Evaluating the suitability of dynamic Web platforms for teaching exception handling

Johnson Dehinbo
Dehinbooj@tut.ac.za
Department of Web & Multimedia Applications, Tshwane University of Technology, Pretoria, 0001, South Africa

Abstract

Due to the availability of many platforms for developing dynamic Web applications, there is the problem of choosing the most appropriate platform for teaching the concepts of web applications development to undergraduate students in tertiary institutions. Students will not perform at their best capacity level if the platform chosen by the institution is not very suitable for teaching the relevant concepts. As part of the framework to determine the most suitable platform for teaching web applications development in tertiary institutions, this study establishes a set of criteria for evaluating the suitability for teaching the exception handling in Web applications. These criteria were tested by evaluating four platforms namely Java Servlets, Java Server Pages, Active Server Pages and PHP using various research methods including descriptive inquiry, document analysis, observations and programming tests. PHP was found to be most suitable.

Keywords: programming, languages, Web development platforms, exception handlings

1. BACKGROUND OF THE STUDY

This study establishes part of a framework containing various criteria that can be used to evaluate dynamic Web platforms to determine a suitable platform for teaching Web applications development in tertiary institutions. This part of the framework is to determine platform that will be suitable for teaching exception handling to undergraduate students. The purpose of the study is not to persuade readers that one platform is better than the other, but to help readers make a more informed decision with regard to the suitability for teaching Web applications development in tertiary institutions.

The context of the problems leading to the study

Today, there are many platforms for implementing the dynamic applications on the World Wide Web. A problem is how to make the best choice. The choice made may have an effect on the on the efficiency of the students and robustness of the developed systems. Web application development students will not perform at their best capacity if the dynamic Web platform chosen by the institution is not well suited for teaching exception handling concepts that forms the backbone of robust programming. This is very important in Web applications where the state of resources scattered over different parts of the Web can not be predicted.

As noted by Lim (2002:1), the advent of the World Wide Web has changed the way computer software is built, but it has not caused many academics to change their way of teaching computing. A good way of teaching Web application development is to teach using a platform that is suitable for the students.

According to Sebesta (1996:2-3), it is widely believed that the depth at which we can think is influenced by the expressive power of the language in which we can communicate our thoughts. Sebesta further indicates that programmers in the process of developing software are similarly constrained. The language in which they develop software places limits on the kinds of control structures, data structures, and abstractions they can use; thus the form of algorithms they can construct are also limited.

Some institutions teach using various programming and scripting languages. These led to elements of repetition and confusion. But for undergraduates learning various aspects of programming, comprehension is very important (Wiedenbeck,1999:5). Therefore, the suitability for teaching relevant concepts that forms the backbone of programming needs to be taken into consideration while choosing the platform to be used.

The research question

The research question is: How do we determine the dynamic Web platform that will be suitable for teaching exception handling to undergraduate students of Web application development?

The objectives of the study

The objectives of the study are therefore given below:

1.) Identify the exception handling concepts that should be taught to undergraduate students of Web applications development in tertiary institutions.

2.) Establish the criteria to determine suitability for teaching the concepts.

3.) Apply the criteria by using them to evaluate the suitability of four specified platforms.

The importance of the study

The main benefit of the study will be the educative choice of suitable dynamic Web platform to enhance the teaching of exception handling concepts. This will thereby increase the students' potentials in such a way that would lead to higher productivity in the Web application development industries. It will therefore be of primary benefit to institutions teaching using the platforms. The secondary beneficiaries will be users that desire and use robust Web applications.

2. LITERATURE REVIEW

Evaluating programming languages, development platforms and tools is very important for understanding the effects on novice programmers, but is difficult to carry out (Wiedenbeck et al, 1999). This could be the reasons why there have been various approaches towards the comparisons of programming languages and platforms. Apte et al. (2003) note that a study of existing literature showed varying conclusions about the superiority of one dynamic Web platform over another. Prechelt (2000) indicates that when it comes to the advantages and disadvantages of various programming languages, programmers and computer scientists usually hold strong opinions. These are illustrated in the presentations of various studies given below.

