

Avantica San Carlos - Signage System

Software Architecture Document

Author: Brian Salazar Sánchez

Major Revision Date: 15/03/2013

Version: 1.0

# Table of Contents

Table of Contents 3

Document Revision History 4

Reviewed By 4

1.0 Introduction 5

1.1 Purpose 5

1.2 System Context Overview 5

2.0 Architecturally Significant Requirements and Constraints 7

2.1 Quality Attribute Checklist 8

2.2 Other ASRs (Functional and Non-functional) 10

3.0 Domain Analysis 11

4.0 System Interactions 11

4.1 Processes and Workflows 12

5.0 Logical Design View 13

6.0 Deployment View 14

7.0 Key Architectural Decisions and Risks 15

7.1 Risks, Dependencies 15

# 

# Document Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Revised By | Revised Date | Version | Comments |
|  |  |  |  |
|  |  |  |  |

# 

# Reviewed By

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Department | Action | Name | Date | Comments |
| Architecture | Approve |  |  |  |
| Product Management | Informed |  |  |  |
| QA | Informed |  |  |  |
| Development Project Manager | Approve |  |  |  |
| Design/Product Owner | Approve |  |  |  |
| Deployment Team | Approve |  |  |  |

# Introduction

The visibility in a company is extremely important to keep all the colleagues joined and informed about what is making the company as important news that must be known, Avantica needs to improve its visibility, this in first instance to show the quality that is being handled within the company, for this is intended to show different screens of information about the projects that are working, who are working and a set of characteristics of Quality on them.

## 1.1 Purpose

Improve the visibility of the colleagues of Avantica, showing, who are the clients, what is being done, who are doing it, what is the quality of the projects. Keep informed to any colleague of Avantica about all the projects and the quality of them searching to maintain company union.

## 1.2 System Context Overview

The project will have to show three different screens:

Projects Overview: A list with all the projects in operation will be displayed and for each project a summary of their different states, such as the build status, the state of the coverage, the commercial status and the execution status.

Project and Team Detail View: It will display a screen with the customer information (a photo and a description), the description of the project, the information of the different states (the build status, the state of the coverage, the commercial status and the execution status) And Information of the colleagues (photo and occupation).

Admin View: It will show the possibility of changing the integration with external systems such as Jenkins, Sonar and database, in addition you will have the possibility to choose which projects to show in the views and configure the time in which they rotate.

It must integrate the Project to Jenkins which is an external System of continuous build to obtain the state of the build of the projects, to sound that it is a code analyzer where the quality coverage will be obtained, and also to the database of Avantica Where the colleagues will be obtained, also the state of execution and the commercial status.

Use Case diagram:

Caso de uso Siganage (1)

# 

# 2.0 Architecturally Significant Requirements and Constraints

|  |  |
| --- | --- |
| ID | Description |
| FR-01 | The system should display a view of all projects showing the build status, coverage status, execution status and business status of each project. |
| FR-02 | The system should display a detailed view with information about a project such as:  A) Customer's photo and description  B) Project description  C) Build status  D) State of coverage  E) Commercial State  F) Status of Execution  G) Photo, name and position of those involved in the project  As additional information, it can be shown if the project is in the warranty phase or if it has already been closed. |
| FR-03 | The system must show the reason why a state is in red or yellow. |
| FR-04 | The system must display an administrative panel where the following characteristics can be configured:   1. A) Complete integration with database, Jenkins and Sonar 2. B) Administrative users 3. C) Time of rotation between views 4. D) Projects to be shown 5. Edit Projects (client logo, client description, project description) |
| FR-05 | The system must rotate views automatically every now and then. |

