**ARCHITECTURE & DESIGN DOCUMENT**

**FOR JUKEBOX SYSTEM**

**Revision: 3.0**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| DOCUMENT CHANGE LOG | | | | | |
| Revision | Description | Author | Date | Approved | Date |
| 1.0 | Initial Draft(1.0) | Huong Nguyen | 23-Jun-10 | <Manager > | <dd-Mmm-YY> |
| 1.1 | Update (1.1):  Introduction (I.)  Appendix A: View template | Huong Nguyen | 26-Jun-10 |  |  |
| 1.2 | Update:  (3.2.) Share d data packet 2: Deposit Money  (3.3.) Shared data packet 3: Play music | Duy Pham | 28-Jun-10 |  |  |
| 1.2 | Update:  (3.4) Shared data packet 4 : Configure Clients Volume Control  (3.5) Shared data packet 5: Adjust clients’ volume | Hong Phan | 28-Jun-10 |  |  |
| 1.2 | Update:  (3.6) Filter – pipe packet 5: List statistic | Nam Vu | 28-Jun-10 |  |  |
| 1.2 | Review& Update all updated packets above | Huong Nguyen | 28-Jun-10 |  |  |
| 1.3 | Update:  (4.1) Decomposition View Packet 1: Jukebox System  (4.2) Uses View Packet 2: Jukebox System  (5.1) Deployment View Packet 1: Jukebox system  (5.2) Implementation View Packet 2: Jukebox system  (5.3) Work Assignment View Packet 3: Jukebox system | Huong Nguyen | 29-Jun-10 |  |  |
| 1.4 | Update:  (1.2) List of Quality attributes  (2.1.) Combined View Packet 1  (3.1) C&C View Packet 1: Manage music store  (3.4) C&C View Packet 4: List statistic  (5.2.1) Deployment View Packet 2: Jukebox System  (6.) Detail Design Draft | Huong Nguyen | 13-July-10 |  |  |
| 1.5 | Update:  (3.3) C&C View Packet 3: Configure/Adjust Clients volume | Hong Phan | 13-July-10 |  |  |
| 1.6 | Update:  (3.2) C&C View packet 2: Select/Play/Adjust volume/Deposit | Duy Pham | 15-July-10 |  |  |
| 1.7 | Update:  (3.5) C&C View packet: Availability of Table side Jukeboxes | Nam Vu | 15-July-10 |  |  |
| 2.0 | Review | Huong Nguyen | 15-July-10 |  |  |
| 2.1 | Review + edit English mistake + group image components | DUY Pham | 19-July-10 |  |  |
| 2.2 | Update 3.1 (Primary presentation)  Update 3.2 (Primary presentation)  Update 6.2 (GUI) | DUY Pham | 21-July-10 |  |  |
| 2.3 | Update : 3.1, 3.2, 3.3  Remove 4.2  Update: 5.1, 5.2, 5.3 | Huong nguyen | 22-July-10  23-July-10 |  |  |
| 2.4 | Update: Class diagram | Duy Pham | 26-July-10 |  |  |
| 2.5 | Update: 3.1.2.3, 3.2.2.3 (sequence diagram for Network socket connector) | Nam Vu | 26-July-10 |  |  |
| 2.6 | Update: 2.1.4, 3.1.4, 3.2.4, 3.3.4, 3.4.4 (Rational for Design) | Huong Nguyen | 30-July-10 |  |  |
| 3.0 | Reviewed and approved | Project Team | 31-July-10 | Project Team | 31-July-10 |
|  | | | | | |

**Table of contents**

[Introduction 7](#_Toc268221962)

[1.1. Project Context 7](#_Toc268221963)

[1.2. System Context 7](#_Toc268221964)

[1.3. Purpose 8](#_Toc268221965)

[1.4. Overview 8](#_Toc268221966)

[II. Combined View 10](#_Toc268221967)

[2.1. Combined View (Deployment and Client- Server) Packet 1: Jukebox System 10](#_Toc268221968)

[2.1.1. Primary Presentation 10](#_Toc268221969)

[2.1.2. Element catalogue 10](#_Toc268221971)

[2.1.3. Context diagram 11](#_Toc268221974)

[2.1.4. Architecture Background 11](#_Toc268221983)

[2.1.5. Related view packet 12](#_Toc268221984)

[III. Dynamic View (C&C View) 12](#_Toc268221986)

[3.1. Shared – Data Packet 1: Manage music store (use cases UCA03 and UCA04) 12](#_Toc268221987)

[3.1.1. Primary Presentation 12](#_Toc268221988)

[3.1.2. Element catalog 13](#_Toc268221991)

[3.1.3. Context diagram 16](#_Toc268221992)

[3.1.4. Architecture Background 16](#_Toc268221993)

[3.2. Shared – Data Packet 3: Select/ Play Music/ Adjust Volume (use cases UCU02, UCU03, & UCU04) 17](#_Toc268221994)

[3.2.1. Primary Presentation 17](#_Toc268221995)

[3.2.2. Element catalog 18](#_Toc268221996)

[3.2.3. Context diagram 21](#_Toc268221997)

[3.2.4. Architecture Background 21](#_Toc268221998)

[3.3. Shared – Data Packet 4: Configure/Adjust clients volume control (use cases UCA05 and UCA06) 22](#_Toc268221999)

[3.3.1. Primary Presentation 22](#_Toc268222000)

[3.3.2. Element catalog 23](#_Toc268222003)

[3.3.3. Context diagram 24](#_Toc268222004)

[3.3.4. Architecture Background 25](#_Toc268222005)

[3.4. Filter and Pipe View packet 6: List Statistic (use case UCA07) 25](#_Toc268222006)

[3.4.1. Primary presentation 25](#_Toc268222007)

[3.4.2. Element catalog 25](#_Toc268222008)

[3.4.3. Context diagram 26](#_Toc268222009)

[3.4.4. Architecture Background 27](#_Toc268222010)

