A Quality Assurance Framework for Business Process Management

Kavya Mallur

Directed By:
Prof. Liam Peyton

Thesis

Submitted to the
Faculty of Graduate and Postdoctoral Studies
in partial fulfillment of the requirements for the degree of

Master of Computer Science

uOttawa

University of Ottawa

Ottawa, Ontario, Canada

March 30, 2015

Abstract

A business process is a defined collection of linked structured tasks, activities, and decisions performed together to produce a desired set of results in order to achieve business goals on behalf of the organization. Companies are increasingly moving their business processes online using Business Process Management (BPM) tools and technologies. With BPM, online business processes are defined by an explicit business process model that flexibly combines and orchestrates forms delivered through a web browser to integrate tasks performed by people, and web services accessible through Internet protocols to integrate tasks performed by software.

Often the approach to quality assurance for online business processes is similar to what would be done with any other web application. This is insufficient since it only provides rudimentary verification of single user behavior whereas the orchestration of tasks across many users and software systems can be quite complex. As well, a simple web application testing approach does not leverage the defined model for a business process to ensure consistency, completeness and enable automation. Nor will such an approach validate that a business process is contributing towards the achievement of business goals. A more systematic approach is required.

This thesis proposes a quality assurance framework to provide a repeatable, systematic, cost-efficient approach to quality assurance for BPM. A prototype framework was implemented and evaluated using two case studies, including one case study that was developed in collaboration with a local hospital to support a business process for cancer care assessment.

Acknowledgements

First and foremost, I wish to thank almighty for giving me strength and courage to complete this thesis.

I thank my beloved supervisor, Dr. Liam Peyton, for his continuous support and encouragement. I would like to express my immeasurable appreciation and deepest gratitude to his guidance throughout this research.

I like to express my sincere gratitude to Bernard Stepien for his tremendous support, and also for sharing his wide industry knowledge on *TTCN-3*.

I owe my sincere gratitude to my husband, Vikram Mallur for his countless love and support throughout my studies. A special thanks to my parents, who shaped me into what I am today. Also, I thank my twin sister and my friends for their encouraging words and prayers for my success.

Finally, I would like to thank IBM, Mitacs and NSERC for funding my research. This gave me a chance to work at one of the local hospitals with their CPM family and acquire the BPM knowledge needed for this research.

Table of Contents

Abstract ii
Acknowledgements iii
Table of Contents iv
List of Figures vii
List of Tables ix
List of Acronyms x
Chapter 1. Introduction 1
1.1. Problem Statement 1
1.2. Thesis Motivation 4
1.3. Thesis Contributions 5
1.4. Thesis Methodology 8
1.5. Thesis Organization 11
Chapter 2. Background 12
2.1. BPM Concepts 12
2.1.1 Web Applications and Web Services 12
2.1.2 Composite Applications and Service Oriented Architecture 13
2.1.3 Business Process 15
2.1.4 Business Process Management 15
2.1.5 Business Process Modelling Notation 18
2.1.6 Business Process Execution Language 21
2.1.7 Business Process Management Tools 21
2.1.8 Business Activity Monitoring 22
2.1.9 Performance Management 23
2.1.10 Metrics 23
2.1.11 Goal Model 24
2.1.12 Business Intelligence 25
2.2. Quality Assurance 27
2.2.1 Verification and Validation 27
2.2.2 Software Development Life Cycle 28
2.2.3 Software Testing Architecture 30

2.2.4 Web Application Testing 33
2.2.5 *TTCN-3* 33
2.2.6 Business Process Testing 37
2.3. Related Work 38
2.3.1 TASSA- framework for Business Process Quality Assurance 38
2.3.2 Business Process Verification with Reset Nets 39
2.3.3 Business Process Validation with SARI 41
2.4. Chapter Summary 44
Chapter 3. Model-Driven Quality Assurance Framework for BPM 45
3.1. Problem Description 45
3.2. Gap Analysis 48
3.3. Evaluation Criteria 50
3.3.1 Methodology 51
3.3.2 Business Process Features 53
3.3.3 Test Case Definition 54
3.3.4 Tool Support 55
3.4. Quality Assurance Framework 57
3.4.1 Framework Overview and Methodology 57
3.4.2 Business Process Model 59
3.4.3 Testing Environment 61
3.4.4 Test Orchestration Work bench 62
3.4.5 Quality Assurance Portal 64
3.5. Chapter Summary 67
Chapter 4. Case Studies 68
4.1.1 JUnit 69
4.1.2 Selenium 70
4.1.3 IBM BPM Testing Asset 71
4.1.4 TTCN-3 72
4.1.5 IBM Cognos BI Suite 74
4.1.6 Framework Prototype 75
4.2. Case Study 1: Library Borrow Book Process 77
4.2.1 Existing Quality Assurance Environment 77

4.2.2 Enterprise Architect 77
4.2.3 Care Process Expert 79
4.2.4 Development Team 80
4.2.5 Quality Assurance Team 81
4.3. Case Study 2: Cancer Care Assessment Process 86
4.3.1 Existing Quality Assurance Environment 86
4.3.2 Enterprise Architect 87
4.3.3 Care Process Expert 88
4.3.4 Development Team 90
4.3.5 Quality Assurance Team 92
4.4. Summary of Experience and Results 96
4.5. Chapter Summary 97
Chapter 5. Evaluation 98
5.1. Framework Evaluation 98
5.1.1 Methodology 98
5.1.2 Business Process Features 99
5.1.3 Test Case Definitions 100
5.1.4 Tool Support 101
5.2. Related Works Evaluation 103
5.2.1 Methodology 103
5.2.2 Business Process Features 104
5.2.3 Test Case Definitions 105
5.2.4 Tool Support 106
5.3. TTCN-3 Evaluation 107
Chapter 6. Conclusions and Future Work 113
6.1. Conclusions 113
6.2. Future Work 114
References 116
Appendix TTCN-3 Library Borrow Book Example 123

List of Figures

Figure 1-1: Design Science Research Methodology (Hevner, March, Park, & Ram, 2004) 8
Figure 2-1: Service Oriented Architecture 14
Figure 2-2: BPM Life Cycle 18
Figure 2-3 : BPMN Language Elements (Wohed, van der Aalst, Dumas, ter Hofstede, & Russell, 2005) 19
Figure 2-4: Sample goal model for *Library Borrow Book* Process 25
Figure 2-6: Black Box Testing 31
Figure 2-7: White Box Testing 32
Figure 2-8: Grey Box Testing (Khan & Khan, 2012) 32
Figure 2-9: XSS attack sequence of events (Stepien, Xiong, & Peyton, 2011). 34
Figure 2-10: *TTCN-3* Separation of Concern (Stepien & Peyton, 2013) 35
Figure 2-12: TASSA tool used for Business Process Testing (Ilieva, Manova, & Petrova-Antonova, 2012) 39
Figure 2-13: Testing Business Processes with a Process Test Specification (Schiefer, Saurer, & Schatten, 2006) 43
Figure 3-1: Current Business Process Testing Approach 46
Figure 3-2: Proposed Business Process Quality Assurance Framework 58
Figure 3-3: Sample Business Process Model 60
Figure 3-4: Proposed Test Environment 61
Figure 3-5: Test Orchestration Workbench 63
Figure 3-6: Quality Assurance Portal displaying Test Report 65
Figure 4-1: Prototype of our Framework 75
Figure 4-2: Service Specification for Library Business Process 78
Figure 4-3: Service Operation Description for Library Business Process 78
Figure 4-4: User Scenarios provided for Library Management 79
Figure 4-5: Goal Model for Library Management Business Process 80
Figure 4-6: Library Borrow Book Business Process Model developed using User Scenarios and SOA Model 80
Figure 4-7: Model-Driven Test Case Generation using IBM's BPM Testing Asset tool 82
Figure 4-8: TTCN-3 test graphical log 84
Figure 4-13: Report displaying number of books borrowed per year 85
Figure 4-14: Report displaying the percentage of books lost 85
Figure 4-15: Report displaying the total amount of fine collected per year 86
Figure 4-16: Service specification for Cancer Care Assessment project 87
Figure 4-17: Service operation description for getPatientDemographics used in Cancer Care Assessment 88
Figure 4-18: User Scenarios for Cancer Care Assessment 89
Figure 4-19: Sample goal model for Cancer Care Assessment project 90

Figure 4-20: Cancer Care Assessment Business Process Model 91
Figure 4-21: Cancer Care Assessment Tests generated using IBM BPM Testing Asset 92
Figure 4-22: TTCN-3 code snippet for the test involving 2 physicians and 2 nurses 93
Figure 4-23: Behaviour of the physician 94
Figure 4-24: TTCN-3 test execution log for Cancer Care Assessment project 94
Figure 4-25: Report showing the average wait time between the Referral and RN Review 96
Figure 5- 1 Reusability of Tri-Send in TTCN-3 code for BPM 111
Figure A-1: Collaboration of multiple user roles in orchestration of multiple services using TTCN-3 123
Figure A-2: TTCN-3 test displaying task allotment for the two student users 124
Figure A-3: Code Snippet Describing Test case written in TTCN-3 125
Figure A-4: TTCN-3 Behavior function of the student 125

List of Tables

Table 2-1: Comparison of Verification and Validation 28
Table 5-1: Framework Evaluation- Methodology 99
Table 5-3: FW Evaluation – Test Case Definitions 101
Table 5-4: FW Evaluation – Tools support 102
Table 5-5: Comparison with Related works- Methodology 103
Table 5-6: Comparison with Related works- Business Process Features 105
Table 5-7: Comparison with Related works- Test Case Definition 106
Table 5-8: Comparison with Related works- Tools support 107
Table 5-9: Comparison of TTCN-3 108

List of Acronyms

Acronym	Definition
BAM	*Business Activity Monitoring*
BI	*Business Intelligence*
BPEL	*Business Process Execution Language*
BPM	*Business Process Management*
BPMI	*Business Process Management initiative*
BPMN	*Business Process Modelling Notation*
DW	*Data Warehouse*
ETL	*Extract Transform Load*
OASIS	*Organization for the Advancement of Structured Information Standards*
OLAP	*Online Analytical Processing*
PTC	*Parallel Test Component*
QA	*Quality Assurance*
REST	*Representational State Transfer*
SOA	*Service Oriented Architecture*
SOAP	*Simple Object Access Protocol*
SQL	*Structured Query Language*
TTCN	*Testing and Test Control Notation*
UDDI	*Universal Data Discovery Interface*
WS	*Web Service*
WSDL	*Web Service Description Language*

Chapter 1. Introduction

1.1. Problem Statement

In any organization, business processes are pervasive. A business process is a defined collection of linked structured tasks, activities, and decisions performed together to produce a desired set of results in order to contribute towards the achievement of business goals on behalf of the organization. Companies are increasingly moving their business processes online using Business Process Management (BPM) tools and technologies and (Olga Levina, 2010). With BPM, online business processes are defined by an explicit business process model that flexibly combines and orchestrates forms delivered through a web browser to integrate tasks performed by people, and web services accessible through Internet protocols to integrate tasks performed by software within a Service-Oriented Architecture (SOA).

Often the approach to quality assurance for online business processes is similar to what would be done with any other web application. This is insufficient since it only provides rudimentary verification of single user behavior whereas the orchestration of tasks across many users and software systems can be quite complex for an online business process. As well, a simple web application testing approach does not leverage the defined model for a business process to ensure consistency, completeness and enable automation. Nor will such an approach validate that an online business process is adequately contributing towards the achievement of business goals. A more systematic approach to verification and validation of online business processes is required.

Business processes are different from typical web applications in that they often involve the collaboration of multiple user roles in parallel activity (Ertugrul & Demirors, 2015). For example, a hospital business process may involve many participating actors: nurses, doctors, patients. There might exist a dependency between these actors within the parts of the process for a particular instance of the process, for a particular patient (e.g. a doctor will re-assess a patient only after all tests have been completed by different nurses and technicians); or there might be a parallel execution of a task by multiple actors in different parts of the process across many instances of the process for different patients (many doctors may be assessing and re-assessing different patients at the same time).

As well BPM development and testing happen in a SOA. BPM naturally integrates web services implemented with different languages and technologies and executed in heterogeneous environment (Ilieva, Manova, & Petrova-Antonova, 2012). The BPM system places calls to these web services for parallel execution of multiple user tasks. For instance, a user task may invoke a service to update information in the patient's electronic health record. There might be invocation of an email or notification service, when another user needs to be alerted of a patient's condition. Another example would be the invocation of web services to interface to the difference software systems throughout the hospital that provide services (e.g. pharmacy, x-ray, blood testing, etc.). Collaborative testing of user interactions in parallel, involving multiple service orchestrations for multiple user tasks, is critical to business process verification.

In addition, one cannot truly address quality assurance for BPM, without it validating on a continuous basis that the online business process is contributing to the

achievement of business goals as intended. Business Activity Monitoring (BAM) integrated with BPM can provide information about business processes (Kang, 2008). BAM promises to ensure round-the-clock monitoring of online business processes. Thus, the quality assurance of business process involves both verification of business process features and validation of online business process against business goals.

In summary, BPM is more complex than simple web applications, and traditional approaches to web application testing are not sufficient to address quality assurance for BPM. Unfortunately, in many organizations today, this is the approach that is taken. In this research, we will address this problem, through the development of a systematic framework for quality assurance of BPM that addresses the complex requirements of BPM, and integrates emerging test tools and technology that are relevant.

1.2. Thesis Motivation

The overall aim of the research is to provide a systematic approach to verify as well as validate online business processes. This work was motivated by the drawbacks in the current BPM testing practices we have observed in practice, and the lack of appropriated guidelines in the marketplace to guide quality assurance for BPM. In particular, traditional web application testing does not evaluate compliance of online business processes with the organization's business goals. The ultimate motivation for adoption of BPM is to improve business performance as a whole (Sánchez, Ruiz, García, & Piattini, 2013). This can only be achieved by validating business processes in addition to verifying them. Performance validation of online business process should result in better quality business process.

Another source of motivation was to address the complexity of multi-user, multi-service orchestration in BPM using *TTCN-3*, international standard for Telecommunications testing. *TTCN-3*, with its Parallel Test Components (PTC) feature, has proven its usefulness in achieving multi-user multi-service collaboration testing in telecommunications but more recently has also been applied to aeronautics and complex Internet Applications. We hypothesized that it could be applied in a similar way to BPM, in order to improve tool support for testing of complex orchestrations.

The third motivation for this research was a desire to leverage model-driven testing, wherein the business process model used in BPM could be used to ensure consistency and completeness of testing, and enable automation. In particular, we sought

to define an integrated Test Orchestration Workbench that could combine tools and approaches to offer an enhanced platform for quality assurance of BPM.

The current approach to quality assurance of online business processes is insufficient as it only provides rudimentary verification of single user-role behavior ignoring the orchestration of tasks across many users and system services. Also, the approach does not leverage the business process model and does not validate that the business process is contributing towards the achievement of goals. So, the overall motivation of this research is to address these gaps (which are discussed in more detail in Chapter 3) through the development of a model-based quality assurance framework for BPM.

1.3. Thesis Contributions

The main contributions of this thesis are as follows:

1. ***A Quality Assurance framework for Business Process Management*** that includes:
    a. *A Model-based Methodology* that defines a systematic approach to quality assurance for business process management that leverages not just the business process model itself, but also well-structured definitions of requirements in terms of Service Oriented Architecture (SOA), User Scenarios, and a Business Goal Model.
    b. *A Test Architecture* which identifies the types of testing for business processes and establishes the relationship between three main

components: business process engine, test workbench, and quality assurance portal.

c. *A Test Orchestration Workbench* which integrates the complete range of testing tools needed to address the expanded quality assurance requirements for online business processes. This includes integration of the *TTCN-3* standard for Telecommunications testing to address complex multi-role, multi-service orchestration testing.

d. *A Quality Assurance Portal* which provides a systematic view of a business process test campaign and validates the business process against business goals in both Test and Production environments.

2. ***An analysis of the applicability of the TTCN3 standard for BPM testing including:***

   a. A review of the typical *TTCN-3* implementation architecture and a proposal for different implementation architecture to better address BPM testing requirements.

   b. A characterization of the advantages of the *TTCN-3* language and abstract layer for addressing complex multi-role, multi-service orchestration testing compared to other tools.

   c. A catalog of issues and challenges that are faced in building a *TTCN-3* "codec" adaptor for BPM to enable testing at the concrete layer.

The following works related to this research have been published

1. **Mallur K.**, Alhaj M., Peyton L. and Bernard S. "*A Systematic Approach to Quality Assurance of a Health Care Business Process*", e-HealthForHumanity workshop at MCETech 2015, Montreal, May 2015
2. Stepien B., **Mallur K.,** Alhaj M. and Peyton L., "*Verification and Validation of Online Business Processes*", MCETech Conference on E-Technologies, Montreal, May, 2015 (tutorial)
3. **Mallur K.,** Valiyapalathingal, R., Alhaj M. and Peyton L. "*A Framework for Dynamic Quality Assurance of Business Process Management*", CASCON 2014, Toronto, November 2014 (poster)
4. **Mallur K.,** Alhaj, M, and Peyton L., Business Process Management (EBC 6230), self-directed laboratory course, University of Ottawa, March 31, 2015 (based on the *library borrow book* process case study from this thesis). Available at https://code.google.com/p/ebc6230-bpm/wiki/TableOfContents?tm=6

1.4. Thesis Methodology

We have followed *Design Science Research (DSR)* methodology in this thesis, which involves construction and evaluation of artifact that is targeted to solve an identified organisational problem (Hevner, March, Park, & Ram, 2004). Figure 1-1 illustrates the framework proposed by Henver et,al for performing DSR.

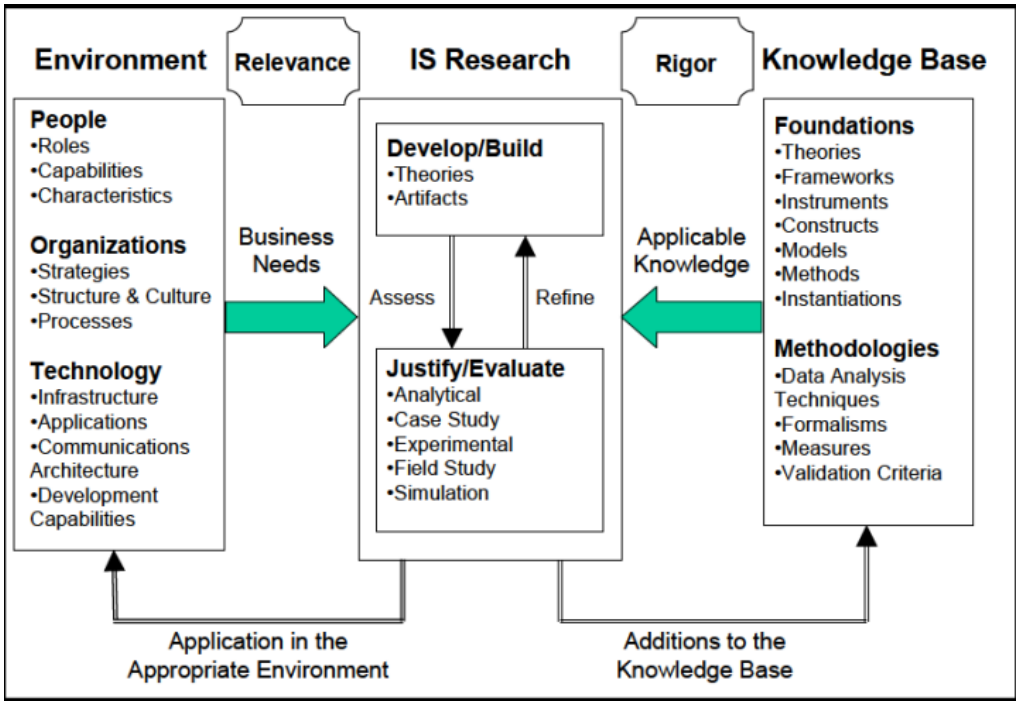


Figure 1-1: Design Science Research Methodology (Hevner, March, Park, & Ram, 2004)

DSR is regarded as a sequence of expert activities that ultimately produces an innovative end product. It combines two research paradigms mainly- Behavioural Science which tries to develop and verify a theory and the Design Science which produces useful solution that solves the identified problem.

Following are the steps followed in our thesis methodology:

- Problem identification based on existing literature and interaction with business process experts:
    - Studying existing business process testing methodologies (literature survey).
    - Interaction with existing business process testing teams on current business process testing methodologies (including IBM, local hospital, and uOttawa research group)
    - Perform gap analyses between existing and proposed solutions to address the problem and develop evaluation criteria.
- Catalog and combine existing testing tools and techniques to design a prototype quality assurance framework for BPM.
    - Identify key features for business process quality assurance.
    - Survey and comparison of available tools for such features.
    - Develop a test workbench for tool integration that would incorporate the complete set of tools needed to address key features for business process quality assurance.
    - Develop a model-based methodology for defining BPM requirements for an online business process, and for structuring the quality assurance activities that would verify that requirements are met.
    - Develop a Quality Assurance Portal for both test campaign management and performance reporting to validated business goals.

- Application of this developed testing framework prototype to two different case studies. Each time results were evaluated using our evaluation criteria, and our quality assurance framework for BPM was refined and updated.
    - *Case Study 1:* A '*Library Borrow Book*' business process implemented for borrowing a book from the rare books collection at the uOttawa library. This case study has now been incorporated into a new graduate course: *EBC6230 Business Process Management and Related Technologies*
    - *Case Study 2:* A '*Cancer Care Assessment*' business process that was developed and deployed at a local hospital.
- Conclusions, future work and publications.

1.5. Thesis Organization

This thesis is organized as follows:

- **Chapter 1** presents an introduction, together with the context and the scope of the research.
- **Chapter 2** provides a background to our research. The chapter provides context for the research problem that we are solving in this work by giving a detailed background on business processes, software testing and business process testing. The chapter also provides literature review from related works. The technologies and tools used in these related works are analysed and compared.
- **Chapter 3** elaborates the research problem, our research model and evaluation criteria. The chapter also proposes the quality assurance framework for business processes.
- **Chapter 4** presents our case studies and provides details of our experiments. The proposed framework is prototypes and used to test two different business processes.
- **Chapter 5** revolves around the evaluation of our research. The framework evaluation against the identified criteria and comparison with current business process test approaches is explained in brief.
- **Chapter 6** briefs the conclusions drawn from our work. And the chapter also speaks about the future work.

Chapter 2. Background

We begin this chapter by explaining the key concepts related to Business Process Management. We then summarize the relevant background related to testing for this thesis. Finally, we survey related works that have attempted to address the type of problem we are concerned with in this thesis. .

2.1. BPM Concepts

2.1.1 Web Applications and Web Services

Gellersen et.al, define 'Web Application' as "Any software application that depends on the Web for its correct execution" (Gellersen & Gaedke, 1999). Web application is a browser-based three-tier client-server application composed of browser-based front end, a server side application model and a database. Traditionally, a web application can be broken down into three components namely: Web server, Web application, running on the web server and the data base.

Web Services is regarded a technology which is used to change application development from being information-oriented to service-oriented (Ziqiang & Hong, 2010). According to Booth, web service is a software system that supports machine-to-machine, interoperable interactions over a network (Booth D. , et al., 2004). Web service technologies (such as SOAP or REST) are platform independent, which allow applications to exchange information regardless of the programming language being used by the applications (Curbera, et al., 2002).

2.1.2 Composite Applications and Service Oriented Architecture

Service Oriented Architecture (SOA) is a business-centric IT architectural approach that provides linked, repeatable services. (Nickul, 2007). SOA is a set of components that can be invoked, and its interface description can be published and discovered (Booth, Haas, & Brown, 2004). It is a way of publishing interfaces for accessing business application services in a systematic manner.

When a SOA is present, web applications can be thought of as composite applications in which the application logic reuses the available services in an integrated fashion (Peyton, Stepien, & Seguin, 2008).So, a composite application can be considered as a piece of software that consumes functionality offered by several services within SOA, to perform defined set of tasks.

In SOA, all the components of the system are treated as services while the users of those services are treated as clients. The system services can be used for all aspects of software including data analysis, information processing and storage (Owen & Raj, 2003). Services are discoverable, and loosely coupled forming the basic constitutional units of SOA.

According to Huhns and Singh, SOA consists of three major components: (i) a service provider, (ii) a consumer that requests for services and (iii) a service registry (Huhns & Singh, 2005). This is illustrated in figure 2-1. The service provider publishes services on the service registry and the consumers browse the registries to identify and invoke the needed services. Emergence of SOA has offered IT departments a new

innovative way to develop business services by reusing the components of existing programs rather than rewriting redundant code from scratch and developing new infrastructures to support them (Mulesoft, 2015).

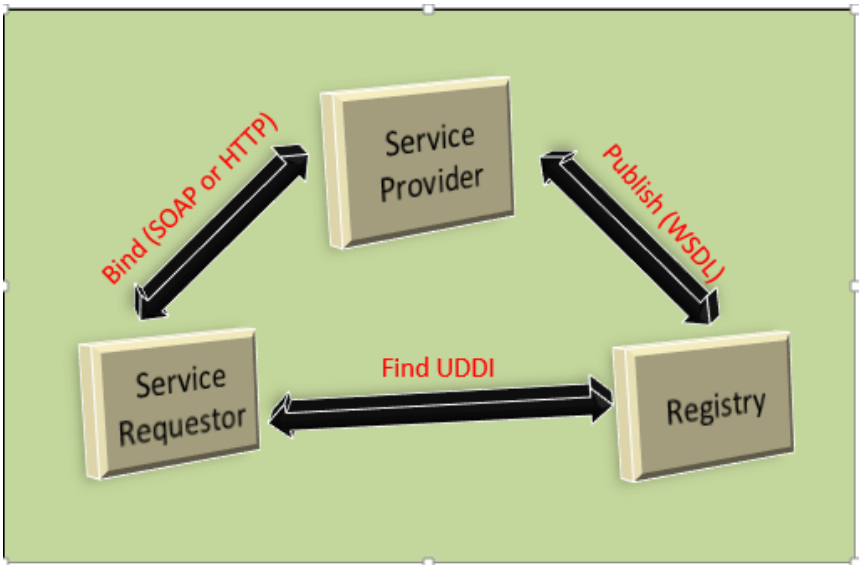


Figure 2-1: Service Oriented Architecture

SOA targets delivery of functionalities through loosely coupled services which can be reused to fulfill business processes. Service Orchestration is the coordination of multiple services, which are exposed as a single aggregate service unit. In orchestration of services, the involved services are controlled by a single end point called the central point, which is regarded as the orchestrator and who is responsible for the coordination of execution of various services (Kacem, Sellami, & Kacem, 2012). Web Service Orchestration controls and directs the action of one or more individual web services so that they work together to perform some useful function (Morris, et al., 2010). SOA provides agility, flexibility and reusability to respond to changing business requirements (Lee, 2009).

2.1.3 Business Process

Business Processes are built in SOA and can leverage the capabilities offered by services to achieve their objectives. A Business Process is a "coordinated chain of activities intended to produce a business result" (Bruce Silver Associates, 2006). It is also defined as "the combination of set of activities within an enterprise with a structure describing their logical ordering and dependence, whose objective is to produce desired result" (Aguilar-Saven, 2003). It is a representation of a sequence of structured activities that can be visualized as a flowchart, and used to deliver a service or product to customers.

