MUSIC RECOMMENDER SYSTEM

PLOYPHET CHIRATCHAYAWAT

RATTIDA WONGSAKTHAWORN

A SPECIAL PROJECT SUBMITTED IN PARTIAL FULFILLMENT

OF THE REQUIREMENT FOR THE DEGREE OF BACHELOR OF SCIENCE

IN COMPUTER SCIENCE (INTERNATIONAL PROGRAM)

FACULTY OF SCIENCE

KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

ACADEMIC YEAR 2011

Thesis Title	Music Recommender System	
Students	Miss Ployphet Chiratchayawat	Student ID: 51051187
	Miss Rattida Wongsakthaworn	Student ID: 51051212
Degree	Bachelor of Science	
Major Program	Computer Science (International Program)	
Academic Year	2011	
Thesis Advisor	Assoc.Prof.Dr.Veera Boonjing	

ABSTRACT

Music Recommender System application was developed to the android operating system for recommending songs to users using two main techniques; the Multicriteria and the Multidimension techniques. Our system works by introducing a song obtaining from the system's database. When the user logs into the system, the system will evaluate and generate a recommended song list for this particular user according to the system's criteria.

.

ACKNOWLEDGEMENTS

This special project Music Recommender System would not have been possible without the help from many people.

Special thanks to our parents for their unconditional loves and supports. Thanks for cheering us up, teaching us and comforting us when we confronted with the problems.

Special thanks to Assoc.Prof.Dr.Veera Boonjing, who gave good advice and be guidance of this project since start until successful.

Special thanks to Assoc.Prof.Dr.Jeeraporn Werapun, who gave a good suggestion on our application. And thanks to Dr. Rungrat Wiangsripanawan for a suggestion on the interface of application.

Last but not least, special thanks to all our friends, and specially Pakorn Phatikornwong for his help on network technique.

Ploypheet Chiratchayawat

Rattida Wongsakthaworn

Contents

Contents (Cont.)

Contents (Cont.)

Contents (Cont.)

Contents (Cont.)

Contents (Cont.)

List of Tables

List of Figures

List of Figures (Cont.)

List of Figures (Cont.)

Chapter 1

Introduction

1.1 Rational and Research motivation

Recommender system is a automatic system that suggests data for users. This system will evaluate from user's behavior, history, profile and appropriate data to each user. For the most of online services, recommender system is important because of a huge data that user can't check all of data before make decision. Recommender system helps users to make decision more easier by recommend data that users interested so it make user more convenience. But some system cannot get the correct result for user, for example sometime system result is unrelated to user's interest. So in this case we use the dataset of music to collect the rate and interest of each user and calculate by filtering of our system to display the most nearly result and meet the user's requirement.

1.2 Object

From dataset of music, recommender system collect the rate and interest of each users and display the result that related and appropriate to each user. By evaluate the information from user behavior and calculate by filtering of our system to display the most nearly result for each user. And make this system more effective by use on android mobile device.

1.3 Scope

1) Study about recommender system

2) Study how to use Collaborative Filtering and Content-based Filtering together

3) Study how to develop Android

4) Software

4.1) Android Developer

Eclipse

4.2) Interfaces Designer

Adobe Photoshop CS5

5) Hardware

5.1) Personal computer or notebook that support software

1.4 Organization

This project consists of 5 chapters

Chapter 1: Introduction

Chapter 2: Background

Chapter 3: Music Recommender System with Collaborate Filtering and Content-based Filtering

Chapter 4: Experiment of this special project

Chapter 5: Conclusion of this special project

Chapter 2

Background

2.1 Recommender System

A recommender system is a system that recommended contents of the service for users. Each of recommender system used the different techniques for the suitable of their data and system. The recommender system can be separated into 4 part :

1. Data for process such as user's profile
2. Input which is user give to system such as rating that user rated to the system. There are 2 kinds of rating :
   - Explicit : represent in term of numerical according to level of rate 1-5, 1-10 or other form depend on usability.
   - Implicit : the information of user's consumption behavior.
3. Part of algorithms which is the most important part that use to process for recommend to users.
4. The last one is the part of presentation to users, and there are 2 types :
   - Top N Recommendation : represent N data that nearest with the user's requirements
   - Predicted Value : system will represent the data with the rating that system have predicted.

