Predefined Email Component for Joomla

Version 1.0

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 25/07/2014 | 1.0 | Initial Document | Malaka G.L. |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

1. Introduction 4

1.1 Purpose 4

1.2 Scope 4

1.3 Definitions, Acronyms, and Abbreviations 4

1.4 References 4

1.5 Overview 4

2. Architectural Representation 5

3. Architectural Goals and Constraints 5

4. Use-Case View 6

4.1 Use-Case Realizations 6

5. Logical View 7

5.1 Overview 8

5.2 Architecturally Significant Design Packages 8

6. Process View 9

7. Deployment View 10

8. Implementation View 11

9. Data View 11

10. Size and Performance 11

11. Quality 11

# Introduction

## Purpose

This document provides a comprehensive architectural overview of the component, using a number of different architectural views to depict different aspects of the component. It is intended to capture and convey the significant architectural decisions which have been made on the component.

## Scope

This document provides a deep description about the component to be created. It will describe the architecture of the component developing and the relation between the prevailing system and the developing component.

## Definitions, Acronyms, and Abbreviations

MVC – Model View Controller

CMS – Content Management System

FOSS – Free and Open Source Software

## References

* Initial Project Proposal Document
* Software Requirement Specification Document

## Overview

This document provides a complete overview of the component in various points of views. These views include architectural representation of the system, architectural goals and constraints, use-case view, logical view, process view, deployment view, implementation view, performance and quality. These descriptions will provide a deep explanation for all the developers about how the core implementation of the component is going to be developed.

# Architectural Representation

This document presents the architecture as a series of views; use case view, logical view, process view, deployment view and implementation view. Each of these views will describe the system in depth as below.

Use case view captures system functionality as seen by users. System behavior, what functionality it must provide, is described well in this part.

Logical view of the system will describe the functionalities that the component provides for the end users. In this system end users will be both Joomla web developers and website visitors.

The process view illustrates the dynamic aspects of the system, explaining the system processes and how they communicate while focusing on the runtime behavior of the system. The process view addresses concurrency, distribution, integrators, performance, and scalability, etc.

Deployment view will describe the environment into which the component will be deployed, including the dependencies the system has on its runtime environment.

Implementation view will capture the architectural decisions made for the implementation.

# Architectural Goals and Constraints

Predefined email component for Joomla will help the Joomla web developers a lot by making site manipulation very easy. This component is built as a FOSS. So the developers are welcome to edit and improve the features. Therefore the goal of this system is to maintain a flexible and reusable code which will provide the utmost user satisfaction.

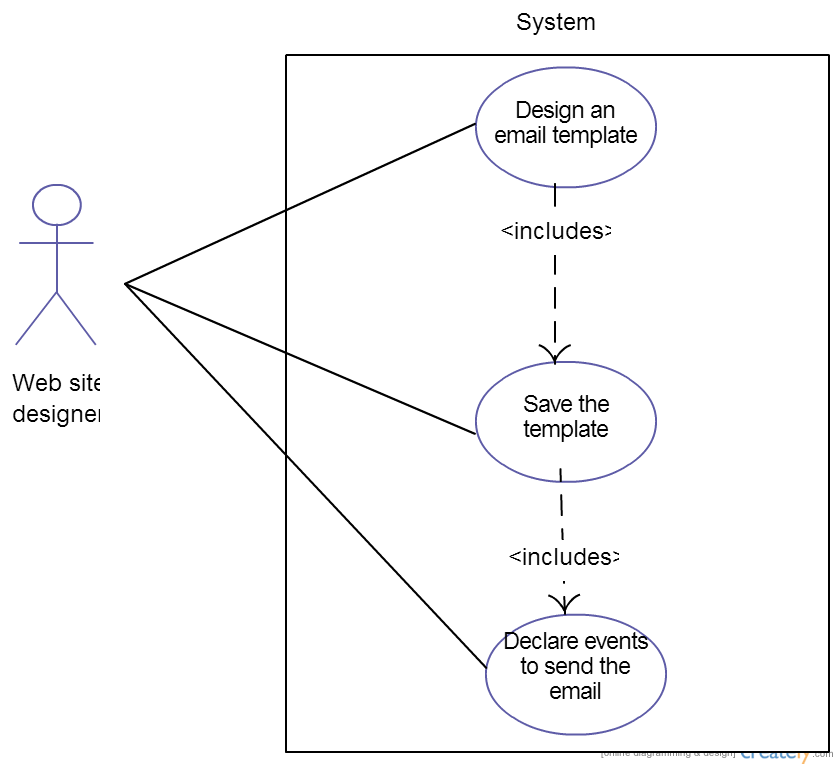
Meanwhile the code and the architecture should be simple enough to understand by the majority of FOSS developers. It should be flexible enough for future development also. Software component should be developed keeping these constraints in mind.