[3.5. Shared – Data Packet 4: Monitor status of Table side Jukeboxes (quality attribute QA01) 27](#_Toc268222011)

[3.5.1. Primary presentation 27](#_Toc268222012)

[3.5.2. Element catalog 27](#_Toc268222013)

[3.5.3. Architecture Background 30](#_Toc268222014)

[IV. Static View 30](#_Toc268222015)

[4.1. Decomposition View Packet 1: Jukebox system 30](#_Toc268222016)

[4.1.1. Primary Presentation 30](#_Toc268222017)

[4.1.2. Element catalog 31](#_Toc268222018)

[4.1.2.1 Elements and their properties 31](#_Toc268222019)

[4.1.1.2. Relationships and their properties 31](#_Toc268222020)

[4.1.3. Context Diagram 31](#_Toc268222022)

[4.1.4. Related view packets 31](#_Toc268222023)

[5. Physic View 32](#_Toc268222024)

[5.1. Deployment View Packet 1: Jukebox system 32](#_Toc268222025)

[5.1.1. Primary Presentation 32](#_Toc268222026)

[5.1.2. Element catalog 33](#_Toc268222027)

[5.1.2.1. Elements and their properties 33](#_Toc268222028)

[5.1.2.2. Relationships and their properties 33](#_Toc268222029)

[5.1.3. Context Diagram 33](#_Toc268222031)

[5.1.4. Architecture Background: NA 33](#_Toc268222032)

[5.1.4.1. Design rationale 33](#_Toc268222033)

[5.1.5. Related view packets 33](#_Toc268222034)

[5.2. Implementation View Packet 2: Jukebox system 33](#_Toc268222035)

[5.2.1. Primary Presentation 33](#_Toc268222036)

[5.2.2. Element catalog 34](#_Toc268222037)

[5.2.2.1. Elements and their properties 34](#_Toc268222038)

[5.2.2.2. Relationships and their properties 34](#_Toc268222039)

[5.2.3. Context Diagram 35](#_Toc268222041)

[5.2.4. Related view packets 35](#_Toc268222042)

[5.3. Work Assignment View Packet 3: Jukebox system 35](#_Toc268222043)

[5.3.1. Primary Presentation 35](#_Toc268222044)

[5.3.2. Element catalog 36](#_Toc268222045)

[5.3.2.1. Elements and their properties 36](#_Toc268222046)

[5.3.2.2. Relationships and their properties 36](#_Toc268222047)

[5.3.3. Context Diagram 36](#_Toc268222049)

[5.3.4. Related view packets 36](#_Toc268222050)

[6. Detail Design 36](#_Toc268222051)

[6.1. Central Jukebox Database 36](#_Toc268222052)

[6.2. Graphic User Interfaces 37](#_Toc268222053)

[6.3. Class diagram 38](#_Toc268222054)

[VII. Appendix A: View template 38](#_Toc268222056)

[7.1. Primary Presentation 38](#_Toc268222057)

[7.2. Element catalog 38](#_Toc268222058)

[7.2.1. Elements and their properties 38](#_Toc268222059)

[7.2.2. Relationships and their properties 39](#_Toc268222060)

[7.2.3. Element Interfaces 39](#_Toc268222061)

[7.2.4. Element behavior 39](#_Toc268222062)

[7.3. Context Diagram 39](#_Toc268222064)

[7.4. Variability Guide 39](#_Toc268222065)

[7.5. Architecture Background 39](#_Toc268222066)

[7.5.1. Design rationale 39](#_Toc268222067)

[7.5.2. Analysis of result 39](#_Toc268222068)

[7.5.3. Assumptions 39](#_Toc268222069)

[7.6. Other Information 39](#_Toc268222070)

[7.7. Related view packets 39](#_Toc268222071)

[Glossary 39](#_Toc268222072)

[References 40](#_Toc268222073)

# Introduction

## Project Context

It is Capstone project that requires Team to develop Virtual Jukebox System with the following constrains:

* Team size: 6 members
* The time for completing: 7 weeks
* Software Development Model assigned: Water fall
* Tools assigned: Database MySQL, Java, and JMF to play music
* Hardware: Laptops

## System Context

Based on the Requirement specification [SEGVN-Team5-RS 10], the implemented Jukebox system will provide a number of functions that help manage and play music efficiently

As Administrator role, He/she can do the following functions:

* Configure Jukebox system in Stand-alone or Distributed system
* Assign Volume control for Users
* Adjust Volume for all Users
* Manage music store (Add/Delete music song & Add genre)
* List statistic (the list of music song with the number of playing times)

As User role, He/she can do the following functions:

* Deposit money from credit card for playing music (check it by available banking system)
* Select/Play music
* Adjust Volume

Here is the list of quality attributes (2.1 of [SEGVN-Team5-RS 10])

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Quality** | **Quality ID** | **Concern** | **Use case** | **Attribute** |
| Performance | QP01 | Response time | Manage music store UCA03 | Table-side Jukeboxes automatically update changing information within 15s |
| QP02 | Delay time | UCU03 - Play music | Delay time between pressing “Play” button of user and the running of the first song is less than 15s |
| Modifiable | QM01 | Easy to configure | UCA03 - Manage music store, UCA06 - Configure clients volume control | Manage music store and configure clients volume control without restarting the system |
| QM02 | Support many music formats | UCU03 - Play music | Support three music’s formats (mp3, wma, wav) |
| Usability | QU01 | Easy to select music | UCU03 - Play music | Easy to search and select music songs filtered by song author, title, or genre and can play the excerpt of a music song |
| QU02 | Easy to setup system configuration | Deployment phase | Easy to setup system as standalone or distributed version |
| QU03 | A pleasing esthetic appearance | UCU03 - Play music | Show pictures |
| Availability | QA01 | Availability of Table side Jukeboxes | During runtime execution | The Jukebox system is always available for Table side Jukeboxes.  When detecting errors, system should send email to administrator |
| Security | QS01 | User’s credit card information | UCU01 - Deposit | Not compromise user information (credit card info) |

## Purpose

This document is intended for:

1. Project Manager and Architect to manage and evaluate if Jukebox system meets function requirements and quality requirements or not,
2. Developers to make coding
3. QA members and Testers to make Testing plan and test cases
4. One that have an interest in documentation of the architectures and design of software systems.