The recommender system that focuses on user's previous consumption behavior, comments, or satisfaction for calculate is called ***Recommender System with Collaborative Filtering (CF)***. On the other hands the system that focuses on the content's properties or descriptions is called ***Recommender System with Content-based Filtering(CB)***. These 2 kinds of system have the different strong point and weak point. The content-based filtering uses the content's properties or descriptions to recommended users. But collaborative filtering uses the

rating that user rated on system to calculate and try to find the other user on system who have the most similar satisfaction then recommended the contents that user interested in to each other. Furthermore, there is the ***Hybrid Recommender System*** which is the system that combined the strong points of collaborative filtering and content-based filtering together.

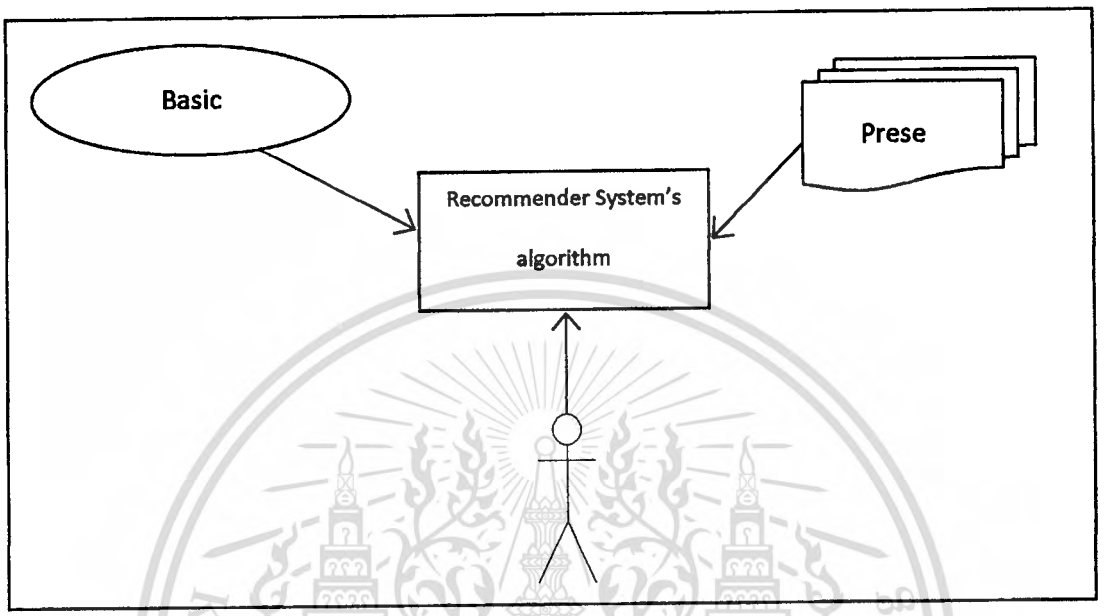


Figure 2.1 *Basic of Recommender System*

2.2 Collaborative Filtering (CF)

Collaborative filtering consider the preference of user that prefer to contents on system with the other users in system who does not prefer those contents yet, but have the similar contents that user ever prefer. The technique prefer to use with collaborative filtering is ***Similarity Measurement***. System will measure the similarity of user's preference of all of users in the system to used for predict the ***Preference Prediction***. Means that the system should convert the preference value into numerically value for calculate. As there are many preferences so the system converted them in term of vector. Then weight the nearest-neighbor with the similarity value of user or content to predict. This technique can use user reference, called ***Item-based Collaborative Filtering***, or content reference, called ***User-based Collaborative Filtering***, or use both of them.