Previous related studies

Various other studies compared programming languages and dynamic Web platforms. In the survey of middleware platforms by Cooper (2001), it was concluded that ColdFusion is fast to learn and fast to use, and Common Gateway Interface (CGI) is also fast to learn. He then mentioned that Servlets are hard to learn and use, even by someone who already knows Java.

Bishop and Hurter (1999) examined some competitors to Java, namely the Scripting languages; Tcl/Tk, Perl and Python and found out that Python incorporated the features of Modula-3 into its scripting syntax thus making it suitable for "programming in the large", unlike Tcl and Perl. Programs in Python were also found to be typically much shorter than equivalent in C or Java for several reasons. Firstly, the high-level dynamic data types allow you to express complex operations in a single statement. Secondly, statement grouping is done by indentation instead of begin-end brackets. Thirdly, no variable or argument declarations are necessary.

In an empirical comparison of seven programming languages, Prechelt (2000) observes that designing and writing programs in the scripting languages, namely Perl, Python, Rexx, or Tcl takes no more than half as much time as writing it in C, C++, or Java. Moreover, the resulting program is only half as long. He therefore concluded that the scripting languages offer reasonable alternatives to other full programming languages, and they may offer significant advantages with respect to programmer productivity, at least for reasonably small programs.

Kruse (2003) illustrates the differences in the strengths and weaknesses of Personal Home Page (PHP) and Java. Klopper (2003) compares Personal Home Page (PHP), Active Server Page (ASP) and Java Server Page (JSP) in terms of their advantages and architectures. However, most of these studies did not use any criterion as a basis for the comparisons. The comparisons in these studies are based on intuition rather than scientific facts. Comparison must be based on a variety of factors supported by scientific facts and results. This is in line with Ashenfelter's (1999:105) assertion that before analyzing tools, it is worth discussing how to evaluate them.

Towards frameworks for performance comparisons

Vinoski (2003) realizes that various comparisons of programming languages concentrate on performance comparisons. Renaud et al (2003) indicate various metrics can be used to measure performance of algorithms in distributed systems. These include response or waiting time, synch delay, number of messages exchanged, throughput, communication delay, node fairness, Central Processing Unit (CPU) cycle usage, and memory usage. They however used response time.

Cooper (2001) estimated the response time for some platforms with ColdFussion having the best performance. In comparing the performance of Java Servlets, Java Server Pages (JSP), Active Server Pages (ASP) and PHP, Dehinbo (2005) also found PHP having the best performance. Marshak and Levy (2003) also evaluated platforms only in terms of user-perceived latency.

It should be pointed out however, that performance has somehow been overemphasized in various studies. This view is shared by Vinoski (2003), who agrees that a suitable framework for comparison should involve other relevant factors. Vinoski (2003) explains that people check only those qualities that are easily measurable, such as performance. He goes on to say that an interesting side effect of this is that it has unintentionally led many programming language users to presume that "high performance" is the same as "high quality". Meanwhile, such works could be entirely meaningless, depending on the nature of one's application. On the other hand, other studies introduce various criteria into their comparisons.

Studies using various criteria in their comparisons

Cecchet et al (2003) evaluate three specific mechanisms namely PHP, Java Servlets, and Enterprise Java Beans (EJB) with respect to performance and ease of development. The study attributes PHP's better performance to the fact that it executes as a module in the Web server, sharing the same process (address space), thereby minimizing communication overhead between the Web server and the scripts. This is unlike Java Servlets which run in a Java Virtual Machine (JVM) as a separate process from the Web server and so can even be placed on a separate machine (Cecchet et al, 2003).