## Overview

Jukebox system will be described as a number of different views:

* (2.) Combined View (Deployment View & Client-Server View): Provide the whole picture of Jukebox system’s Architecture
* (3.) Dynamic view (Component and Connector View) : provide the picture of runtime entities and potential interactions for each function (implement some use cases described in Requirement specification)
  + (3.1) Shared – data view Packet 1: Manage music store
  + (3.2) Shared – data view Packet 2: Select/Play/Adjust volume
  + (3.3) Shared – data view Packet 3: Configure/Adjust clients volume
  + (3.4) Filter and Pipe view Packet 4: List statistic
  + (3.5) Share – data view Packet 5: Availability of Table side Jukeboxes
* (4.) Statistic View (Module View):
  + (4.1) Decomposition View Packet 1: Jukebox system (4.1) to describe the modular structures of a Jukebox system’s software
* (5.) Physic View (Allocation View):
  + (5.1) Deployment view Packet 1: Jukebox system (5.1) to describe mapping of the software architecture onto its environments
  + (5.2) Implementation view packet 2: jukebox system (5.2) to describe where modules are allocated
  + (5.3) Work assignment view packet 3: Jukebox system (5.3) to describe which module will be implemented and tested by whom

Each view is presented as a number of related view packets. View packet is the small cohesive bundle of information about system. View template will be described in Appendix A

* (6.) Detail Design, including:
  + (6.1) Central Jukebox Database
  + (6.2) Graphic User Interface
  + (6.3) Class diagram
  + (6.4) Sequence diagrams

Glossary and References will be listed in the end of document

# Combined View

## Combined View (Deployment and Client- Server) Packet 1: Jukebox System

### Primary Presentation

Table-side Jukeboxes (Clients: UI and Control)



Central Jukebox (Database Server)

KEY

Network connection

Tier

A B: A Request B

A B: B Response A





### Element catalogue

#### Elements and their properties

|  |  |
| --- | --- |
| **Element** | **Responsibility** |
| Central Jukebox | Database Server MySQL and File Server (music files) |
| Table side Jukeboxes | Provide UI to provide the functionality of the system:  For Administrator: Login, Logout, Manage music store, Adjust clients’ volume, Configure clients volume control, List statistic  For User: Play music, Select music, Deposit money, and Adjust volume |

#### Relationships and their properties

|  |  |
| --- | --- |
| **Element** | **Responsibility** |
| Request/response (client/server) | Via Network connection (wire or wireless) , Table side clients connect to Central Jukebox to access Music Database |



Select music

Adjust Volume

Play music

Deposit

Configure Clients Volume Control

Adjust Clients Volume

Login

Logout

Mange music store

**Jukebox system**

**Users**

**(1..n)**

**Administrator**

List statistics

**Banking**

**System**

Credit Card



### Context diagram



### Architecture Background

#### Design rationale

Jukebox System uses the Laptops and requires the distributed configuration, so Client – Server is rational. Client – Server model has some following advantages compared with Peer To Peer:

* + Modifiability:
    - It is possible to replace, repair, upgrade, or even relocate a server while its clients remain both unaware and unaffected by that change.
    - Since data storage is centralized, updates to that data are far easier to administer in comparison to a P2P paradigm. In the latter, data updates may need to be distributed and applied to each peer in the network, which is both time-consuming and error-prone, as there can be thousands or even millions of peers.
    - It functions with multiple different clients of different capabilities.
  + Security: All data is stored on the servers, which generally have far greater security controls than most clients. Server can better control access and resources, to guarantee that only those clients with the appropriate permissions may access and change data.

#### Assumptions

* The function related to Credit card is available by Banking System

### Related view packet

* Parent: None
* Children: None
* Siblings: None of this view. Other view that express the same scope are:

Decomposition View Packet 1: Jukebox system (4.1)

Uses View Packet 2: Jukebox system (4.2)

Deployment View Packet 1: Jukebox system (5.1)

Implementation View Packet 2: Jukebox system (5.2)

Work Assignment View Packet 3: Jukebox system (5.3)



# Dynamic View (C&C View)

## Shared – Data Packet 1: Manage music store (use cases UCA03 and UCA04)

### Primary Presentation

Music Dict.

Music Dict.

**KEY** Call – Return

Request – Reply (SQL-JDBC)

Request – Reply (Network socket)

Software Component

Database File Screen UI

Add new genre

Admin music store control (DB)

Central Database

(Music Dictionary + Configuration)

Auto update

(each 10 s)

Add genre/Add or Delete song

Music Dictionary

Configuration

Add song UI

Add genre UI

Delete song UI

Brow Music files

Write Music files stination

Select/Play Music UI

Add new song

Delete song

Music source

Music destination

Write music Dict.

Music Store Control (DB)

Read music Dict.

Write Configuration

Configure VolumeControl (DB)

Read Config.