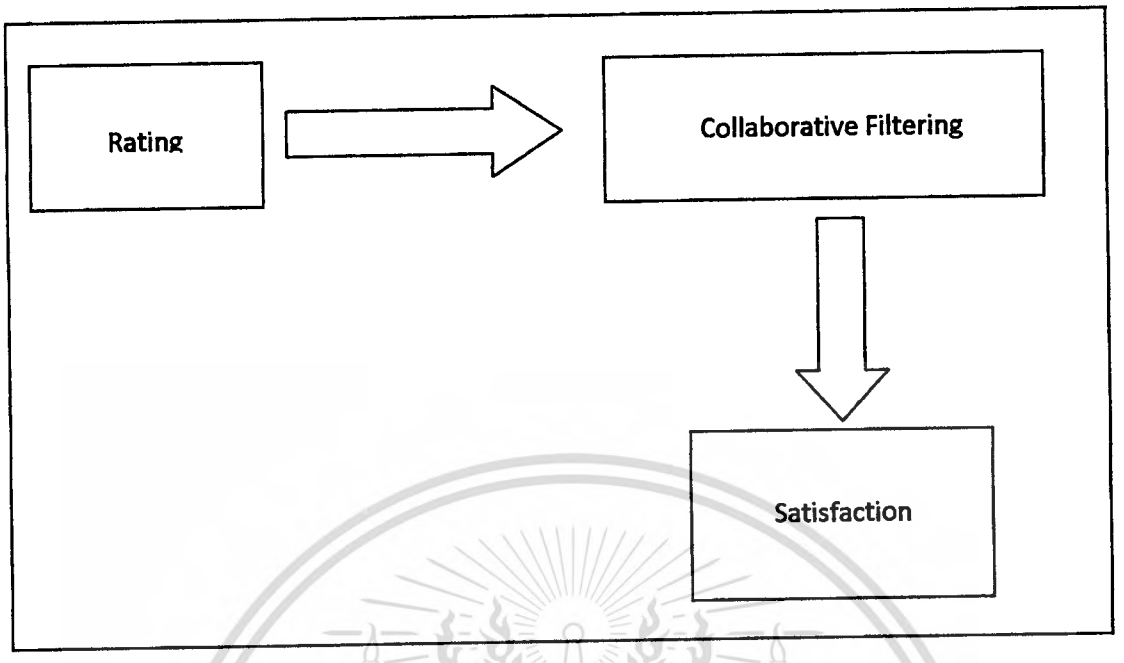


Figure 2.2 Collaborative Filtering Process

2.2.1 Item-based Collaborative Filtering

Item-based Collaborative Filtering is the technique try to find the similar items for each item. The system looks into the set of items the target user has rated and computes how similar they are to the target item. The prediction is computed by taking a weighted average of the target user's ratings on those similar items. The similarity computation can be got offline because the similarity between items is more stable than users.

2.2.2 User-based Collaborative Filtering

User-based Collaborative Filtering is the another technique that use with the recommender system. This technique compute nearest-neighbor set of active user through comparing similarity with other users according to input ratings data. Then, can predict the missing data according to the nearest-neighbor set, and get recommendation set.

2.2.3 Weak points of CF

The important weak points of this technique are ***Cold Start Problem*** and ***First Rater Problem***. Because CF uses rating that user rated in system to calculate but when new user get into the system, system cannot recommend any content to them. In the similar case, if there are any content into the system, they also will not recommended to any user too. And another important weak point is ***Sparsity Problem*** which is the case that there is a few rating in the system maybe users in system do not rated the content enough for calculate. In the other hands, the content is not rated by any user for compare with the similarity value.

2.2.4 Strong points of CF

Different contents in the system can be recommended to users even if they never use it before. Because system will compare them with other users who ever used those contents and have similar preference by measurement of similarity value. So the system can recommended to users many ways.