Business Processes contain related tasks that aim to achieve a set of business goals. The scope of business process can be limited to a particular department or may span several divisions within an enterprise, or it may require enterprise collaborations (Leymann, Roller, & Schmidt, 2002). The design of business process involves instrumentation of flow model parts which include activities, and transition conditions to measure relevant metrics (such as, time taken to carry out an activity).

2.1.4 Business Process Management

BPM is defined as "a generic software system that is driven by explicit process designs to enact and manage operational business processes" (van der Aalst, 2013). It is the discipline that combines knowledge from information technology and knowledge from management sciences, and applies this to operational business processes (van der Aalst, 2004). BPM is composed of techniques, methods and tools that support the design, development, management and analysis of online business processes

that involve humans, organizations, applications, documents and other information sources. SOA provides an ideal infrastructure for BPM, where in the applications are invoked using interfaces and protocols (Steen, 2007).

Online business processes are implemented in SOA. It is an innovative architectural approach where software applications can be developed and deployed as a set of reusable services (Alhaj, 2011). It provides a service ecosystem in which composite business centric services can be assembled on the fly as requirements change (Li, Tan, Liu, Zhu, & Mitsumori, 2008). Web service forms a basis for SOA and makes business processes accessible within an enterprise or across different enterprises operating in a heterogeneous environment (Leymann, Roller, & Schmidt, 2002). Business processes integrate the web services that execute in heterogeneous environments and are built in different languages and technologies (Ilieva, Manova, & Petrova-Antonova, 2012)

BPM enables organizations to make faster and better decisions and improve the process to contribute better business results for them (Radovan, 2012). BPM is used in many organizations to move existing paper-based business processes online in order to improve the quality of provided services and reduce costs. BPM activities are iterative and continuous for improving the business process to continuously meet business goals. The main objective of BPM is to provide integrated solutions to people, processes, systems and customers which are identified as four pillars that are crucial for the success of an organization (Jufuru, 2007). Automatic services can be integrated with human tasks to form complex business processes that span across departments.

According to Wetzstein et.al, BPM is a top-down approach, designed to organize, manage, analyze and reengineer the processes running in an organization (Wetzstein, et al., 2007). Figure 2-2 depicts the iterative life cycle of BPM. In the **'Design & Model'** phase, business process requirements are captured analysed and a business model is designed. A good design should reduce process complexity and enhance efficiency. The designed business process is then modeled. There are several modelling languages and notations that help in modelling a business process. The modelling tools support graph-based representation and the process can be modelled by specifying the order of tasks. The process models generated at this stage are high-level models with no technical details associated with them. The next phase is the **'Deploy & Execute'** phase wherein the process models generated in Design phase are transformed into an executable model. In this phase, the business process will be assembled, deployed and managed by integrated applications. Following this phase is the **'Monitor'** phase wherein the business activity information generated on execution of business process is fed into monitoring applications for analysis. This stage helps in the identification of any errors or loopholes in the design. The main goal of the monitoring phase is to provide information necessary for optimization of the process model. The final stage is the **'Analyze and Optimize'** phase where issues or potential improvements in business processes are identified as input to the '**Design & Model"** phase of the next iteration of the lifecycle.

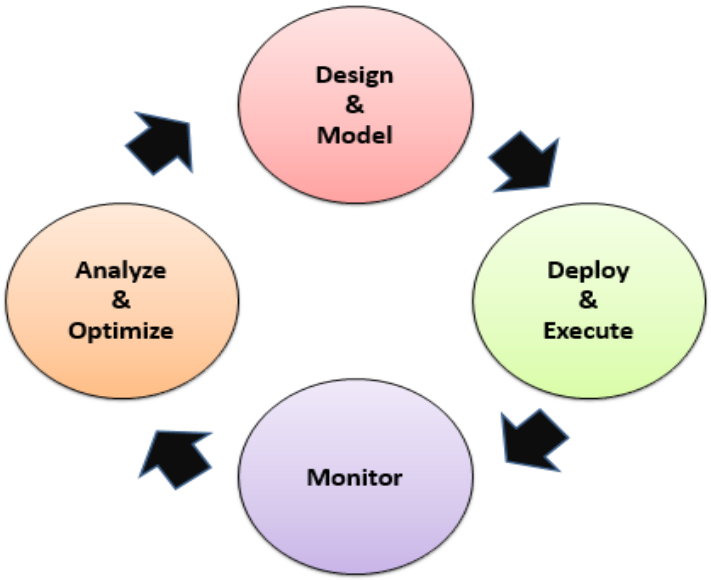


Figure 2-2: BPM Life Cycle

2.1.5 Business Process Modelling Notation

Business Process Modelling Notation (BPMN) provides graphical representation of business procedures that help organizations to understand and communicate business procedures effectively. BPMN is a standard developed by Business Process Management Initiative (BPMI) with a primary goal to provide easy notation that is readily understandable by all the business users ranging from Business Analysts to Business users (White & Miers, 2008).

BPMN is a core enabler for BPM (Owen & Raj, 2003). BPMN provides a systematic approach to communicate visually a wide breadth of information to different audiences and helps in reducing the fragmentation that exists with the myriad of process modelling tools and notations. Figure 2-3 describes the BPMN language elements (Wohed, van der Aalst, Dumas, ter Hofstede, & Russell, 2005)-

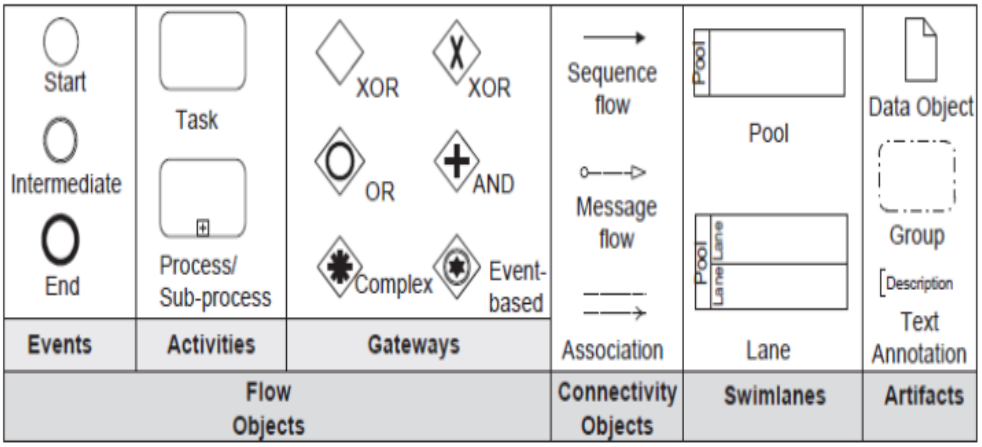


Figure 2-3 : BPMN Language Elements (Wohed, van der Aalst, Dumas, ter Hofstede, & Russell, 2005)

The Flow objects include-

***Events**: Events are used to indicate that something outside the process has occurred, and this event occurrence affects the process behavior. The three commonly used events are- *Start event* indicates the beginning of the process, *End event* indicates the end of the process and the *Intermediate event* occurs in the middle of the process flow.

***Activities**: Represents a unit of work to be performed by an actor as a part of the process. An activity can be a simple task or a sub-process, which is a logical sequence of activities making up small process. With Sub-Processes, logically related steps can be encapsulated within the parent process.

***Gateways**: They help in organizing the process control flows. There are different kinds of gateways (AND, OR, XOR, etc.)

The Connectivity Objects include-

***Sequence flows**: This indicates the ordering of activities. The arrow headed flows connect the activities together in a process.

***Message flows**: This indicates the flow of message between the entities in the business process. The entity at the tail of the flow is regarded as the sender, while the one at the arrow head is the receiver of the messages.

***Associations**: They are used to indicate links between the objects and the data information.

The Swim lanes include the following-

***Pool**: This is regarded as the participant in the business process.

***Lane**: This is a subset of pool and is generally an actor/user role.

The Artifacts include-

***Data Object**: Used to specify the nature of data inputs and outputs for an activity involved in a given process. It is basically used in the representation of data involved in the process.

***Text Annotations**: This serves like a comment box, allowing the attachment of textual note to the components of the process.

***Groups**: These are used to group related elements in the process and are not constrained to swim lanes.

2.1.6 Business Process Execution Language

According to Wieland et al, Business Process Execution Language (BPEL), the shorter version of Web services Business Process Execution Language (WS-BPEL), is the most broadly used OASIS standard for specifying actions within business processes (Wieland, Kopp, Nicklas, & Leymann, 2007). Once the business process is modeled, the BPM engine should be able to execute the model. BPEL is an xml-based language, whose kernel is composed of a number of communication primitives combined with control flow constructs representing sequences, branching, parallelism, etc. (Ouyang, et al., 2007).

2.1.7 Business Process Management Tools

According to Strobl et.al, "BPM tools are software products supporting the acquisition, modeling, analysis, simulation, evaluation, and deployment of business processes" (Strobl, Mullner, & Rausch, 2009). Examples of BPM suites available on the market include IBM Websphere BPM, Bizagi BPM suite, HP Process Automation, Tibco ActiveMatrix, and Oracle Business Process Management Suite.

These suits provide tools to model and draw business processes, and implement them as an orchestration of user interface forms for human interaction and web service invocations for leveraging software services provided through a SOA. They also provide a business process execution engine that can execute the modeled business process assigning, sequencing and coordinating tasks, and invoking user interface forms and web services to execute tasks (Reynolds, et al., April 2014).

In our case studies, we used IBM Websphere Business Process Manager 8.5.5 to model, analyse and evaluate business processes. It supports BPMN 2.0. The history of BPM support at IBM started in 2005 when IBM released WebSphere Process Server (WPS), which supported core SOA. In 2010, IBM acquired Lombardi, a 3rd party company that marketed TeamWorks and IBM renamed TeamWorks to WebSphere Lombardi Edition. IBM released the first version in June 2010. Teamworks provided enhanced support for human interaction through user interface forms (called coaches) as well as better IDE support for building, deploying and monitoring business processes. However, it was not until the release of IBM Websphere Business Process Manager 8.5.5 that IBM fully integrated the two product offerings in a streamlined fashion. (Kolban, 2014).

2.1.8 Business Activity Monitoring

The term 'Business Activity Monitoring' (BAM) was coined by Gartner Group. And it is defined as- "BAM is the real-time reporting, analysis and alerting of significant business events, accomplished by gathering data, key performance indicators and business events from multiple applications" (Gartner, 2013). BAM allows organizations to observe what is happening in the business in real time. BAM can be used to improve speed and effectiveness of business operations by tracking the current happenings and thus enabling analysis of critical business performance indicators based on real-time data. BAM is gaining more popularity these days and is regarded as one of the major components of a complete Business Process Management System.

Process monitoring is used for monitoring events from processes and other applications using real-time dashboards. Process workspace and monitoring dashboards help in managing and monitoring processes in real-time.

Online business processes need to be monitored continuously to ensure that the business process is meeting its goals. The key outcome of continuous monitoring, controlling and analysis of business processes is process optimization and improved decision making (Kronz, 2006).

2.1.9 Performance Management

According to Olsen Erica, Performance Management is regarded as a process by which organisations align their resources, systems and employees to strategic objectives and priorities (Olsen, 2014). Performance management is used to tie evaluation of the success and compliance of business processes directly to the data collected about them, in terms of metrics (Middleton, Peyton, Kuziemsky, & Eze, 2009). The user interfaces for reporting, monitoring, and functionality analysis of the performance management system is typically provided by a portal (Peyton, Zhan, & Stepien , 2008).

2.1.10 Metrics

A metric is used to measure how well a process is working and what a process is doing. Metrics can also help in identification of problems or progress towards goals. Goals are the targets set in terms of metrics (David Walden, 1994). A metric can be defined as "A quantitative measure of the degree to which a system, component, or process possesses a given attribute" (Villar, 2010). The overall performance of a business process can be measured using the most critical high level metrics. For example, *average*

wait time for borrowing book in library is one example of a critical metric, used to assess the quality of Library business process.

2.1.11 Goal Model

Business Processes are tightly coupled with business goals. Goals are objectives that a business or organization is expected to achieve through integration of people, technology and operational processes (Liu & Yu, 2005). Business processes are designed to help the organization achieve its goals.

A goal model structures the goals and captures dependencies and interactions between the goals and identifies the key stakeholders and actors responsible for goals (Kuziemsky, Liu, & Peyton, 2010). There should exist at least one metric critical for each goal and it should measure to what degree the organization is achieving the goal. Goal model indicates the relationship between a system and its environment, connecting requirements to design. Goals help in clarification of requirements and play a major role in resolving any requirement conflicts (Yu & Mylopoulos, 1998). Goal models capture the objectives of stakeholder and businesses, means of meeting these objectives, and their impact on quality of businesses (Amyot Daniel, et al., 2012) and Goal modelling is an important part of requirements engineering activities. Business Performance Management helps organisations to understand the status of the business processes in terms of goals and business trends and help improve business operations.

Figure 2-4 depicts a sample goal model for a university library business process (see section 4, for a full discussion of our *Library Borrow Book* case study). As seen here, there are three defined goals for a Library business process namely: G1: To encourage

borrow of books; G2: Availability of books, to ensure books are always available for borrow; and G3: To manage the cost involved in this process. These three goals are measured by identified metrics. For example, G1: To encourage borrow book can be measured using the metric total number of books borrowed per year. Similarly, Availability of books can be successfully measured by the metric Number of overdue books.

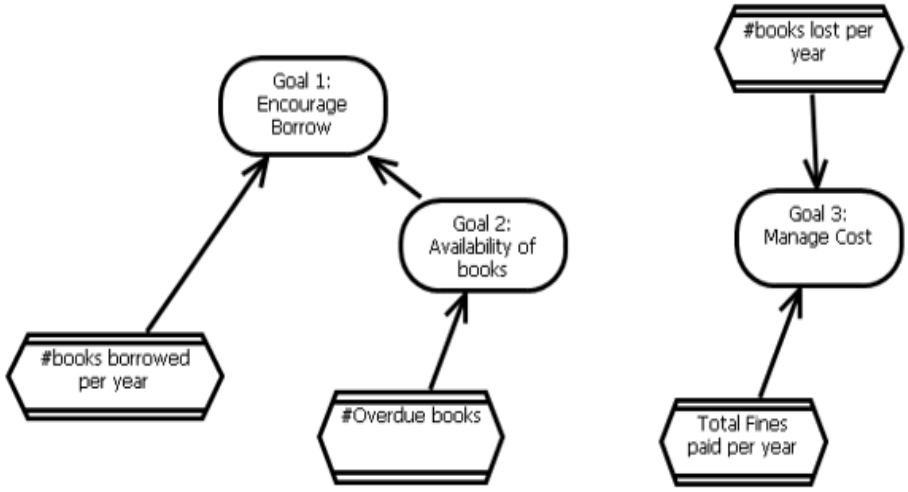


Figure 2-4: Sample goal model for *Library Borrow Book* Process

2.1.12 Business Intelligence

Business Intelligence (BI) serves as a bridge between the organisation's operational data and business performance information (Kruppke & Bauer, 2006). BI, defined as, "the application of various advanced analytic techniques to data to answer questions or solve business problems" (Trkman, McCormack, de Oliveira, & Ladeira, 2010) can be used as an integrated solution to provide easy access to business data (Moss & Atre, 2003). BI is the ability of an organisation to take all its capabilities and convert them

into knowledge, ultimately getting the right information to the right people, at the right time via the right channel which produces large amount of useful information that can lead to development of new opportunities in the organisation (Rud, 2009). BI technologies provide historical, current and predictive views of business operations.

BI services often access the data needed from a Data Warehouse which is a "Subject-Oriented, integrated, time-variant, and non-volatile collection of data that helps in supporting decision making process by data analysts in an organisation" (Inmon, 2005). A Data warehouse contains consolidated historical data from which the organisation can analyse its business. It is a copy of transaction data specifically structured for query and analysis (Kimball & Ross, 2013). A data warehouse is typically maintained separately from an organization's operational databases (Alsquor, Owoc, & Ahmed, 2014) and supports online analytical processing (OLAP), which focuses on the more read-oriented queries and statistical analysis.

2.2. Quality Assurance

2.2.1 Verification and Validation

Software Testing is an essential part of software quality assurance. It is the fundamental aspect of Software Engineering, but is often forgotten or given least importance in today's fast development culture. According to Jim Collofello and Kalpana Vehathiri, "Software Testing is a time-honored approach for evaluating the software in terms of correctness, robustness, efficiency, functionality, and ease of use" (Collofello & Vehathiri, 2005). Software should be predictable and consistent, offering no surprises to users.

Software Testing is an important activity that an organisation should perform throughout the software development life cycle. According to Manjit Kaur and Raj Kumari, the main aim of Software Testing is to identify all the defects that exist in a software product. Software Testing as a process of exercising and evaluating a system or system components by manual automatic means to verify that it satisfies specified requirements or to identify differences between expected and actual results. Testing is the measurement of software quality (Kaur & Kumari, 2011). We measure how closely we have achieved quality by testing the relevant factors such as correctness, reliability, usability, maintainability, reusability and testability. (Kaur & Gupta, 2013).

Software Testing poses two main crucial questions- What exactly is the software supposed to do? And is the software really doing what it is supposed to do? This introduces two key terms commonly used in software engineering: Verification & Validation (see table 2-1).

TABLE 2-1: COMPARISON OF VERIFICATION AND VALIDATION

Verification	Validation
• Does our system design meet the specifications? • Does our implementation meet the specifications? • Is the problem statement and specification consistent?	• Does the system account for all the needs of the customers? • Is the delivered system doing what we claimed it would do? • Is our problem statement complete and captures the real existing problem?

Verification

Verification helps a tester to answer the question- "*Are we building the product correctly?*" Verification is concerned with checking if the software is well engineered, error-free and meets specifications, and helps the testers to determine if the software is of high quality (Easterbrook, 2010). It involves activities like- testing, inspection, design analysis, specification analysis that are involved in producing a high quality software (Easterbrook & Callahan, 1998).

Validation

Validation is the process of checking if the specification captures the customer's needs and helps to answer the question- "*Are we building the right system?*". Activities like- Requirements modelling, prototyping and user evaluation are included in validation of a system.

2.2.2 Software Development Life Cycle

Iterative and incremental software methodology is widely used, which can survive any change and uncertainty during the development process and delivers software as increments containing a set of implemented features (Stoica, Mircea, & Ghilic, 2014). Some of the commonly employed iterative and incremental methodologies include:

Rational Unified Process (RUP) that uses Unified Modeling Language (UML) to deliver high quality software within the given time frame and quoted budget (Kruchten, 2004); Agile Software Development, that can handle changing business requirements and deliver the promised quality of software at a faster pace (Biju, 2008). Of all the existing agile methods, Extreme Programming (XP), that involves breaking of user stories-based requirements into tasks and prioritization of these tasks by the customers for the purpose of implementation (Beck & Andress, 2004); and Scrum, which demands the participation of customers in the software development (Biju, 2008); are known to be used extensively (Cohen, Lindvall, & Costa, 2004). *Test Driven Development* (TDD) is another agile method that involves creation of automated test cases for a new functionality, before its implementation. This demands the developers to thoroughly understand the requirements before writing the code. It involves development of program units without a large amount of designing upfront (Gupta & Jalote, 2007). It is reported by IBM that adoption of TDD has reduced the number of defects by 50% (Maximilien & Williams, 2003).

Behavior Driven Development (BDD) is an emerging methodology in which the stake holders determine specifications of the system's desired behavior, in detail. The main objective of BDD is to bring together different members of the projects which includes- customers, developers, and testers , provide a collaborative environment to ensure all the participants are in sync with the requirements understanding (Soeken, Wille, & Drechsler, 2012). Scenarios are designed to assist the stake holders and the development team consisting of the business analysts, developers and the testers to write their tests (Solis & Wang, 2011). BDD involves user stories for features and scenarios that describe the expected behavior of the system and each scenario is mapped to one test

method and hence success of all the test methods imply success of a scenario (Solis & Wang, 2011).

In this research, we have followed agile principles that are model-driven and also behavior driven involving user scenarios in order to describe the behavior of the system, to ensure its validation against these to-be satisfied scenarios.

2.2.3 Software Testing Architecture

There exist three main types of Software Testing Architecture relevant to testing software:

Black box Testing

Black box testing is the verification of an item by applying test data derived from the specified functional requirements without considering the underlying software architecture or composition (Collofello & Vehathiri, 2005). Black box testing involves infusion and extraction of data through a software interface (Coulter, 1999).

Black box testing is also regarded as functional testing which ignores the internal mechanism of a software system or a component and focuses only on the outputs generated in response to the fed inputs and execution conditions. So, the testers performing black box testing have no access to source code. Figure 2-6 illustrates black box testing. The system or component under test is viewed as black box and identify the circumstances in which the system or component does not behave according to requirement specifications (Myers, 2004).

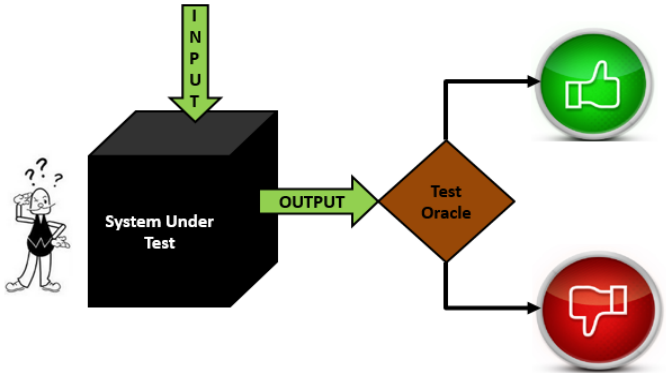


Figure 2-6: Black Box Testing

White box Testing

White box testing is also regarded as structural testing and glass box testing. The name itself indicates that it is a transparent testing that which takes into account the internal mechanism of the software system or component. This type of testing depicted in Figure 2-7, allows the testers to look at the code and write test cases. The software testers will have access to all the internal code/structure for the verification of a system or component. White box testing permits you to examine the internal structure of the program and derives test data from the examined program's logic (Myers, 2004). White box testing helps in revealing errors in hidden code and spotting the dead code, which is never reached. However, this type of testing demands testers to have in-depth technical knowledge and skills to implement test cases.

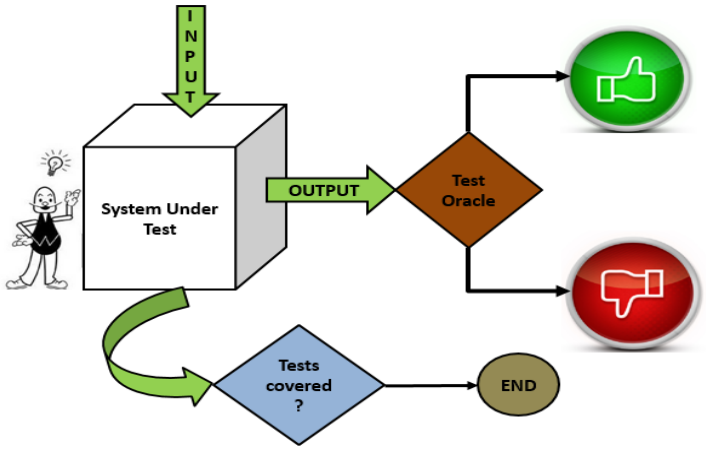


Figure 2-7: White Box Testing

Gray box Testing

This test approach depicted in Figure 2-8, has a combined flavor of black box and white box testing. According to Khans, "Gray box is a technique to test an application with limited knowledge of internal working of an application and with the knowledge of fundamental aspects of the system" (Khan & Khan, 2012). It is especially appropriate for testing in the context of SOA. When using gray box testing in the context of a SOA, the individual services are treated as black boxes and the application or online business process that is making calls to the services is treated as a white box.

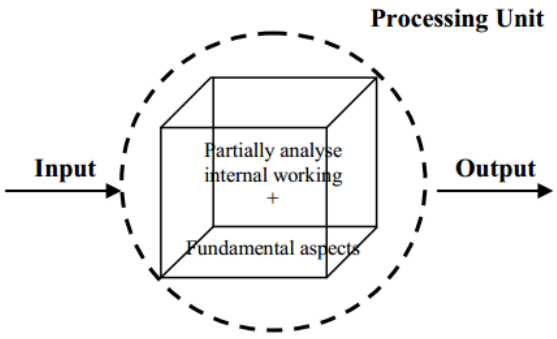


Figure 2-8: Grey Box Testing (Khan & Khan, 2012)

2.2.4 Web Application Testing

The aim of Web Application testing is to identify any failures in the required functionality and to ensure the application behaves in a way the requirement demands it to be. The web application testing consists of executing the application using combinations of input and states, to reveal failures (Lucca & Fasolino, 2006). The main objective of web application testing is to identify failures in the required services, and to verify the conformance of the behavior of the application with the provided requirement specifications.

The web application testing traditionally has used manual or automated interface testers for the GUI and the unit testers for the underlying services and application objects. The unit testers typically test one service at a time using unit testing tools like JUnit (JUnit, 2014), an open source framework; HTTPUnit and ServletUnit (Stepien, Peyton, & Xiong, 2008). Using JUnit and DBUnit, testers can compose integration test cases in java which ensures in-flight data modifications as expected (Nogueras & Olivieri, 2013). Selenium is a portable open source software tool, commonly used in testing browser forms. Selenium tests can be written in a number of programming languages and can run directly in most Web browsers. It helps to record the user actions performed, which can then be rerun for functional testing by tweaking the parameters.

2.2.5 *TTCN-3*

Testing and Test Control Notation version 3 (*TTCN-3*) is a test specification and test implementation language created by industry and academia experts at the European Telecommunications Standards Institute (ETSI) (TTCN-3, 2014). It is a powerful

scripting language and has been successfully employed in testing composite applications enabled in SOA (Peyton, Stepien, & Seguin, 2008). It has been extended to web penetration testing that involves- SQL injection and XSS attacks (Stepien, Xiong, & Peyton, 2011).

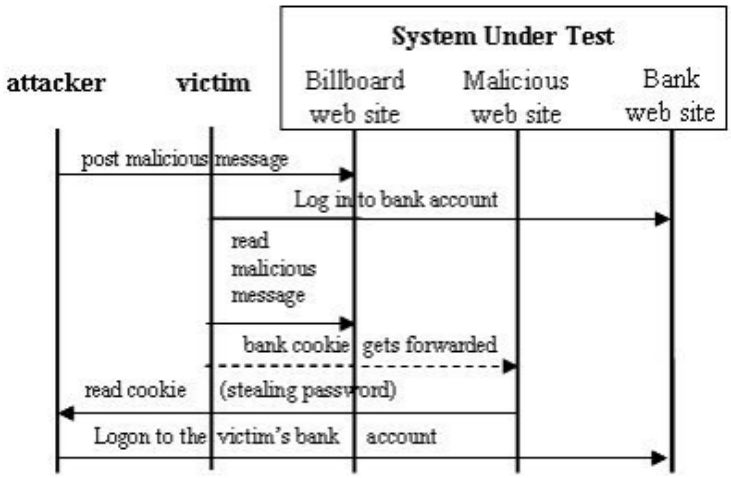


Figure 2-9: XSS attack sequence of events (Stepien, Xiong, & Peyton, 2011).

TTCN-3 has a fundamental model of separation of concerns which is clearly depicted in the Figure 2-10. This separates the Abstract Test Suite (ATS) from the coding/decoding and communication, and presentation details; providing powerful matching mechanism that separates behavior and the conditions governing the behaviors and there by promoting a systematic approach to testing. This separation of concern provides full portability of test suites, making them independent of platform implementation (Stepien, 2015).