### Element catalog

#### Elements and their properties

|  |  |  |
| --- | --- | --- |
| **Element** | **Type** | **Description** |
| Add song UI | Screen UI | Allow the Administrator specify where the music source is and select a song to add and specify its attributes (author, title, genre) |
| Delete song UI | Screen UI | Allow the administrator select a song to delete |
| Add genre UI | Screen UI | Allow the Administrator to specify a new genre to add |
| Admin music store control (DB) | Software Component | This component can:  Brow music files in Central Jukebox  Write selected music files to the given destination in Central Jukebox  Read music dict. from Database  Write music dict. to Database |
| Central Database | Database | Where available music dictionary and Configuration are stored |
| Music source | Driver/Directory where music source files are stored | Where “Music store control” will read music files (mp3, wav, wma ) in Central Jukebox |
| Music Destination | Driver/Directory where available music files are stored | Where “Music store control” will store available music files for the user to play |
| Auto update | Software Component | This component each 10s:   * + - 1. Forces “Configure Volume Control”, and “Music Store Control” to load Music Dictionary and Configuration from Database (in Central Jukebox) into Music Dictionary and Configuration (in Table side Jukebox)       2. Force “select/play music songs UI” to update data in “Music Dictionary” and “Configuration” |
| Music Dictionary | Object | Where Music dictionary is stored in Table side Jukebox |
| Configuration | Object | Where Configuration is stored and in Table side Jukebox |
| Select/Play music UI  (List 10 the newest music songs) | Screen UI | Display the list of 10 newest music songs |
| Music Store Control(DB) | Software Component | This component to read music Dict. from Database (central Jukebox) into music Dictionary (Table side Jukebox) |
| Configure Volume Control (DB) | Software Component | This component to read Configuration from Database (central Jukebox) into Configuration (Table side Jukebox) |
| Server Listener | Network socket connector (Brow Music files and Write Music Files) | To listen the requests from Table side Jukebox |
| File Brower | Network socket connector (Brow Music File and Write Music Files) | It can:   * To brow drivers/directories/music files in Central Jukebox * To copy music file chosen by the user to the location specified in Configuration table in Central Jukebox |

#### Relationships and their properties

The relation of this view is attachment, dictating how components and connectors are attached to each other. The relations are shown in the primary presentation; there are no additional ones

#### Element behavior:

1. Add genre

Admin music store control

Add genre

Read genres

Music Dictionary

Add new genre

Read genres

1. Add song

Add song

Admin music store control

Music Dictionary

Music source

Music Destination

Read music files

Write music files

Write song Attributes (file name, tile,

Author, genre, singer)

Read genres

2.1) Select Music File

List Music File From folder

Add Song

Music store control

Request MusicList (path)

Server Listener

Open Connection

File Browser

create

2.2)Copy Music File

Write music file

Add Song

Music store control

Request copy file

Server Listener

Close Connection

File Browser

1. Delete song

Delete

song

Admin music store control

Music Dictionary

Read Music songs

Delete Music songs

### Context diagram

View Packet described in this section

Select music

Adjust Volume

Play music

Deposit

Configure Clients Volume Control

Adjust Clients Volume

Login

Logout

Mange music store

**Jukebox system**

**Users**

**(1..n)**

**Administrator**

List statistics

**Banking**

**System**

Credit Card

### Architecture Background

#### Design rationale

Design decisions:

* All music files are stored in Central Jukebox’s disks
* Central Music dictionary is stored in Central Jukebox’s Database (MySQL)
* Each 10s central music dictionary and Volume Configuration will be updated in Table side Jukebox from Central Jukebox

The rationale for this decomposition is to further divide into 3 software components: GUI, Control (to connect with Server), and Auto update:

* The Producer (Create/update music store in Server) is separated from the Consumer (read music store from Server into client) that will promote concurrent communication.
* GUI (for presenting data) is separated from Control (for connecting Database and music files in Server) that will promote modifiability

Why our design meets quality attributes will be explained in the following table

|  |  |  |  |
| --- | --- | --- | --- |
| **Quality** | **Quality ID** | **Concern** | **Response measure** |
| Performance | QP01 | Response time | table-side Jukeboxes automatically update within 15s |
| **Rationale**: Auto Update is the timer that each 10s will read music store in Server and write to Clients | | | |

## Shared – Data Packet 3: Select/ Play Music/ Adjust Volume (use cases UCU02, UCU03, & UCU04)

### Primary Presentation

Central Database

(Played Music history + User Account + Music Dictionary + Configuration)

Music Dictionary

Configuration

Music store control (DB)

User Account control (DB)

Adjust Volume UI

Select Music UI

Load Configuration

Select Song

Play Music UI

Buy/Play demo

Music destination

Read

Music Files

Read Available Songs

Deposit Money UI

Deposit Money

Pend/withdraw money

Call-return

Request-Response (SQL-JDBC)

Request-Response (Network Socket)

Software component

Database

Music files

Screen UI

**KEY**

Play music control (JMF + DB)

Configure Volume Control (DB)

Read music file name

Read Volume Value

/Music file location

Add music History

Set

Volume

Value

### 

### Element catalog

#### Elements and their properties

|  |  |  |
| --- | --- | --- |
| **Element** | **Type** | **Description** |
| Select music UI | Screen UI | Allow Jukebox user to select music song(s) that he wants to play |
| Play music UI | Screen UI | Allow Jukebox user to ask system to buy selected music song(s) or Play demo |
| User music store control (DB) | Software component | Read all available songs from Database (in Central Jukebox) into Music Dictionary (in Table side Jukebox) and write into Music Dictionary (in Table side Jukebox) |
| Central Database  (Music Dictionary + User Account + Configuration + Music History) | Database | Where all available music songs, user Account, Configuration, and music history are stored in Central Jukebox, |
| Music destination | File | Where all available music files are stored (mp3, wav, wma ) in Central Jukebox |
| Music Dictionary | Object | Where available music, selecting, and playing songs are stored in Table side Jukebox |
| Configuration | Object | Where Configuration (user permission, volume value, and music file location) is stored in Table side Jukebox |
| Configure Volume Control (DB) | Software component | Check permission of the user in Configuration from Database (in central Jukebox) |
| Play music control (DB) | Software component | Play music songs based on given Playing songs (in Music Dictionary) and Volume value (in Configuration)  Then add Played music history into Database (in central Jukebox) |
| Adjust Volume UI | Screen UI | Allow the user to adjust Volume and update this volume value in Configuration in Table side Jukebox (if the user has this permission) |
| Deposit Money UI | Screen UI | Allow Jukebox user to specify amount he/she wants to deposit from credit card |
| User Account Control (DB) | Software component | Load balance/Deposit/Pend/Withdraw money for the user account |
| Sever Listener | Network Socket connector (Read Music file) | To listen requests from Table side Jukebox |
| File Loader | Network Socket connector  (Read Music file) | To load music file from Central Jukebox |