In terms of ease of development, Cecchet et al (2003) explain that PHP scripts are easy to write because they can be seen as an extension of the HTML language that embeds code directly into an HTML page. However, he mentions that of concern is the fact that the database interfaces of PHP are ad hoc and code maintenance for database is awkward because new code needs to be written for each new database to which the scripts need access. On the other hand, Java Servlets access the database using JDBC which makes them easily portable between databases.

Hartman (2001) examined some tools for creating dynamic websites namely ASP, PHP and ASP.NET. He mentioned some factors that complicate choosing a scripting environment. First, there is the issue of culture among developers which has a lot to do with the ideological camps to which they belong. If they love to tinker with source code because it lets them develop solutions that are more efficient than off-the-shelf products, and if their cubicles are embellished with defaced portraits of Bill Gates, it is a good bet that they will prefer to use PHP. If they love the convenience and efficiency of existing integrated technology solutions, they would probably prefer to use ASP. He further mentioned that he has encountered very few developers who are equally willing to use both, or who can talk about "the other" technology without a trace of disdain.

The second factor that complicates choosing a scripting environment is the website's future scalability and functional requirements, although hard to predict. The choice between JSP, PHP and ASP (or its successor ASP.NET) might restrict which servers and platforms the site could run on or impact the feasibility of developing future features, such as database-linked connectivity with extranet partner sites (Hartman, 2001).

This section has highlighted the need to evaluate platforms using various criteria. However, the most important criterion is the suitability for doing the job for which a tool is needed. This view is shared by the following studies.

Studies emphasizing suitability for achieving the purpose of systems

According to Lim (2002) information systems / computer science departments need to reexamine their curricula in order to prepare students to face the challenge of being productive in a computing world that is now swamped with web technologies. The choice of dynamic Web platform for teaching students needs to be backed with evidence from relevant literatures, information from practicing web developers and empirical experimental programming results. This will lead to critical evaluation of the dynamic Web platforms, in line with the ideas, put forward by Ashenfelter in the statement below.

Web development tools need to be analyzed in terms of its purpose (what it is designed to do), technology (ease of use, robustness, scalability, security, performance, etc.), support (portability, cost, ISP support), and how well it works in the real world. (Ashenfelter, 1999:105). Similarly, in the process of choosing a language or platform that is usable and suitable for teaching introductory programming, Holt et al. (1997) lists the following criteria:

- ➢ It should be appropriate for introducing programming concepts used in the real world such as in business, science and government.
- ➢ It should encourage systematic problem solving.
- ➢ It should be small, convenient and easy to master.
- ➢ It should be easy to implement on economical processors and compilers.

Furthermore, Kolling and Rosenberg (1996) suggest the following sets of principles when making decisions about language issues:

- ➢ No conceptual redundancy. Achieving the same thing in a variety of ways can mean flexibility to the expert, but is usually confusion to the beginner.
- ➢ Clean concepts. The concepts of the language should be presented in a way that directly reflects the theoretical model.
- ➢ Readability. Achievable by favoring expressive keywords. This enables students to learn example programs and to re-read their own programs.
- ➢ Software Engineering support. This is necessary to avail mechanisms and guidelines that support good program development.

Moreover, Hadjerrouit (1998) evaluates the suitability of Java as a first programming language using the following criteria:

- ➢ Programming concepts to be taught. These include problem solving skills, algorithmic thinking and structured programming.
- ➢ Novice usability. How sufficiently simple the language is.
- ➢ Marketability. Student would want to be taught using languages that sells well.
- ➢ Use in subsequent courses. Knowledge gained should be useful in later studies.
- ➢ Programming paradigm support. The language should support the desired paradigm that students need to be exposed to.
- ➢ Motivation. There should be some enthusiasm about the language.

These studies emphasize the need for the suitability for doing the job for which a tool is needed, which in this case is the teaching of Web applications development. This however involves the suitability for teaching exception handling concepts.

