Technical Architecture Design Spec (TADS)

|  |  |  |  |
| --- | --- | --- | --- |
|  | |  | |
| Document Reference: | |  | |
| Project: | | VMS | |
| Document Title: | | Volunteer Management System | |
| Version: | | 1.0 | |
| Date: | | 08 April 2011 | |
| Author: | | Team 08 | |
|  | |  | |
|  | |  | |
|  | |  | |
|  | |  | |
|  | |  | |
|  | |  | |
|  | |  | |
|  | |  | |
| **©** 2001  The information contained in this document is the property of ISS. The contents must not be copied in whole or in part for purposes other than which it has been supplied without the consent of ISS, or, if it has been furnished under contract to another party, as expressly authorised under that contract, then ISS shall not be liable for any errors or omissions. |  |
| ***Institute of Systems Science, 25 Heng Mui Keng Terrace,  Singapore 119615*** | |

Version History Record

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Review & Approval Form Number | Date | Brief description of amendments and affected pages, paragraph |
| 1.0 | ISS/VMS | 19/03/11 | First Issue |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

1. DOCUMENT OVERVIEW 4

1.1 Goals 5

2. APPLICATION ARCHITECTURE DESIGN 6

2.1 Technical Application Architecture Overview 6

2.2 Objectives 6

2.3 Organization 6

2.4 Scope 6

2.5 Definition of Terms 7

3. OVERVIEW OF REQUIREMENTS 8

3.1 Introduction 8

3.2 System Perspective 8

3.3 System Functions 8

3.4 User Characteristics 9

3.5 General Constraints 9

4. FUNCTIONAL REQUIREMENTS 10

4.1 Processing Requirements 10

4.2 Dio: User Interface Requirement 17

5. OPERATIONAL and Quality REQUIREMENTS 18

5.1 Operating Environment 18

5.2 Development Constraints 18

5.3 Performance 18

5.4 Reliability 18

5.5 Availability 18

5.6 Capacity and Expandability 18

5.7 Security 18

5.8 Future Growth 19

# DOCUMENT OVERVIEW

This document describes the development architecture for the VMS application’s components and development patterns adopted. VMS system is used by the NGO staff and volunteers and is designed as a public web based application.

## Goals

### The following table lists the goals for the VMS.

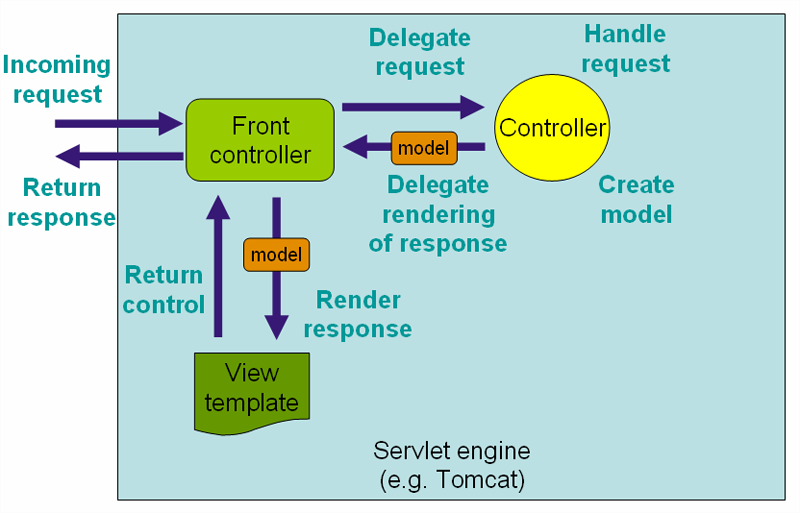
|  |  |
| --- | --- |
| **Goals** | **Indicators** |
| Reliability | The system should be able to resist failure and to recover from failure with minimum loss of data. |
| Scalability | The system can be expanded to meet user loads without significant modifications. |
| Performance | The system must be able to meet the performance required for volunteers’ management operations. |
| Modularity | Changes to individual module should have minimum impact to other modules |
| Reusability | Common system components could be reused or enhanced for systems. |

# APPLICATION ARCHITECTURE DESIGN

## Technical Application Architecture Overview



### Spring Web MVC



The multi-layered design provides the following benefits:

Development by teams can be better planned, disseminated, controlled and tested because of the logical segmentation of the system.

Complex problems can be broken into smaller more manageable pieces via the divide-and-conquer approach.

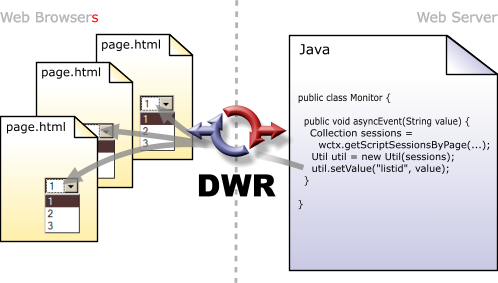
### Presentation Layer

The VMS presentation layer will consist of basic web interface method such as html java script and css. And it will enhance with YUI and DWR to support the user friendliness.

### Web Flow Layer

### DWR Architecture Overview

DWR is a RPC library which makes it easy to call backend Java functions API from JavaScript and to call JavaScript functions from Java (a.k.a Reverse Ajax).



DWR consists of two main parts. One is the Java Servlet running of the server that process the request and sends back to the browser. And the other one is the java script running in the browser that send request and can dynamically update the web page. Server side generate value such as calendar and other data selection drop down will mainly use dwr.