#### Relationships and their properties

The relation of this view is attachment, dictating how components and connectors are attached to each other. The relations are shown in the primary presentation; there are no additional ones

#### Element behavior:

1. Adjust Volume

Configuration

Configuration

Adjust Volume UI

Configure Volume Control (DB)

Is user permission

Adjust volume value

Play music control (JMF + DB)

1. Select Music

Select Music UI

Music Store Control (DB)

Music Dictionary

Available Music Songs

Load Available Songs

Write Available Songs

Select Song

1. Play Music

Play Music UI

Play Music Control (JMP + DB)

Configuration

Music Dictionary

Played Music History

User Account

Music Files

User Account Control (DB)

Buy songs

Pend money

Read music file name

Read Volume Value, music location

Read music file

Play music file

User Account

Withdraw money from user account

Add music History

3.1)Read Music File

Music Files

Play Music Control

Request Music File (filename)

Server Listener

Open Connection

File Loader

create

Read file

1. Deposit Money

Deposit

Money UI

User account control (DB)

User account

Deposit money (cardnumber, amount)

Balance

Cardno: Hashing(cardnumber)

Deposite (cardno, amount)

### Context diagram

View Packet described in this section

Select music

Adjust Volume

Play music

Deposit

Configure Clients Volume Control

Adjust Clients Volume

Login

Logout

Mange music store

**Jukebox system**

**Users**

**(1..n)**

**Administrator**

List statistics

**Banking**

**System**

Credit Card

### Architecture Background

#### Design rationale

The rationale for this decomposition is to further divide into 4 software components: GUI, Control (to connect with Server), Music dictionary (client’s memory), and Play music control (JMF):

* Presenting data is separated from connecting Database will promote modifiability
* Selecting music songs is separated from playing music songs by using local memory (Music Dictionary) will promote concurrency. It will support the user to select a new song while another song is playing

Why our design meets the required quality attribute will be explained in the following table

|  |  |  |  |
| --- | --- | --- | --- |
| **Quality** | **Quality ID** | **Concern** | **Response measure** |
| Performance | QP02 | Delay time | Delay time between pressing “Play” button of user and the running of song less than 15s. |
| **Rationale**: Music songs will be played based on stream based technology using network socket in TCP with no limit, while in UDP, size is limited by 64 kilobytes on datagrams. And all available data are read immediately in the same order in which they are received.  Network socket technology also provides speed better than RMI technology that require overhead to implement | | |
| Modifiability | QM02 | Support many music formats | Support many music’s formats (mp3, wma, wav) |
| **Rationale**: JMF (Java Media Framework) supports playing .mp3, .wma, and .wav extension files | | |
| Usability | QU01 | Easy to select music | Easy select music by author, title, or genre and can play the excerpt of a music song |
| **Rationale**: We support user to play 10s of selected song as demonstration before buying it. | | |
| QU03 | A pleasing esthetic appearance | Show picture |
| **Rationale**: Show colorful background image for digital jukebox | | |
| Security | QS01 | User’s credit card information | Not compromise user information |
| **Rationale** : Account number will be created by one way hashing of Credit card number, so it will ensure that user information is not compromised | | |

## Shared – Data Packet 4: Configure/Adjust clients volume control (use cases UCA05 and UCA06)

### Primary Presentation

Music Dict.

Music Dict.

**KEY** Call – Return

Request – Reply (SQL-JDBC)

Software Component

Database Screen UI

Enable/Disable Volume Control

Configure Volume Control

Central Database

(Music Dictionary + Configuration)

Auto update (each 10 s)

Controlling Permission/Value of VC

Read Music Dict

Music Dictionary

Configuration

Configure Volume UI

Configure Volume UI

Select/Play Music UI

Adjust Client’s Volume Control

Music Store Control (DB)

Configure Volume Control (DB)



### Element catalog

#### Elements and their properties

|  |  |  |
| --- | --- | --- |
| **Element** | **Type** | **Description** |
| Configure Volume UI | Screen UI | Allow the Administrator to configure/adjust clients’ volume control |
| Configure Volume Control | Software Component | This component can:   * Assign the permission of controlling volume control to clients * Adjust clients’ volume control. |
| Central Database (Volume Configuration) | Database | Where available music dictionary and Configuration are stored |
| Auto update | Software Component | This component each 10s:   1. Forces “Configure Volume Control”, and “Music Store Control” to load Music Dictionary and Configuration from Database (in Central Jukebox) into Music Dictionary and Configuration (in Table side Jukebox) 2. Force “select/play music songs UI” to update data in “Music Dictionary” and “Configuration” |
| Music Dictionary | Object | Where Music dictionary is stored in Table side Jukebox |
| Configuration | Object | Where Configuration is stored and in Table side Jukebox |
| Select/Play music UI  (List 10 the newest music songs) | Screen UI | Display the list of 10 newest music songs, and Volume Value |
| Music Store Control(DB) | Software Component | This component to read music Dict. from Database (central Jukebox) into Music Dictionary (Table side Jukebox) |
| Configure Volume Control (DB) | Software Component | This component to read Configuration from Database (central Jukebox) into Configuration (Table side Jukebox) |

#### Relationships and their properties

The relation of this view is attachment, dictating how components and connectors are attached to each other. The relations are shown in the primary presentation; there are no additional ones

#### Element behavior:

* **Assign permission of controlling volume control to clients:**

Configure Volume Control UI

Configure Volume Control

Central Database (Configuration)

Update control permission

Read control permission

* **Adjust clients’ volume control:**

Configure Volume Control UI

Configure Volume Control

Central Database (Configuration)