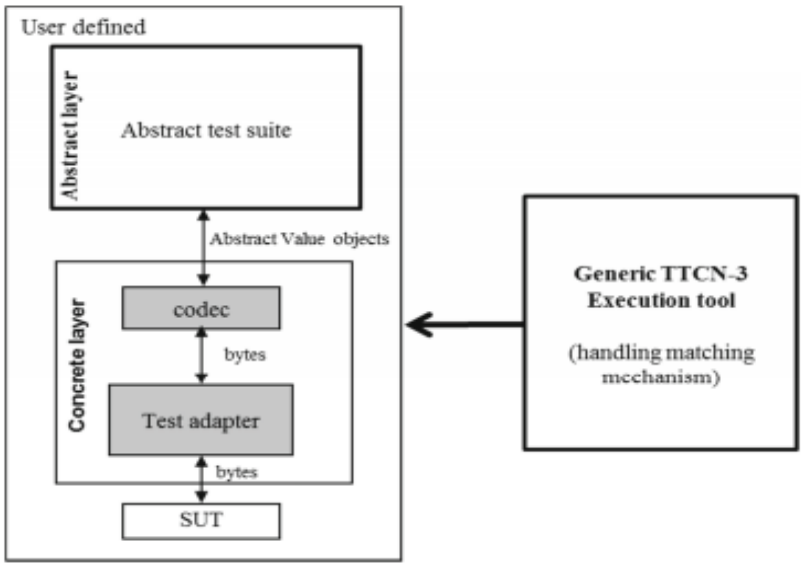


Figure 2-10: *TTCN-3* Separation of Concern (Stepien & Peyton, 2013)

TTCN-3 provides powerful re-usable test oracle specification language construct called a template and corresponding built-in matching mechanism for test oracles verification. The template concept is based on data structures and allows for the specification of multiple test oracles in one single operation. The *TTCN-3* concept of template is a double edged concept that is used both to define data to be sent to an SUT and test oracles to be matched against incoming messages. A template resembles a structured variable data assignment as follows:

```
type record MyType {
integer a,
charstring b
}

template MyType temp_1 := {
a := 5,
b := “abcd”
}
```

The above template can be used to send its data to the SUT using the *TTCN-3* send keyword as follows:

```
myPort.send(temp_1);
```

It can also be matched against incoming messages using the *TTCN-3* receive keyword as follows:

```
myPort.receive(temp_1);
```

This concept allows powerful re-usability compared to the usual assertions of Junit for example. The *TTCN-3* concept of template actually enables the separation of concerns between test behavior and conditions governing behavior (Peyton, Stepien, & Seguin, 2008). This avoids the usual pitfalls of spot checking with its trail of errors and omissions. *TTCN-3* also provides a wide selection of tools that offer powerful test execution results inspection features that enable the tester to focus on individual elements of a web page.

TTCN-3 is an international standard that was originally designed for testing telecommunication protocols which consists of discrete messages between communication entities (Stepien & Peyton, 2013). It is a strongly typed test scripting language which was successfully employed in conformance testing of communicating systems, where in discreet events like sending/receiving of data units of protocol and abstract service primitives are used. Conformance test suits are developed based on the

protocol specifications (Schieferdecker, Stepien, & Rennoch, 1997). *TTCN-3* is also employed in testing avionics systems that send or receive huge series of periodic messages with an identical payload at precise time intervals (Stepien & Peyton, 2013).

TTCN-3's matching mechanism enable multi-user matching of request/responses with precise detection and location of faults and quality of service issues. The test specification approach used in *TTCN-3* supports parallel testing and promotes reusability of data types tied to reusable components (Stepien, Peyton, & Xiong, 2008).

2.2.6 Business Process Testing

Business Process Testing (BPT) involves testing the behavior of developed business processes (Jarrar, Al-Mudimigh, & Zairi, 2000). Business process often sends data to various systems, fetches data from other systems, integrates with web services, creates events, sends notifications, and assigns workflow tasks that may have their own rules. It is the responsibility of testers to extensively test the business process work flow before it is deployed in a live environment. According to Robic (Robic, 2010), there are two important things for testing workflow instances.

- *Checking if the workflow is in a state as it should be according to business rule.*
- *Verifying if the workflow related properties are set to expected values.*

There has been much work that defines formal semantics for process flow, and verifies the correctness of the work flow in terms of the semantics of the orchestration of the business process (Wynn & Kyaw, 2006) (Dijkman, Dumas, & Ouyang, 2008).

Traditionally, there has not been an emphasis on testing of user interfaces (including usability and adoption) and validation that the business process is achieving the desired business results (e.g. improving business performance) (Kuziemsky, Liu, & Peyton, 2010).

2.3. Related Work

In this section, we summarize related works that have attempted to address quality assurance of BPM.

2.3.1 TASSA- framework for Business Process Quality Assurance

Ilieva *et al*, propose a SOA-based framework called *TASSA* that supports automatic test case generation for path coverage functional testing (Ilieva, Manova, & Petrova-Antonova, 2012). The approach also aims at providing fault injection technique for negative functional testing. The framework methodology proposes isolation of business processes from web services and targets end-to-end business process testing.

The framework encompasses many tools:

- ***Isolation tool***, which targets removal of BPEL dependencies from one or more external web services;
- ***Fault Injection tool***, whose main task is to generate faults for negative testing;
- ***Value Generation tool***, whose goal is to create valid values for all the participating field variables;

- ***Data Dependency Analysis tool***, whose main duty is to generate a list of needed variables for a path in business process;
- ***Test Case Generation tool*** provides with test cases for all the identified executable paths of the business process and help to promote regression testing.

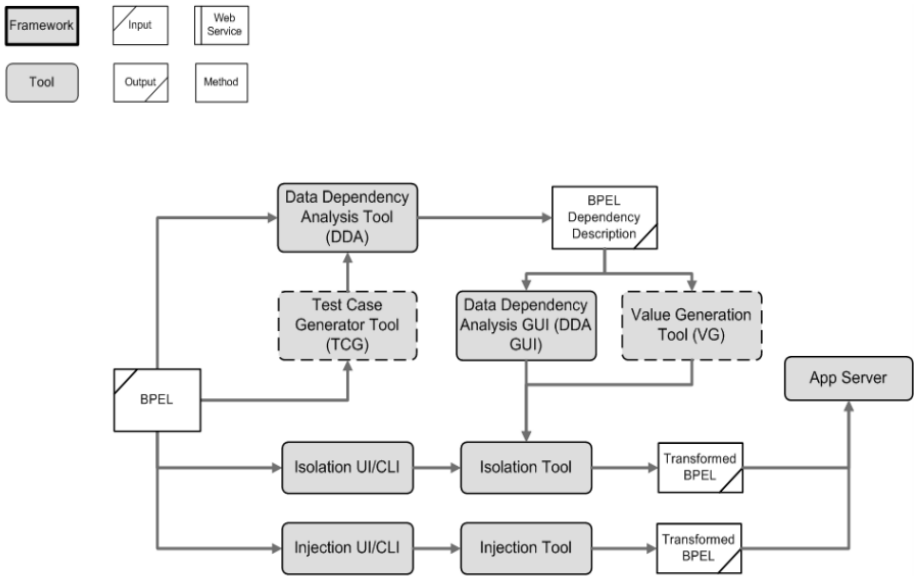


Figure 2-12: TASSA tool used for Business Process Testing (Ilieva, Manova, & Petrova-Antonova, 2012)

2.3.2 Business Process Verification with Reset Nets

Wynn et,al define business process verification as a process to determine, in advance, if the developed business model exhibits certain desirable behaviors (Wynn, Verbeek, van der Aalst, ter Hofstede, & Edmond, 2009). Verification during design time helps in identifying potential problems and thus helps in correcting the model before taking it online. The verification approach proposed by the authors, focus on two main features:

- ***Cancellation***, which captures the interference of an activity in the execution of others resulting in either stopping currently running activities or disabling some scheduled activities, and
- ***OR-Joins*** which are suitable in '*Wait and See*' modelling situations for assessing the correctness of real-life models and its semantics require synchronization depending on an analysis of future execution paths (Wynn, van der Aalst, ter Hofstede, & Edmond, 2006).

The authors have proposed four desirable properties that can be verified for a business process with the Cancellation and OR-Joins:

- *Soundness*, which is composed of three desirable sub properties, namely: Process when started, can always complete (option to complete); Proper completion with no other tasks being active (proper complete execution) and the tasks that will never be executed (no dead transitions).
- *Weak Soundness*, which indicates completion of the business process in some cases.
- *Irreducible Cancellation Regions*, where in the process contains elements that can never be cancelled during the execution of the task.
- *Immutable OR joins* property of a process ensure non-existence of convertible OR joins (the joins that can be replaced by XOR or AND joins).

The proposed verification approach exploits coverability and reachability notions from reset nets (Wynn & Kyaw, 2006). The approach relies on using formal methods that

employ mapping of a business process to reset net and performing state space analysis. The work is evaluated on a business model that depicts Australian Visa application process, constructed using *YAWL*, workflow language (Wynn, Verbeek, van der Aalst, ter Hofstede, & Edmond, 2009).

2.3.3 Business Process Validation with SARI

Joseph et.al, have proposed a test framework for business process testing based on process test specifications, a skeleton for defining functional tests (Schiefer, Saurer, & Schatten, 2006). It consists of:

- ***Simulation Models***, employed to generate data for the entire business scenario.
- ***Mocks***, which are created for systems and services that has enough code to fool the process and track its behavior.
- ***Test Communication*** supports test scripts and offers two components:
    - Injector service to pass input data from the simulator to the system interface within the process.
    - Collector service to collect data generated during the execution of process.
- ***Event Patterns***, used to generate assertions to identify sequences of events and describe the selected events for the execution path of business process.
- ***Assertions*** to test the assumptions made about a business process.
- ***Test Scripts*** to invoke a portion or the entire business process.

The framework is developed to address various business process quality issues such as, validation and correctness of business process model, detection and diagnosis of issues in business process performance, identification of malfunctioning steps in business processes.

The framework includes different test classes at different abstraction levels- 'Code and Unit Tests' forms the basis of framework and is responsible to ensure correct working of every implementation unit constituting the business process application. 'Component Tests' aims at testing communication scenarios with the identified interfaces of functional and logical components. 'Interaction Tests' are responsible for testing integration of different parts of business process system. And finally the 'End-to-End Tests' which aim at testing complete sense and flow of business process to verify process's correct behavior from an end user perspective. The Figure 2-13 indicates the elements of process test specifications and describes how they are linked to with the business process model. The proposed framework is used, after defining a process test specification, to make process descriptions testable.

The proposed framework is tied to ***Sense and Respond Infrastructure*** (SARI), an agile business process management platform (Schiefer, Saurer, & Schatten, 2006), whose main target is to help organizations in controlling and monitoring business processes. SARI allows modelling and execution of different kinds of sense-respond processes. Events are successfully captured and transmitted to SARI for initial unification before the processing of data. The events are later transformed into more meaningful business details (like, metrics, business situations, etc). This generated business information is then

analysed against the business goals to find best possible solutions to improve current enterprise situations (Schiefer & Seufert, 2005).

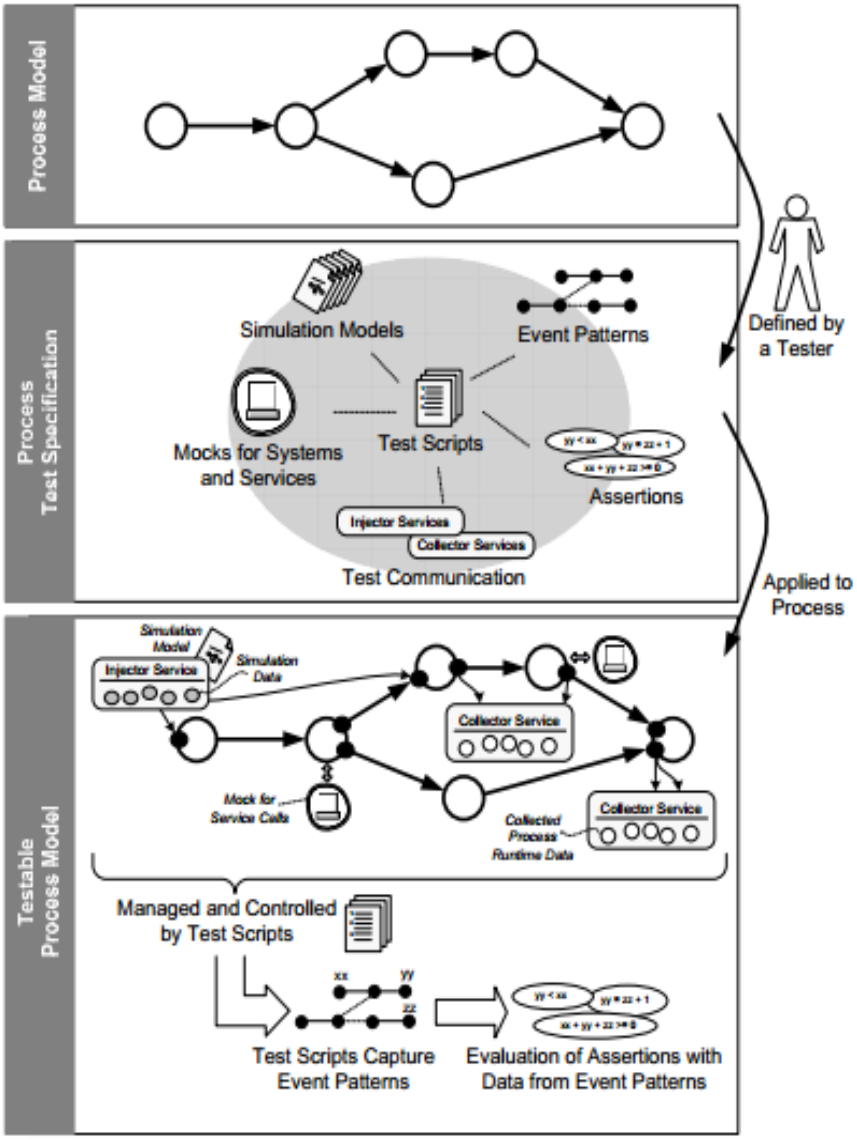


Figure 2-13: Testing Business Processes with a Process Test Specification (Schiefer, Saurer, & Schatten, 2006)

2.4. Chapter Summary

This chapter has presented the key terms and concepts used in this thesis. We have reviewed the basic concepts of BPM and testing prior to the presentation of our quality assurance framework. This chapter provided a brief overview of business process testing related approaches and technologies. We have presented a literature review on existing academic works related to business process testing. In the next chapter, we will present our business process testing approach and explain the proposed quality assurance framework.

Chapter 3. Model-Driven Quality Assurance Framework for BPM

In this chapter, we present our quality assurance framework for BPM. The chapter begins by characterising the problem and providing a gap analysis of current practice and related work. A set of evaluation criteria are identified that can be used to evaluate the proposed approach to address quality assurance for BPM. Then we provide an overview of our framework and its methodology for quality assurance of BPM, before discussing the different components in detail.

3.1. Problem Description

Companies are increasingly taking their business processes online using BPM tools and technologies which allow them to model explicitly the orchestration and interaction of tasks performed by roles (people) and systems (web services) that define a business process, and execute that process as a web application that provides forms for roles to interact with and service interfaces to the organizations SOA. Often quality assurance for such online business processes are handled as just another case of testing a web application,

Figure 3-1 illustrates the typical approach in current practice in terms of the interactions between the process experts defining the business process for an organization, and the development team that builds the online business process, and the quality assurance team that tests the business process.

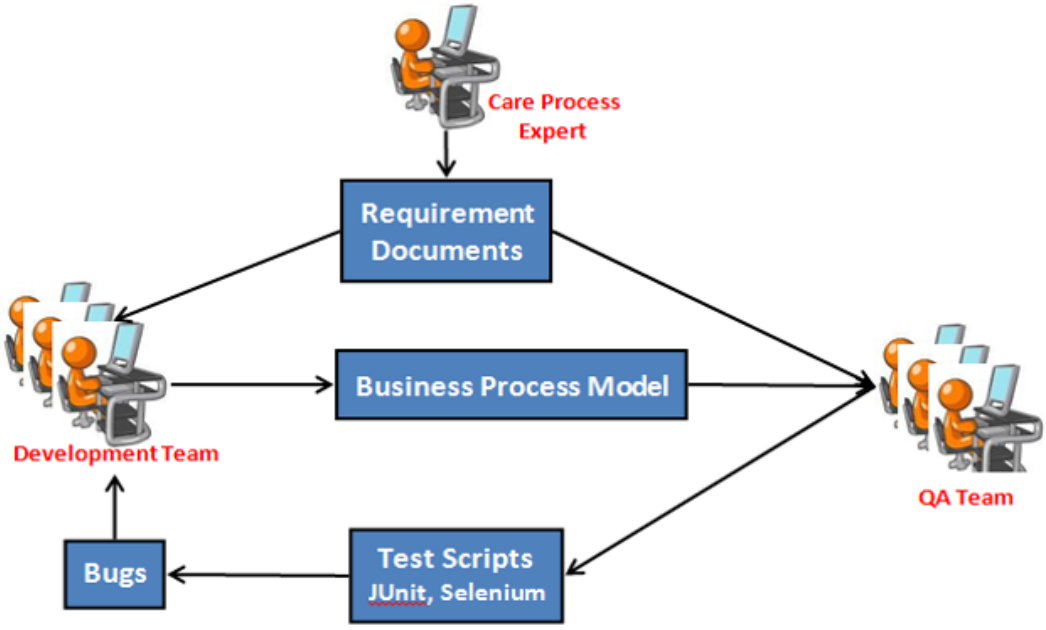


Figure 3-1: Current Business Process Testing Approach

Process Experts are typically either stakeholders or managers or expert participants in the business process who can articulate how the business process is supposed to work, or they are a business analyst who interacts with the stakeholders to capture requirements. The challenge is that they must be very familiar with how the As-Is process currently works, but also be able to understand how the To-Be process needs to be adapted to take advantage of online technology. Their understanding of how the To-Be process should work is captured in Requirements Documents that are read and interpreted by both the development team and the QA Team. Typically, these are natural language documents, structured in a standardized manner that is used for the development of any web application. There may be diagrams and even high level models of the business process included but they are not grounded in the specific details of the SOA of the organization. Nor are they specific in the details of the forms the different roles might use. Often, there is no direct mention of the goals that the business process is supposed to

support, or of the metrics that will be used to measure performance. Typically, these are either implicit or mentioned as high level objectives without detail.

Development Team are typically web application or web service developers who are introduced to BPM as new tool they can use to build applications. They use a BPM tool to create a BPM model that can be executed by the BPM engine to satisfy the requirements given to them by the process experts. There are preliminary discussions and negotiations around the requirements, but after that most interactions are around reviews of the running process once it has been implemented and released. The quality assurance team provides most of the feedback through the logging of bugs. Typically, the development team builds web services as well as the business process model, as needed in order to interface to legacy systems within the organization. They also build the forms that are used by the different roles to interact with the business process.

Quality Assurance Team tests the execution of the business process implemented by the development team, and logs bugs based on their interpretation of the requirements documents. Typically, the quality assurance team tests the business process in the same way they would test any other web application. Often, there is ad hoc testing of the system by testers manually interacting with forms in the manner they assume the different roles would. Interface testing tools, like Selenium, can be used to automate some of this interface testing. Unit testing tools like Junit can be used to test the underlying web services as well, independent of the business process, although this is often done by the development team.

Unfortunately, typical web application testing is focused on testing an application for a single task, from the point of view of a single user whereas the orchestration of tasks across many users and software systems can be quite complex for an online business process. This issue is compound by the fact that the requirements documents are difficult to interpret and there is ambiguity or confusion about how the business process is supposed to work. In addition, the current approach to testing fails to take advantage of the fact that there is an explicit detailed model of the business process. Model-driven testing approaches should be employed to ensure consistency and completeness. Perhaps most significantly, testing a web application as if it were a standalone system, ignores the fact that the business process model has been designed to contribute positively towards the business goals of the organization and its performance should be measured and analyzed in that context. A more systematic approach to verification and validation of online business processes is required

3.2. Gap Analysis

In summary, the following gaps exist with the current approach to quality assurance for BPM:

- Requirements captured are unclear, ambiguous, vague, and difficult to understand.
- Business process model is not used to drive quality assurance.
- The approach to defining and running test cases is ad-hoc and labour intensive.

- Tools are not used systematically to address business process model execution in the context of an SOA
- Failure to verify the complexity of multi-role, multi-server orchestration for BPM in a systematic way
- Failure to validate the business process against the business goals.

Recent research has attempted to address the above identified gaps.

TASSA framework described in section 2.3.1 (Ilieva, Manova, & Petrova-Antonova, 2012), supports model-based testing of business process and has attempted to test orchestration of multiple roles independently of the services used. The framework offers tools support for the generation of tests from a BPEL definition. The framework does not address multi-role, multi-service orchestration in parallel. The framework does not address validating the business process against business goals.

The work by Wynn et.al, described in section 2.3.2 (Wynn, Verbeek, van der Aalst, ter Hofstede, & Edmond, 2009) has suggested design-time business process verification to identify issues before the process is taken into production. The work suggests formal verification approaches for service orchestration using Reset nets. Run-time issues like performance are not addressed. It also does not address multi-role, multi-service orchestration, in parallel, nor does it address validation against business goals.

The test framework (Schiefer, Saurer, & Schatten, 2006) proposed by Joseph et.al, described in section 2.3.3 support unit testing of services and end-to-end testing involving service orchestration. The framework addresses validation of business processes against

goals and has demonstrated this by coupling their framework to an agile business process management platform called *SARI*. However, it does not address form testing, or multi-role, multi-service orchestration in parallel.

In summary, it is learned that all of the above works have addressed some of the identified gaps and have potentially missed verifying combination of multiple roles and multiple service orchestration. Our proposed quality assurance framework, will attempt to close all these identified gaps in a more systematic manner.

3.3. Evaluation Criteria

We have identified criteria that can be used as a basis for evaluating any approach to quality assurance for BPM. In particular, in chapter 5, we will use them to evaluate our proposed framework and compare our framework with the related work. The set of evaluation criteria are deduced based on the following factors-

- Careful study and analysis of literature review related to quality assurance of business processes including related works.
- Feedback obtained from BPM experts and testing experts, and process experts that we have interacted with during our research. This includes: Bernard Stepien who is a TTCN-3 consultant with 20 years' experience, Randy Giffen from IBM who is both a BPM expert and a medical doctor, Shahid Shamsi who is a BPM development manager at a large teaching hospital, as well as several business analysts from the same hospital.

- Experiences gained from our two case studies: ***library borrow book*** process and ***hospital cancer care assessment*** process, described in Chapter 4.

We have grouped the criteria into four categories: Methodology, Business Process Features, Test Case Definition and Tool Support.

3.3.1 Methodology

A systematic approach to quality assurance for BPM is needed so that the complexity of business processes can be addressed in a model-driven fashion. The following are key elements related to quality assurance methodology that any framework solution should support.

SOA

This criterion comes from the related work (Ilieva, Manova, & Petrova-Antonova, 2012) and the experience we had with the case studies. A well-defined SOA, should be systematically leveraged by the methodology supported by a quality assurance framework.

Goal Model

This criterion comes from one of the related works (Schiefer, Saurer, & Schatten, 2006), that perform business process validation against goals. It is familiar from the literature (Amyot Daniel, et al., 2012) that goal models capture the objectives of stake holders and are important to business process quality. And hence, goal model should be

used in the definition of requirements against which the business process will be validated.

Scenarios

This criterion comes from the related work on behavior-driven development (Soeken, Wille, & Drechsler, 2012). From our case studies and experiences shared by business process testers, we have identified that requirements gathering and documentation for any business process should be in terms of user scenarios which are unambiguous and easy to understand and model.

Workbench

This criterion comes from related works (Ilieva, Manova, & Petrova-Antonova, 2012) and (Schiefer, Saurer, & Schatten, 2006) which involve frameworks that support tool integration to assist business process testing. Business processes are complex and there are many different aspects involved in quality assurance, so it is important to be able to flexibly integrate tools as needed.

Portal

This criterion comes from the literature survey (Peyton, Zhan, & Stepien , 2008) and also from the experiences gained from our case studies. It is necessary for a framework to provide timely details on the status of testing. In addition to testing and logging of defects, there should exist an interface for reporting, monitoring and analysis of process functionality.

Production

This criterion comes from one of our related works (Schiefer, Saurer, & Schatten, 2006). It is important to validate the business process against business goals, even when the business process development life cycle is complete and the process is taken online. It is from our case studies that, we have learned the importance of monitoring business processes in production to ensure the process is constantly meeting the business goals.

3.3.2 Business Process Features

This criteria category targets various business processes testing, including the validation of process against business goals.

Model-driven

This criterion comes from the related work (Ilieva, Manova, & Petrova-Antonova, 2012), where business process model is used to drive business process testing. This criterion accounts for the fact that in BPM, the business process model is central and quality assurance should be driven by that model. In addition, the environment in which business processes operate should be well-defined in terms of both the SOA, which is leveraged by an online business process and the goal model, which captures the business results that the business process is expected to contribute to.

Forms

This criterion comes from the literature survey (Reynolds, et al., April 2014). It is important to be able to test the different forms that are used by different roles at different points in the business process.

Services

This criterion comes from the study of literature (Steen, 2007). Business processes are defined in the context of a SOA and interact with services. It is important to be able to test them in a systematic and complete manner in the manner that they will be used, when the business process is executing.

Multi-Role, Multi-Service orchestration

This criterion comes from the related work on *TTCN-3* related to composite applications (Peyton, Stepien, & Seguin, 2008). It is relevant to ensure verification of the full complexity of interactions in an online business process when multiple instances of multiple roles accessing multiple services in parallel are taking place.

Goal Validation

. This criterion comes from the related work (Schiefer, Saurer, & Schatten, 2006), where in the business process is validated against the business goals to ensure the process is compliant with the business needs.

3.3.3 Test Case Definition

This describes the effort and skills needed to generate, execute and maintain test scripts for business process verification.

Test script effort

This criterion comes from the related works (Ilieva, Manova, & Petrova-Antonova, 2012) which involve generation of test scripts for business process verification. This criterion is used to determine the effort and skills involved in the

creation of test scripts. Test script creation can be completely automatic, semi-automatic or completely manual. Further, different languages for test scripts have different expressiveness which can affect the effort of creating test scripts. This criterion was also determined by our two case studies.

Test coverage

This criterion comes from the literature study (Lee, 2009). It is important to ensure that tests cover all paths, and boundary conditions for the business process.

Level of coding

This criterion comes from study of literature on *TTCN-3* (Stepien, 2015). It is also derived from the experiences gained from our case studies. It measures the amount of coding involved in generation of test cases for the quality assurance of business process. In addition, experiences gained from the case studies' test implementation have contributed to this criterion.

Test Reusability

This criterion come from literature study (Lee, 2009) and work related to *TTCN-3*, whose main feature is to support reusability of tests among different applications (Stepien, Peyton, & Xiong, 2008). Reusability of tests is one important criterion to be considered when making a decision on testing approach and methodologies.

3.3.4 Tool Support

This criteria category is used to measure and compare the features involved in the tools and effort involved in using these tools for execution of tests.

Test Execution

This criterion comes from the related work (Ilieva, Manova, & Petrova-Antonova, 2012). This criterion is used to describe the execution of tests during quality assurance of business processes. It describes if the execution of tests are automated with the support of tools or to be manually performed.

