2015 Final year Computer Science project proposal by Symbiotics

13 March 2015
### Table of Contents

**Introduction**

**Project vision and objectives**
- Project vision
- Problem statement
- Main objectives

**Project owner**

**Project scope**

**Functional requirements**

**Use Cases**
- User Login
- New Project Registration

**Architectural requirements**

**Quality requirements**
- Performance
- Scalability
- Reliability
- Security
- Code coverage

**Integration and access requirements**

**Technology preferences**
- Programming language
- Database
- Application server
- Build automation
- Version control
- Testing tools
- Code quality

**Skills requirements**
- Agile and fast learners

**Project deliverables**
- Source code
- Repository
- Documentation

**Client commitments**
- Enterprise Infrastructure
- Available resources to teams
- Meetings
Project name

Project Management Life-Cycle Automation

Introduction

Software development companies regularly start up new client projects, projects which require a large infrastructure to support both the project itself as well as the developers working on it. This task often involves configuring several tools for the project, these tools can include: a project management tool (eg. JIRA), a project document tool (eg. Confluence), a tool for automated project build and deployment (eg. Jenkins). This initial phase also involves setting up deployment environments (dev, int, and QA) for the project as well as creating a scaffold for the project code and a repository to store the project. A task like this would be made much easier if it could be automated.

The goal of this project is to create a system that allows for a software development company to easily set up everything they require for a new project.

Project vision and objectives

Project vision

We would like to explore a potential solution for coordinating the life-cycle of a customer project, from inception all the way through to completion. This solution would efficiently solve the problem of coordinating a project across the phases of the project and across multiple infrastructure systems.

Problem statement

The coordination required to manage the full life-cycle of a customer project uses up valuable resource time.

Main objectives

To develop an automated process for creating and managing a customer project, which then ultimately reduces the time spent to create and manage a customer project. To provide the team with an opportunity to get a feel of "commercial" Agile and Software Engineering processes, as well as expose students to the infrastructure tools commonly used in the software development industry.

Project owner

The project owners for this project are Poloko Mphahlele and Deon Taljaard, graduates from the University of Pretoria. While their main focus will be to act as clients for the project, they
also aim to guide and advise the team on bridging their academic knowledge and skills to commercial use.

<table>
<thead>
<tr>
<th>Project owner</th>
<th>Email</th>
</tr>
</thead>
<tbody>
<tr>
<td>Poloko Mphahlele</td>
<td><a href="mailto:polo.mphahlele@symbiotics.co.za">polo.mphahlele@symbiotics.co.za</a></td>
</tr>
<tr>
<td>Deon Taljaard</td>
<td><a href="mailto:deon.taljaard@symbiotics.co.za">deon.taljaard@symbiotics.co.za</a></td>
</tr>
</tbody>
</table>

**Project scope**

To manage the full life-cycle of a customer project a number of key aspects need to be coordinated, namely:

- Registering an initiative (i.e sales opportunity) and subsequently managing the preparation and submission of a proposal / tender
- Recording the contract and key deliverable & milestone components of the contract
- Co-coordinating key information required by the customer, such as:
  - Deliverable / Milestone sign-offs
  - Accounting information, e.g:
    - Contract reference
    - Purchase order number
    - Cost centre number
    - Project reference

This information and associated references need to be aligned and reflected within Symbiotics' project delivery and control systems to enable seamless execution. Examples include ensuring that tasks and associated time capture enable correct invoicing as per specific customer requirements.

As we work across multiple large corporates, each with their own processes and documentation demands, we need to have a commonality that underpins the processes while still ensuring we meet critical requirements that are specific to each customer as this facilitates timely approvals and invoice payments.

The integration of the information flow must not only be within administrative systems but must extend to software engineering process and execution toolsets to ensure an efficient coordinated approach. Examples include automated setup of JIRA, Confluence, Jenkins, SVN through to Integration test infrastructure.

**Functional requirements**

1. The application should be web-based, allowing administrative employees to access its functionality from a browser on any device connected to an appropriate VPN.
2. The application should require users to log in before accessing any functionality.
a. The username and password of the user should be authenticated against an LDAP instance.
b. The permissions of the user should be determined by the groups the user is in on LDAP.