Summary of the literature review

Choosing a suitable tool should involve exhaustive evaluations of various options based on various relevant criteria that are backed by scientific facts and results. Some previous comparisons of programming languages and platforms are based on intuition, while others simply concentrate on performance. Other comparisons were not aimed at a specific need. This study fills this gap in the body of knowledge by being unique in the following ways:

This study does not agree that it is sufficient to list the advantages and strengths of each programming language or platform. Rather, the advantages and strengths should be examined and ranked in the light of certain desired qualities relevant to specific use. For example, the ease of learning of a platform increases productivity in institutions training middle-level workers than the flexibility of the platform.

Most importantly, the study addresses the need for comparisons to be done with reference to specific use such as the suitability for the teaching of Web applications development. Therefore, we find it necessary to first identify the desirable concepts that will provide valuable programming background for the students. We can then evaluate the platforms according to the suitability for teaching these concepts as well as the satisfaction of other important constraints.

This study therefore establishes part of a comprehensive and yet specific framework. The framework addresses suitability for teaching other concepts such as structured programming, object-oriented programming, Web techniques, remote database management, file processing and XML support. The constraints include ease of use, performance, affordability and portability.

3. RESEARCH DESIGN AND METHODOLOGY

The research design will involve elements of descriptive and comparative studies. McMillan and Schumacher (2001:33) state that while a descriptive study describes a system with the aim of characterizing it as it is, by using numbers, comparative study investigates the differences, thereby taking descriptive study a step further. We therefore use a descriptive approach to characterize the platforms in order to compare them with the goal of determining the most suitable platform.

Research methods adopted to obtain the results

Evaluation of the platforms using a framework will involve the use of modeling. According to Bowling (2002:141), models are abstract representations of the essential characteristics of phenomena of interest, thereby making explicit, the relationships and or comparison between the characteristics. The form of modeling used in this study will consist of a set of criteria that will be established to measure the suitability for teaching the exception handling concepts.

The measurement of the suitability for teaching the exception handling concepts by the platforms will involve the use of descriptive modes of inquiry to characterize the features of the platforms. We find it necessary to first identify the exception handling concepts. This will enable us to establish some criteria to evaluate and compare the platforms with respect to the suitability for teaching the identified concepts. We can then evaluate the platforms according to the suitability for teaching these concepts. We eventually find out the features of the specified platforms from various sources and other established body of knowledge, and assign scores to the platforms based on the availability of the necessary features.

Sources of information to be used: To accomplish the above, we seek answers to the questions and the availability of features that will serve as the criteria. We seek these answers from established texts, journals and authoritative websites, which include those written by the designers of the platforms. These will be augmented by authoritative websites for the applicable web servers such as IIS and tomcat.

We also physically examine the handling of exceptions in the various platforms. Also important is the presentation of the error reports to users in the various Web platforms.

Measuring scale to be used: Using close-ended "Yes/No" questions, the measuring tool has values on a scale of 1 to 3, where:
3 = "Yes",
2 = "Not quite / with some workaround",
1= "No".

We have used a scale of 1 to 3 to avoid subjective situations where it will be difficult to distinguish between, for example, a score of 3 or 4 in a scale of 1 to 5. The use of the 1 to 3 scale therefore reduces the situation as to whether or not a facility is available, or in between.

Establishing reliability and validity: It is important that a measuring scale or instrument be consistent and reliable. It should produce more or less the same accurate results every time it is applied, even when applied by different persons (Coertze & Heath, 1997:78). Also, Coertze and Heath (1997:79) indicate that validity is concerned with soundness or the effectiveness of the measuring instrument. Validity raises questions such as: Does the test measure what it is supposed to measure? How well? How comprehensively? How accurately does it measure?

As a way of increasing validity, answers to the criteria questions will be sought from established and recognized textbooks (including text written by designers of the platforms), authoritative websites, and journal articles. We provide the accompanying references so that interested readers can verify or seek more information. This is supplemented with practical experiences and program tests confirming the satisfaction of some of the criteria established.