Test Environment

This criterion comes from the careful analysis of related works (Ilieva, Manova, & Petrova-Antonova, 2012) and experience from case studies. This criterion measures the initial effort and time involved in setting up the test environment for the quality assurance of business processes. This is also derived by considering the feedback tester's feedback who was involved in our case studies.

Test report and logging

This criterion comes from the study of literature (Stepien, 2015) that describes reporting and logging as an important characteristic feature of the testing tool- *TTCN-3*. This criterion is also devised from the feedback provided by the testers regarding the tools supported by our framework.

Ownership

This criterion comes from the careful analysis of related works (Ilieva, Manova, & Petrova-Antonova, 2012) that define custom-built internal tools to drive business process testing. This criterion is used to compare the ownership and to measure the availability of the tools.

Cost

This criterion comes from the related work (Ilieva, Manova, & Petrova-Antonova, 2012). This criterion describes the cost of the tools and technologies used in business process testing.

Skills

This criterion comes from the related works study (Ilieva, Manova, & Petrova-Antonova, 2012) and (Schiefer, Saurer, & Schatten, 2006) which involve use of custom build tools for addressing various business process testing. This describes the effort that a tester needs to invest in mastering tools, programming languages and the testing environment. This criterion is also used to measure the learning curve associated with the usage of framework and its tools and technologies.

3.4. Quality Assurance Framework

In this section, we give an overview of our framework in terms of a testing methodology that specifically addresses BPM, in which we detail the key roles and artifacts used to communicate between the roles. We then give a detailed description of each of the key components of our framework which supports the work of the key roles involved in quality assurance.

3.4.1 Framework Overview and Methodology

The Figure 3-2 depicts the proposed framework and shows how different participating key actors work together around well-defined modeled artifacts to provide model-driven quality assurance for BPM.

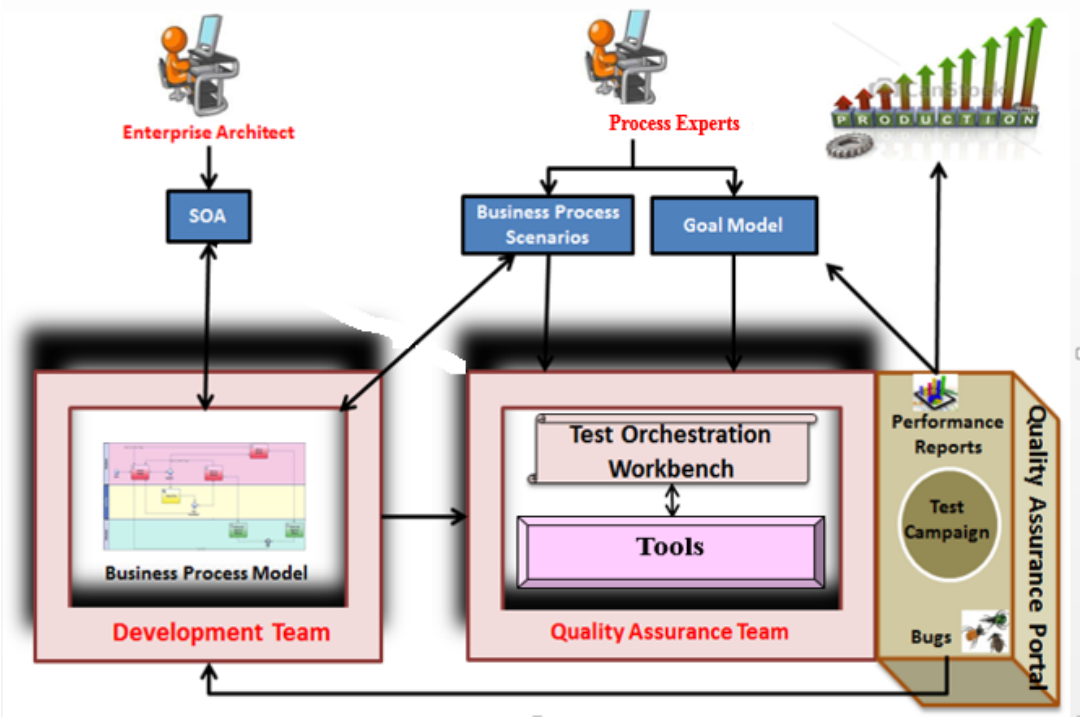


Figure 3-2: Proposed Business Process Quality Assurance Framework

The ***Process Experts*** are responsible for giving a clear goal model and a set of user scenarios. The requirements gathered are clearly translated into a goal model and a set of business process scenarios. Process Experts are also responsible for validating the business performance reports in the production to verify business goals are met.

The ***Enterprise Architect*** manages SOA which defines the list of services that can be used in a business process development. They interact with the development team to help them understand which services to use and also to identify if any new service needs to be created for the business process. The services created are clearly documented and handed off to the development team.

The ***Development Team*** receives the SOA document from enterprise architect and the user scenarios from the process experts and begins to develop the business process

model. The team is also responsible for analysing and fixing bugs, encountered during the business process development life cycle.

The ***Quality Assurance Team*** receives user scenarios and goal model from the process experts, business process model from the development team and begins to test the business process. The team leverages test workbench that supports model-driven testing of business process using testing tools to perform quality assurance of the business process. The team also uses test campaign for planning and executing the test cases. In addition to business process verification, the QA team is also responsible for validating the process against business goals. The team uses goal model, understands the business needs and identifies what metrics map to the goals. The generated business performance reports are then validated against the business goals. Any identified bugs during the process verification are logged in the bug tracking unit offered by the test portal. There is an aspect of quality assurance that can only be validated in production. The team uses test campaign to generate test reports that summarize the test runs.

3.4.2 Business Process Model

The Business Process Model defines a business process in terms of services used from the SOA, and forms used by the participating roles and the sequences of tasks and decisions that are performed based on the information input and output from services and forms. Our framework is setup so that all quality assurance activities are driven by the model (in relationship to the SOA, Scenarios and Goal Model).

An example of a business process model is shown in Figure 3-4. The process model is built for inventory management and involves two user roles- clerk and manager.

The model has a start point in the clerk's lane, which means the process gets started by the clerk's activity. The clerk and the manager actors have forms through which the information exchange happens. Also, the model has a decision box and hence there is a minimum of two user scenarios identified in this model. The model is built with an inventory management goal- to avoid stock-outs. The model is built leveraging services which help in fetching data from the database and also writing records to database.

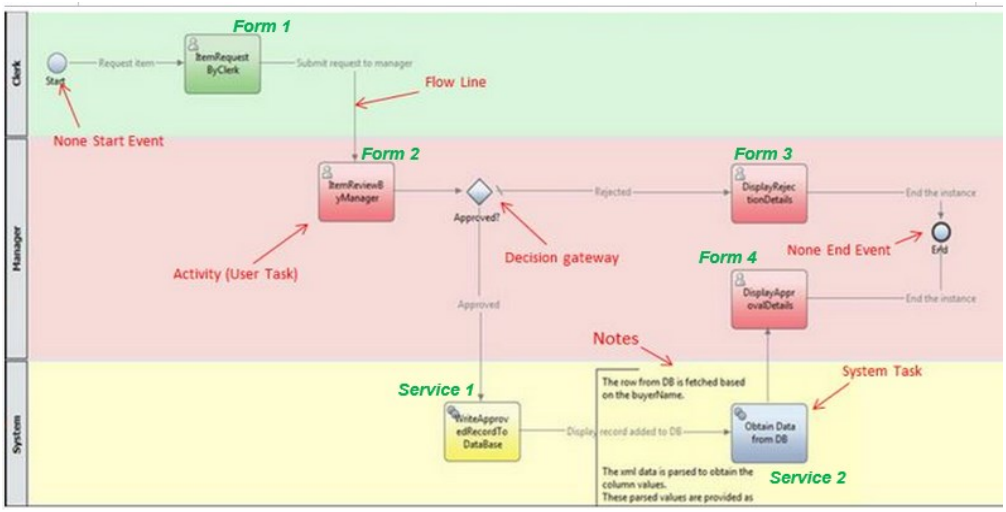


Figure 3-3: Sample Business Process Model

The development team consults the enterprise architect to figure out what services exist, and are suitable for the business process development. If no appropriate services exist, the SOA team builds the needed service, which is leveraged by the development team. The development team uses the user scenarios and identifies the execution paths by mapping each scenario to the flow thread. All the identified paths are to be integrated into a single model. The quality assurance team is responsible for validating the quality based on the scenarios and goal model, and must set up a test environment to do so. This is explained in the next following section.

3.4.3 Testing Environment

Figure 3-4 illustrates the test environment we have defined for our framework.

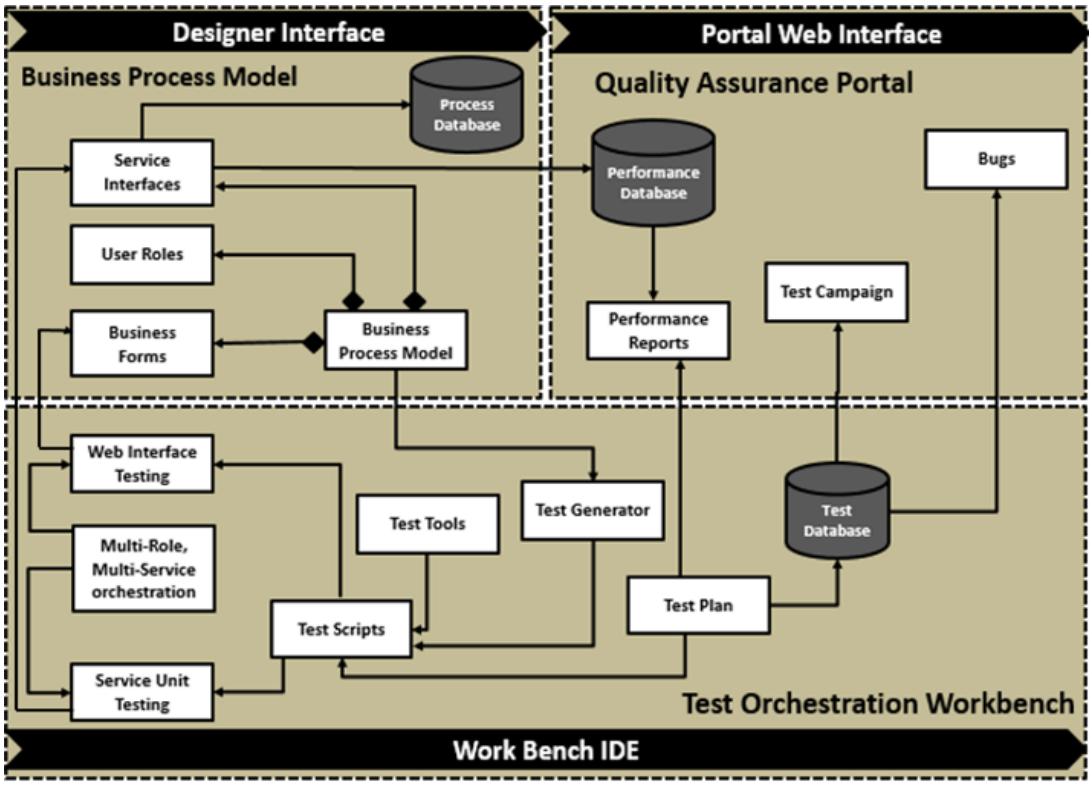


Figure 3-4: Proposed Test Environment

There are three main components: The Business Process Model, which developers have a user interface to work on, the test orchestration workbench which the testers have work bench IDE to work on and the QA Portal which the testers access via Portal web interface.

The *Business Process Model* drives testing. In particular, there exists a test generator in the *Test Orchestration Workbench* that can generate complete set of test scripts for web interface testing and service unit testing. Multi-role and multi-service orchestration testing is defined based on the scenarios given by the process experts. The

test generator generates an initial set of test scripts which can be modified to do multi role multi service orchestration. The scheduling of test execution happens via test plan. And the test plan is refined as necessary. There exists a process DB that supports business process model execution and is populated or read through the service interfaces. There is a test DB that stores all the test results and test data, used for testing. There is a *QA Portal* that communicates with everybody about the current status of the business process quality. It shows the current plan for the test campaign and the current status of the test campaign in case of test runs. It shows the current status of the bugs that needs to be fixed/already fixed. It supports performance reports that show whether or not the goals are being met.

3.4.4 Test Orchestration Work bench

Test Orchestration Workbench is a key unit in the proposed quality framework. The workbench is composed of tools identified suitable for complete business process testing. The Figure 3-5 explains the Test Orchestration Workbench used in our framework:

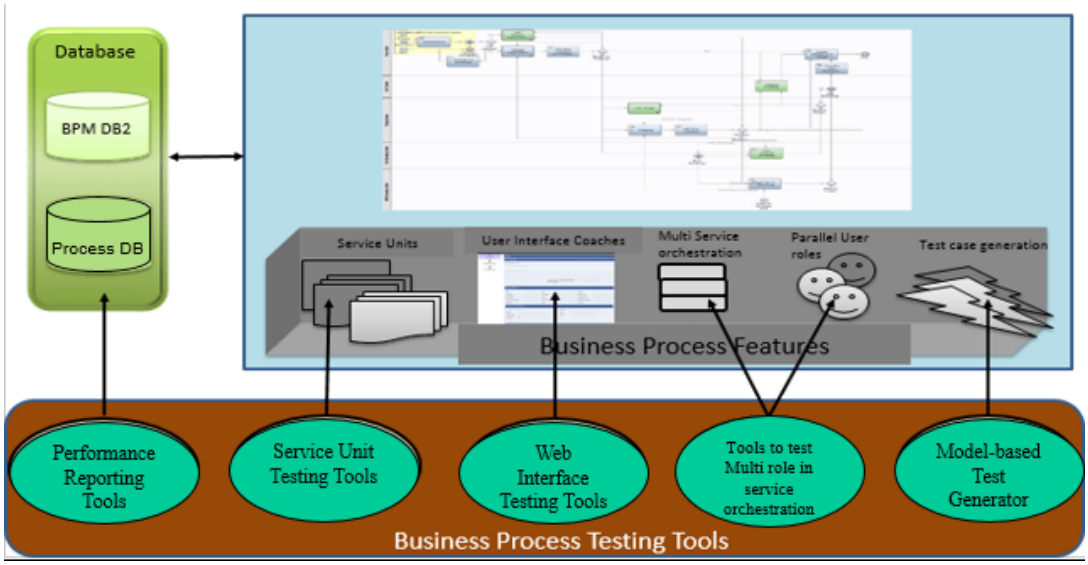


Figure 3-5: Test Orchestration Workbench

In Figure 3-5, the workbench consists of a class of tools for serving testing of different business process features. It is possible, there can be one single tool that can be used to drive quality assurance of business process, but so far in the market, such tool does not exist. The workbench supports a ***model-driven test generation tool*** that can automatically generate test cases for unit testing of services and forms, based on the business process model; an ***orchestration tool*** that supports testing of multi-user roles collaboration in multiple service orchestration, in parallel; and a ***performance reporting tool*** that can generate business process performance reports for the validation of process against business goals.

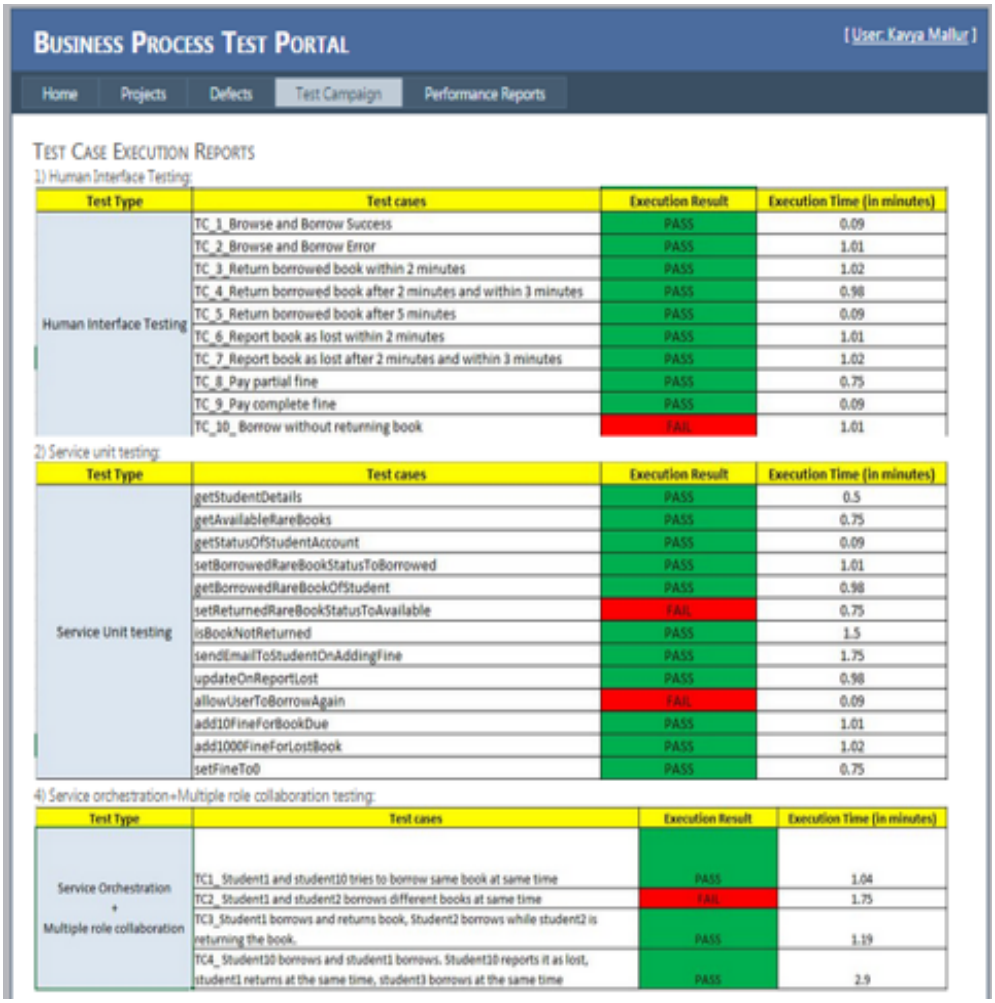
The model-driven test generation tool generates test cases for services and forms unit testing based on the business process model. IBM BPM Testing Asset is one such tool.

The orchestration tool can be successfully used to test multi role, multi service orchestration together, which is one of the identified gaps that has not been addressed. There exists a telecom standard called TTCN-3 that has been successfully employed to test multiple roles in multiple service orchestration in parallel, in the telecom domain. This tool is identified suitable for testing the same feature in business processes. However, the main challenge with this tool is to facilitate interaction with the BPM engine to test forms and services individually. It is possible to translate user scenarios that define the model paths into TTCN-3 test scripts.

Performance reporting tools can be used to generate business process performance reports. ***Cognos***, an IBM's product is one such reporting tool available in the market. Business Process performance reports can be generated using Cognos and can then be analyzed and compared against the goals and metrics for performance validation.

3.4.5 Quality Assurance Portal

The QA portal communicates quality for BPM projects, in terms of bugs, test campaign, and performance management. It is accessible by the Enterprise Architect, Process Experts, Development team and QA team. The following figure illustrates the QA portal built in asp.net



BUSINESS PROCESS TEST PORTAL [User: Kavya Mallur]

Home | Projects | Defects | Test Campaign | Performance Reports

TEST CASE EXECUTION REPORTS

1) Human Interface Testing:

Test Type	Test cases	Execution Result	Execution Time (in minutes)
Human Interface Testing	TC_1_Browse and Borrow Success	PASS	0.09
	TC_2_Browse and Borrow Error	PASS	1.01
	TC_3_Return borrowed book within 2 minutes	PASS	1.02
	TC_4_Return borrowed book after 2 minutes and within 3 minutes	PASS	0.98
	TC_5_Return borrowed book after 5 minutes	PASS	0.09
	TC_6_Report book as lost within 2 minutes	PASS	1.01
	TC_7_Report book as lost after 2 minutes and within 3 minutes	PASS	1.02
	TC_8_Pay partial fine	PASS	0.75
	TC_9_Pay complete fine	PASS	0.09
	TC_10_Borrow without returning book	FAIL	1.01

2) Service unit testing:

Test Type	Test cases	Execution Result	Execution Time (in minutes)
Service Unit testing	getStudentDetails	PASS	0.5
	getAvailableRareBooks	PASS	0.75
	getStatusOfStudentAccount	PASS	0.09
	setBorrowedRareBookStatusToBorrowed	PASS	1.01
	getBorrowedRareBookOfStudent	PASS	0.98
	setReturnedRareBookStatusToAvailable	FAIL	0.75
	isBookNotReturned	PASS	1.5
	sendEmailToStudentOnAddingFine	PASS	1.75
	updateOnReportLost	PASS	0.98
	allowUserToBorrowAgain	FAIL	0.09
	add10FineForBookDue	PASS	1.01
	add1000FineForLostBook	PASS	1.02
	setFineTo0	PASS	0.75

4) Service orchestration+Multiple role collaboration testing:

Test Type	Test cases	Execution Result	Execution Time (in minutes)
Service Orchestration + Multiple role collaboration	TC1_Student1 and student10 tries to borrow same book at same time	PASS	1.04
	TC2_Student1 and student2 borrows different books at same time	FAIL	1.75
	TC3_Student1 borrows and returns book, Student2 borrows while student2 is returning the book.	PASS	1.19
	TC4_Student10 borrows and student1 borrows. Student10 reports it as lost, student1 returns at the same time, student3 borrows at the same time	PASS	2.9

Figure 3-6: Quality Assurance Portal displaying Test Report

The portal has three aspects

- ***Test Campaign***

This manages the process of running test scripts against different releases of a BPM project and reporting the results. Test Campaign offers a platform for the planning and execution of test cases across all types of BPM testing and testing tools. It provides test reports that summarizes the test result of test runs and includes statistics related to business process testing. Test reports can include graphs which convey information

regarding test success/failure rate and time spent executing a test. Overall progress of the test campaign is communicated in terms of how many test cases are successfully executed, how may failed, and how many are still pending and needs execution.

- ***Performance Reports***

The performance reports for the business process are generated based on the data written to the performance data base while the business process is taking place. They include charts and metrics that communicate how well the business process is contributing to business goals. The accuracy and usability of the reports are tested and verified correct in the test environment as defined by the goal model, but the actual performance of the business process can only be validated when the process is deployed and being used in production. Reports can be generated on a scheduled basis to monitor the performance of business process, but they can also be run dynamically and used to generate alerts if the business process is underperforming.

- ***Bug Tracking***

The quality assurance team executes business process tests to perform verification and validation of the developed business process. Any bugs identified during this testing process are logged. The bugs logged by testers are addressed by the development team. This way, the quality assurance portal serves as a communication bridge between the quality assurance and development teams.

3.5. Chapter Summary

In this chapter, we have identified the gaps in business process testing and presented our QA framework for BPM. We successfully identified a set of criteria that are considered for the evaluation of our framework. The chapter provided a brief overview of proposed framework and quality assurance methodology. In the next chapter, we evaluate our framework using two case studies.

Chapter 4. Case Studies

The proposed quality assurance framework was evaluated in two case studies: *Library Borrow Book* and *Cancer Care Assessment*. In both case studies a prototype of our framework was used to test an existing BPM project which had implemented a business process using IBM BPM 8.5.5. In the *Library Borrow Book* case study, a process for ensuring rare books were returned when borrowed and fines paid when owed was developed and implemented for use in the uOttawa graduate course EBC6230 Business Process Management and Related Technologies. In the *Cancer Care Assessment* case study a process for managing assessment of cancer patients throughout their treatment was designed, implemented and deployed at a local hospital. The author of this thesis was a member of the development team on that project for one year as part of a Mitacs research internship.

Both processes at the time their case studies were started were experiencing persistent quality issues and there was general dissatisfaction with the results of the quality assurance that had been done. Both had been tested in a rudimentary way as web applications (more or less along the lines of what is described as typical of current practice in section 3.1 Problem Description). For both case studies, we first took a systematic approach to evaluating commercially available tools that were representative of the types of testing needed for BPM: *Junit* for unit testing web services, *Selenium* for interface testing of forms, *IBM BPM Testing Asset* for model-driven generation of test cases, *TTCN-3* for multi-role, multi-service orchestration testing, and *IBM Cognos BI Suite* for performance reporting. Each tool was evaluated independently to see how well

it could meet our evaluation criteria on its own when testing the business process for each case study. Then we implemented a prototype of our framework, and evaluated how well it worked.

All testing was done by the author of this thesis, who was familiar with Junit, Selenium and Cognos BI, but who had no previous experience with IBM BPM Testing Asset or TTCN-3. The *library book borrow* process was exactly as used in the EBC6230 course since this was already available at the university. The *Cancer Care Assessment* process was very similar but somewhat simplified from what is actually run at the hospital. This is so that it could be set up and analyzed at the University, where it was not practical to duplicate the exact SOA, networks and user groups that were in place at the local hospital.

In the rest of this section, we share the results of evaluating the individual tools (which were very similar between the two case studies) and explain the implementation of our framework prototype. Then in the rest of the chapter, we explain in detail how our framework performed and the results obtained when applied in each of the two case studies.

4.1.1 JUnit

JUnit is an open source tool used extensively in industry, for unit testing. In both case studies, it was observed that JUnit was successful and straight forward to use in unit testing the services used by each of the two processes. It took approximately *1 week* for the tester to create and run the unit tests for the services in the *Library Borrow Book* case

study, and approximately *2 weeks* for the tester to create and run the unit tests for the *Cancer Care Assessment* case study.

It was observed that JUnit could be used to unit test forms, but it required a lot of low-level script writing and manual effort.

There was no model-driven support for generating JUnit test scripts.

An attempt was made to write a script for a single multi-role, multi-service orchestration scenario. In theory, this should be possible, but in practice it was very difficult and complex to write and the attempt was abandoned after two weeks of effort. The major difficulty was coordination of the different scripts for each role.

Similarly, it would be possible in theory to write a JUnit script that communicated via JDBC to run a SQL query to report on a metric, but in practice this does not make sense for performance reporting.

4.1.2 Selenium

Selenium is an open source tool extensively used in industry for web interface testing. In both case studies, it was observed that Selenium was successful and straight forward to use for unit testing the forms used by each of the two processes. It had a nice graphical interface for recording form interactions that was much better than Junit. It took approximately *2 person days* to create and run the unit tests for the forms in the *Library Borrow Book* case study, and approximately *5-6 person days* to create and run the unit tests for the *Cancer Care Assessment* case study.

It was observed that Selenium could be used to unit test services, but it required a lot of awkward low-level configuration, script writing and manual effort compared to Junit.

There was no model-driven support for generating Selenium test scripts.

Like Junit, in theory it is possible to write scripts for multi-role, multi-service orchestration scenarios in Selenium but in practice it would be even more difficult and complex to write than in Junit.

Similarly, it would be possible in theory to write a Selenium script for performance reports that communicated via JDBC to run a SQL query to report on a metric, but in practice this did not make sense.

4.1.3 IBM BPM Testing Asset

BPM Testing Asset is an IBM tool for generating test scripts from business process model. It generates Selenium scripts. It was observed that it could generate a complete set of unit tests for services, and a complete set of unit tests for forms. It took about *2 man hours* to generate and run all such tests for the *Library Borrow Book* case study. It took about *3 man hours* to generate and run all such tests for the *Cancer Care Assessment* case study. The set of tests generated was far superior to the unit tests that had been written manually using JUnit or Selenium in terms of:

* The completeness of the test scripts generated especially for boundary conditions.