2.3 Content-based Filtering

This technique focuses on content's information or description. System will recommend contents which are the same or similar with content that users ever used or rated. The information of contents should be the data that can learn by the computer process only.

2.3.1 Weak points of CB

Details of user and content that can use with CB should be the data that can learn by computer process only. Then use the information by ***Information Retrieval*** technique for compare the similarity and difference between user and content. If that content depend on opinion of user, such as joke, this technique cannot distinguish which one is joke or not. The result from this kind of technique has the *overspecialization*.

2.3.2 Strong points of CB

Even if the contents never used or rated by users before, they can be recommended to users. Also include the contents which is the new into the system. Because the system will consider by the nearest content.

2.4 Hybrid Recommender System

This kind of system combined the strong points of both of collaborative filtering and content-based filtering together. And there are many way to combined the CF and CB :

2.4.1 Weighed Hybrid Recommender System

Weighed Hybrid Recommender System is the system that sum up preferences that predicted from each of technique in term of *linear combination*. And those preferences could be calculated in the linear combination system and complaint with each other.

2.4.2 Switching Hybrid Recommender System

Switching Hybrid Recommender System will choose the method to predict. Problem of this system is the criterion that use for decide the method. Because quality of prediction depended on situation and criterion that system chose.

2.4.3 Mixed Hybrid Recommender System

This system will separate the preference calculation to each method. The process will combine all preference of each method together.

2.4.4 Feature Combination Hybrid Recommender System

Feature Combination Hybrid Recommender System uses the properties of on recommender system to be an input for another recommender system for prediction.

2.4.5 Feature Augmentation Hybrid Recommender System

Feature Augmentation Hybrid Recommender System is similar to feature combination hybrid recommender system. But this system use new data to be an input.

2.4.6 Cascade Hybrid Recommender System

Is the system that separate in to 2 system, major system and minor systems In case of the predicted value in major system are equal, minor system will used for value configuration.

2.4.7 Meta-Level Hybrid Recommender

Whole of major and minor system will predicted the value in the same time. Then they choose the best one to use for system.

2.5 Recommender System with Multicriteria Rating

Recommender System that use until now still find only one value from rating which user rated to content. So the neighbor which system used for recommend to user may not appropriate.

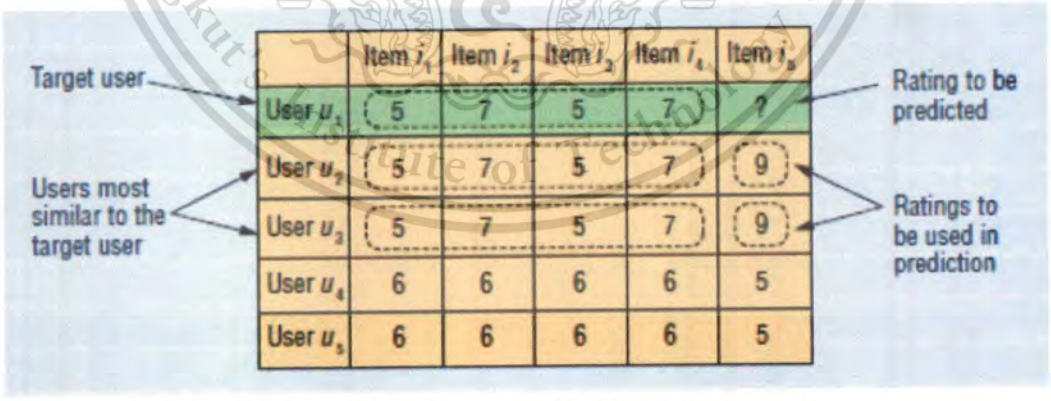


***Figure 2.3** Rating Prediction of today Recommender System*

From *figure 2.3* user U_1 need the value of item i_5, after consideration, user U_2 and U_3 have rated item i_1-i_4 in the same way with user U_1 so the system that used now a day will predict the value for item i_5 of user U_1 is 9 same with user U_2 and U_3. On the other hand, if the value was increased just only 1 point make user U_1 may not have the value like the system predicted.