# Use-Case View

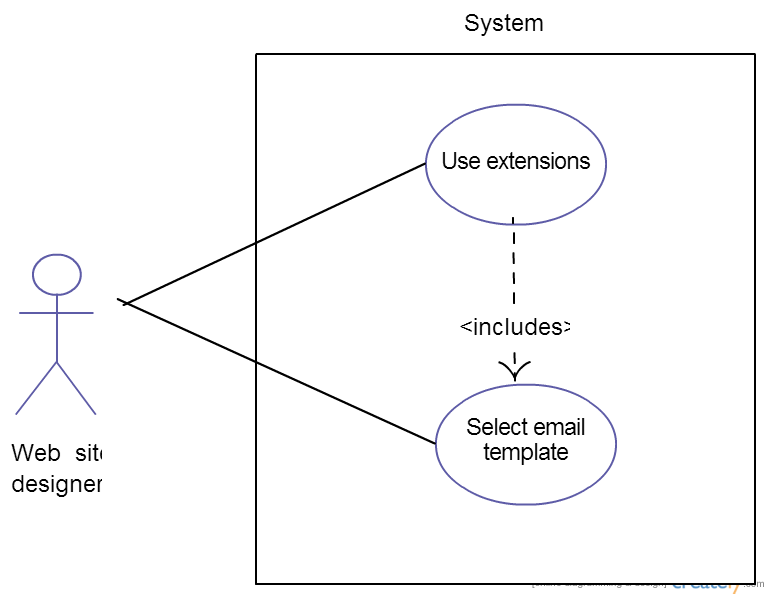
In this component basically there are only two use cases. The web site designer creates email templates and save them is the first use case. The other use case is the web site designer uses the saved email templates in his extensions.

## Use-Case Realizations

Creating and saving an email template

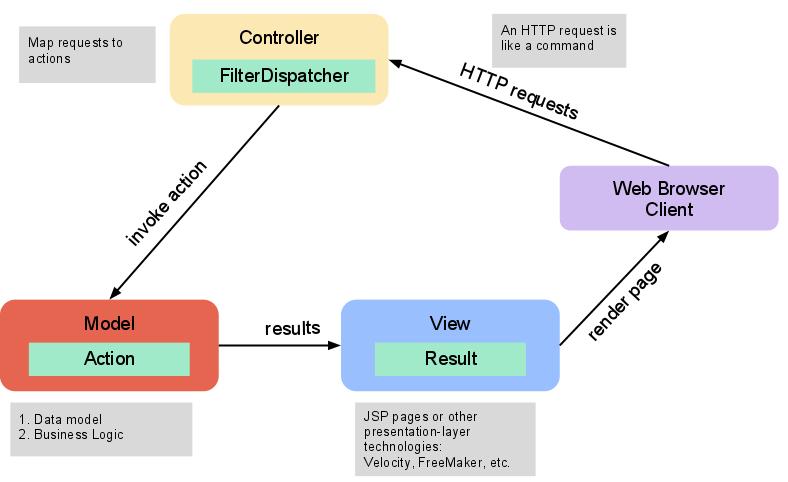


Using the saved templates in an extension



# Logical View

Joomla uses MVC pattern. In this pattern we have the following basic structure for each component.



## Overview

Each Joomla component is basically divided into two main packages. One is the site and the other is the admin package. Each of these packages manages the front end and the backend of the Joomla CMS for that specific component. Each package has the basic subcomponent packages needed by MVC pattern either it is a backend or frontend package. That is both admin and site packages has models, views and controllers classes.