* The consistency of the tests generated.

* The ease of managing and maintaining the scripts.

BPM Testing Asset, however, offered no support whatsoever for either multi-role, multi-service orchestration testing, or for performance reporting. However, BPM Testing Asset does automatically generate very good testing reports including charts of times taken and errors occurred.

4.1.4 TTCN-3

As described in chapter 2.2.5 *TTCN-3* is a standard for Telecommunications testing that can test multi-user, multi-service orchestration scenarios. It was observed that it can be successfully applied to BPM projects in general for this purpose, and it was successfully used for both case studies. However, there was a significant investment in time and effort, to define a 'codec' or concrete layer implementation for BPM in general and to customize it for each project. It took *3 weeks* to create an initial codec for the *Library Borrow Book* case study, which included a general library for BPM. And it took an additional *1 week* to extend the 'codec' for the *Cancer Care Assessment* case study. The codec allowed complex multi-role, multi-user scenarios to be written in the high level *TTCN-3* specification language and then automatically translated into a combination of Junit scripts for service tests and HTML Units scripts for forms tests.

It took *3 days* to write multi-role, multi-user orchestration scripts that successfully tested all *6 scenarios* as multi-user multi-service orchestration scripts for the *Library Borrow Book* case study. And it took *5 days* to write multi-role, multi-user orchestration

scripts that successfully tested *5 of the 17 scenarios* for *Cancer Care Assessment* in combination.

The effort to write scripts is proportional to the number of user scenarios and the length of the scenarios (in terms of business process steps). Once the scenarios are specified it is a simple matter of configuration to do multi-role, multi-service orchestration testing of all scenarios in parallel. The scenarios for *Cancer Care Assessment* were longer than the ones for *Library Borrow Book*. It took roughly 2 days each to capture the details of a single scenario in *TTCN-3*. However, once all scenarios are captured, it took only an hour or two to specify complete testing of all possible multi-role, multi-user orchestration combinations across all scenarios. It is estimated that another *26 days* would be required to complete all multi-role, multi-user orchestration testing for the *Cancer Care Assessment* case study. It should be noted, that the tester was the author of this thesis, who had only a few days training in *TTCN-3* while doing this project. The effort required by an experienced *TTCN-3* tester, like Bernard Stepien would probably be an order of magnitude less (i.e. measured in hours rather days).

It is also much easier to write unit tests for services in *TTCN-3* than in Junit, once the codec is set up, but it is still a lot more manual effort than simply using BPM Testing Asset.

As is explained in more detail below, it was not possible to truly unit test all the functionality of forms in *TTCN-3* (although verification of inputs when the form was launched and outputs when the form was submitted could be tested).

There is no support for model-based generation of *TTCN-3* test scripts from the business process model and scenarios, but in theory this could be done in the same manner that IBM BPM Testing Asset generates unit tests from the business process model for Selenium.

There is no support for performance reporting against business goals with *TTCN-3*. However, *TTCN-3* does automatically generate very good testing reports including charts of times taken for the execution of tests, errors occurred and graphical logging of each step in the script.

It should be noted that it was not possible to test the user interfaces of business process forms directly using *TTCN-3*. As we mentioned in the concrete layer, the test adapter for testing forms was written using the open source testing tool htmlUnit. Because of the way that IBM BPM 8.5.5 dynamic load web page contents with the background execution of JavaScript it was not possible to interact with the web page directly. Instead the Rest API provided by IBM BPM 8.5.5 was used to test the transfer of data to the form when it was presented to the user and the transfer of data from the form when the user submitted. This is sufficient for the purposes of multi-role, multi-service orchestration testing, but means it was not possible to use *TTCN-3* to unit test forms.

4.1.5 IBM Cognos BI Suite

IBM Cognos BI Suite was used to generate business performance reports. It took about *2 person days* of manual effort to develop reports for the library project based on the provided goal model. Similarly it took *2 person days* to develop reports for the *Cancer Care Process*. Cognos is not an open source tool and involves steep learning

curve as it incorporates complex queries for sophisticated reports. However, once written it supports enterprise distribution of reports (including alerts, and scheduled reports) and the reports are available in various forms (HTML, PDF, and Excel). Though the validation was not model-driven, it was performed based on the given goal model.

4.1.6 Framework Prototype

After systematically evaluating the various tools, it was time to build a prototype of our quality assurance framework for BPM and evaluate it. The following figure depicts our prototype implementation.

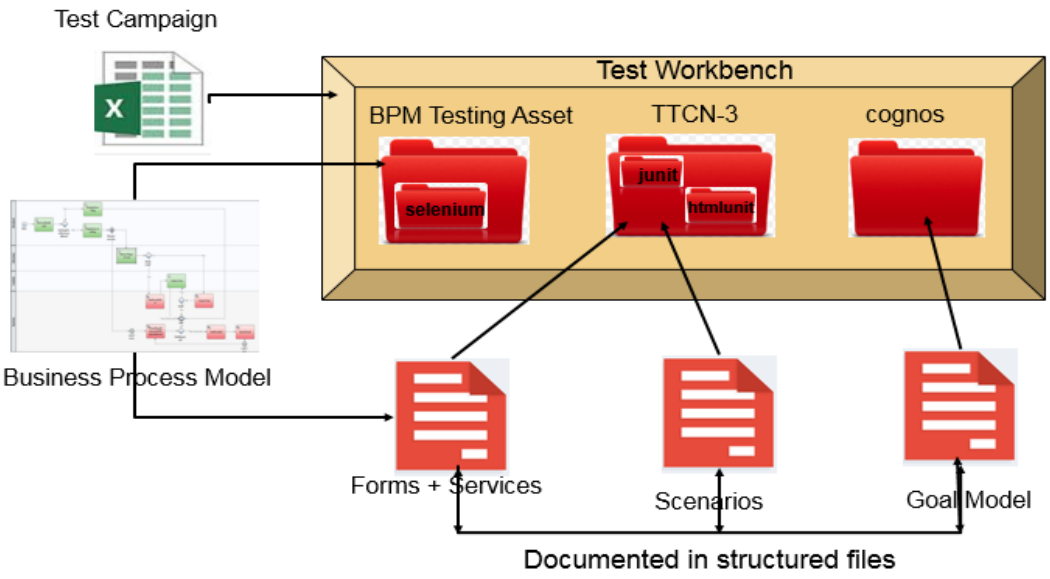


Figure 4-1: Prototype of our Framework

The business process model was implemented using the IBM BPM 8.5.5 tool and the suite of tools used in our test workbench consisted of IBM BPM Testing Asset (which embedded Selenium) and the TTCN-3 tool from Testing Tech (which embedded JUnit and Selenium) as well as IBM Cognos BI Suite 10.0 for performance reporting.

In both our case studies, the SOA was simulated by mock services, so we simply extracted a structured list of the web services used and their interfaces along with all forms from the Business Process Model that was captured in a structured document used for TTCN-3. IBM BPM Testing Asset inferred this information automatically from the Business Process model.

User Scenarios were modeled in structured documents with a listing of the exact data processed at each step. This was used to write the TTCN-3 scripts for multi-role, multi-service testing.

A Goal Model was defined for each case study using URN to model the goals, metrics and business process tasks where the data for metrics was captured. This was used to identify the performance reports needed from Cognos BI.

Finally, a systematic test plan was defined to manage, plan, and execute the test scripts generated by BPM Testing Asset, written in TTCN-3 and the performance reports written in Cognos BI. A web portal was constructed to manage and display the test campaign (including reporting of test results), bugs, and the performance reports.

We only used three tools (Model-driven IBM BPM Testing Asset, TTCN-3, and IBM Cognos), not five (as shown in Figure 3-4), when implementing the prototype of our framework. It would be nice if both TTCN-3 and Cognos were embedded into the model-driven test generation tool, like the way Selenium is embedded now.

4.2. Case Study 1: Library Borrow Book Process

4.2.1 Existing Quality Assurance Environment

The 'Library Borrow Book' process manages borrow and return processes for a Rare Books Collection at uOttawa. The process involves four participating roles: Student, Librarian, Cashier and the Library System. Before the case study started, requirements were captured in natural language documents. Both the developer and tester had issues understanding the requirements. The QA team tested the process manually as a web application. The process was remodelled and released six times but there was persistent ongoing quality issues and confusion about what the correct behaviour should be.

At that point we commenced the case study using the prototype implementation of our framework. The first step was to more systematically capture the requirements based on structured documents for SOA, User Scenarios and Goal Model.

4.2.2 Enterprise Architect

A master's student with a good development background served as an enterprise architect in order to create the services needed for the library process. As was mentioned earlier, there was no true SOA since the case studies were set up in a university lab. The list of Services and Forms used by the process was extracted automatically from the business process model by the BPM Testing Asset tool. A Services and Forms document was extracted manually from the business process model to guide the testing done by TTCN-3. The document provided a brief overview and description of the service and its operations, including input and output parameters including the WSDL -defined

interface to the service. Figures 4-2 and 4-3 are a snippet from the prepared SOA document.

Service Specification for Library Project

Two Services:

(a) Data Access Service
Description: Provides operations that can be used to read/write from/to DB.
Interface: WSDL *https://bpm2.site.uottawa.ca:9443/teamworks/webservices/RBC/EBC6230Project_RareBookCollection.tws?WSDL*
Message Format: XML; **Communication Protocol:** HTTPS
Operations: *getStudentDetails, getStatusofStudentAccount, getavailableRareBooks, setBorrowedBooktoBorrowed, setReturnedBooktoReturned, setReportedLostToLost, allowUserToBorrowAgain, add10FineForDue, add1000FineForLost, setFine0*

(b) Email Service
Description: service to send reminder emails to the students, when a fine is added to their account.
Interface: Internal to BPM tool
Message Format: XML; **Communication Protocol:** SMTP
Operations: *sendEmail*

Figure 4-2: Service Specification for Library Business Process

Service Specification for Library Project

Two Services:

(a) Data Access Service
Provides operations that can be used to read/write from/to DB.
WSDL: https://bpm2.site.uottawa.ca:9443/teamworks/webservices/RBC/EBC6230Project_RareBookCollection.tws?WSDL
Operations: *getStudentDetails, getStatusofStudentAccount, getavailableRareBooks, setBorrowedBooktoBorrowed, setReturnedBooktoReturned, setReportedLostToLost, allowUserToBorrowAgain, add10FineForDue, add1000FineForLost, setFine0*

Operation: getStudentDetails
Description: This operation can be used to obtain details of a student.
Input: userName (STRING)
Eg: student4
Output: studentDetails (Business Object: StudentDetails)
Eg: student4|Kavya|Mallur|Computer Science|AccountBlocked

Figure 4-3: Service Operation Description for Library Business Process

4.2.3 Care Process Expert

The expert or business analyst for the library process was Dr. Liam Peyton, who is the professor responsible for the course EBC6230 that used the library process. He provided the user scenarios (Figure 4-4) and the goal model (Figure 4-5).

Scenario 1: Student browses available books and then borrows a book and returns in on time, before the due date.
Scenario 2: Student browses and borrows a book and returns it after (a) 2 minutes, (b) 3 minutes and (c) 5 minutes and then pays fine for a) and b), but not c) and then borrows another book.
Scenario 3: Student browses and borrows a book and reports it as lost before the due date and then borrows another book without paying fine.
Scenario 4: Student browses available books and borrows a book. He reports it as lost after 5 minutes and then pays fine, then borrows another book.
Scenario 5: Student borrows a book and reports it as lost within 1 minute and then pays fine and again borrows another book.
Scenario 6: Student tries to borrow a book without returning the previously borrowed book.
Scenario 7: Student tries to pay partial fine to the cashier. An error message is displayed.

Figure 4-4: User Scenarios provided for Library Management

Figure 4-5 describes the sample goal model provided for the *Library Borrow Book* case study. There are three main goals of the project: increase the number of borrow transactions; enhance the availability of books: and manage costs. There are four metrics that help measure process performance for those goals: Number of books borrowed per year, Number of overdue books per year, the Number of books lost per year and the total amount of fines collected per year.

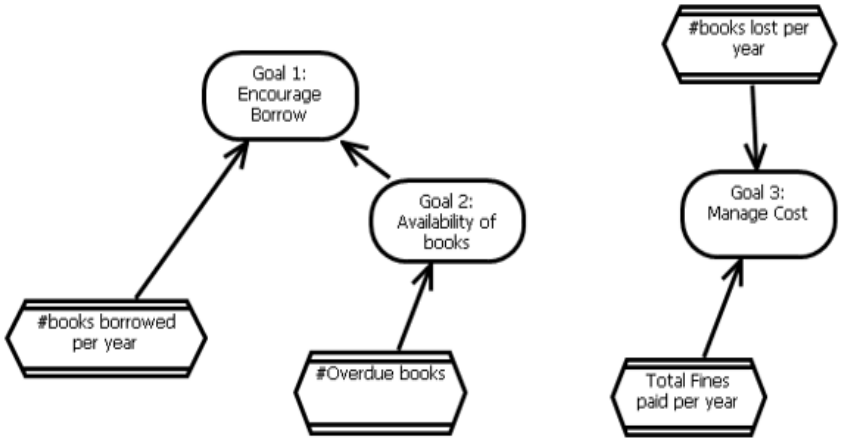


Figure 4-5: Goal Model for Library Management Business Process

4.2.4 Development Team

A single developer Mohammad Alhaj, a post-doctoral fellow with a good knowledge of IBM BPM created the business process model shown in Figure 4-6.

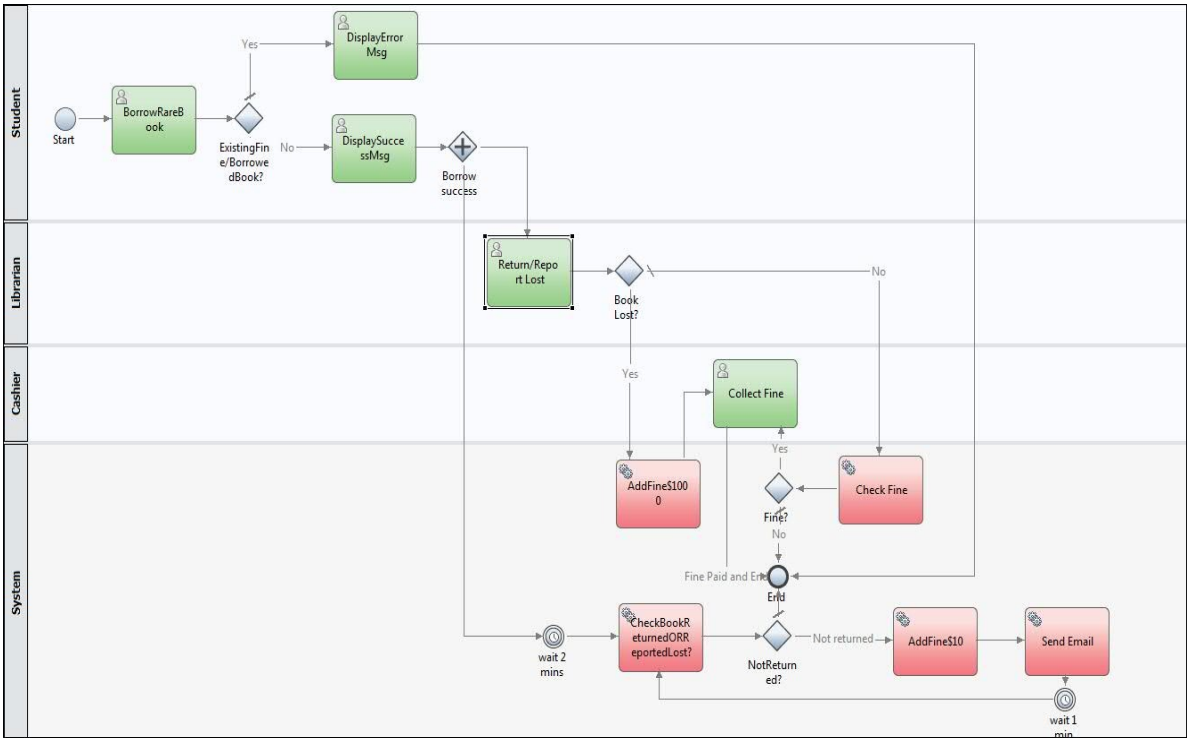


Figure 4-6: Library Borrow Book Business Process Model developed using User Scenarios and SOA Model

The process begins when a student attempts to borrow a book and ends when the book is returned or reported lost and the student owes no fine. The student is restricted to borrow only one book. i.e., the student can make another borrow transaction only after previous borrow transaction is complete. There is another intertwined process that wakes up soon after a book is borrowed by the student. This process wakes up every minute to verify if the book is overdue and also calculates a fine and ends when the fine is paid.

4.2.5 Quality Assurance Team

The QA team was the thesis author supported by a TTCN-3 consultant and researcher Bernard Stepien. Below is a detailed description of the testing done using the Test Orchestration Workbench.

IBM BPM Testing Asset

IBM's BPM Testing Asset was used to automatically generate test cases. The tool generated sets of test cases covering the services and user-interface forms involved in the process which tested boundary conditions and default conditions automatically. As mentioned in 4.1.3 this was a very successful, efficient and effective model-driven mechanism that required little effort to completely address all unit testing for the business process in a few hours.

The tool offered an additional capability to identify possible paths in a business process and generate a default test script template for each path identified. This capability, though, was very immature and difficult to use.

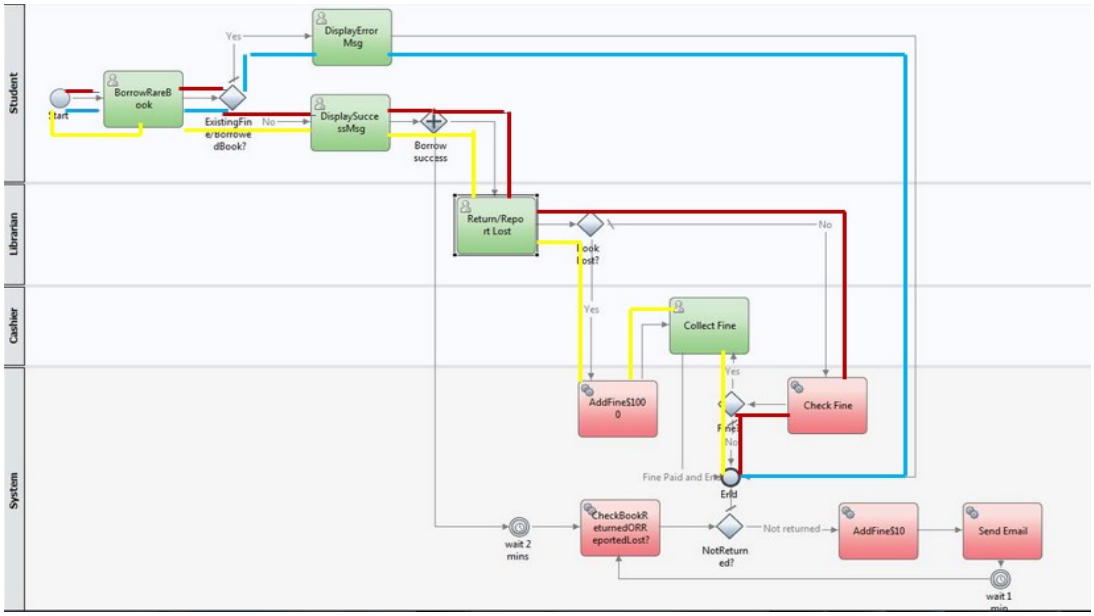


Figure 4-7: Model-Driven Test Case Generation using IBM's BPM Testing Asset tool

Figure 4-7 shows the paths identified by BPM Testing Asset for our case study in blue, red and yellow. Each path is based on taking a different decision at the decision gateways "*ExistingFine/BorrowedBook?"* and "*Book Lost?*" book. However, this was not useful because of the following problems:

- It failed to identify any of the paths launched by timers related to "*NotReturned?*"
- The test script templates needed to be manually edited to specify values input and timing constraints.
- Without values and timing constraints the correspondence between paths and the scenarios in figure 4-4 was tenuous at best.

- Even with input values and timing constraints there was no way to test if the output values were correct at each form and service orchestrated along the path.
- Even if multi-role multi-service orchestration could have been tested along one path, there was no way to coordinated and verify correct behavior when many scenarios were run in parallel.

TTCN-3

In the library process, several activities involving different services were executed simultaneously to ensure right calls are made to appropriate services. *TTCN-3* was used to verify multi-role, multi-service orchestration testing with verification of results at critical points throughout the architecture. For example, testing is performed to ensure BPM system invokes database services correctly with correct results whenever, processed data needs to be written to database. Similarly, the use of the email service is verified when an email is sent to a student when a book is overdue. A detailed example of a TTCN-3 test case is given in Appendix A. TTNCN-3 has powerful features that can display test cases graphically; in addition to good reporting features for summarizing test results.The entire execution of the test is graphically displayed in figure 4-8.

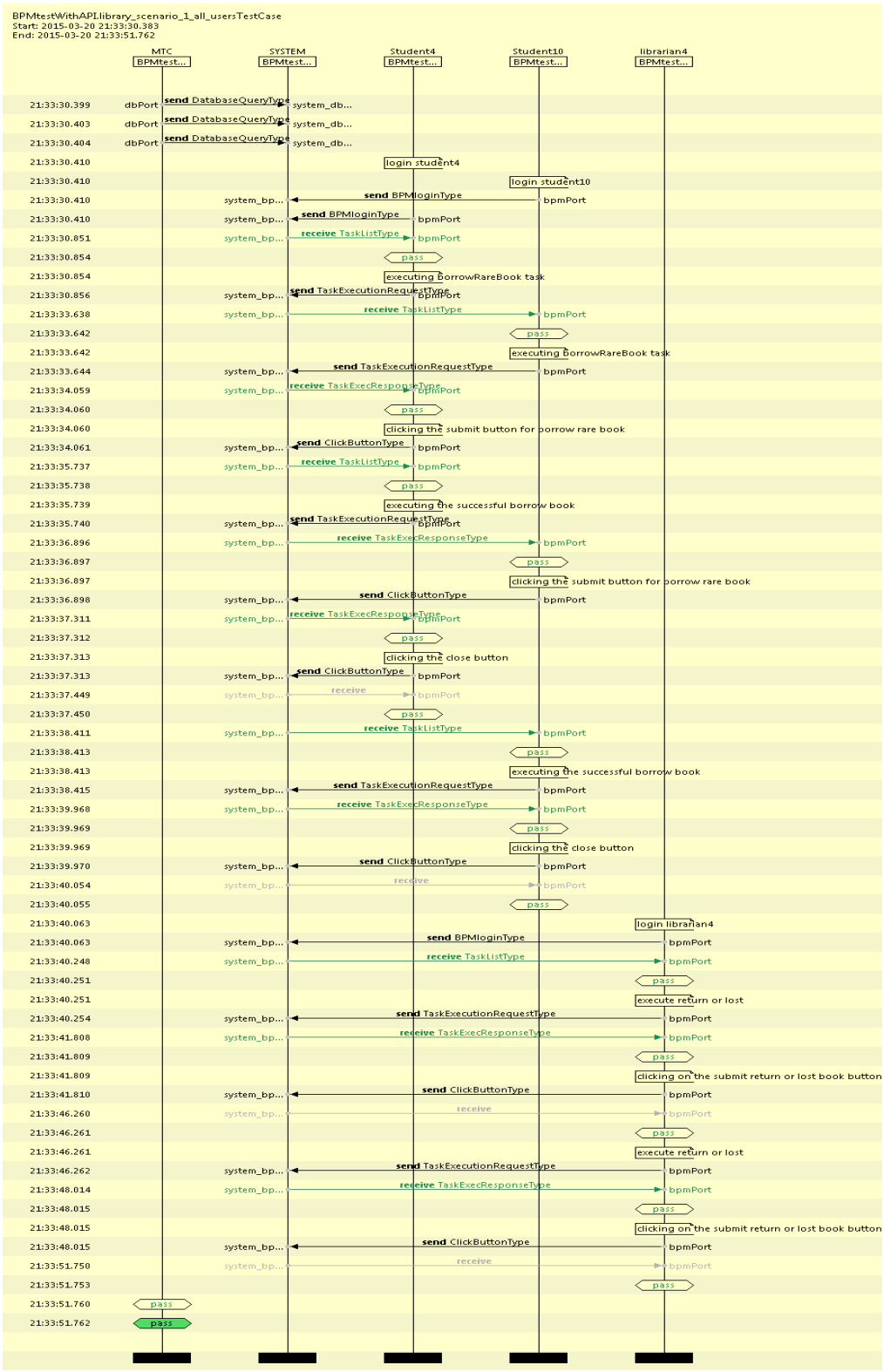


Figure 4-8: TTCN-3 test graphical log

IBM Cognos

The business process performance reports were generated using IBM's Cognos. The reports were generated by the development team and handed off to the QA team. And with the reports, the QA team validated the performance against the goals. Sample reports generated using IBM Cognos for Library Project are as shown in Figure 4-13, Figure 4-14, and Figure 4-15.

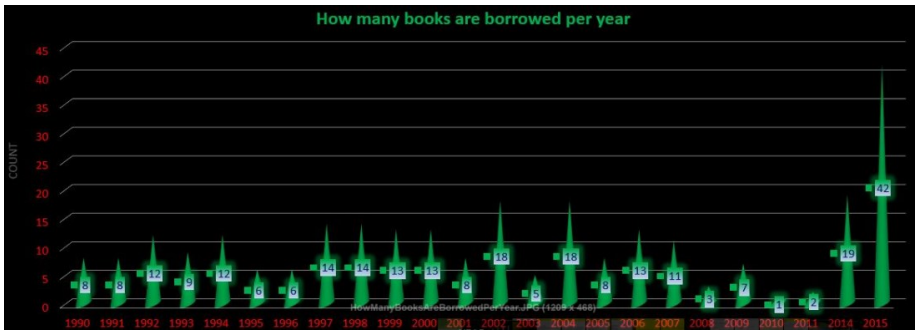


Figure 4-13: Report displaying number of books borrowed per year

The report shown in Figure 4-13 is used to verify if the defined goal G1: Encourage Borrow Book is met successfully. Similarly two reports (Figure 4-14 and Figure 4-15) are used to measure if the goal G3: Manage cost is met.