Update clients’ volume value permission

Read control Permission

### Context diagram

Select music

Adjust Volume

Play music

Deposit

Configure Clients Volume Control

Adjust Clients Volume

Login

Logout

Mange music store

**Jukebox system**

**Users**

**(1..n)**

**Administrator**

List statistics

**Banking**

**System**

Credit Card

View Packet described in this section

### Architecture Background

#### Design rationale

The rationale for this decomposition is to further divide into 3 software components: GUI, Control (to connect with Server), and Auto Update that will promote modifiability and concurrent communication (see 3.1.4.1 for the same reason)

Why our design meets the required quality attribute will be explained in the following table

|  |  |  |  |
| --- | --- | --- | --- |
| **Quality** | **Quality ID** | **Concern** | **Response measure** |
| Modifiable | QM01 | Easy to configure | Without restarting system |
| **Rationale**: Each 10 s, Volume configuration will be updated in Table side Jukebox by Update auto. So it will ensure that all configuration stored in Database will be updated in all Table side Jukeboxes | | | |

## Filter and Pipe View packet 6: List Statistic (use case UCA07)

### Primary presentation

Statistic Control

Central Database

(Playing Music History)

**KEY**

call-return

Request-Reply (SQL-JDBC: Call stored procedure in Database)

Statistic UI

Software Component Screen UI Database

### Element catalog

#### Elements and their properties

|  |  |  |
| --- | --- | --- |
| **Element** | **Type** | **Description** |
| Statistic UI | Screen UI | Allow the administrator:   * Chooses the filter conditions to list music songs played by the users * Gets the statistic results |
| Statistic Control | Software component | Receive the request from Statistic Interface, call stored procedure to query data from Central Database (Music Played music History) |
| Central Database  (Played Music History) | Database | Where Playing Music History (records of music song played by Jukebox Users) is stored in Central Jukebox . |

#### Elements behavior

Statistic UI

Statistic

Control

Playing Music History

Read playing songs (From date, To date)

### Context diagram

View Packet described in this section

Select music

Adjust Volume

Play music

Deposit

Configure Clients Volume Control

Adjust Clients Volume

Login

Logout

Mange music store

**Jukebox system**

**Users**

**(1..n)**

**Administrator**

List statistics

**Banking**

**System**

Credit Card

### Architecture Background

#### Design rationale

The rationale for this decomposition is to further divide into 2 software components: GUI, and Control (connect to Server). It will promote modifiability.

The rationale for calling stored procedure to receive report on music history in Statistic Control component:

* Precompiled execution: MySQL complies each stored procedure once and then reutilizes the execution plan. This result in tremendous performance boots when stored procedure are called repeatedly
* Reduced client/server traffic: If network bandwidth is a concern, stored procedure can reduce long SQL queries to a single line that transmitted over Network connection
* Efficient reuse of code and programming abstraction: stored procedure can be used by multiple users and client programs
* Enhance security controls: Allow grant users permission to execute the stored procedure independently of underlying table permissions

Why our design meets the requirement will be explained as below:

* Every time a music is played, this information will be stored in Music Playing Historical Data
* Music Playing Historical is stored in the central database
* The result is based on the number of time the music playing, not relate with money.

## Shared – Data Packet 4: Monitor status of Table side Jukeboxes (quality attribute QA01)

### Primary presentation

Automatic Update Control

Central Database

(Client Status)

**KEY**

Request-Reply (SQL-JDBC)

Automatic Monitor Control

### Element catalog

#### Elements and their properties

|  |  |  |
| --- | --- | --- |
| **Element** | **Type** | **Description** |
| Automatic Update Control (DB) | Software Component | This control is activated by the timer in Table side Jukebox. Every 10s, it’s activated and update its status in to database |
| Automatic Monitor Control (DB) | Software component | This control is activated by the timer of Central Jukebox. Every 30 minutes, it will connect to database to request all clients’ status. If there’s some timeout-status, it will alert administrator via email. |
| Central Database  (Clients Status) | Database | Store the status of Table side Jukeboxes. |

#### Elements behavior

* **Update Status:**

**ALT:** if not exist

Automatic update Control (DB)

ClientsStatus

queryStatus

Insert into new status

Update status

openConnection

closeConnection

* **Monitor status:**

**LOOP:** has another client-status

Calculate time

Send alert email

**ALT:** if is time-out

Automatic Monitor Control

ClientsStatus

Query All Client Status

Get a client-status

openConnection

closeConnection

Update Client-status

### Architecture Background

#### Design rationale

Why our design meets the requirement will be explained as below:

|  |  |  |  |
| --- | --- | --- | --- |
| **Quality** | **Quality ID** | **Concern** | **Response Measure** |
| Availability | QA01 | Availability of Table side Jukeboxes | The Jukebox system is always available for Table side Jukeboxes. If there is a problem an alert Email will be sent to Administrator |
| **Rationale:**   * Every 10s, the Automatic Update Control will connect to database-server to request some new information (music dictionary, volume-status) and update its status to database. The information will be updated: TIME (connect to server). There’s 2 case:   + Just turn-on client: a new record will be inserted   + Is running-client: update current status (time) * Every 30 minutes, the Automatic Monitor Control will connect to database-server to request all running-client’s status. If there’s any timeout status the alert email will be sent to administrator, and this control will mark this client has been alerted.   Timeout status = (current time – time stamp in ClientsStatus) > 5 minutes | | | |

# Static View

# Decomposition View Packet 1: Jukebox system

## Primary Presentation

|  |  |
| --- | --- |
| **System** | **Module** |
| **Jukebox System** | User Main UI |
| Deposit money UI |
| Select/Play music UI |
| Adjust Volume UI |
| List Statistic UI |
| Add Genre UI |
| Add Song UI |
| Delete Song UI |
| Admin Main UI |
| Configure Volume UI |
| Admin Music Store Control (DB) |
| Music Store Control (DB) |
| List Statistic Control (DB) |
| User Account Control (DB) |
| Play Music Control (JMF + DB) |
| Configure Volume Control (DB) |
| Music Dictionary |
| Configuration |
| Auto update |
| Automatic Update Control (DB) |
| Automatic Monitor Control (DB) |
| Music File Loader (Network socket connector) |
| Server Listener (Network socket connector) |
| File Brower (Network socket connector) |

## Element catalog

### Elements and their properties

All elements are described in the above view packets

### Relationships and their properties

The relation type of concerns in this view is *is-part-of*. Every module is part of exactly one Jukebox system



## Context Diagram

See context diagram in Combined View Packet 1: Jukebox system (Section 2.1.3.)

