of the times it was heuristic number five – "errors prevention". Other often violated heuristics were – "match between system and the real world" and "consistency and standards". The only heuristic that was not broken in the system was heuristic number three – "user control and freedom" (Fig. 4).

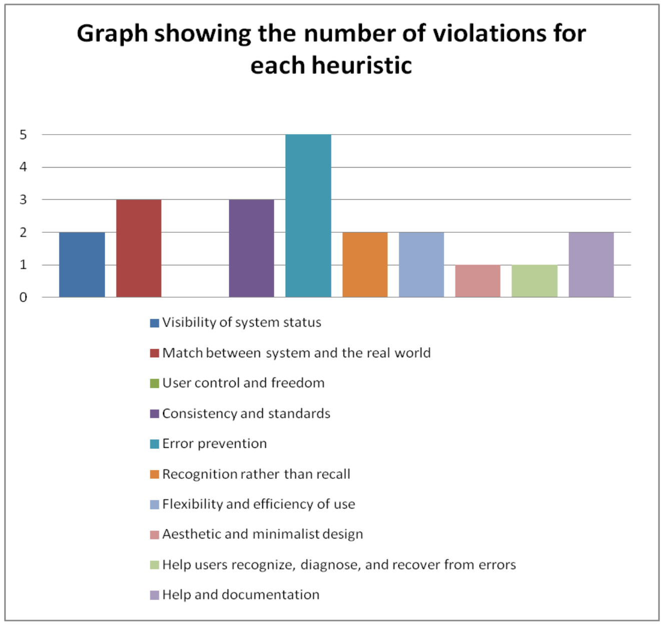


Fig. 4. Graph showing the number of violations for each heuristic

The last test was the evaluation using the most popular validators. HTML 5 and CSS syntax validator reported only few errors and other tests had positive output – the application was readable, applied colors were visible and all the links and anchors were valid.

Analysis of Results

After gathering the results we were able to assess how each of the test methods has helped us in evaluating the user interface in a SOA-based system.

First of all the most valuable tests in terms of interface usability problems found, were focus group and expert evaluation. We have compared those both methods in terms of their effectiveness (accuracy, cost, completeness and time) and focus group turned out to be slightly more effective for our system [4]. However both methods can be used together to ensure that the interface usability was thoroughly evaluated. Instead of a focus group we could perform individual test with users, but nonetheless it is important to design such test in a way, that users would focus on testing the execution of web services with various input data.

As for the structural tests, they are an effective way to test SOA-based systems, because they allow to easily check each implemented path in the interface. However it is not easy to automate this process, because in the interfaces which consist of many web services, with various loading times and input data required, it is hard to create test scripts that would go through every path without problems. For smaller systems it might is easier just to perform cognitive walkthrough [1].

Application portability testing has shown that it is very hard to create SOA-based systems that would be compatible with every environment. It is a problem that developers will face in the near future, when more and more mobile applications will be written according to the SOA paradigm.

The test in which we used various validators did not provide any interesting results and can be treated as an addition to other tests.

Heuristic evaluation has only confirmed the problems that were found during expert evaluation. Comparing its efficiency with expert evaluation using the method described in [4], we obtained similar score. During expert evaluation the evaluators often have Nielsen heuristics in mind, so both of these methods are pretty much the same and can be used interchangeably.

Summary

By adapting various software testing techniques to the unique demands of SOA-based systems, valuable insights can be gained to help improve their usability and quality. While performing this evaluation we have found many usability issues and we were able to vastly improve Platel system after this research. Moreover we have obtained many interesting conclusions and testing strategies for SOA-based systems user interface evaluation

In conclusion, we can propose a testing strategy for usability evaluation of SOA-based system, which consists of:

- test with users (focus group or individual test),
- structural test,
- expert evaluation.

During such research we should focus on functionality and on proper testing of collaboration between web services and the user interface. SOA-based systems usually have simple UI and their main usability problems lie in error prevention and data visualization.

In addition, if our application is indented for various environments we should also perform portability testing.

SOA paradigm in web/mobile applications design is a relatively new trend, so there is still plenty more to research in this area. This work, along with the article concerning only usability testing of such systems [2], presents a first take on SOA-base system user interface evaluation, and can be a good starting point for future works on this subject.

References

[1] Barnum C.M. Usability testing and research, Longman (2002)

[2] Beaton J., Myers B.A., Stylos J., Jeong S.Y., Xie Y. Usability Evaluation for Enterprise SOA APIs. SDSOA'08, May 11, 2008, Leipzig, Germany (2008)

[3] Black R. Managing the Testing Process (2nd ed.). Wiley 4 Publishing (2002)

[4] Chynał P. A method for comparing efficiency of the different usability evaluation techniques. In: Information systems architecture and technology : Web Engineering and High-Performance Computing on Complex Environments Oficyna Wydawnicza Politechniki Wrocławskiej, 2012. pp. 51-57 (2012)

[5] Duchowski A. T. Eye tracking methodology: Theory and practice, London, Springer-Verlag Ltd (2007)

[6] International Standard ISO 9241-11. Ergonomic requirements for office work with visual display terminals (VDTs) – Part 11: Guidance on Usability. ISO 1997

[7] Kobsa, A. Personalized Hypermedia and International Privacy. Communications of the ACM, vol. 45, no 5, pp. 64-67 (2002)

[8] Kruger, R.A., Casey, M.A. Focus groups: A practical guide for applied research. Thousand Oaks, CA: Sage, pp 107 – 113 (1999)

[9] Newcomer, E., Lomow, G. Understanding SOA with Web Services, Addison Wesley Professional, (2004)

[10] Nielsen, J. Heuristic evaluation. In Nielsen, J., and Mack, R.L. (Eds.), Usability Inspection Methods, John Wiley & Sons, New York, NY (1994)

[11] Pearrow M. Web Site Usability Handbook. Charles River Media (2007)

[12] Stelmach, P. Juszczyszyn, K., Prusiewicz, A., Świątek, P. Service Composition in Knowledge-based SOA Systems. New Generation Computing, vol. 30, no 2&3 pp. 165-188 (2012)