5th Slovakian-Hungarian Joint Symposium on Applied Machine Intelligence and Informatics
January 25-26, 2007 ░ Poprad, Slovakia

Testing Implementation of SeaFM Facility Management System

Márta Seebauer, Zoltán Balogh, Gréta Sifter, Andrásné Kovács

BMF SzGTI, seebauer.marta@szgti.bmf.hu; balogh.zoltan@szgti.bmf.hu
SeaCon Europe Ltd., sifter.greta@seacon.hu
BMF SzGTI, kovacs.andrasne@szgti.bmf.hu

Abstract: The purpose of the SeaFM Facility and Property Management System is combining the resources and activities of any organization or company dealing with facility and property management. At the corporate level, it contributes to the delivery of strategic and operational objectives. The system has a modular structure integrated in a complex of subsystems providing a global support for all of functionalities of the property management.
In the Client-Server architecture, it is more vital than ever to test throughout the life cycle of the development project. Testing can not be done toward the end or as an afterthought to development. Testing must be carefully planned, designed, prepared and executed. It requires quantifiable specifications, complex testing environments, architectures and tools, trained testers, and explicit exit criteria.
The V- Model of testing provides a structured testing framework throughout the Client-Server architectures development process. It is emphasizes quality from the initial requirements stage through the final testing stage. The V-model is a testing framework that has been successfully applied to various types of business applications, using many different development approaches. V-Model of verification, validation and testing is demonstrated through the Case study of the SeaFM Facility.

Keywords: facility and property management, software architecture, testing implementation, Test Stages, V-Model

1 Introduction

Software testing is any activity aimed at evaluating an attribute or capability of a program or system and determining that it meets its required results. [1] The software systems are becoming more complex according as an introduction of the several modelling applications is required in the course of designing and developing of effecient software systems. This statement applies to the testing functions, too. Well designed and implemented software products need an extensive testing process. The several test supporting tools are evolving parallel to the testing methods.

Software testing should be built-upon the correctness, completeness, security, and quality of developed computer software. Testing can never completely establish the correctness of arbitrary computer software, testing furnishes a criticism or comparison that compares the state and behaviour of the product against a specification. An important point of view is that the software testing should be distinguished from the separate discipline of software quality assurance, which encompasses all business process areas, not just testing.

The main software architecture types of computing systems will be reviewed at the first chapter. The testing methodology of the complex software architecture will be demonstrated at the second chapter, like the V-Model. The testing implementation of SeaFM Facility Management System will be described at the last chapter.

2 Architectures and Frameworks of Computing Systems

2.1 Enterprise Information System Architecture

The Enterprise Information System Architecture (EISA) framework provides a starting point for understanding what is meant by the various architecture types under consideration. [2] The EISA framework contains the next levels:

2.1.1 Business Solution

When it comes time to decide what technical architecture to use, many of answers are found by looking at the business solutions arhitecture. The decisions made for the application and data architectures drive the requirements of the technical architecture and platform.

2.1.2 Application Architecture

The applications architecture layer can be defined as services that perform business functions on the computer. It represents the components that provide the automation support for a business function or activity in the business process. This layer is best compontent of the EISA.

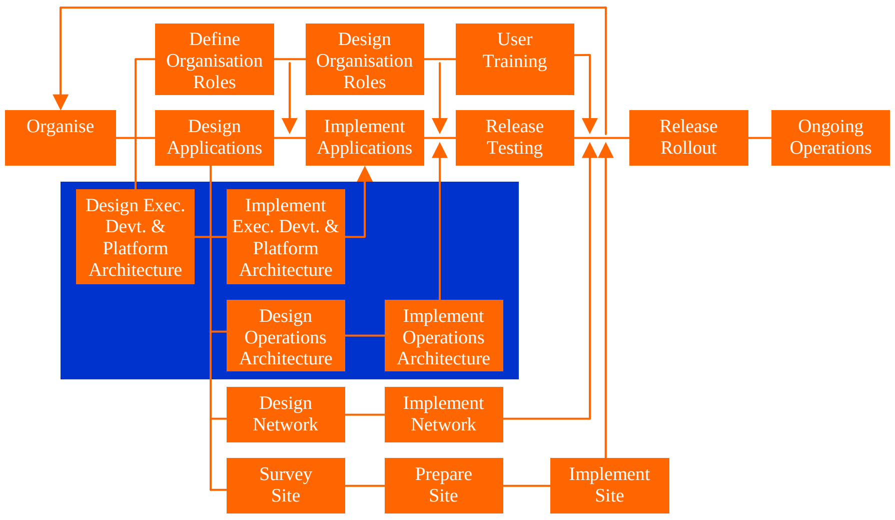
Define Organisation Roles
Design Organisation Roles
User Training
Organise
Design Applications
Implement Applications
Release Testing
Release Rollout
Ongoing Operations
Design Exec. Devt. & Platform Architecture
Implement Exec. Devt. & Platform Architecture
Design Operations Architecture
Implement Operations Architecture
Design Network
Implement Network
Survey Site
Prepare Site
Implement Site