## Architecturally Significant Design Packages

When developing Joomla components we have to use the in-built core classes in Joomla. Using these classes we create code segments in following parts.

* *models* They manage the data
* *controllers* They perform tasks, set and get the states of the models and ask the views to display
* *views* They display the content according to the type (*error*, *feed*, *html*, *json*, *raw*, *xml*) and the layout chosen by the controllers

In this project controllers include the following code segments.

* Email : to manipulate email objects
* Comment : to add comments for use by admin side
* Log : to keep track of email component logs
* User : to keep track of the user
* Updates : to save and update email objects

Models part includes,

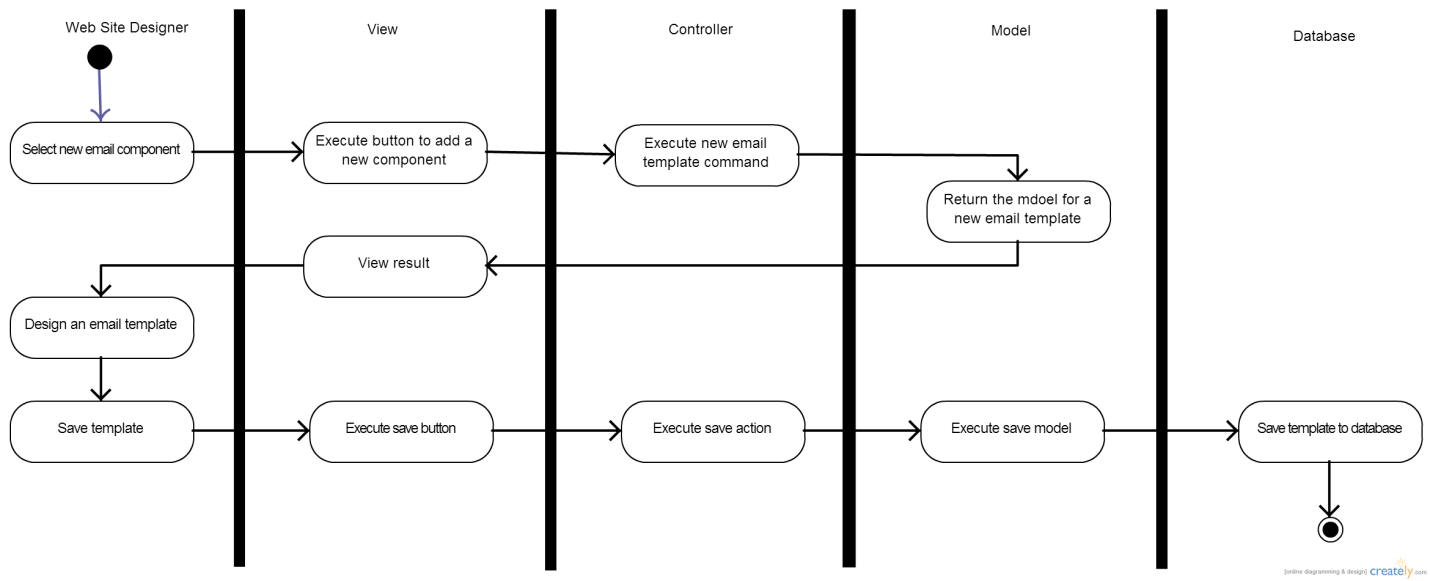
* Forms : to generate form models, which can be filled by the user
* Rules : to constraint user input
* Fields : to hold user input

Views part includes code segments with the same captions used in controllers part. These provide a user interface manipulate the controller.