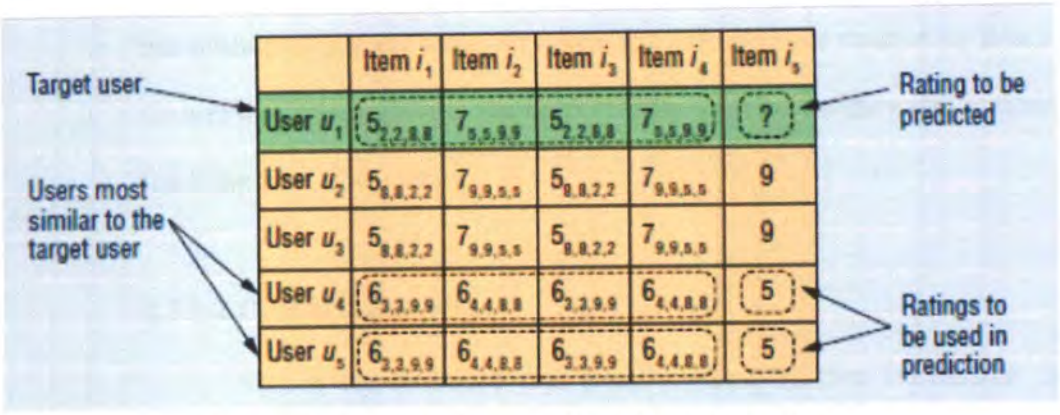


Figure 2.4 *Rating Prediction with Multicriteria Rating*

From *figure 2.4*, preference value of each property of item $i_1 - i_4$ of user U_1 absolutely difference from user U_2 and U_3. But user U_1 has preference value of each property of item $i_1 - i_4$ nearby with user U_4 and U_5. So rating prediction of system should be the same with user U_4 and U_5. We called this method ***Multicriteria Rating***.

2.6 Recommender System with Multidimension

Three of recommender system, Content-based, Collaborative and Hybrid, always consider with the traditional two-dimensional, user and item. So for recommendation, must have the dataset of preference value of each item of user.

2.7 Recommender System Technique

Recommender system uses many techniques to adapt with the system idea.

2.7.1 Techniques in Collaborative Filtering

There are 3 techniques that use in recommender system with collaborative filtering

2.7.1.1 Search as Filter

This technique is the easiest way to recommended to user. When user input *key word* to search the content, system will list all of contents in system

that relate with that key word. So system will give the same result every time if the uses the same key word. And never change, adapt or configure with contents that bring from other users.

2.7.1.2 Clustering as Filter

This is the technique that group contents by their information or descriptions. In this technique users should rated to system first then system will recommend group of contents that match with users preferences.

2.7.1.3 TF-IDF

TF-IDF is the most popular technique to use in the information searching. Idea is any word or phrase that represent in description of many contents those of word or phrase is the best to use for explain the contents. Sometime word or phrase that often represent in description can be used in many contents.

2.7.2 Techniques in Content-based Filtering

There are 2 techniques that use in content-based filtering

2.7.2.1 Memory-based Collaborative Filtering

This is the process that uses all of data in the system or some data to predict. The data that use in the system is the preference of users or contents that have the similarity to predict, can called *Neighborhood-based*. The system will calculate the similarity of users or contents. Then weight with the similarity from *Similarity Measurement* which has 3 ways to compute :

- Correlation-Based Similarity
- Vector Cosine-Based Similarity
- Adjusted Cosine Similarity

2.7.2.2 Model-based Collaborative Filtering

Model-based Collaborative Filtering is the process that system uses all of data in the system or some data to predict. The data that use in the system is the preference of users or contents that have the similarity to predict. Then system will create the model from user's preferences by *Learning Mechanism*, such as Bayesian network, Clustering, and Rule-based System etc, to recommend to users.