3. The application should allow an authorised user to register a new project, coordinating the following information
   a. Customer details
   b. Contract details
   c. Specific Scope of Work
   d. Customer Purchase Order and related project preferences (e.g. cost centre, project reference number, etc.)
   e. Project resources and project specific rates information (where T&M)

4. The application should set up and configure the initial Confluence space for Project Documentation Management of the newly registered project
   a. Based on project type template
   b. Initial documents created within standard structure, ready for enhancement with project specific details

5. The application should set up and configure a project in the JIRA project management tooling for the newly registered project
   a. Create Project in toolset
   b. Configure initial agile board
   c. Populate agile board with Project Inception tasks

6. The application should set up and configure the automated build and deployment tools (Jenkins or Bamboo) for the newly registered project
   a. Internal and Customer Reporting
   b. Source Code Management Repository
   c. Continuous Integration Toolset and processes
   d. Integration testing environment setup and configuration

7. The application should set up access for the relevant personnel
   a. Grant personnel access to the various tools configured for the newly registered project
   b. Notify personnel of the granted access

**Use Cases**

The two main use cases for the system is to have a user log in and then to register a new project. These goals are depicted below.
Architectural requirements

This section layouts what we expect with regards to the solutions quality, the integration and access requirements as well technology preferences that should be used to implement a solution.

Quality requirements

The source code of the project will be analysed on a regular basis by SonarQube. This will provide the quality metrics that will be used during this project.

Performance

The application should be responsive at all times under normal working conditions. The performance scalability is not a large concern as the application user base will likely be limited to administrative personnel.

Scalability

The architecture of the solution should be modular and allow for integration points to be added and removed with minimal impact to the rest of the system.

Reliability

The application should gracefully handle all failures and errors, providing the user with intuitive feedback. All failures should be appropriately logged for auditing purposes.

Security

The application will be secured using Keycloak, integrated with an instance of LDAP. The application will likely only be accessible via a VPN connection.

Code coverage

The source code of the project should have a line coverage of at least 70% and a branch coverage of at least 60%.

Integration and access requirements

The team will be provided access to all the required infrastructure (Jenkins, SonarQube, etc) to develop the solution. Also see the Enterprise Infrastructure section under Client Commitments. In terms of integration points, the team will be provided with the APIs to use to develop the adapters.

Technology preferences

We would like the team to use the latest technologies that are used to augment the Software Engineering process in the industry. Although these technologies are used by Symbiotics, we
are open for suggestions to alternatives. You can assume that the version of the technologies mentioned below should be the latest stable version at the time development begins.

Programming language

The core business logic of the project MUST be Java based, utilising JEE 7 functionality such as CDI, EJB and JPA. Java is platform (operating system) agnostic, thus there is no specific platform to target. The view end of the project should preferentially be built with Angular JS, but this is open for negotiation as well. However, we would like to see REST and JSON being used as communication between the view and the core logic of the solution.

Database

If a database is needed for the solution, we would prefer PostgreSQL, but this is open for negotiation. Also, if a database is needed for the application, we would like to see Liquibase used to manage changes and migration to the database model.

Application server

The final deployable deliverable should run on the latest version of the Wildfly application server.

Build automation

We would like the solution to be Mavenised, meaning it should use Maven for dependency management as well building the deployable deliverable. In conjunction with Maven, Jenkins should be utilised for continuous delivery.

Version control

We use SVN for version control, but are open to negotiate other version control technologies such as Git or Mercurial.

Testing tools

For unit tests, we would like to see JUnit used in conjunction with Mockito. Where needed and where appropriate, Arquillian should be used for integration tests. JaCoCo should be used to analyse code coverage.

Code quality

We expect SonarQube to be used as a static code analysis tool to enforce sufficient code quality.
Skills requirements

Agile and fast learners

The developers on this project should be willing to learn new technologies as well as participate in an Agile development process. We emphasise attitude over aptitude.

Project deliverables

Source code

The source code of the project should be delivered at regular intervals during the project.

Repository

Our organisation has a preference towards SVN, however, students may make use of Github if they would prefer.

Documentation

As an output of the project processes, the following documents should be produced

A. Functional Requirements Specification
B. Architectural Specification
C. User Manual
D. Test Reports
E. Quality Reports

Client commitments

We will run this as governed by the standard project methodology adopted by the project (guided by Symbiotics' standard approach), so we would expect to have;

- a formal showcase in week 2 and 3 of each sprint,
- the project team involve the product owners as required (we will commit to min 1 meeting at UP per week).

Enterprise Infrastructure

The developers of the project will have access to the infrastructure tools used by Symbiotics. This includes the following set of tools:

A. JIRA and Confluence for project management
B. Jenkins for project build, test and deployment automation
C. SonarQube for project analysis
Available resources to teams

The product owners can meet the team at a location convenient to them. This will probably be on the premises of the university.

Meetings

We will be available for weekly or once every second week (every third week at worst) update meetings, which will take the form of a showcase (or demo). The frequency all depends on the team's availability. We don't expect working deliverables in the planning stage of the project, but would like to see improvements in documentation and research and development (proof of concept) in this phase. The team can use this opportunity to demonstrate suggestions, enhancements or improvements they would like to make with regards to the initial functional specification.