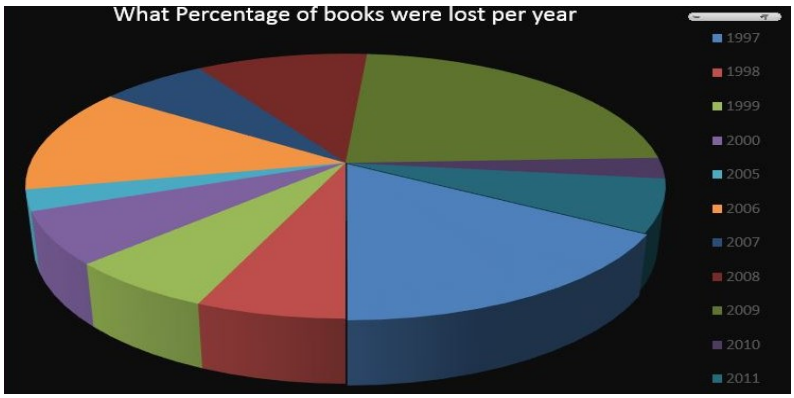


Figure 4-14: Report displaying the percentage of books lost

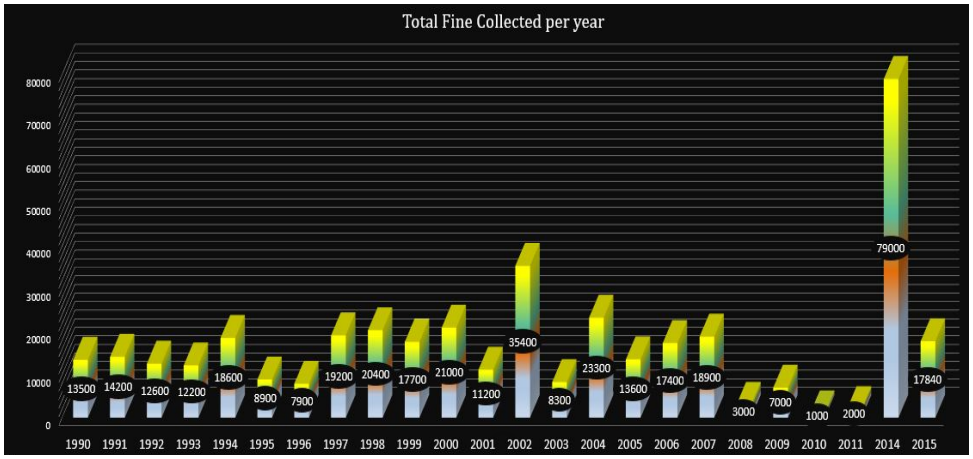


Figure 4-15: Report displaying the total amount of fine collected per year

4.3. Case Study 2: Cancer Care Assessment Process

4.3.1 Existing Quality Assurance Environment

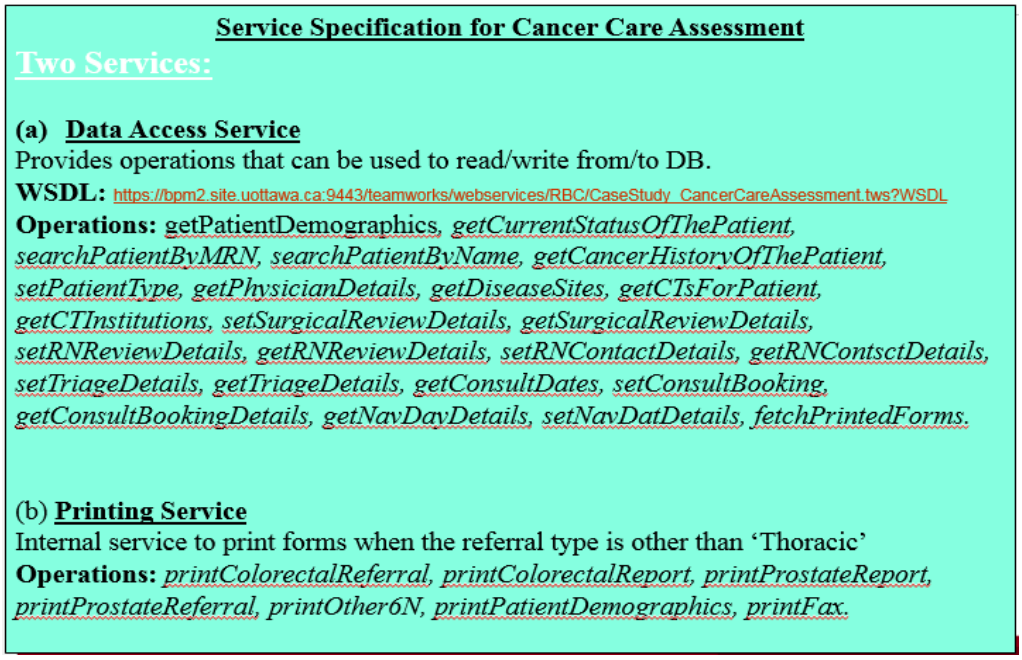
The 'Cancer Care Assessment' process manages wait and service times for cancer patients and was built at a local hospital to provide better quality of treatment to the cancer patients. The process involves nine participating roles: Clerk, Printing Clerk, Thoracic Nurse, Colorectal Nurse, Prostate Nurse, Assigned Nurse, and Physician. Before the case study started, requirements were captured in natural language documents. Both the development and QA teams had issues understanding the requirements. The QA team tested the process manually as a web application. There were persistent ongoing quality issues and confusion about what the correct behaviour should be.

At that point we commenced the case study using the prototype implementation of our framework and a version of the process that was set up in our test environment at the

university. The first step was to more systematically capture the requirements based on structured documents for SOA, User Scenarios and Goal Model.

4.3.2 Enterprise Architect

A master's student created the "mock" services needed for the process (since we could not duplicate the exact SOA of the hospital). The list of Services and Forms used by the process were interpreted from the business process model by the BPM Testing Asset tool. A Services and Forms document was extracted manually from the business process model to guide the testing done by TTCN-3 The document provided a brief overview and description of the service and its operations, including input and output parameters including the WSDL -defined interface to the service. Figures 4-16 and 4-17 are a snippet from the prepared SOA document (clicking on an operation, popped up a description of its parameters.



Service Specification for Cancer Care Assessment

Two Services:

(a) Data Access Service

Provides operations that can be used to read/write from/to DB.

WSDL: https://bpm2.site.uottawa.ca:9443/teamworks/webservices/RBC/CaseStudy_CancerCareAssessment.tws?WSDL

Operations: getPatientDemographics, *getCurrentStatusOfThePatient*, *searchPatientByMRN*, *searchPatientByName*, *getCancerHistoryOfThePatient*, *setPatientType*, *getPhysicianDetails*, *getDiseaseSites*, *getCTsForPatient*, *getCTInstitutions*, *setSurgicalReviewDetails*, *getSurgicalReviewDetails*, *setRNReviewDetails*, *getRNReviewDetails*, *setRNContactDetails*, *getRNContsctDetails*, *setTriageDetails*, *getTriageDetails*, *getConsultDates*, *setConsultBooking*, *getConsultBookingDetails*, *getNavDayDetails*, *setNavDatDetails*, *fetchPrintedForms*.

(b) **Printing Service**

Internal service to print forms when the referral type is other than 'Thoracic'

Operations: *printColorectalReferral*, *printColorectalReport*, *printProstateReport*, *printProstateReferral*, *printOther6N*, *printPatientDemographics*, *printFax*.

Figure 4-16: Service specification for Cancer Care Assessment project

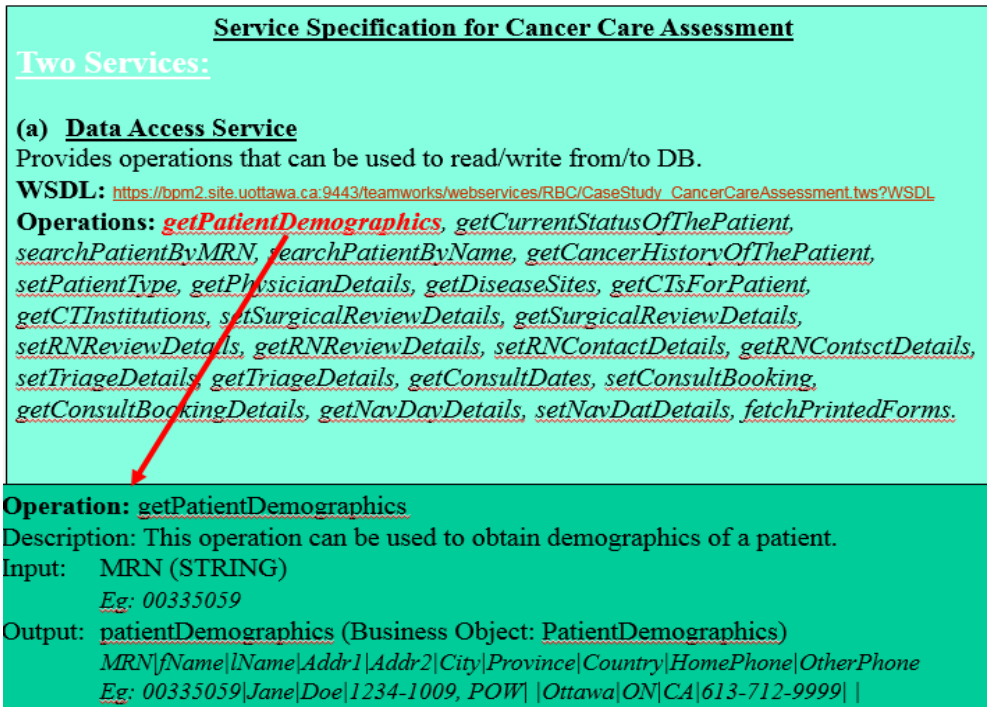


Figure 4-17: Service operation description for getPatientDemographics used in Cancer Care Assessment

4.3.3 Care Process Expert

The care process expert for Cancer Care Assessment was the business analysts at the local hospital. The thesis author was on the development team at the local hospital. When the case study was started, she interacted with the business analysts to define the user scenarios (Figure 4-18) and the goal model (Figure 4-19).

Scenario 1: The CAC clerk faxes a referral and enters the MRN. The MRN is not found in the system, so she sends the referral to NPR clerk to have the MRN created.
Scenario 2: The CAC clerk faxes a referral and indicates that the fax is non-medical.
Scenario 3: The CAC clerk faxes a referral and indicates that the fax is a non-CAC patient report.
Scenario 4: The CAC clerk faxes and submits a colorectal referral. The CAC nurse then completes the review and prints the fax along with the patient's demographics.
Scenario 5: The CAC clerk faxes and submits a prostate referral. The CAC nurse then completes the review and prints the fax along with the patient's demographics.
Scenario 6: The CAC clerk faxes and submits a referral of type "other". After the CAC nurse completes the review, it is sent to the printing work list where the clerk can print the fax along with the cover page.
Scenario 7: The CAC clerk faxes and submits a thoracic referral for the CAC nurse to review. If CT results are being awaited, the instance goes to the Awaiting CT work list and stays there until the results become available. The instance then goes back to thoracic review.
Scenario 8: The CAC clerk faxes and submits a thoracic referral for the CAC nurse to review. If CT results are being awaited, the instance goes to the Awaiting CT work list and stays there until the clerk does an override. The instance then goes back to thoracic review.
Scenario 9: The CAC clerk faxes and submits a thoracic referral for the CAC nurse to review. From there, the instance then goes to Physician Review and then RN Contact. If the nurse is unable to contact the patient after 3 attempts, she can choose to try again or send the referral for clerical printing.
Scenario 10: The CAC clerk faxes and submits a thoracic referral for the CAC nurse to review. There, the Physician Review is marked as not needed, and the referral is sent to RN Contact.
Scenario 11: The CAC clerk faxes and submits a thoracic referral for the CAC nurse to review. There, the Physician Review is marked as not needed, and the referral is sent to clinical triage.
Scenario 12: The CAC clerk faxes and submits a thoracic referral for the CAC nurse to review. There, the Physician Review is marked as not needed, and the referral is sent to Consult Booking. There, the patient is contacted and an appointment booked.
Scenario 13: The CAC clerk faxes and submits a thoracic referral for the CAC nurse to review. There, the Physician Review is marked as not needed, and the referral is sent to Consult Booking. There, the booking clerk is unable to reach the patient and the referral is sent for clerical printing.
Scenario 14: The CAC clerk faxes and submits a thoracic referral for the CAC nurse to review. From there, the instance then goes to Physician Review and then RN Contact. If the patient is eligible for a navigation day, the nurse books a navigation slot.
Scenario 15: The CAC clerk faxes and submits a thoracic referral for the CAC nurse to review. From there, the instance then goes to Physician Review and then RN Contact. If the results of the tests ordered at Physician Review have not arrived yet, the nurse sends the instance to Pending Results.
Scenario 16: The CAC clerk faxes and submits a thoracic referral for the CAC nurse to review. From there, the instance then goes to Physician Review and then RN Contact. The nurse sends the instance to clinical triage where the patient is diagnosed. Once the triage is complete, an appointment is booked.
Scenario 17: The CAC clerk faxes and submits a thoracic referral for the CAC nurse to review. From there, the instance then goes to Physician Review and then RN Contact. The nurse sends the instance to clinical triage where the patient is diagnosed and sent to both CAC and NPR appointment bookings.

Figure 4-18: User Scenarios for Cancer Care Assessment

Figure 4-19 illustrates the sample goal model provided for the Cancer Care Assessment project. There is one main goal to provide better care for cancer patients. There are two sub-goals: Reduce wait time and Reduce service time that together contributes to the main goal. There are seven metrics that help measure the process

performance for those goals and it includes: Time from Referral End to Nurse Review Start; Time from RN Review End to Physician Review Start; Time from Referral End to Nurse Review Start; Time from RN Review Start to RN Review End; Time from Physician Review Start to End; Time from RN Contact Start to End; Time from Triage Patient Start to End.

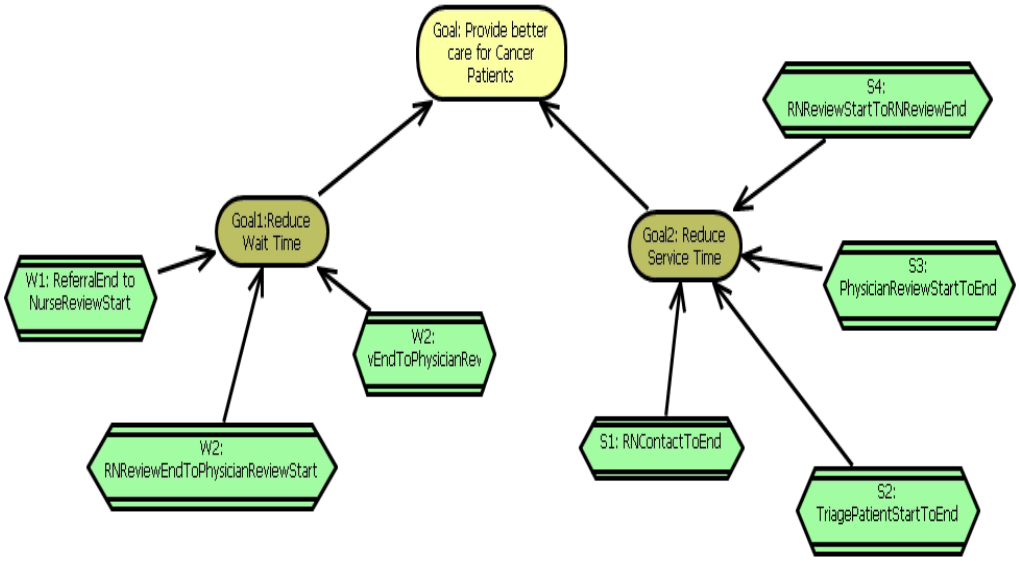


Figure 4-19: Sample goal model for Cancer Care Assessment project

4.3.4 Development Team

The development team consisted of a single developer Mohammad Alhaj, a post-doctoral fellow with a good knowledge of IBM BPM who was a member of the development team at the local hospital as well. Figure 4-20 illustrates the final model developed using the provided user scenarios.

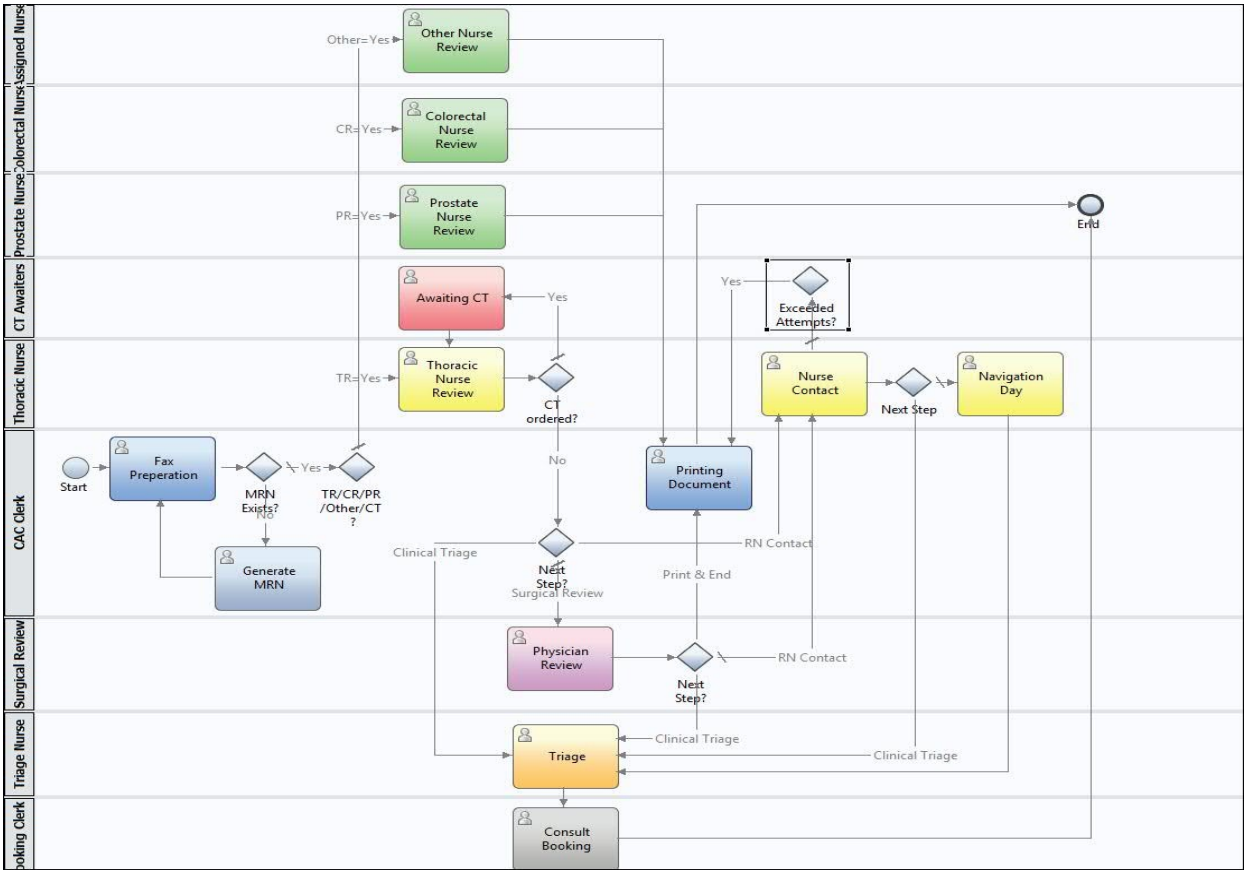


Figure 4-20: Cancer Care Assessment Business Process Model

In reality, the process modelled at the local hospital starts on receiving a fax. For the simplicity of the process, we have modelled it in a way that the CAC Clerk starts the process assuming the fax is received and looks for the patient in the data base either by providing the patient's name or by providing the Medical Record Number (MRN). As seen here, there are several process flows. For testing the process, we have considered the happy path (longest route) where in the process ends by booking consult appointment for the patient. We have also considered the shortest path, where in the process ends after printing a document for identifying the referral as Prostate.

4.3.5 Quality Assurance Team

The QA team consisted of the author of thesis, supported by a TTCN-3 consultant and researcher Bernard Stepien.

IBM BPM Testing Asset

IBM's BPM Testing Asset was used to automatically generate test cases, based on the business process model. The tool generated test cases and covered all the services, and user-interface forms involved in the process. The tool also generated tests covering some of the user scenarios, but was not really useful as it ignored timing and input values. In addition, there was no possibility to run these scenarios in parallel nor did the tool offer a capability to check the values along the test execution. Figure 4-21 illustrates the development of test cases based on possible work flow paths.

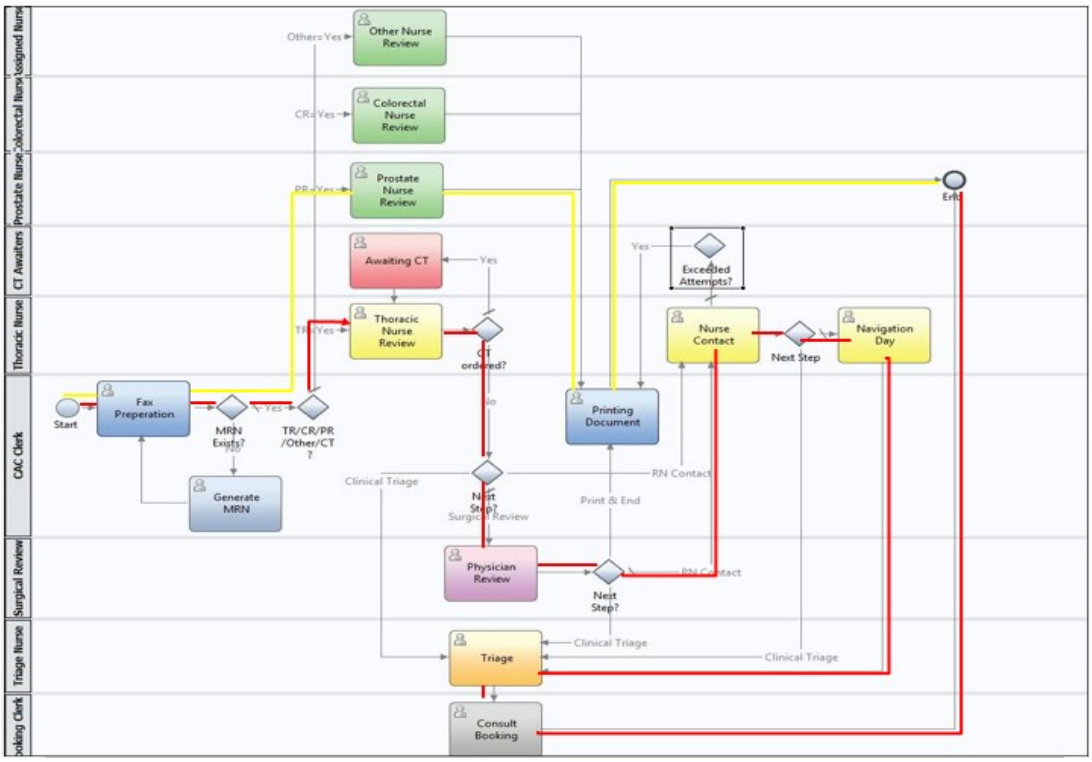


Figure 4-21: Cancer Care Assessment Tests generated using IBM BPM Testing Asset

As seen in Figure 4-21, the yellow line maps to user scenario 5, and the red line maps to scenario 16. Using this model-driven tool, the tester successfully achieved complete coverage of unit testing of forms and services.

TTCN-3

In the Cancer Care Assessment process, several activities performed by different roles involving different services were executed simultaneously to ensure right calls are made to appropriate services. *TTCN-3* was used to verify service orchestration and collaboration of multiple user roles in parallel. For example, testing is performed to ensure BPM system invokes database services whenever processed data needs to be written to database and data needs to be fetched from database. Similarly, the system invokes print service when patient demographics form needs to be printed. The following figure shows the creation of PTCs in TTCN-3 abstract scripting language. The test involves four users: 2 Thoracic RNs and 2 Review Physicians, who are working in parallel. Code snippet of the test is shown in Figure 4-22.

```
testcase myTestCase() runs on MTCComponentType system mySystemType {
   //Creation of component instances
 var clientComponentType RN := clientComponentType.create("ThoracicRN");
 var bpmServerComponentType BPM := bpmServerComponentType.create("BPM");
 var clientComponentType PHY := clientComponentType.create("Physician");

   //Connecting the components with each other
   connect (RN:ports, BPM:portclient);
   connect (BPM:portclient, PHY:ports);
   //Starting the components
   BPM.start(behaviorBPM(RN,PHY));
   PHY.start(behaviorPHY());
   RN.start(behaviorRN());.

   //wait until all the parallel test components are executed
   all component.done;
   setverdict(pass);
}
```

Figure 4-22: TTCN-3 code snippet for the test involving 2 physicians and 2 nurses

The behavior function of the physician (PTC instance PHY) specifies that the physician chooses to respond to the second message before responding to the first message. This is indicated clearly in the Figure 4-23.

```
//Behavior of Physician
        function behaviorPHY() runs on clientComponentType
        {
            ports.receive(messageRN1);
            ports.receive(messageRN2);

            ports.send(messagePHY2);
            ports.send(messagePHY1);
            setverdict(pass);
        }
```

Figure 4-23: Behaviour of the physician

The execution of the above test case is shown in Figure 4-24.

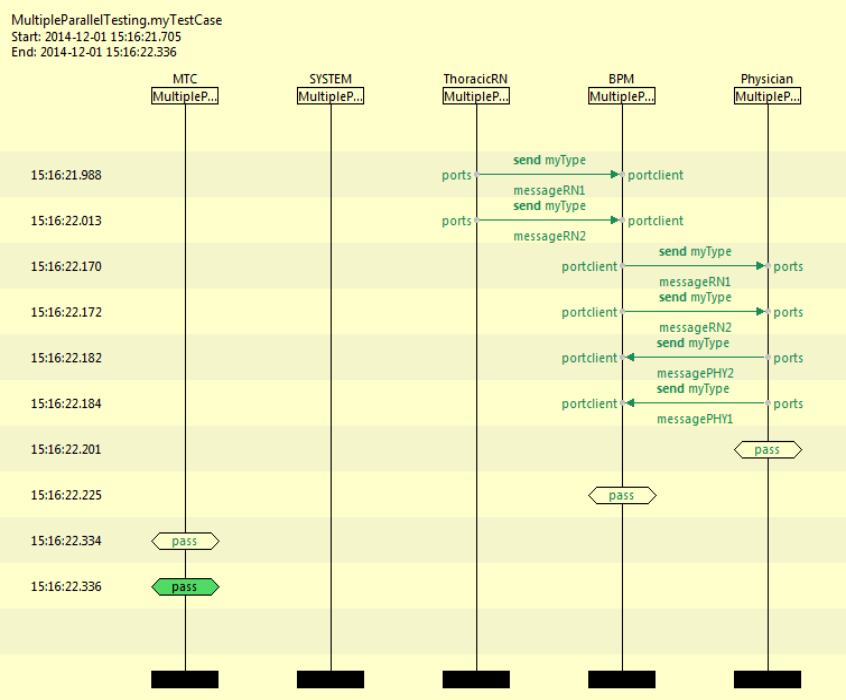


Figure 4-24: TTCN-3 test execution log for Cancer Care Assessment project

An important difference with TTCN-3 testing is the use of a monitor. Thus the traditional black box approach is inappropriate. Here, we need a process that collects all the messages occurring in the system and match them to a predefined test oracle.

IBM Cognos

The tester (author of this thesis) interacted with the business analysts at the local hospital in order to have an explicit goal model defined for the process. The existing project at the hospital already had 3 reports defined, but through the interaction of building the goal model, the business analysts realized that there was an additional report that was missing, which would be needed to validate that the process was meeting goals. This clearly showed the importance of an explicit goal model to define requirements, as well as the need for the QA team to take responsibility for ensuring that the right set of performance reports exist in order to validate goals.

IBM Cognos was used to generate process performance reports. The development team generated the reports and handed off to the QA team. The tester validated that the each process performance report measured progress accurately for the goals it was intended to measure. This is to ensure the reports would be useful once transferred to production. A sample report generated for the Cancer Care Assessment process is depicted in Figure 4-25. This report is used to verify if the goal: Reduce wait times is successfully met.