2.8 Android

2.8.1 Why Android?

Google announced the Open Handset Alliance and the Android platform in November of 2007, releasing he first beta version of the Android Software Development Kit (SDK) at the same time. Within a matter of a few months, over 1 million people had downloaded versions of the SDK from Google's website. In the United States, T-Mobile announced the G1 Android mobile phone in October of 2008, and estimates are that several hundred thousand G1s were sold before the end of that year. There are already several competing mobile phone software stacks in the market.

Android has the potential for removing the barriers to success in the development and sale of a new generation of mobile phone application software. Just as the standardized PC and Macintosh platforms created markets for desktop and server software, Android, by providing a standard mobile phone application environment, will create a market for mobile applications—and the opportunity for applications developers to profit from those applications.

Android gives developers a way to develop unique, creative applications and get those applications in the hands of customers. Hundreds of thousands of Android mobile phone users are already there, looking for the next clever or useful application.

2.8.2 The Open Handset Alliance

Google and 33 other companies announced the formation of the Open Handset Alliance on November 5, 2007. According to the joint press release from that day:

> *This alliance shares a common goal of fostering innovation on mobile devices and giving consumers a far better user experience than much of what is available on today's mobile platforms. By providing developers a new level of openness that enables them to work more collaboratively, Android will accelerate the pace at which new and compelling mobile services are made available to consumers.*

For user who as mobile application developers, that means developers are free to develop whatever creative mobile applications they can think of, free to market them (or give them, at their option) to Android mobile phone owners, and free to profit from that effort any way we can. Each member of the Open Handset Alliance has its own reasons for participating and contributing its intellectual property, and they are free to benefit.

2.8.3 Android Architecture

Android Architecture is separated into ***Layer***. Each of layer will called the services from below layer. There are important 4 layers

1) Linux Kernel
2) Library
3) Application Framework
4) Application

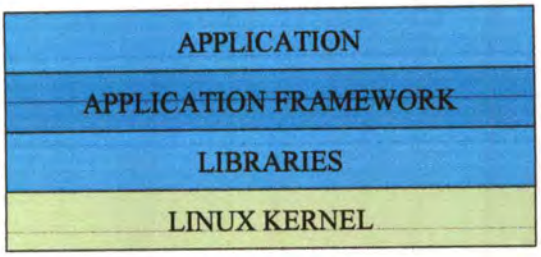


***Figure 2.5** Android Architecture*

2.8.3.1 Application

This layer is the top of the *Android Structure* that is the part of application which developed for use. Program in this layer must be in form of .apk file.

***Figure 2.6** Application Layer Architecture*

2.8.3.2 Application Framework

Android was designed for support case of developer use android by ***API (Application Programming Interface).*** To decrease the duplicate called of *Application Component*. The examples of application framework:

2.8.3.2.1 View System

View System is the part that control works of application building

2.8.3.2.2 Location Manager

Location Manager is the part that control location of devices.

2.8.3.2.3 Content Provider

Control accessibility of *Share Data* between different applications.

2.8.3.2.4 Resource Manager

Handle the accessibility of any data that does not use the code.

2.8.3.2.5 Notification Manager

Control all of events that showed on *Status bar*.