Also, to increase reliability, the quantitative characterization and evaluation using numbers will enhance clarity in the choice of platform with the highest score. This is unlike just using qualitative sentences to evaluate the platforms, at the end of which it is difficult to say which platform is really more suitable. Furthermore, the range of values applicable between 1 and 3 instead of say between 1 and 5 will avoid subjective situations where it will be difficult to distinguish between, say, a score of 3 or 4 in a scale of 1 to 5 and therefore increase reliability.

Data analysis: The analysis for the data for the study was done using simple statistical parametric analysis, such as sums and means. The scores for all the criteria were summed up for each web based dynamic platform to obtain a total score.

4. DEVELOPMENT OF THE CRITERIA FOR THE EVALUATIONS

It is of primary importance that a suitable platform must be capable of satisfying the purpose for which it is needed. This, in this case implies that it must be suitable for the purpose of teaching desirable concepts. This involves identifying the exception handling concepts and then evaluating the platforms according to the suitability for teaching these concepts.

Exception handling in Web applications

One of the main difficulties of writing a program is making sure that it is robust. That is, when something unexpected happens (e.g. an invalid piece of data has been typed in by the user or an access to the Internet has failed), we have to ensure that the program can detect what has happened and do something about it (Garside & Mariani, 2003:351). The ability of a program to intercept run-time errors (as well as other unusual conditions detected by the program), take corrective measures and then continue is a great aid to reliability (Sebesta, 1996:17).

According to Farrell (2003:542), an exception is an error or unexpected condition. The program can generate many types of potential exceptions, such as when a command is issued to read a file from a disk, but the file does not exist there or data items are to be written to a disk, but the disk is full or unformatted. Other sources of exception includes when a program attempts to divide a value by zero, access an array with an invalid subscript, or calculate a value that is too large for the answer's variable type.

These errors are called exceptions because, presumably, they are not usual occurrences; they are "exceptional". A reliable language should be able to handle different types of errors highlighted below.

HTTP errors: A common type of error in Web applications is the class of HTTP errors. According to Sun Educational services (2002: Module 9:3), an HTTP response sent from the Web server to browsers includes a status code which indicates whether the request was successful (status code 200) or not. When unsuccessful, the status codes in the 400-500 range indicate the errors types.

By default, the Web browser generates a message based on the status code, and displays it as an HTML message to the user. This is a generic HTTP Error page (Sun Educational services, 2002: Module 9:4). However, as noted by Sun Educational services (2002, Module 9:7), the generic HTTP Error pages generated by the Web browsers (for HTTP error codes) and the Web container (for the programming exceptions) are often ugly and not very informative to the user. Sometimes, they are even frightening. Students will therefore be taught:

- How to extract the status code from the HTTP response.
- How to create new error pages with informative messages and allow the Web container to handle the forwarding to these pages.

Other programming errors: In addition to HTTP errors, a Web application can generate exceptions to indicate a problem with the processing of the HTTP request. Students will be exposed to various programming techniques to handle such errors. These include:

- How to intercept and handle runtime errors.
- How to specify the errors that the program is allowed to leave unhandled.
- How to exit gracefully without stopping the program execution.

Logging exceptions: In addition to handling errors interactively, an exception might be written to a log file. This is necessary because, given that the standard screen is limited to about 25 lines per display, the errors sometimes scroll up, leaving the user with the last set of errors. Moreover, programmers are often very tense when debugging. Therefore, when the exception is written to a log file, the programmer can look at it later in a more relaxed mood. Students will be taught:

- How to create a file on the server from the executing Web application.
- How to log exceptions to the created file.