## Related view packets

* Parent: None
* Children: None
* Siblings: None of this view. Other view that express the same scope are:

Combined View Packet 1: Jukebox system (2.1)

Uses View Packet 2: Jukebox system (4.2)

Deployment View Packet 1: Jukebox system (5.1)

Implementation View Packet 2: Jukebox system (5.2)

Work Assignment View Packet 3: Jukebox system (5.3)

# Physic View

# Deployment View Packet 1: Jukebox system

## Primary Presentation

Network connection

Auto update

User Account Control (DB)

Play music Control (JMF+DB)

Configure

Volume Control (DB)

List Statistic Control (DB)

Central Database

Table side Jukebox

Central Jukebox

Music Dictionary

Configuration

Music Files Source

Deposit UI

Select /Play

UI

Adjust

Volume UI

Add genre

UI

Add song

UI

Delete

song

UI

Configure

Volume UI

List

statistic UI

User

Main

UI

Admin

Main UI

Music Store Control (DB)

Admin Music Store Control (DB)

Automatic Update Control

(DB)

Automatic Monitor Control (DB)

Server Listener

Music File Brower

Music File Loader

Music Files Destination

## Element catalog

### Elements and their properties

The same as Decomposition View Packet 1: Jukebox system

### Relationships and their properties

The relation type of concerns in this view is *execution migrates to.* An element is *execution migrates to* a computer



## Context Diagram

See context diagram in Combined View Packet 1: Jukebox system (Section 2.1.3.)

## Architecture Background: NA

### Design rationale

Why our design meet the required quality attributes will be explained in the following table

|  |  |  |  |
| --- | --- | --- | --- |
| **Quality** | **Quality ID** | **Concern** | **Response measure** |
| Usability | QU02 | Easy to setup of system configuration (standalone or distributed) | Can setup in standalone or distributed mode at deployment time |
| **Rationale:** At deployment time, the software components will be installed as below:   1. Standalone: all software components must be installed in Central Jukebox 2. Distributed: Software components will be installed in Central Jukebox and Table side Jukeboxes as shown in 5.1.1 | | | |

## Related view packets

* Parent: None
* Children: None
* Siblings: None of this view. Other view that express the same scope are:

Combined View Packet 1: Jukebox system (2.1)

Decomposition View Packet 1: Jukebox system (4.1)

Uses View Packet 2: Jukebox system (4.2)

Implementation View Packet 2: Jukebox system (5.2)

Work Assignment View Packet 3: Jukebox system (5.3)

# Implementation View Packet 2: Jukebox system

## Primary Presentation

|  |  |  |  |
| --- | --- | --- | --- |
| **System** | **Module** | **Where stored** | **Files** |
| **Jukebox System** | User Main UI | Table side Jukebox | .\Interface\userMainInterface |
| Deposit money UI | Table side Jukebox | .\Interface\userDepositMoneyInterface |
| Select/Play music UI | Table side Jukebox | .\Interface\userSelectMusicInterface |
| Adjust Volume UI | Table side Jukebox | .\Interface\userVolumeControlInterface |
| List Statistic UI | Table side Jukebox | .\Interface\listMusicInterface |
| Add Genre UI | Table side Jukebox | .\Interface\userPlayMusicControl |
| Add Song UI | Table side Jukebox | .\Interface\adminAdjustClientVolumeInterface |
| Delete Song UI | Table side Jukebox | .\Interface\adminConfigureClientVolumeInterface |
| Admin Main UI | Table side Jukebox | .\Interface\adminManageMusicStoreInterface |
| Configure Volume UI | Table side Jukebox | .\Interface\adminAddGenreInterface |
| Admin Music Store Control (DB) | Table side Jukebox | .\Business\adminMusicStoreControlDB |
| Music Store Control (DB) | Table side Jukebox | .\Business\musicStoreControlDB |
| List Statistic Control (DB) | Table side Jukebox | .\ Business \statisticDB |
| User Account Control (DB) | Table side Jukebox | .\Business\userAccountControlDB |
| Play Music Control (JMF + DB) | Table side Jukebox | .\Business\playMusicControlDB |
|  | Configure Volume Control (DB) | Table side Jukebox | .\ Business \volumeConfigurationDB |
|  | Music Dictionary | Table side Jukebox | .\Business\MusicDictionary |
|  | Configuration | Table side Jukebox | .\ Business \Configuration |
|  | Auto update | Table side Jukebox | .\Business\autoUpdate |
|  | Automatic Update Control (DB) | Table side Jukebox | .\Business\autoUpdateControlDB |
|  | Automatic Monitor Control (DB) | Central Jukebox | .\autoMonitorControlDB |
|  | Music File Loader | Central Jukebox | .\musicFileLoader |
|  | Server Listener | Central Jukebox | .\serverListener |
|  | File Brower | Central Jukebox | .\fileBrower |

## Element catalog

### Elements and their properties

The same as Decomposition View Packet 1: Jukebox sytem

### Relationships and their properties

The relation type of concerns in this view is *allocated to.* An element is *allocated to* computer (in files)



## Context Diagram

See context diagram in Combined View Packet 1: Jukebox system (Section 2.1.3.)