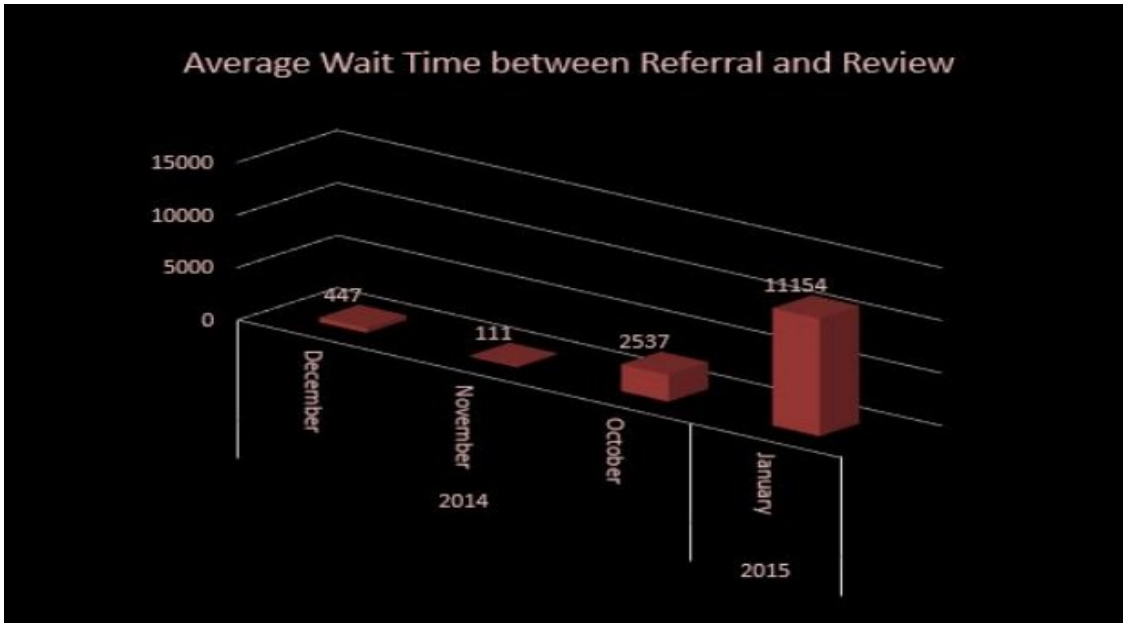


Figure 4-25: Report showing the average wait time between the Referral and RN Review

4.4. Summary of Experience and Results

It is observed that with the QA framework prototype, it was possible to achieve all the necessary testing in both case studies. The BPM Testing Asset tool was the simplest and easiest to use because of the way it directly generated test scripts from the Business Process Model. TTCN-3 was complex to set up but very powerful at providing complex multi-role, multi-service orchestration testing. Writing of tests would have been greatly improved if it could have been model-driven but would have required modeling of user scenarios as well as the business process model. Similarly, Cognos is complex to set up and use, but could also have been greatly improved if it could have been model-driven but would have required modeling of goals and metrics as well as the business process model.

4.5. Chapter Summary

This chapter has provided a concrete implementation prototype of our QA framework. The framework has been designed, applied and evaluated against two case studies, including the *Cancer Care Assessment* project that was originally built at the local hospital. We have successfully demonstrated the use of testing tools for addressing different aspects of business process testing including *TTCN-3* for sequential and interleaved testing involving racing condition during multiple roles collaboration and parallelism in multiple service orchestration. The next chapter will present the evaluation of our framework based on the criteria defined in section 3.4. The chapter also compares our framework against the related works described in section 2.3.

Chapter 5. Evaluation

In this chapter, we evaluate our proposed framework using the evaluation criteria identified in Chapter 3.3. In section 5.1, we evaluate our proposed framework in comparison to individual tools using our case study. In section 5.2, we compare our proposed framework to the related works identified in chapter 2.3. And finally, in section 5.3 we evaluate the TTCN-3 testing tool and standard and the challenges faced and benefits obtained when using it for quality assurance of BPM.

5.1. Framework Evaluation

In this section we evaluate our framework in comparison to commonly used testing tools based on our case studies. There are four tables, one for each category of criteria identified in chapter 3.3. Green indicates feature is supported, yellow indicates partial support while red indicates it is not supported.

5.1.1 Methodology

In Table 5-1, we evaluate coverage according to our methodology. SOA, Scenarios and Goal Model drive quality assurance developed in Work Bench, reported in Portal and Production. BPM Testing Asset automatically tests at the service layer (SOA) but Junit and TTCN3 can be manually configured to do so as well. Selenium is awkward for service testing. Only Cognos leverages the goal model to identify the business goals and metrics to be measured and even then only manually.

TTCN-3 can drive multi-role, multi-service orchestration tests in parallel from the user scenarios as can our framework. BPM testing Asset also leverages scenarios but

only for unit testing of forms (there is no verification of multi-service orchestration and there is no testing of scenarios in parallel). Only TTCN-3 with its concrete layer code offers the possibility of integrating tools in a systematic way in a workbench, although BPM Testing Asset has a hard coded integration with Selenium.

Cognos provides a portal for report generation and management in both test and production environments.

TABLE 5-1: FRAMEWORK EVALUATION- METHODOLOGY

Feature	Junit	Selenium	BPM Testing Asset	TTCN-3	Cognos	Proposed QA Framework
SOA	Yes. Ad-hoc.	No.	Yes.	Yes. But manually inferred from Business Process Model.	No.	Yes. Manually for TTCN-3.
Goal Model	No.	No.	No.	No.	Yes. Inferred manually from Goal Model	Yes. Inferred manually from Goal Model
Scenarios	No	No	No.	Yes. Scenarios drive multi-role multi-service orchestration.	No.	Yes. Scenarios drive multi-role multi-service orchestration
Work Bench	No.	No.	Limited. Selenium	Yes Manually incorporate in Concrete architecture	No.	Yes.
Portal	No.	No.	No.	No.	Yes. Reports only.	Yes.
Production	No.	No.	No.	No.	Yes	Yes.

5.1.2 Business Process Features

In Table 5-2, we evaluate coverage according to Business Process Features. To address BPM, the framework needs to be model-driven and cover forms, services, multi-role, multi-service orchestration and business goal validation.

Table 5-2: FW Evaluation – Business Process Features

Feature	JUnit	Selenium	BPM Testing Asset	TTCN-3	Cognos	Proposed QA Framework
Model Driven	No.	No.	Yes. Completely model driven.	Yes partially. BPM specific concrete layer to support abstract layer.	No.	Yes. Partially
Forms	Yes but not practical	Yes.	Yes.	Yes, but only input/output.	No.	Yes.
Services	Yes	No	Yes, Completely.	Yes.	No.	Yes.
Multi role, Multi service orchestration	No.	No.	No	Yes.	No.	Yes.
Goal validation	No.	No.	No.	No.	Yes.	Yes.

IBM BPM Testing Asset drives complete unit testing of forms and services based on the business process model whereas *TTCN-3* can model the BPM architecture but tests are not directly driven by the business process model. Similarly, Junit, Selenium and BPM Testing Asset can only test forms directly, but *TTCN-3* had to be configured to test simply the inputs to and outputs from forms via the BPM Rest API. Only *TTCN-3* could handle multi-role, multi-service orchestration testing and only Cognos could handle goal validation. As a result, only our framework had complete coverage of all business process features.

5.1.3 Test Case Definitions

In Table 5-3, we evaluate criteria related to Test Case Definitions. In particular we evaluate effort, coverage, reusability and level of coding required.

With the experiences gained from our case studies and experiments, it is well observed that both Junit and Selenium involved lot of effort in manually coding the tests

with poor coverage and reusability, in part because of the low level of implementation coding (Junit) but really because they were solely focused on unit testing (no concept of business process). IBM BPM Test Asset was the best in this category because it was completely driven by the business process model (but lacked coverage because of no support for multi-role multi-service orchestration and goal validation). *TTCN-3* offered the best coverage but was not model driven. If model-driven support could be developed to drive *TTCN*-3 and Cognos, then our proposed framework would address all criteria.

TABLE 5-3: FW EVALUATION – TEST CASE DEFINITIONS

Feature	Junit	Selenium	BPM Testing Asset	TTCN-3	Cognos	Proposed QA Framework
Test Script Effort	High Manual coding of scripts.	High Manual coding of scripts.	Automatic for unit tests	Medium (after codec created)	Medium for reports (after initial setup)	Automatic / Medium
Test Coverage	Low Services (partial)	Low forms (partial)	Medium (Unit tests).	High	Low Performance Reports	Complete, but not fully automated
Level of coding	Low Level	High Level	Very High (model generated)	Medium.	Medium.	Medium to Very High
Test Re-usability	No.	No	No need to reuse – just regenerate	Yes. Codec can be reused to test another application.	No.	Partial

5.1.4 Tool Support

In Table 5-4, we evaluate criteria related to tool support. To address BPM, the framework needs to provide a test environment with built-in support for BPM in which tests can be executed efficiently, with good reporting and logging. Ideally cost would be

low, with open source preferable, and the need for specialized skills (other than BPM) minimized.

TABLE 5-4: FW EVALUATION – TOOLS SUPPORT

Feature	JUnit	Selenium	BPM Testing Asset	TTCN-3	Cognos	Proposed QA Framework
Test Execution	Simple but manually organized.	Simple but manually organized	Simple and automatic.	Simple, but manual. Powerful language can specify many tests running in parallel with expected results.	Simple, fully ability to schedule and define alerts.	Simple Systematic.
Test Environment	Low	Low	Automatic	High first time.	High initially	Very High.
Test Report & Logging	Partial. Tracing may need classical debugger	Partial. Tracing may need classical debugger	Yes. Built in reports and logging Simple & easy to understand	Yes, Built in reports and logging.	Yes	Yes. Built in reports and logging.
Ownership	Eclipse public license.	Eclipse public license.	IBM. proprietary	ETSI Standard. Multiple Vendors proprietary	IBM. proprietary	Mixed.
Cost	Free	Free	Low if purchased IBM BPM 8.5.5	High	High.	High
Skills	Developer (Java)	Tester (UI)	BPM Engineer	TTCN-3 developer	Reports developer.	BPM+TTCN3+Reports.

Execution of tests is simple as with Selenium and Junit, but need to be organized manually. However, the effort to set up the environment for them is minimal and the tools are freely available and open source keeping costs low. On the other hand, IBM BPM Testing Asset is very easy to use for the execution of tests and is automatic. Its costs are similarly low if you have already purchased IBM BPM and the only skill needed is to understand BPM. And it provides good test reports and logging. However it is specific to the IBM tool.

TTCN-3 currently requires significant setup, but a built-in codec for BPM could easily be developed which would greatly reduce set up time. Unfortunately, our framework probably provides the worst tool support, since it is combining several tools in order to provide full coverage of the features needed for BPM testing. Future work, should address providing an open source, model-driven framework similar to BPM Testing Asset that integrates the required tools seamlessly.

5.2. Related Works Evaluation

In this section we evaluate our framework in comparison to the related works based on the criteria identified in chapter 3.3. There are four tables, one for each category of criteria.

5.2.1 Methodology

TABLE 5-5: COMPARISON WITH RELATED WORKS- METHODOLOGY

Feature	TASSA	Testing with reset nets	Testing with SARI	Proposed QA Framework
SOA	Yes.	Yes.	Yes.	Yes. Via business process model.
Goal Model	No.	No.	Yes.	Yes.
Scenarios	Yes.	Yes.	Yes.	Yes. Scenarios drive multi-role, multi-service orchestration
Work Bench	Yes.	Yes.	Yes.	Yes.
Portal	No.	No.	No.	Yes.
Production	No.	No.	Yes.	Yes.

TASSA framework is based on isolation of web services from the business process. It leverages SOA to learn about the underlying services and generates test harness to replace these services. It uses scenarios to understand the work flows which in turn help in generation of data variables, variable values, faults by leveraging tools supported by the workbench. The work does no process validation against business goals.

The work on testing with Reset Nets leverage SOA to perform formal verification of orchestration of services. It uses scenarios to drive end-to-end testing and does not cover validation of process. It offers no support to testing in production.

The work on testing with SARI uses SOA to understand the services needed for unit testing. It uses scenarios to understand the process flows and leverages goal model to achieve validation of process in production.

Our framework leverages SOA via business process model and uses scenarios to drive business process testing. It offers a workbench supporting tools for complete business process testing, including the validation of process against goals derived by leveraging goal model.

5.2.2 Business Process Features

As is shown in table 5-6, TASSA drives model-based business process testing. Business process model is used to generate tests and injection of faults to the process. The work involving Reset Nets is partially model-driven. The work targets formal verification of model but no model-based test scripts generation. The work on SARI is partially model-driven and leverages simulation models to generate inputs for the scenarios. TASSA addresses verification of forms and does not cover testing of services individually or coupled with collaboration of user roles, and validation of business process.

None of the related works perform multiple roles, multiple service orchestration testing in parallel. Our framework addresses this gap by leveraging the telecom standard

TTCN-3. SARI and Reset Nets address unit testing of services and do not support verification of forms.

TABLE 5-6: COMPARISON WITH RELATED WORKS- BUSINESS PROCESS FEATURES

Feature	TASSA	Testing with reset nets	Testing with SARI	Proposed QA Framework
Model-Driven	Yes. Generation of test cases. Model is used for fault injection.	Yes. Partially. Formal verification of model but not generated	Yes. Partially. Simulation Models are used to generate inputs for given scenarios.	Yes. Partially.
Forms	Yes.	No.	No.	Yes.
Services	No.	Yes.	Yes.	Yes.
Multi-Role, Multi-Service orchestration	No. Multi-Role only.	No. Multi-Service only.	No.	Yes.
Goal Validation	No.	No.	Yes.	Yes.

5.2.3 Test Case Definitions

As can be seen in Table 5-7, TASSA involves minimum effort and uses business model to generate tests. On the other hand, Reset Nets-based and SARI-based test approaches do not automatically generate tests. Our framework successfully handles automatic generation of tests using BPM Testing Asset (and to a lesser extent test generation from *TTCN-3* abstract specifications), but more automation driven by User Scenarios and Goal Model would be beneficial.

TASSA, Reset Nets, and SARI based do not address complete testing of business processes. All the three works fail to handle multi roles, multi services orchestration testing in parallel. The level of coding for TASSA and Sari is high-level in terms of the business process whereas BPM Testing Asset makes much of our coding very high level (simply generate from business process model). Testing with reset nets requires medium

level of coding, similar to how our use of *TTCN-3* while powerful also requires medium level of coding

TABLE 5-7: COMPARISON WITH RELATED WORKS- TEST CASE DEFINITION

Feature	TASSA	Testing with reset nets	Testing with SARI	Proposed QA Framework
Test Script Effort	Low. Model-generated.	High.	Medium-High. Manual but test data injected.	Automatic - Medium
Test Coverage	Medium -No Multi-Role, Multi-service. No Performance reports	Medium -No multi role Multi-service. No Performance reports.	Medium -No multi-role, multi-service.	Complete Not fully automated.
Level of coding	High.	Medium.	High.	Medium to Very High.
Test Reusability	Partial.	No.	Partial.	Partial.

5.2.4 Tool Support

As can be seen in table 5-8, tool support is lacking in all related works as well as our framework. This is, of course, due to the academic nature of the compared frameworks, but it does highlight a lack of focus on tools in current research.

The test execution is automated in all the three related works. Our framework provides a simple yet systematically organised execution of tests. Reset Nets-based work provides no support for reporting or logging, while it is open source and is available for no cost.

For a person to test business process using TASSA framework, they should have knowledge about BPM and the tools supported by the framework. Knowledge and experience using Reset nets and Yawl editor is a must for one to use Reset Nets-based approach for business process testing. And work involving SARI demands one to possess C# development skills to use the proposed approach for business process testing.

TABLE 5-8: COMPARISON WITH RELATED WORKS- TOOLS SUPPORT

Feature	TASSA	Reset Nets	SARI	Proposed QA Framework
Test Execution	Automated.	Automated.	Automated.	Simple and systematically organized.
Test Environment	Medium.	High.	High.	Very High.
Test report and logging	Yes.	No.	Yes.	Yes. Built in reports and logging.
Ownership	Proprietary	Open	Proprietary	Mixed.
Cost	High	Free	High	High
Skills	BPM+ TASSA tools. High.	Reset nets+ Yawl editor. Very High.	C# developer. Very High.	BPM+TTCN3+Reports. Very High.

5.3. TTCN-3 Evaluation

Table 5-9, below, is a comparison of *TTCN-3* usage in telecom and BPM testing. Testing BPM with *TTCN-3* is as straight forward and as powerful as testing telecom where *TTCN-3* is well established and is successfully used throughout the industry, once the necessary code translation and concrete layer architecture is established. Moreover, the standardization of interfaces to forms and services means that construction of a

generic code and concrete layer can be standardized making *TTCN-3* even more attractive for BPM.

TABLE 5-9: COMPARISON OF TTCN-3

TTCN-3 Feature	Telecom	BPM	Comments
Key Language Features	Alt Sequences, Templates, Matching, Objects	Alt Sequences, Templates, Matching, Objects	Use of TTCN-3 language is the same but BPM specific objects are abstracted at the abstract layer.
Abstract Layer Objects	Network Functions	Forms, Services, Roles	Domain level testing enabled by implementation of codec translation using concrete layer components.
Codec Translation	Individual Network Elements. Formatted messages.	Forms Input & Output, Roles Behavior, Service Input & Output	Difficult to handle forms, but standardization of services/forms in BPM in terms of I/O means generic BPM codec possible (simple configuration for specific forms, services)
Concrete Layer Components	Different for each protocol	Junit, REST API for services, HTMLUnit for forms	A standardized open concrete layer can be defined for BPM
Protocols	Many: open and proprietary	HTTP (SOAP or REST)	Protocol communication is standardized on SOAP or REST for BPM
System Under Test	Large number of network devices	Small number of somewhat standardized BPM engines	There are variations between each BPM engine but the basic architecture is the same as are the protocols used. Incentive would be for BPM vendors to provide high-end model-driven tool support.

The key language features used in business process testing remained similar to those used in telecom testing. However, the abstract layer objects in BPM testing involved user-interface forms, services and user roles, while telecom testing dealt with network functions at the abstract layer. Codec translation for BPM testing involved only form inputs/outputs, behavior functions of user roles, and the service inputs/outputs. We were able to establish a generic codec with basic and standard concrete layer components involving JUnit for services, and HTMLUnit for the interface forms. This was not the

case with telecom testing, which used different concrete layer components for different protocols.

However, there are still many details and techniques specific to testing BPM that have to be understood and documented. As described in section 4.4.1, *TTCN-3* has proved suitable for testing multi role, multi service orchestration in parallel which other testing tools failed to address. This however was not an easy task. For instance, in *Library Borrow Book* process, student tasks - browse and borrow rare book, and return borrowed book - are meant to be run by the same student. In other words, if Student1 has borrowed a book he is solely responsible for the return of this borrowed book to the librarian. This means Student4 cannot execute Student1's instance. However, the librarian task can be executed by any librarian (librarian1 or librarian2), who is part of user group 'librarian' regardless of which librarian was first involved in the process and similar for the cashier. It depends on the role. But this is problematic, because BPM will show tasks meant for different parallel users in the same response when interacting with *TTCN-3*. Thus, there is ambiguity when a test behavior instance has to pick similar tasks that belong to it. We eventually addressed this using one of *TTCN-3*'s features, 'Racing Condition', in our test cases.

This is highlighted even more strongly in *Cancer Care Assessment* project, where there is a case in which a stimulus will produce a response that is applicable to all roles, each of which is handled by a separate *TTCN-3* PTC, at once instead of one at a time. In other words, a response from the SUT must be shared by all PTCs. This is a radically different architecture from what the *TTCN-3* PTC has been initially designed for. In

telecommunication systems testing always consists essentially of stimuli-response pairs and strict association with a given user.

It is also important to emphasize the reusability made possible by the *TTCN-3* codec for BPM. This is illustrated in Figure 5-1. In our case study implementation, the tests developed at the abstract layer invoke user behavior that in turn sends calls to functions in the concrete layer. For example, in Figure 5-1 the behavior 'LibraryTestLibrarianBehavior', called in the test case, makes a call to java triSend( ) method. This method is responsible for fetching the tasks and driving their execution, and remains generic for any other application. So, the entire codec built for *Library Borrow Book* is reusable for *Cancer Care Assessment*.

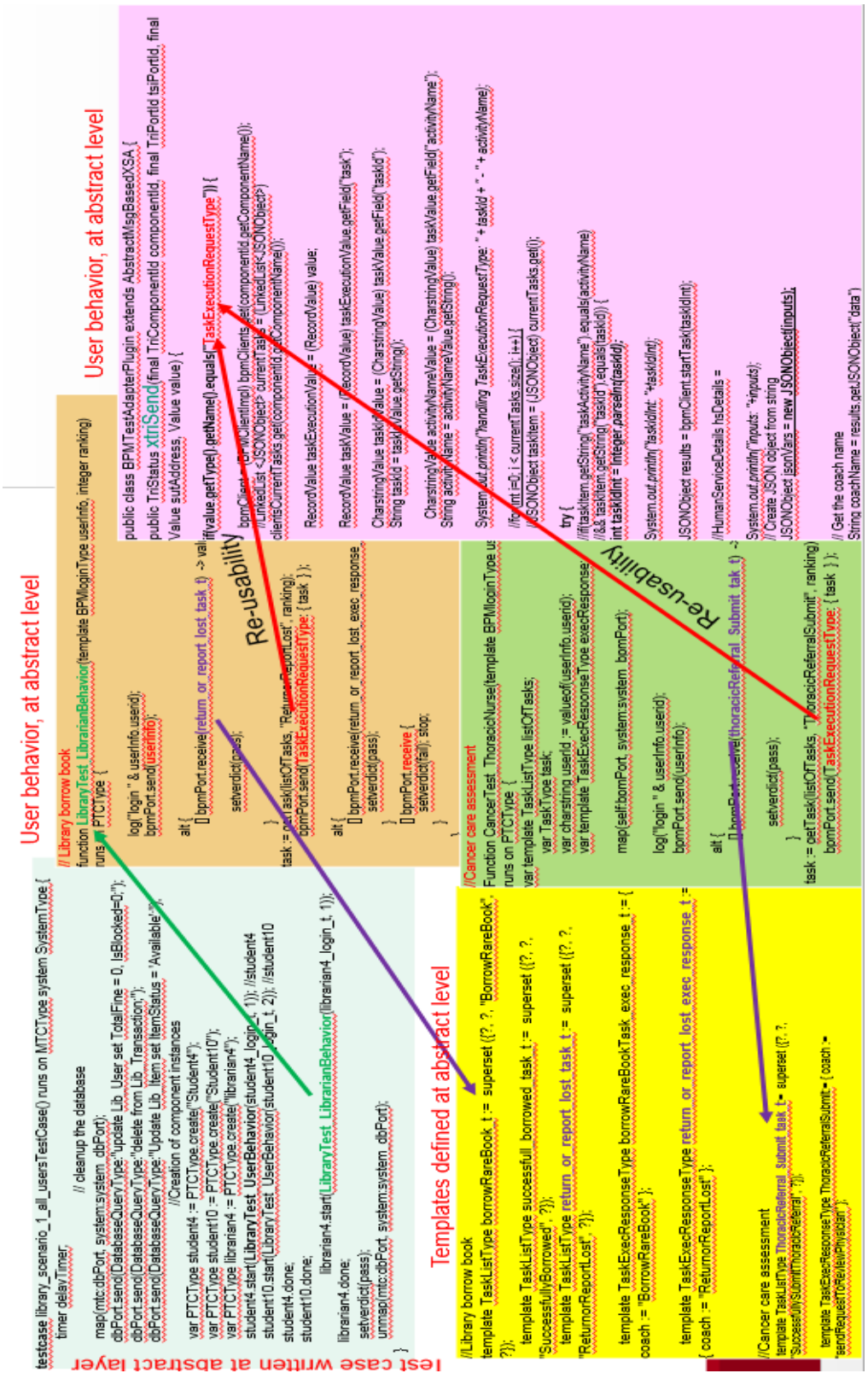


Figure 5- 1 Reusability of Tri-Send in TTCN-3 code for BPM

There was lot of manual effort involved in building the codec and the test effort was proportional to the number and length of the scenarios. However, this effort can be minimised if there exists support for model-based generation of *TTCN-3* test scripts from the business process model and scenarios. For example, in our case studies, it would have been nice if *TTCN-3* had been integrated with IBM BPM Testing Asset in place of Selenium. Then the different paths through the business process model, identified by IBM BPM Testing Asset (described in section 4.2.5) could be edited with sets of timing constraints and expected values to define clear user scenarios, and *TTCN-3* test scripts could be generated to provide complete multi-role, multi-service orchestration testing in parallel.

Chapter 6. Conclusions and Future Work

6.1. Conclusions

In this thesis, we have introduced a quality assurance framework for BPM that leverages three types of tools: model-driven unit testing for BPM services and forms; multi-role, multi-service orchestration testing; and performance reporting. In particular, we highlighted the challenges and opportunities in applying the *TTCN-3* testing standard to BPM. We have demonstrated the success of leveraging a test workbench which offers a suite of tools: model-based test generation tool, an orchestration tool and a performance reporting tool, for quality assurance of BPM.

The framework prototype was implemented and validated against two case studies. For the two case studies, *Library borrow book and Cancer care Assessment,* our framework was clearly better than current testing practices, addressing complex business process features. With the framework it was possible to address all the aspects of BPM quality assurance. In particular, integrating *TTCN-3* proved suitable to address the most technically challenging aspect of BPM testing- multi role, multi service orchestration. Obviously, more case studies in different industries are needed, but the preliminary results are promising. More work is also needed to ensure that the framework will work with business processes modeled in other tools besides BPM 8.5.5. Despite the BPEL standard that is loosely followed, there are variations amongst the architectures of different BPM tools and hence there might be problems or difficulties in using this framework.

We have clearly characterized the types of tools available and their effectiveness and shown the need for an integrated workbench to make the right tool available for the job, but there is still a long way to go. Our systematic approach to modelling SOA, scenarios and goals, lays the groundwork and shows the potential for integrating tools more seamlessly in a model-driven fashion.

6.2. Future Work

The next step of the research is to evaluate our framework with more sophisticated business processes to ensure the proposed framework is practical and feasible for industry use. More industry case studies are needed of course before we can make definitive claims about how well the framework would favor the BPM tester. In order to make this evaluation practical, though, better tool support and integration is needed.

Another future work is to support performance reporting for BPM by supporting model-driven generation of reports from business process model and goal model. Lastly, we would like to expand the model-driven support to generation of test campaign from BPM, scenarios and goal model.

A final future work of this research is to expand support of *TTCN-3* for BPM which includes development of a complete generic codec for BPM, and fully leverage business process model and user scenarios to automatically generate *TTCN-3* tests.

It would be worthwhile to explore the creation of an open source project that provided full coverage of BPM quality assurance driven by the business process model, plus goal models, plus user scenarios.

References

Aguilar-Saven, R. S. (2003). Business process modelling: Reviewand framework. *Int. J. Production Economics 90 (2004)*, 129-149.

Alhaj, M. (2011). Automatic Generation of Performance Models for SOA Systems. *Proceedings of the 16th international workshop on Component-oriented programming.*

Alsquor, M., Owoc, M. L., & Ahmed, A. S. (2014). Data Warehouse as a Source of Knowledge Acquisition. An emperical study. (pp. pp.1421-1430). Warsaw: IEEE.