Criteria for the evaluation of suitability for teaching exception handlings

The previous section discussed the various exception handling techniques concepts to be taught to second year students. As a contribution to the body of knowledge, we develop the criteria below to ensure the suitability of the platforms for teaching exception handling techniques to undergraduate students:

- Ability to extract the status code from the HTTP response: this is necessary to extract the status of the error detected by the server.
- Ability to translate error codes and display pages that will present the errors in a meaningful language: this will assist the user in understanding what went wrong and to take corrective measures.
- Ability to specify the errors that the program is allowed to leave unhandled: this is necessary to allow the system to continue operations in situations of minor or expected errors.
- Ability to disable or suppress error messages for a single expression: this is also necessary to allow the system to continue operations in situations of minor or expected errors.
- Ability to turn off error reporting entirely, possibly temporarily, to enable initial concentration on the logic of the program: students need to get the logic right before handling syntax errors.
- Ability to intercept and handle runtime errors in the programs such that the program can recover from the errors: this illustrates how to handle runtime exceptions.
- Ability to exit gracefully without abruptly stopping the program; this illustrates how to perform necessary "housekeeping chores" to ensure consistency of program states.
- Ability to log error on files on disk: this demonstrates the techniques of keeping permanent records of errors encountered in order to gain more understanding of them and corrective actions later.

Interested users of these criteria can assign weights to the criteria. A weight of zero can be used to exclude some of the criteria. A weight that is greater than those for other criteria can be used to indicate the relative importance of such criterion over others. Moreover, users can add their own criteria.

5. SUMMARY OF THE RESULTS OF APPLYING THE CRITERIA

We applied the criteria to evaluate the suitability of four platforms namely Java Servlets, JavaServer Pages (JSP), Active Server Pages (ASP) and Personal Home Page (PHP) for teaching the exception handling concepts. Table 1 below reflects the scoring for the platforms. Formed from the review of various literatures and from programming experience, a summary of the results for the four platforms is also presented below.

An HTTP response sent from the Web server to browsers includes a status code which indicates whether the request was successful (the status code 200) or unsuccessful (Sun Educational services, 2002, Module 9:3). All the platforms have facilities to extract information from the HTTP request-headers, and to embed or set information into the HTTP response headers. The Java based platforms use a series of "set" methods for this purpose (Sebesta, 2003:429-455) while ASP uses the "AddHeader" method (Chapkanov, 1999, Chapter 12:11) and PHP uses the header () function (Lerdorf & Tatroe, 2002:175-176).

Table 1. Scoring for the platforms based on the criteria on exception handling capabilities.

	Criteria questions	Servlet	JSP	ASP	PHP
1	Is it possible to set and extract the status code representing error status stored by the server?	3	3	3	3
2	Does the platform translate error codes and display pages that will present the errors in a meaningful language?	2	2	3	3
3	Are there facilities for specifying the errors that the program is allowed to leave unhandled?	3	3	2	3
4	Are there facilities to disable or suppress the errors messages for a single expression?	2	2	2	3
5	Are there facilities to disable or turn off error reporting completely?	1	1	1	3
6	Are there facilities for intercepting and handling run-time errors in the programs such that the program can recover from these errors?	3	3	3	3
7	Are there facilities for exiting gracefully without abruptly stopping the program, thereby ensuring consistency of program states?	3	3	3	3
8	Are there facilities for logging or recording errors on files on disks?	2	2	2	3
	TOTAL	19	19	19	24

Scale: 3 = "Yes", 2 = "Not quite or with some workaround", and 1= "No".

The status code is sometimes meaningless or even confusing to the programmer (Deitel et al., 2001:646). Anyone who has used the Java-based platforms will confirm that they provide terrifying error diagnostics. We note that ASP and PHP translate error codes and display pages that will present the errors in a more meaningful message.

To speed up testing, it is sometimes necessary to specify the errors that the program is allowed to leave unhandled. In the Java-based platforms, one can declare the uncaught exceptions by listing them in the throws clause (Wigglesworth, 2000:248). ASP does not allow errors to be left unhandled (Deitel et al., 2001:646), except when the author specifically "comments out" these statements to avoid their execution. PHP lists the error conditions that are caught. It also has facilities to disable the error messages for a single expression (Lerdorf & Tatroe, 2002:304).