## 2.1 Quality Attribute Checklist

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Is applicable?** | **Goal/Requirement** |
| Security | Yes | The system must offer the best practices to protect the information. |
| Authentication | Yes | The system will have an administrator user that is verified with username and password to access the administrative panel |
| Authorization | Yes | To access to the externals applications is requirement an authorization |
| Auditing | No |  |
| Data Sensitivity (at rest/in transit) | Yes | The system must handle the data in a secure way, since it has important data for the company |
| Performance | Yes |  |
| End to end performance (user experience of timing for a particular operation) | Yes | The system must respond quickly because not handle large amounts of information |
| Typical throughput | Yes | The system must be stable. |
| Peak throughput (transactions per second) |  |  |
| Report performance (average rendering time for a report) |  |  |
| Batch job performance (maximum time) |  |  |
| Scalability | No |  |
| Number of users | No |  |
| Data size projections | No |  |
| Multi-tenancy (multiple customers, same DB) | No |  |
| Workload distribution (multiple data centers? Multiple servers?) | No |  |
| Fail-over/Resiliency | Yes | The system must have error handling to prevent the application is over |
| Globalization |  |  |
| Supportability |  |  |
| Error Traceability |  |  |
| Monitoring |  |  |
| Installability/Uninstallability |  |  |
| Upgradability/Updatability |  |  |
| Trainability |  |  |
| Configurability | Yes | The system must be able to modify the different integration with the external systems. |
| Auditability |  |  |
| Level of Documentation | Yes | The system must present a clear and specific documentation. |
| SLA |  |  |
| Maintainability | Yes | The system must recover easily from bugs |
| Lifespan |  |  |
| Extensibility | Yes | The system should allow to add more functionality without any problem. |
| Flexibility | Yes | The system must be able to adapt easily to changes. |
| Modularity | Yes | The system should be divided into modules that facilitate the understanding and development of the system. |
| Testability | Yes | The system testability must be high, since unit tests are implemented. |
| Interoperability | Yes | The system must be integrated with external systems (Jenkins and Sonar) |
| API | Yes | The integration will be through the API offered by those systems. |
| Pluggability | No |  |
| Degree of Dependency or Reuse |  |  |
| Usability | No |  |
| Regulatory Compliance |  |  |
| **Extend with other QuAts if needed** |  |  |

## 2.2 Other ASRs (Functional and Non-functional)

|  |  |
| --- | --- |
| **ASR Name** | **Requirement Statement** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

# 

# 3.0 Domain Analysis

Introduction: The domain is the global communication in a way, to show important information to a large number of people can be of great benefit in this case for the company.

General knowledge about the domain: When communicating a message to a large number of people, it is necessary to take into consider the different methods that exist to find the one that is indicated, to motivate the receptors so that they are always attentive to the channel where they are and Will receive the message, as this is one of the biggest problems.

Customers and users: The users in this case will mainly be the receptors, who will be the colleagues of the company that are mostly in their cubicles in the rooms, so it is necessary that the information is displayed in strategic places, in a visible channel whenever it enters or leaves the room and is easily accessible and visibility.

Tasks and procedures currently performed: Project information is currently displayed through two screens located in each room, and important information is sent through the mails.

# 4.0 System Interactions

The system interacts with 3 components, two external such as Jenkins and Sonar and an internal one that is basically the Avantica database.

Jenkins: It is a external system that works as an automation server generating several benefits such as automatic build of projects. The system must be integrated with jenkins to take the state of the build of each project, if the state of the build is correct or has errors.

Sonarqube: It is an external system that works as a code analyzer and is responsible for detecting different types of vulnerabilities in the code, the state of quality coverage and many more features, in this case sonarqube will take the percentage of coverage of the code that contain unit Test, the percentage must be more than 70% in order for the coverage state to be optimal.

Avantica database: It is the internal database of the company, the system must be integrated with it to obtain the information of the projects and the colleagues.

## 4.1 Processes and Workflows

# Activity Diagram

# 5.0 Logical Design View

Untitled Diagram (2)

The system is divided into three sections, the data source section is responsible for providing the data that will be shown in the views through calls to the APIs, the views section is responsible for showing that data to the user, while the Data Base section is in charge of storing some necessary data as the data for the integration of the different systems. The configuration panel is the only one that is responsible for saving data in the Data Base but the stored data is used in the other views.

# 6.0 Deployment View

Deployment View

The system can be accessed through two forms, from a web browser and from the Concerto System, to access the system send an http request and the front-end in this case React displays a web page with information from the back-end( nodejs, express ), that is Requests through an API, at the same time the back-end requests information to external systems like Sonar, Jenkins and Dashboard, this through an API also, finally with the URL insert it in the Concerto System, Concerto will be in charge to show the pages Web In the different screens.

# 7.0 Key Architectural Decisions and Risks

The system was divided into two separate projects, one to develop the back-end and another to develop the front-end, this to have a more evident separation and to avoid problems between the configuration of the routes of React and the express routes.

## 7.1 Risks, Dependencies

|  |  |  |  |
| --- | --- | --- | --- |
| **Risk** | **Probability (1-5)** | **Severity (1-5)** | **Mitigation** |
| Changes in the APIs | 1 | 4 | Use good programming practices to avoid having to make big changes. |
| Time | 1 | 3 | Follow the schedule. |
| Confusing requirements | 3 | 4 | Good communication with the client. |