## Related view packets

* Parent: None
* Children: None
* Siblings: None of this view. Other view that express the same scope are:

Combined View Packet 1: Jukebox system (2.1)

Decomposition View Packet 1: Jukebox system (4.1)

Uses View Packet 2: Jukebox system (4.2)

Deployment View Packet 1: Jukebox system (5.1)

Work Assignment View Packet 3: Jukebox system (5.3)

# Work Assignment View Packet 3: Jukebox system

## Primary Presentation

|  |  |  |
| --- | --- | --- |
| **System** | **Module** | **Members** |
| **Jukebox System** | User Main UI | Duy Pham |
| Deposit money UI | Duy Pham |
| Select/Play music UI | Duy Pham |
| Adjust Volume UI | Huong Nguyen |
| List Statistic UI | Huong Nguyen |
| Add Genre UI | Duy Pham |
| Add Song UI | Duy Pham |
| Delete Song UI | Duy Pham |
| Admin Main UI | Duy Pham |
| Configure Volume UI | Huong Nguyen |
| Admin Music Store Control (DB) | Duy Pham |
| Music Store Control (DB) | Duy Pham |
| List Statistic Control (DB) | Huong Nguyen |
| User Account Control (DB) | Duy Pham |
| Play Music Control (JMF + DB) | Nam Vu |
| Configure Volume Control (DB) | Huong Nguyen |
| Music Dictionary | Duy Pham |
| Configuration | Huong Nguyen |
| Auto update | Nam Vu |
| Automatic Update Control (DB) | Nam Vu |
| Automatic Monitor Control (DB) | Nam Vu |
| Music File Loader | Nam Vu |
| Server Listener | Nam Vu |
| File Brower | Nam Vu |

## Element catalog

### Elements and their properties

The same as Decomposition View Packet 1: Jukebox system

### Relationships and their properties

The relation type of concerns in this view is *allocated to .*An element *is allocated to* members to implement



## Context Diagram

See context diagram in Combined View Packet 1: Jukebox system (Section 2.1.3.)

## Related view packets

* Parent: None
* Children: None
* Siblings: None of this view. Other view that express the same scope are:

Combined View Packet 1: Jukebox system (2.1)

Decomposition View Packet 1: Jukebox system (4.1)

Uses View Packet 2: Jukebox system (4.2)

Deployment View Packet 1: Jukebox system (5.1)

Implementation View Packet 2: Jukebox system (5.2)

# Detail Design

## 6.1. Central Jukebox Database

Image2

## Graphic User Interfaces



## Class diagram

## 



# Appendix A: View template

## Primary Presentation

It shows the elements and their relationships that populate the view packet

## Element catalog

### Elements and their properties

List elements with their properties

Depend on the view type, properties will be different, for example, elements in a module decomposition view have the property of “responsibility”. in Shared data view have the properties “type, description”, and in Deployment view have the properties “Processing speed MIPS, Communication speed MBPS”…

### Relationships and their properties

List all relationships and their properties

### Element Interfaces

Interface is a boundary across which elements interact or communicate with each other. This section is where element interfaces are documented

### Element behavior

Some elements have complex interactions with environment and for the purpose of understanding or analysis, the element’s behavior is documented (can use sequence diagram)



## Context Diagram

IT shows how the system depicted in the view packet relates to its environment

## Variability Guide

It shows how to exercise any variation points that are a part of the architecture shown in the view packet

## Architecture Background

It explains why the design reflected in the view packet came to be

### Design rationale

It explains why the design decisions reflected in the view packet were made and gives a list of rejected alternatives and why they were rejected

### Analysis of result

This documents the results of analyses that have been conducted, such as the results of performance or security analyses, or a list of what would have to change in the face of a particular kind of system modification

### Assumptions

This documents any assumptions the architect made when crafting the design

## Other Information

This section includes non architectural and organization specific information

## Related view packets

This section will name other view packets that are related to the one being described in a parent/child or sibling capacity

# Glossary

**Architecture style** a specialization of element and relation types, together with a set of constrains on how they can be used

**Combined view** a view that contains elements and relationships that come from two or more other views

**Components** the principal computational elements and data stores that execute in a system

**Connector** a runtime pathway of interaction between two or more components

**Context diagram** a representation of what’s in and what’s out of the system under consideration and the external entities with which the system interacts

**Interface** a boundary (GUI) across which system’s users interact or communicate with

**Layer** a collection of code that forms a virtual machine and interacts with other layers only according to predefined roles

**Tier** a mechanism for system partitioning. Usually applied to client-server-based systems, where the various parts (tiers) of the system (user interface, database, business application logic…) execute on the different platform

**View** a representative of a set of system elements and the relationships among them

**View packet** the smallest cohesive bundle of documentation that you would give to a stakeholder

**View type** the element types and relation types used to describe the architecture of a software system from a particular perspective

# References

[Anthony 09] Anthony J. Lattanze, 2009 “Architecting software intensive systems. A Practitioner’s Guide”

[Bass 03] Len Bass, Paul Clements, Rick Kazman, 2003, “Software Architecture in Practice”

[Clements 02] Paul Clements, Felix Bachmann, Len Bass, 2002 “Documenting Software Architectures”

[Erich 00] Erich Gamma, Richard Helm, [Ralph Johnson](http://www.amazon.com/s/ref=ntt_athr_dp_sr_3?_encoding=UTF8&sort=relevancerank&search-alias=books&field-author=Ralph%20Johnson), [John M. Vlissides](http://www.amazon.com/s/ref=ntt_athr_dp_sr_4?_encoding=UTF8&sort=relevancerank&search-alias=books&field-author=John%20M.%20Vlissides) 2000, “Design Patterns. Elements of Reusable Object – Oriented Software”

[Mary 96] Mary Shaw, David Garlan 1996, “Software Architecture. Perspectives on an emerging discipline”

[SEGVN-Team5-RTM 10] SEGVN Team 5, 2010, “Requirement Traceability Matrix for Jukebox System V2.0”

[SEGVN-Team5-RS 10] SEGVN Team 5, 2010 “Requirement Specifications for Jukebox System V3.0”

[SEGVN-Team5-DD 10] SEGVN Team 5, 2010 “Database Design for Jukebox System V3.0”