Amyot Daniel, S. A., Amyot, D., Shamsaei, A., Kealey, J., Tremblay, E., Miga, A., . . . Cartwright, N. (2012). Towards Advanced Goal Model Analysis with jUCMNav. *3rd Int. Workshop on Requirements, Intentions and Goals in Conceptual Modeling (RIGiM'12)* (pp. pp. 201-210). Florence, Italy: Springer.

Beck, K., & Andress, C. (2004). Extreme Programming Explained: Embrace Change (2nd Edition ed.). *Addison-Wesley Professional*.

Bernard Stepien, L. P. (June 2014). Innovation and evolution in integrated web application testing with TTCN-3. *International Journal on Software Tools for Technology Transfer*, pp. 269-283.

Biju, S. (2008). Agile Software Development. *E- Learning , 5 (1)*, pp. 97-102.

Booth, D., Haas, H., & Brown. (2004). *A Web Service Glossary: Technical Report.* World Wide Web Consortium (W3C), http://www.w3.org/TR/ws-gloss/.

Booth, D., Haas, H., McCabe, F., Newcomer, E., Champion, M., Ferris, C., & Orchard, D. (2004, February 11). *Web Services Architecture* . Retrieved from W3C Working Group Note: http://www.w3.org/TR/ws-arch/#whatis

Bruce Silver Associates. (2006). *The 2006 BPMS report: Understanding and evaluating BPM suite.* BPMInstitute.org. Retrieved May 25, 2012, from http://doc.mbalib.com/view/d8d13d5f85e8735de2229bfb94a37d2f.html

Cohen, D., Lindvall, M., & Costa, P. (2004). An Introduction to Agile Methods. *Advances in Computers*, pp 1-66.

Collofello, J., & Vehathiri, K. (2005). An Environment for Training Computer Science Students on Software Testing. *Frontiers in Education, 2005. FIE '05. Proceedings 35th Annual Conference* (pp. T3E - 6). Indianopolis, IN: IEEE.

Collofello, J., & Vehathiri, K. (2005). An Environment for Training Computer Science Students on Software Testing. *Frontiers in Education, 2005. FIE '05. Proceedings 35th Annual Conference* (pp. T3E - 6). Indianopolis, IN: IEEE.

Coulter, A. C. (1999). Graybox Software Testing Methodology: Embedded Software Testing Technique. *Digital Avionics Systems Conference, 1999. Proceedings. 18th (Volume:2)* (pp. 10.A.5-1 - 10.A.5-8 vol.2). St Louis, MO: IEEE.

Curbera, F., Duftler, M., Khalaf, R., Nagy, W., Mukhi, N., & Weerawarana, S. (2002). Unraveling the Web Services Web. *IEEE*, vol. 6, no. 2, pp. 86–93.

Dijkman, R. M., Dumas, M., & Ouyang, C. (2008). Semantics and analysis of business process models in BPMN. *Information and Software Technology*, pp. 1281-1294.

Dresner, H. (2003). Business Activity Monitoring. *Gartner Symposium, November*, (pp. pp.4-7). Cannes, France.

Easterbrook, S. (2010). The difference between Verification and Validation., (pp. Retrieved on March 21, 2015).

Easterbrook, S., & Callahan, J. (1998). Formal Methods for Verification and Validation of partial specifications: A Case Study. *Journal of Systems and Software, vol. 40, (3)*, pp. 1-13.

Ertugrul, A., & Demirors, O. (2015). An exploratory study on role-based collaborative business process modeling approaches. *S-BPM ONE '15 Proceedings of the 7th International Conference on Subject-Oriented Business Process Management.* New York, NY, USA.

Gartner, I. (2013). *IT Glossary, Business Activity Monitoring*. Retrieved from Gartner Research: http://www.gartner.com/it-glossary/bam-business-activity-monitoring

Gellersen, H.-W., & Gaedke, M. (1999). Object Oriented Web Application Development. *IEEE Internet Computing, vol. 3(1)*, pp. 60-68.

Gupta, A., & Jalote, P. (2007). An Experimental Evaluation of the Effectiveness and Efficiency of the Test Driven Development. *First International Symposium on Empirical Software Engineering and Measurement*, pp. 285-294.

Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). Design Science Information Systems. (pp. pp. 75-105.). MIS Quarterly.

Huhns, M. N., & Singh, M. P. (2005). Service-Oriented Computing: Key Concepts and Principles. (pp. vol. 9, no. 1, pp. 75-81). EEE Internet Computing.

Ilieva, S., Manova, I., & Petrova-Antonova, D. (2012). Towards a Methodology for Testing of Business Processes. (pp. pp.1315 - 1322). Wroclaw: IEEE.

Inmon, W. H. (2005). *Building the data warehouse* (4th ed.). Indianapolis: Wiley Publishers.

Jarrar, Y. F., Al-Mudimigh, A., & Zairi, M. (2000). ERP implementation critical success factors -the role and impact of business process management. *IEEE International Conference on Management of Innovation and Technology* (pp. 122 - 127 vol.1). IEEE.

Jufuru, V. (2007, September). *Business Activity Monitoring – Economic Impact on Industry Verticals.* Retrieved from BPTrends: www.bptrends.com

JUnit. (2014, December 09). Retrieved from http://junit.org/

Kacem, H. H., Sellami, W., & Kacem, A. H. (2012). A formal approach for the validation of web service orchestrations. *21st International WETICE* (pp. pp. 42-47). IEEE.

Kang, J. G. (2008). A Business Activity Monitoring System Supporting Real-Time Business Performance Management. *Third International Conference on Convergence and Hybrid Information Technology.* (pp. pp.473-478). Busan: IEEE.

Kaur, H., & Gupta, G. (2013). Comparative Study of Automated Testing Tools: Selenium, Quick Test Professional and Testcomplete. *Journal of Engineering Research and Applications*, pp1739-1743.

Kaur, M., & Kumari, R. (2011). Comparative Study of Automated Testing Tools: TestComplete and QuickTest Pro. *International Journal of Computer Applications (0975 – 8887) Volume 24*.

Khan, M. E., & Khan, F. (2012). A Comparative Study of White Box, Black Box and Grey Box Testing. *(IJACSA) International Journal of Advanced Computer Science and Applications,Vol. 3, No.6*, pp.12-15.

Kimball, R., & Ross, M. (2013). *The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling* (3rd ed.). John Wiley & Sons.

Kolban, N. (2014). *Kolban's book on IBM Business Process Management.*

Kronz, A. (2006). Managing of Process Key Performance Indicators as Part of the ARIS. *Corporate Performance Management* (pp. pp.31–44). Springer.

Kruchten, P. (2004). The Rational Unified Process: An Introduction. Addison-Wesley Professional.

Kruppke, H., & Bauer, T. (2006). No Business Intelligence Without Process Intelligence. In *Corporate Performance Management* (pp. pp.77-98). Springer Berlin Heidelberg New York.

Kuziemsky, C., Liu, X., & Peyton, L. (2010). Leveraging Goal Models and Performance Indicators to Assess Health Care Information Systems. *2010 Seventh International Conference on the Quality of Information and Communications Technology*, (pp. pp.222-227).

Lee, Y. (2009). An Implementation Case Study: Business Oriented SOA Execution Test Framework. *Fifth International Joint Conference on INC, IMS and IDC.* IEEE.

Leymann, F., Roller, D., & Schmidt, M. T. (2002). *IBM2, Web services and Business Process Management.* IBM Systems Journal.

Li, Z. J., Tan, H. F., Liu, H. H., Zhu, J., & Mitsumori, N. M. (2008). Business-process-driven gray-box SOA Testing. *IBM SYSTEMS JOURNAL, VOL 47, NO 3*, pp. 457-472.

Liu, L., & Yu, E. (2005, October 8). Designing Information Systems in Social Context: A Goal and Scenario Modelling Approach. *Information systems 29.2 (2004): 187-203.* Retrieved from Wikipedia: http://en.wikipedia.org/wiki/Goal_modeling

Lucca, G. A., & Fasolino, A. R. (2006). Testing Web-based applications: The state of the art and future trends. *Information and Software Technology 48 (2006) 1172–1186.* Elsevier B.V.

Maximilien, E. M., & Williams, L. (2003). Assessing Test-Driven Development at IBM. *25th International Conference on Software Engineering*, pp. 564-569.

Middleton, G., Peyton, L., Kuziemsky, C., & Eze, B. (2009). A framework for continuous compliance monitoring of eHealth Processes. *World Congress on Privacy, Security, Trust and the Management of e-Business*, (pp. pp. 152-160).

Morris, E., Anderson, W., Bala, S., Carney, D., Morley, J., Place, P., & Simanta, S. (2010). *Testing in Service-Oriented Environments.* Research, Technology, and System Solutions (RTSS) Program.

Moss, L. T., & Atre, S. (2003). Business Intelligence Roadmap.

Mulesoft. (2015, February 22). Retrieved from Service Orchestration and SOA: https://www.mulesoft.com/resources/esb/service-orchestration-and-soa

Myers, G. J. (2004). *The Art of Software Testing, Second Edition, ISBN 0-471-46912-2* . John Wiley & Sons, Inc., Hoboken, New Jersey. .

Nickul, D. (2007, December). Service Oriented Architecture (SOA) and Specialized Messaging Patterns. *Adobe technical paper*.

Nogueras, J. R., & Olivieri, R. (2013, May). *Automating your business process application test cases using Java in IBM Business Process Manager*. Retrieved December 09, 2014, from http://www.ibm.com/developerworks/bpm/bpmjournal/1212_olivieri/1212_olivieri.html

Olga Levina, O. H.-R. (2010). Extracting Business Logic from Business Process Models. (pp. 289 - 293). Chengdu: IEEE.

Olsen, E. (2014, April 11). *Performance management*. Retrieved from M3 Planning: http://mystrategicplan.com/resources/section/glossary/

Ouyang, C., Verbeek, E., van der Aalst, W. M., Breutel, S., Dumas, M., & ter Hofstede, A. H. (2007). Formal Semantics and Analysis of Control Flow in WS-BPEL*. *Science of Computer Programming, Volume 67, Issues 2–3*, pp. 162-198.

Owen, M., & Raj, J. (2003). *BPMN and Business Process Management: Introduction to the New Business Process Modeling Standard.* Retrieved from http://www.omg.org/bpmn: http://www.omg.org/bpmn/Documents/6AD5D16960.BPMN_and_BPM.pdf

Peyton, L., Stepien, B., & Seguin, P. (2008). Integration Testing of Composite Applications. Waikoloa, HI: IEEE.

Peyton, L., Zhan, B., & Stepien , B. (2008). A Case Study in Integrated Quality Assurance for Performance Management Systems. *MVSVEIS.*

Radovan, A. (2012, September 07). *IBM1, Business Process Management IBM's Way*. Retrieved from http://www.croz.net/eng: http://www.croz.net/eng/business-process-management-ibms-way/

Reynolds, J., Collins, M., Ducos, E., Frost, D., Knapp, D., Kornienko, I., . . . Pfau, G. (April 2014). *Leveraging the IBM BPM Coach Framework in Your Organization (First Edition).* ibm.com/redbooks.

Robic, D. (2010). Workflow testing . *MIPRO, 2010 Proceedings of the 33rd International Convention* (pp. pp.320 - 323). Opatija, Croatia: IEEE.

Rud, O. P. (2009). *Business Intelligence Success Factors: Tools for Aligning Your Business in the Global Economy.* John Wiley & Sons Date published: 2009 ISBN-13: 9780470392409 ISBN: 0470392401.

Sánchez, L. G., Ruiz, F., García, F., & Piattini, M. (2013). Improving Quality of Business Process Models. *Evaluation of Novel Approaches to Software Engineering Communications in Computer and Information Science Volume 275*, pp 130-144.

Schiefer , J., & Seufert, A. (2005). Management and Controlling of Time-Sensitive Business Processes with Sense & Respond. *Computational Intelligence for Modelling, Control and Automation* (pp. pp. 77 - 82). Vienna: IEEE.

Schiefer, J., Saurer, G., & Schatten, A. (2006). Testing Event-Driven Business Processes. *Journal of Computers, Vol 1, No 7*, pp.69-80.

Schieferdecker, I., Stepien, B., & Rennoch, A. (1997). PerfTTCN, a TTCN language extension for performance testing. *IWTCS.* Cheju Island, Korea.

Soeken, M., Wille, R., & Drechsler, R. (2012). Assisted Behavior Driven Development Using Natural Language Processing. *50th International Conference on Objects, Models, Components, Patterns*, 7304, pp. 269-287.

Solis, C., & Wang, X. (2011). A Study of the Characteristics of Behaviour Driven Development. *37th EUROMICRO Conference on Software Engineering and Advanced Applications*, pp. 383-387.

Steen, B. (2007). BPM on Top of SOA: Experiences from the Financial Industry. pp. 96-111.

Stepien, B. (2015, February 14). *Testing and Test Control Notation.* Retrieved from TTCN-3 in a Nutshell: http://www.site.uottawa.ca/~bernard/ttcn3_in_a_nutshell.html

Stepien, B., & Peyton, L. (2013). Challenges of Testing Periodic Messages in Avionics Systems Using TTCN-3. *25th IFIP WG 6.1 International Conference, ICTSS 2013* (pp. pp. 207-222). Istanbul, Turkey: Springer.

Stepien, B., Peyton, L., & Xiong, P. (2008). Framework testing of web applications using TTCN-3. *Int J Softw Tools Technol Transfer (2008) 10:371–381* (pp. pp. 371-381). Springer-Verlag.

Stepien, B., Xiong, P., & Peyton, L. (2011). A Systematic Approach to Web Application Penetration Testing Using TTCN-3. *MCETECH* (pp. pp. 1–16). Berlin Heidelberg: Springer-Verlag.

Stoica, M., Mircea, M., & Ghilic, M. B. (2014). Software Development: Agile vs. Traditional. *Informatica Economică vol. 17*, pp. 64-76.

Strobl, R., Mullner, T., & Rausch, T. (2009). Web-Based Process Portals: Powering Business Process Management within Large Organisations. *Commerce and Enterprise Computing, 2009. CEC '09.* (pp. P312-316). Vienna, Austria: IEEE.

Trkman, P., McCormack, K., de Oliveira, M. P., & Ladeira, M. (2010, June). The impact of business analytics on supply chain performance. *Decision Support Systems, 49*(3), 318-327. doi:10.1016/j.dss.2010.03.007

TTCN-3. (2014, December 09). Retrieved from TESTING AND TEST CONTROL NOTATION VERSION 3 (TTCN-3): http://www.ttcn-3.org/

van der Aalst, W. M. (2004). Business process management demystified: a tutorial on models, systems and standards for workflow management. *in Lectures on Concurrency and Petri Nets, J. Desel, W. Reisig, and G. Rozenberg, Eds., vol. 3098 of Lecture Notes in Computer Science* (pp. pp. 1–65). Berlin, Germany: Springer-Verlag.

van der Aalst, W. M. (2013). Business Process Management: A Comprehensive Survey.

Villar, C. (2010). *A Goal-Driven Methodology for Developing Health Care Quality Metrics.* Masters Thesis in Electronic Business Technologies, University of Ottawa, Telfer School Of Management, Ottawa, Ontario.

Wetzstein, B., Ma, Z., Filipowska, A., Maczmarek, M., Bhiri, S., Losada, S., . . . Cicurel, L. (2007). Semantic Business Process Management: A Lifecycle. *Workshop on Semantic Business Process and Product Lifecycle Management (2007)*, (pp. P1-11). Innsbruck, Austria.

White, S. A., & Miers, D. (2008). *BPMN Modeling and Reference Guide.* Lighthouse Pt, FL: Future Strategies Inc. Retrieved from http://www.bizagi.com/.

Wieland, M., Kopp, O., Nicklas, D., & Leymann, F. (2007). Towards Context-aware Workflows. *Proceedings of the Workshops and Doctoral Consortium*, pp 11-15.

Wohed, P., van der Aalst, W. M., Dumas, M., ter Hofstede, A. H., & Russell, N. (2005). *Pattern-based Analysis of BPMN - an extensive evaluation of the Control-flow, the Data and the Resource Perspectives (revised version).* BPM Center Report BPM-06-17, BPMcenter.org.

Wynn, & Kyaw, M. T. (2006). Semantics, Veri¯cation, and Implementation of Work°ows with Cancellation Regions and OR-Joins. *PhD Thesis*. Brisbane, Australia.

Wynn, M. T., van der Aalst, W. M., ter Hofstede, A. H., & Edmond, D. (2006). Verifying Workflows with Cancellation Regions and OR-Joins: An Approach Based on Reset Nets and Reachability Analysis. (pp. pp 389-394). Berlin Heidelberg: Springer-Verlag.

Wynn, M. T., Verbeek, H. M., van der Aalst, W. M., ter Hofstede, A. H., & Edmond, D. (2009). Business process verification – Finally a reality! *Business Process Management Journal, Vol. 15*, pp.74-92.

Yu, E., & Mylopoulos, J. (1998, March). *Why Goal-Oriented Requirements Engineering.* Retrieved from http://www.cs.toronto.edu/pub/eric/REFSQ98.html

Ziqiang, Z., & Hong, C. (2010). Research on Business Process Management Framework Based on BPEL. *Information Technology and Applications (IFITA)* (pp. pp.45 - 48). Kunming: IEEE.

Appendix TTCN-3 Library Borrow Book Example

Figure A-1 explains one of the test cases developed for collaborative testing of multiple user roles in parallel involving multiple service orchestrations for multiple user tasks in parallel. The test involved two users: student4 and student10.

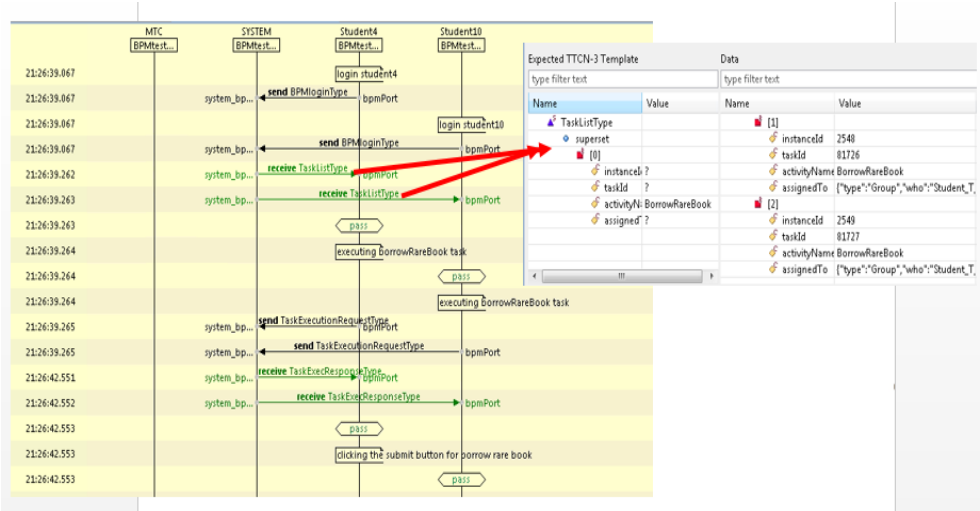


Figure A-1: Collaboration of multiple user roles in orchestration of multiple services using TTCN-3

In Figure A-1, the test has created two instances, one for student4 and the other one for student10. After login in, the BPM will return exactly the same list of BorrowRareBook tasks, one for each user while the next step will consist in each user to execute one of the two BorrowRareBook tasks but not both by the same user. The Figure A-2 describes the tasks picked by the two students:

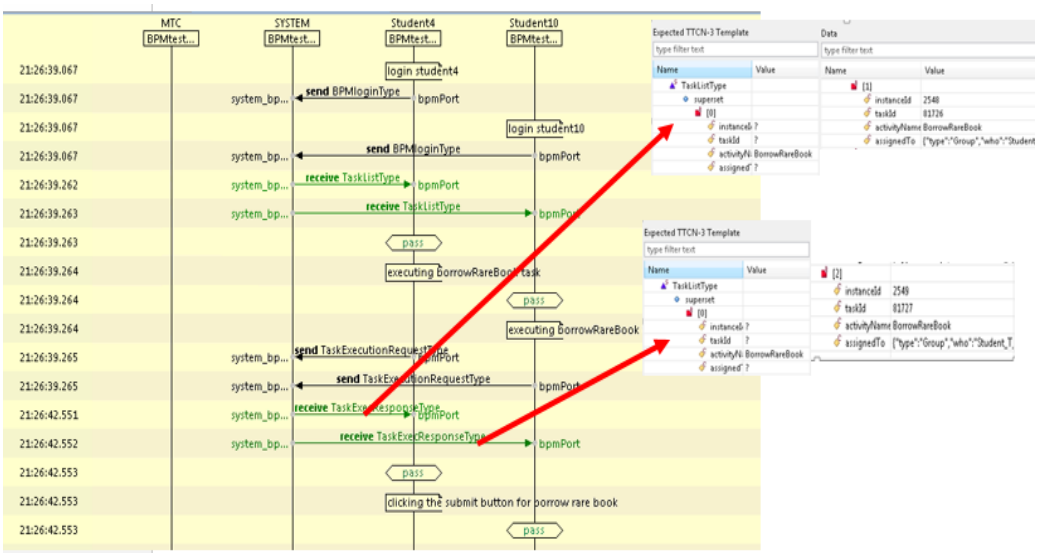


Figure A-2: TTCN-3 test displaying task allotment for the two student users

As seen in Figure A-2, student4 picks a task and student10 picks another task. This is achieved by introducing ranking which ties the tasks to the users in the test. Each user has a sequential rank. For example user4 will have the rank 1 and user 2 will have rank 2. Thus, using this rank, a user will pick the identical labelled task according to its rank. User 4 will pick the first instance of BorrowRareBook task while user10 will pick the second. Without this feature and using strictly the task name, both users ended up picking the first instance of this task. The above figures are illustrations of one of the features of *TTCN-3* execution tools to produce very detailed logs showing the sequences of steps involved in tests and the matching. The matching window displays two columns, the test oracle on the left showing the subset of tasks received from SUT on the right.

Figure A-3 indicates the code snippet for this test. As seen in the code snippet, the two student users: student4 and student10 perform their test behavior in strict parallel. The librarian4 then begins with his behavior. In this interleaved example, the two independent student user threads are instantiated and programmed to execute

simultaneously. all component.done is used to verify all the activities of both the users have completed. And Figure A-4 describes the behavior of a student.

```
testcase library scenario 1 all usersTestCase() runs on MTCType system SystemType {
         timer delayTimer;
               // cleanup the database
         map(mtc:dbPort, system:system_dbPort);
        dbPort.send(DatabaseQueryType:"update Lib User set TotalFine = 0, IsBlocked=0;");
        dbPort.send(DatabaseQueryType:"delete from Lib Transaction;");
        dbPort.send(DatabaseQueryType:"Update Lib_Item set ItemStatus = 'Available';");
                              //Creation of component instances
         var PTCType student4 := PTCType.create("Student4");
         var PTCType student10 := PTCType.create("Student10");
         var PTCType librarian4 := PTCType.create("librarian4");
         student4.start(LibraryTest_UserBehavior(student4_login_t, 1)); //student4
         student10.start(LibraryTest_UserBehavior(student10_login_t, 2)); //student10
        student4.done;
        student10.done;
          librarian4.start(LibraryTest_LibrarianBehavior(librarian4_login_t, 1));
        librarian4.done;
         setverdict(pass);
         unmap(mtc:dbPort, system:system_dbPort);
     }
```

Figure A-3: Code Snippet Describing Test case written in TTCN-3

```
function LibraryTest_UserBehavior(template BPMloginType userInfo, integer ranking) runs on PTCType  {
          timer waitForReturningBook;
// variables are defined here
          log("login " & userInfo.userid);
          bpmPort.send(userInfo);
          alt {
              [] bpmPort.receive(borrowRareBook_t)  -> value listOfTasks {  setverdict(pass);  }
              [] bpmPort.receive {
                  setverdict(fail); stop;
              }
          }
          log("executing borrowRareBook task");
          task := getTask(listOfTasks, "BorrowRareBook", ranking);
          task.assignedTo := "nil";

          bpmPort.send(TaskExecutionRequestType: { task });

          alt {
              [] bpmPort.receive(borrowRareBookTask_exec_response_t) -> value execResponse {
                  setverdict(pass);
              }
              [] bpmPort.receive {
                  setverdict(fail); stop;
              }
          }     // Similarly other tasks are executed

      }
```

Figure A-4: TTCN-3 Behavior function of the student

The tests constructed in *TTCN-3* involved the following specification:

Login specification is described in the user login template as illustrated in the below screenshot and is used to send the user login information to the BPM.

```
// Template
template BPMloginType
student4_login_t := {
   userid := "student4",
   password := "password"
}
```

```
//datatype
type record BPMloginType {
   charstring userid,
   charstring password
}

bpmPort.send(userInfo);
```

Checking List of Task Response enables us to check that the particular task is present in the returned list of tasks which is stored in a variable, as illustrated in the below snippet

```
// Template
template TaskListType
borrowRareBook_t :=
        superset ({?,
"BorrowRareBook", ?});
```

```
// datatype
type record TaskType {
   charstring taskId,
   charstring activityName,
   charstring assignedTo
}

type set of TaskType
TaskListType;
```

```
//received tasks in listOfTasks
alt {
      []
bpmPort.receive(borrowRareBook_
t)
                              ->
value listOfTasks {
            setverdict(pass);
      }
        ...
```

Task Execution Request is to ask the test adapter to execute a task that the tester has picked The following code snippet describes the *getTask()* function details. The function is used to search for a particular task. In case there exists more than one task with the same name, then their occurrences are checked against the rank of the user.

```
log("executing borrowRareBook task");

Task := getTask(listOfTasks, "BorrowRareBook",
ranking);
Task.assignedTo := "nil";

bpmPort.send(TaskExecutionRequestType: {
task });
```

```
function getTask(template TaskListType listOfTasks, charstring taskName, integer ranking) return
TaskType {
                var TaskType foundTask := null;
                var integer numberOfOccurences := 0;
        for(var integer i :=0; i < sizeof(listOfTasks); i := i + 1) {
                    if(listOfTasks[i].activityName == taskName) {
            foundTask := valueof(listOfTasks[i]);
            numberOfOccurences := numberOfOccurences +1;
            if(ranking == numberOfOccurences) {     break;      }
                } }  return foundTask;
```

Task Execution Validation is performed to ensure if the form displayed is as expected.

```
//datatype
type record TaskExecResponseType {
  charstring coach
}
```

```
//check if the form is : 'BorrowRareBook'

template TaskExecResponseType
    borrowRareBookTask_exec_response_t
:=
                        { coach :=
"BorrowRareBook" };
```

Form Content Submission is used to submit the user form.

```
//Template definition
template ClickButtonType click_
   submit_bottow_rare_book_button_t
                (charstring requestedTaskId)
:= {
      taskId := requestedTaskId,
      buttonRealName := "submit",
      buttonName := "ButtonGroup0_Button0"

 };
```

```
//Data Type
type record ClickButtonType {
   charstring taskId,
   charstring buttonRealName,
   charstring buttonName
}
```

```
     bpmPort.send
(click_submit_bottow_rare_book_button_t(task.taskId));
```

Test Behavior Variables are used to store the list of tasks returned by BPM and the task selected for execution.

```
var template TaskListType listOfTasks;
 var TaskType task;
```

```
bpmPort.receive(borrowRareBook_t)
                                -> value listOfTasks
```