Figure 1
Phases of development of Enterprise Information System

2.1.3 Technical Architecture

The technical architecture is where the system software are combined with the special infrastructure. It is comprised of the execution, development and operations architectures, wich are discussed subsequently. The technical architecture layer can be segmented so into three primary components:

- An execution architecture describes the components requiered when an application executes.
- A development architecture describes the components requiered to create the execution architecture.
- An operations architecture describes the components requiered to operate and manage the system.

2.1.4 Platform

The bottom layer is the platform architecture layer in the EISA model. It is often called: ‘The things you can see.’ The platform architecture provides a framework for the platform components requiered: the servers, workstations, operation systems and networks. This framework represents the all technology platform for the implementation and deployment of the execution architecture, development architecture, operations architecture and of course, the applications.

3 Testing Framework for Client-Server Architectures

3.1 The Test Strategy

The test strategy is developed to determine and communicate precisely how the end product, complete with new hardware, software, workflows, and procedures, will be tested. When developing the test strategy, developers must be sure to apply the V-Model concepts not only to the applications under development but also to the technical and application architectures, the training and documentation, and the organization and work processes.

Traditional test strategies focused only on testing the application code and basically ignored the architecture. But in today's world the architecture components are too complex and too integrated. So the modern test strategy must be focused on overall architecture layers and components.

3.2 Defining Test Cycles

Definition of test cycles requires addressing the issue of overall test design. Test design considerations to be applied when outlining cycles include the balance between sequential and parallel cycle definition and the inclusion of test cycles to cover ancillary tests.

3.2.1 Sequential Test Cycles

The sequential test cycles tend to imply single-thread executions that can lenghten the elapsed time needed to complete. Also, the built-in data dependencies leaves a team at risk that a defect in early cycles can halt execution of the entire test. Data dependencies also make the test model more likely to incur significant rework if changes to the data model occur in early test cycles. A positive consideration for sequential test is the ability to bundle all test conditions of a particular type into a single cycle, so reducing the number of cycles to be regression tested when a particular area changes.

3.2.2 Parallel Test Cycles

Parallel test cycles are designed by small amount of data, followed by multiple and concurent test cycles breaking off and testing groups of test conditions. Parallel test cycles reduce the risk that one error stops all test execution in its tracks. The parallel test cycles can be used independent data. Therefore, the inability to progress along one path will not necessarily halt others.

3.2.3 Mixed Test Cycles

The sequential and parallel test cycles are mixed cycles. It must be used when there are dependent data and independent data. So, the sequential test can be suitably used a flow of data and events through the application, because it supports the testing of true business processes. But the parallel test cycles can be applied when there are multiple executors active at a time, reducing risk and elapsed time for execution.

3.3 The V-Model for Testing

The V-Model of testing (Figure 2) provides a structured testing framework throughout the networking systems development process. It emphasises quality from the initial requirements stage through the final testing stage. All solution components, including application programs, conversion programs, and technical architecture programs, are required to proceed through the V-Model development process.