In other situations, it is better to disable or turn off error reporting completely. Only PHP allows one to turn off error reporting entirely (Lerdorf & Tatroe, 2002:305). This ensures that, regardless of the errors encountered while processing and executing script, no errors will be sent to the client except parse errors which cannot be suppressed (Lerdorf & Tatroe, 2002:305).

For better control of errors than simply hiding them, one can use the error handler. The error handler is called when a condition of any kind is encountered. It can do anything one requires, from logging to a file to printing error messages. Using the Java-based platforms, Wigglesworth (2000:253) lists some of these methods which include "trying" code and "catching" exceptions by encapsulating them in try blocks.

In order to handle errors more elegantly in ASP using VBScript, one can use the ONERROR event to launch error-handling codes when an ONERROR event is triggered. This can be used to write error messages to the status bar of the browser (Deitel et al., 2001:646). As explained by Lerdorf & Tatroe (2002:305), the basic process in PHP creates an error-handling function and registers it with the set_error_handler( ) method. This intercepts and handles run-time errors in the programs such that the program can recover from the errors.

Sometimes, it is necessary to exit gracefully without stopping the program abruptly, thereby ensuring consistency of program states. The Java based platforms use the System.exit() function to circumvent displaying the error message because the program ends before the error occurs, thus ending the application gracefully (Wigglesworth, 2000:675). In ASP, the exit function exits the program (Deitel et al., 2001:1169). In PHP, according to Lerdorf and Tatroe (2002:54), the exit () function which is also an alias for die () can be used to print out messages before ending the execution of a script when necessary.

For further reference and perusal, it is necessary to log or record errors on files on disks. This is typically done by the associating Web server, and most Web servers have such facility. However, in addition to this and unlike other platforms, PHP provides a built-in function, error_log(), to log errors to files and even to email addresses (Lerdorf & Tatroe, 2002:306).

6. CONCLUSIONS

It is important to ensure that a platform selected for teaching Web applications development is suitable for teaching the exception handling concepts. This is because exception handling serves as a solid foundation for robust Web applications development. These ideas form the basis of this study as part of a developed framework to evaluate the suitability of dynamic Web platforms for teaching Web application development in tertiary institutions.

The other components of the framework ensure the suitability for teaching other concepts such as structured programming, object-oriented programming, remote database management, file processing and XML support. In addition, the framework also ensures satisfaction of constraints such as ease of use, performance, affordability and portability.

This study and the use of the overall framework in general is expected to contribute to the body of knowledge with regard to the choice of suitable platform for teaching Web application development concepts (including exception handling) to undergraduate students in tertiary institutions.

9. REFERENCES

Apte, V., Hansen, T. and Reeser, P. (2003) "Performance comparison of dynamic Web platforms." Computer Communications. 26 (8) pp. 888 – 898.

Ashenfelter, J.P. (1999) Choosing a Database for Your Web Site. John Wiley & Sons, New York.

Bishop, J. and Hurter, R. (1999) "Competitors to Java: Scripting languages" (Paper read at the South African Computer Lecturers Association -SACLA- conference, June, Golden Gate, South Africa). Unpublished.

Bowling, A. (2002) Research methods in health. 2nd edition. Open University Press, Buckingham.

Cecchet, E., Chanda, A., Elnikety, S., Marguerite, J. and Zwaenepoel, W. (2003) "Performance Comparison of Middleware Architectures for Generating Dynamic Web Content." Lecture Notes in Computer Science: Middleware. 2672 (2003) pp. 242-261.

Chapkhanov, K. (1999) "Working with Active Server Pages: Chapter 12: Enhancing Interactivity with Cookies, Headers, and the Server Object" [Online]. Available: http://makecashonline.tripod.com/ch12.html [Accessed: 09/09/2004].