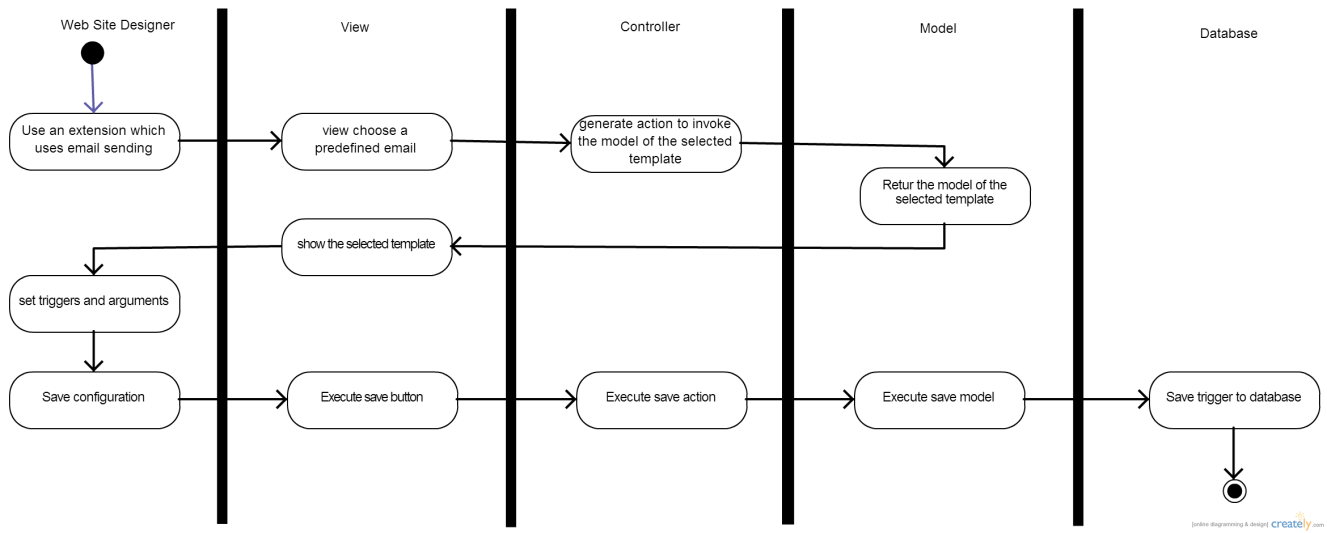
# Process View

This component really processes in a very simple way. As mentioned repeatedly above, the process can be shown in a diagram as below.

This is the process of creating a new email template.

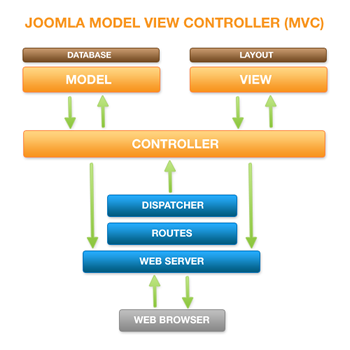


Using a predefined email template through an extension.



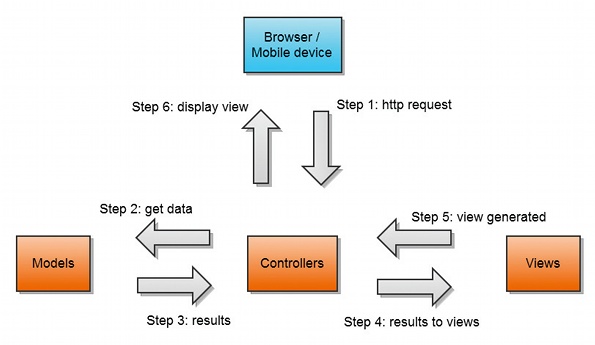
# Deployment View

This Joomla component is added to the Joomla CMS when it is deployed. Then, it runs as a usual Joomla component. Below is a diagram how the Joomla CMS is structured. The proposed component is going to spread in the orange colored parts when it is installed in the system.



# Implementation View

Implementation of the component is based on three layers. They are view, logic and data layer. Below is a diagram which shows the basic flow of a Joomla website.



# Data View

In this component predefined emails are needed to save in a database for later usage.

# Size and Performance

This component will behave as a simple usual component used in Joomla. So it would not be a very large component compared to the existing components in the Joomla CMS. At the same time it would fulfill the need of a huge amount of extensions and extra coding to settle confirmation emails in a web site.

In case of performance, the website performance will be as usual since the component triggered only when an external event is generated. The speed of the action would depend on the mail server and database access speed of the web site service provider.

# Quality

Since this is a FOSS, this should have a clear and flexible structure. A clear and well formatted code should be done following Joomla coding style and standard.