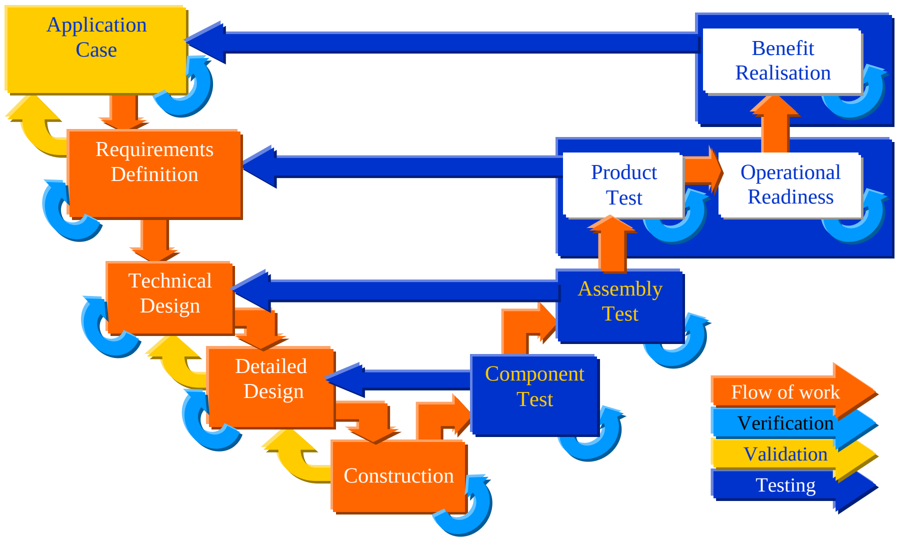

Figure 2
The V-Model for verification, validation, and testing

Verification is a process of checking that, as a deliverable is created at a stage of development, it is complete and correct and properly derived from the inputs of that stage. In addition, verification checks that the process to derive the deliverable was followed and that the output conforms to the standards set in the project's quality plan.

Validation checks that the deliverables satisfy requirements specified in the previous stage or in an earlier stage and that the application case continues to be met. In other words, the work product contributes to the indented benefits and does not have undesired side effects. Given the top-down nature of systems specification, validation is critical to ensuring that the decisions made at each successive level of specification continue on track to meet the initial application needs.

Testing is designed to ensure that the specification is properly implemented and integrated. Ideally, testing activities are not in place to ensure that the solution was properly specified, that activity is done via verification and validation. Rather testing activities associated with each level of the specification ensure that the specifications are properly translated into the final solution.

3.3.1 Component Test

The component test verifies that the component specification has been properly implemented. When testing components are in isolation, upfront planning is required along with the development of generic stub and driver logic that can be used throughout the component testing process.

3.3.2 Assembly test

The objective of assembly testing is to test the interaction of related components to ensure that the components function properly after integration. During assembly testing, application designers verify data is passed correctly between modules and as required, that messages are passed correctly between a client and a server. This is essentially a technical test, functional testing is limited to verifying that the flow through the assembled components supports the defined transanctions.

3.3.3 Product Test

The product testing focuses on the entire application to ensure that the system has met all functional and quality requirements. Product testing may occur at multiple levels. For example, the first level tests assemblies within an application. The next level tests applications within a system, and a final level tests systems with in a solution.

The product test focuses on the actual function of the solution as it supports the user requirements: the various cycles of transactions, the resolution of suspense items, and the work-flow within organizational units and across these units.

The product test determines whether the system is ready to be moved to operational readiness testing and subsequently into rollout. At this stage, the business worker becomes heavily involved in the testing process.

3.3.4 Operational Readiness Test

The objective of operational readiness testing is to ensure that the application can be correctly deployed, and those responsible for systems operations are prepared for rolling out and supporting the applications and associated data in both a normal mode and an emergency mode.

In addition, systems sometimes must be installed and operated by personnel at local sites who may have limited knowledge of complex application systems. In these cases, in particular, it is essential that the system be tested as an installable whole prior to being rolled out to the user community.

Operational readiness testing has four aspects:

1 Rollout test: Ensures that the rollout procedures and programs can install the application in the production enviroment.

2 Operations test: Ensures that all operational procedures and components are in place and acceptable, and that the personnel responsible for supporting production can operate the production system.

3 Service-level test: Ensures that when the application is rolled out, it provides the level of service to the users as specified in the service level agreement.

4 Rollout verification: Ensures that the application has been correctly rolled out at each site. This test, defined by the work cell or team performing operational readiness testing, should be executed during each site installation by the work cell or team in charge of actual rollout of the application.

3.3.5 Benefits Realization Test

The benefits realization test ensures that the business case for the system will be met. The emphasis here is on measuring the benefits of the new system, such as increased productivity, decreased service times, or lower error rates. If the business case is not testable, the benefits realization test becomes more of a bayer sign-off.