<bean id=*"vmsDwr"* class=*"mtech.se1008s.service.dwr.operation.RtDwr"* >

<property name=*"manager"* ref=*"baseManager"* />

<property name=*"sessionBean"* ref=*"sessionBean"* />

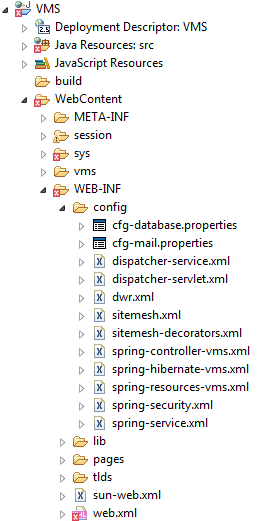
</bean>

### YUI Architecture Overview

## Development Framework

### Configuration

All the configuration files can be found under the WEB-INF/config as follows.



### Spring Configuration

#### spring-controller-vms.xml to configure all the business logic beans/ page beans and dwr beans

#### spring-hibernate-vms.xml to configure hibernate session factory

#### spring-security.xml to configure the spring security interceptor

### Email Configuration

Email configuration file (cfg-mail.properties) is to set email user that vms will be use to send email alert as follow.

sys.server.mail.send.host=vms.com.sg

sys.server.mail.send.username=volunteers

sys.server.mail.send.password=user123

sys.server.mail.send.port=25

sys.server.mail.receive.host=vms.com.sg

sys.server.mail.receive.username= volunteers

sys.server.mail.receive.password=user123

sys.server.mail.receive.port=110

### Database Configuration

cfg-database.properties is to configure the database connection parameters such as JNDI, number of cached statements, timeout of pool connection and validation SQL. The logger will save the information into the file and file will be splitted into different file based on the file format

(“vms .log.2011-03-25.log”).

### Logger Configuration

log4j.properties file can be found under the src folder. The application log flag, log level and log location configuration can be configured.

# Direct log messages to stdout

log4j.appender.stdout=org.apache.log4j.ConsoleAppender

log4j.appender.stdout.Target=System.out

log4j.appender.stdout.layout=org.apache.log4j.PatternLayout

log4j.appender.stdout.layout.ConversionPattern=[VMS] %d [%-5p] [%t] %c.%M(%-L) | %m%n

# Root logger option

#log4j.rootLogger=INFO, stdout

log4j.logger.org.springframework=ERROR

#log4j.rootCategory=error, stdout, R

log4j.rootCategory=DEBUG, stdout, R

log4j.appender.R=org.apache.log4j.RollingFileAppender

log4j.appender.R.File=logs/vms.log

log4j.appender.R.MaxFileSize=10000KB

#detail log filter

log4j.logger.mtech.se1008s.service.DispatcherServlet=ERROR

### Web Flow Configuration

### DWR Configuration

### Application Exception Handler

## Database Performance

### Connection Pool

A prefilled connection pool will be configured on the application server to ensure minimum overhead when making database call. Each connection of the pool will be authenticated via a pre-set JAAS configuration on the application server.

Below are the initial parameters:

|  |  |
| --- | --- |
| **Parameter** | **Value** |
| Prefill | Yes |
| MAX Pool size | 200 |
| Idle timeout | 300 seconds |
| Min Pool size | 20 |

### Stored Procedures / Functions

By design, the business logic is kept at the business tier. No business logic is to be coded in the data tier. At the time of design, there are no requirements that require mass movement of data or database related logic. Hence no Stored procedures / Functions will be coded in the program.

### Database Security

#### Database Connection

The application server will be pre-configured with a capable connection pool. Connections in the pool will be instantiated with reference to a JAAS configuration on the application server. The username and password will be configured and encrypted on the application server. A JNDI lookup will be made in request for a connection from the pool.

No username and password is to be kept in the program.

**Appendix**

**Hardware/Network Layout**

**Web Server 1**

|  |  |
| --- | --- |
| **IP** |  |
| **Hostname** |  |
| **Disk space** |  |
| **CPU** |  |
| **RAM** |  |
| **F/O** |  |

**Web Server 2**

|  |  |
| --- | --- |
| **IP** |  |
| **Hostname** |  |
| **Disk space** |  |
| **CPU** |  |
| **RAM** |  |
| **F/O** |  |

**Application Server 1**

|  |  |
| --- | --- |
| **IP** |  |
| **Hostname** |  |
| **Disk space** |  |
| **CPU** |  |
| **RAM** |  |
| **F/O** |  |
| **Application instance name** |  |

**Application Server 2**

|  |  |
| --- | --- |
| **IP** |  |
| **Hostname** |  |
| **Disk space** |  |
| **CPU** |  |
| **RAM** |  |
| **F/O** |  |
| **Application instance name** |  |

**Database Server 1**

|  |  |
| --- | --- |
| **IP** |  |
| **Hostname** |  |
| **Disk space** |  |
| **CPU** |  |
| **RAM** |  |
| **F/O** |  |
| **Database instance name** |  |
| **Data file** |  |

**Database Server 2**

|  |  |
| --- | --- |
| **IP** |  |
| **Hostname** |  |
| **Disk space** |  |
| **CPU** |  |
| **RAM** |  |
| **F/O** |  |
| **Database instance name** |  |
| **Data file** |  |

**Software**

|  |  |
| --- | --- |
| **Name** | **Version** |
| **IIS** | **7.5** |
| **Windows** | **2008 R2** |
| **…** | **…** |