Coertze, D. and Heath, R. (1997). Research Methodology for Technikon Students: A practical Approach. Technikon Natal publishing, Durban.

Cooper, R. (2001) "Software for managing websites." Proceedings of the South African Institute of Computer Scientists and Information Technologists (SAICSIT) Annual conference. 5-8 September 2001. Pretoria, South Africa.

Dehinbo, J. (2005) "The performance of Web-based 2-tier middleware systems." Journal of Issues in Informing Science and Information Technology. 2 (2005) pp. 757-769 [Online]. Available from: http://2005papers.iisit.org/I59f23dehin.pdf [Accessed: 13/05/2005].

Deitel, H.M., Deitel, P.J. and Nieto, T.R. (2001) e-Business & e-Commerce: how to program. Prentice Hall, New Jersey.

Farrell, J. (2003) Java Programming. 2nd edition. Thomson Learning, Boston, Mass.

Garside, R. and Mariani, J. (2003) Java: first contact. 2nd edition. Thomson Learning, Toronto.

Hadjerrouit, S. (1998) "Java as first programming language: a critical evaluation." ACM SIGCSE Bulletin inroads. 30 (2) pp. 43-47.

Hartman, H. (2001) "Tools for dynamic websites: ASP vs PHP vs ASP.NET." Seybold Report Analysing Publishing Technologies, 15339211 (12) pp.1-12.

Holt, R., Wortman, D., Barnard, D. and Cordy, J.R. (1977) "SP/k: a system for teaching computer programming." Communications of the ACM. 20 (5) pp. 301-309.

Klopper, S. (2003) "Comparing the three scripting languages: PHP, ASP and JSP with each other, in order to use the best option for a specific application." Technologiae. 1 (2003) pp. 17-22.

Kolling, M. and Rosenberg, A. (1996) "Blue: A language for teaching object-oriented programming." Proceedings of the 27th SIGCSE Technical Symposium on Computer Science Education, Philadelphia. New York. pp.190-194.

Kruse, W. (2003) "A comparison of PHP and J2EE." Technologiae. 1 (2003) pp. 110-117.

Lerdorf, R. & Tatroe, K. (2002) "Programming PHP." O'Reilly & Associates Inc. Sebastopol, California.

Lim, B.L. (2002) "Teaching Web development technologies: Past, present, and (near) future." Journal of Information Systems Education. 13 (2). pp. 117-123.

Marshak, M. and Levy, H. (2003) "Evaluating Web user perceived latency using server side measurements." Computer Communications. 26 (8) pp. 872-887.

McMillan, J.H. and Schumacher, S. (2001) Research in education. 5th edition. Addison Wesley Longman, New York.

Prechelt, L. (2000) "An empirical comparison of seven programming languages." Computer. 33 (10) pp. 23-29.

Renaud, K., Lo, J., Bishop, J., Van Zyl, P and Worrall, B. (1999) "Algon: A framework for supporting comparison of distributed algorithm performance." Proceedings of PNDP conference, February 2003. Genoa. Rome.

Sebesta, R.W. (1996) "Concepts of Programming Languages, 3rd edition. Addison-Wesley, Reading, Mass.

Sun Educational Services. (2002) "Web Component Development With Java Technology." SL-314 Student Guide. Revision A.1, Sun Microsystems, New York.

Vinoski, S. (2003). "The performance presumption." IEEE Internet Computing. 7 (2) pp. 88-90.

Wiedenbeck, S., Ramalingam, V., Sarasamma, S. and Corritore, C.L. (1999) "A comparison of the comprehension of object-oriented and procedural programs by novice programmers." Interacting with Computers. 11 (3) pp. 252-282.

Wigglesworth, J. and Lumby, P. (2000) Java programming: advanced topics. Thomson Learning, Cambridge.