Ideally, benefits realization testing occurs prior to complete deployment of the system and utilizes the same environment that was used for the service-level testing component of the operational readiness test. Tools are put in place to collect data to verify the business case, such as ‘count costumer calls’. A team of people is still needed to monitor the reports from tools and to prove that the business case is going to be achieved. The size of the team depends on the number of users and the degree to wich tools can collect and report the data.

4 Testing Methods for SeaFM System

For three main modules of the SeaFM facility management system were tested, there are

- SYS module for the system administration,
- REG module for the real estates and equipments maintaining,
- REM modules for the management of these objects.

The V-Model was important aspect by the SeaFM testing. The testing of system had been dissolved by right of the V-Model, so efficiency of the testing methods could be succesfully increased. The testing may be disassembled the follow functions: the components test, the assembly test for the componenets collaboration, product and operational test for requirements satisfaction, and benefits realization test for the business logic satisfaction.

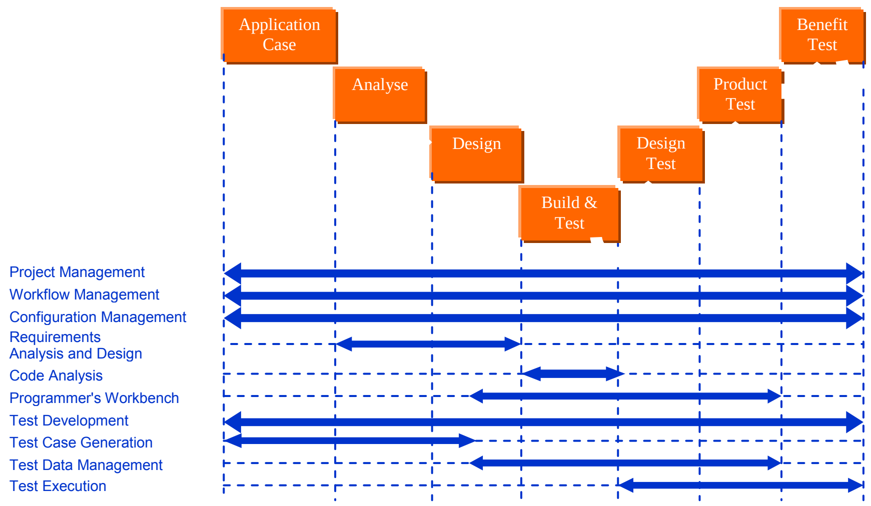

Figure 3
Implementation of the V-Model for SeaFM System

For the testing function a testing team was founded with the next participants: test manager, test designer, , test administrator, and testers. The effective testing began at the first phase of the system designing and was followed until the end of it. The test activities covered

- the test development,
- the test case generation,
- the test data management and
- the test execution.

Functionality (exterior quality)	Engineering (interior quality)	Adaptability (future quality)
Correctness	Efficiency	Flexibility
Reliability	Testability	Reusability
Usability	Documentation	Maintainability
Integrity	Structure	

Table 1
Typical software quality factors [1]

An outline of the main software quality factors investegeted in testing process are summerized in the Table 1. The testing phases were executed parallel on each of the modules. The testing results were recorded in testing protocols and regulary presented to software engineering and developing team.

In some opinion the testing could be never finished but each project is limited on the time and budget. The SeaFM system passed the operational readiness test and now is in benefits realization testing phase.

Conclusions

It can be concluded that the complex business systems need not only well thought-out, extensive designing and developing functions but even well considered testing methods. The testing methods may be decomposed by right of the system's layers.

The V-Model as a testing method uses this approach. The correctly operation of the business logic is standard expectation at the systems, but in addition the capacity of the platforms and runtime enviroments must be tested. If and only if it may be told with absolute security that the systems suits to the specification requirements, if the all testing methods has been succesfully executed.

The business systems in our days becomes more and more complex, so the testing methods is more important in the sotfware developing. The graver results can be only really obtained, if the testing influences on the all components of system.

References

[1] Hetzel, William: The Complete Guide to Software Testing, Second Edition, Wiley 1988, ISBN 0471565679

[2] Goodyear, Mark: Enterprise System Architecture, CRC Press LLC, 2000, ISBN 0849398363