2.8.3.2.6 Activity Manager

Control the *Life Cycle* of application

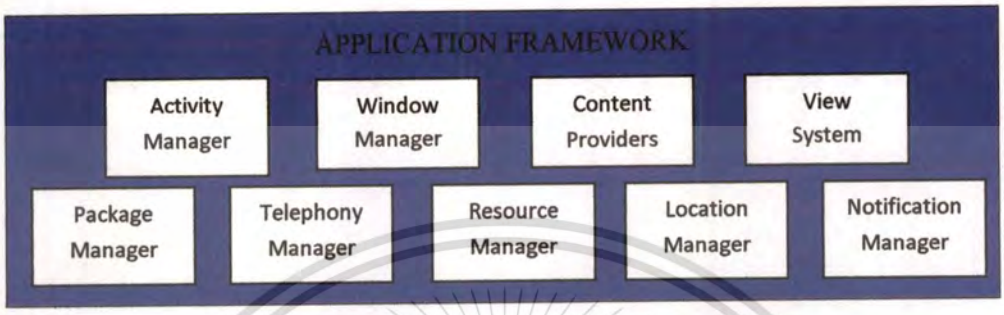


***Figure 2.7** Application Framework Layer Architecture*

2.8.3.3 Library

Android amass groups of libraries that important and necessary for developing program. The libraries were written in C and C++ language. Examples of libraries:

2.8.3.3.1 System C library

Group of basic libraries based on basic C language library (libc)

2.8.3.3.2 Media Libraries

Group of task that related with multimedia.

2.8.3.3.3 Surface Manager

Group of interface management, interface drawing.

2.8.3.3.4 2D/3D library

Group of 2D graphic or SGL (Scalable Graphics Library) and 3D graphic or OpenGL.

2.8.3.3.5 FreeType

Group of *Bitmap* and *Vector* for Image Render.

2.8.3.3.6 SQLite

Group of *Database* that is the same with Firefox and Apple iPhone. And also developer can use this database to collect the data of applications.

2.8.3.3.7 Browser Engine

Group of displaying on web browser based on *Webkit* which is similar to Google Chrome, Safari and Nokia s60

Layers in *Library* must call the upper level to use. In Library, Android has the minor part called Android Runtime which is separated into 2 parts:

- Dalvik VM (Virtual Machine)

This part was specifically written in *Java* language for use with devices. However the difference of ***Java VM (Virtual Machine)*** and ***Dalvik VM*** is Dalvik VM can run *.dex file* which is compiled from *.class file* and *.jar file*. Then Java class will compress by the tool named ***dx***.

- Core Java Library

The part that contained the basic libraries but differ from library of Java SE (Java Standard Edition) and Java EM (Java Mobile Edition).

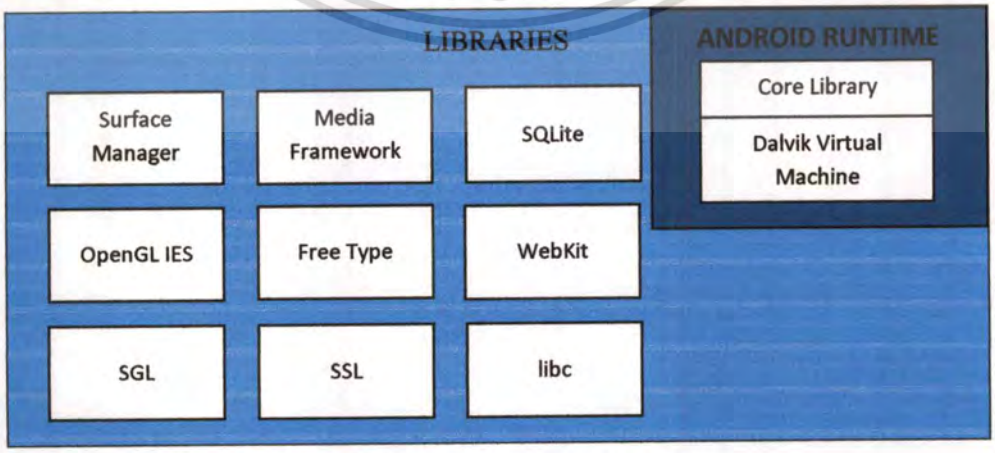


Figure 2.8 *Library Architecture*

2.8.3.4 Linux Kernel

Android system based on *Linux OS*. There are many functions in this level. Each of function was developed in form of C language. The developers cannot directly access to this part but they can access with the ***Command Prompt***.

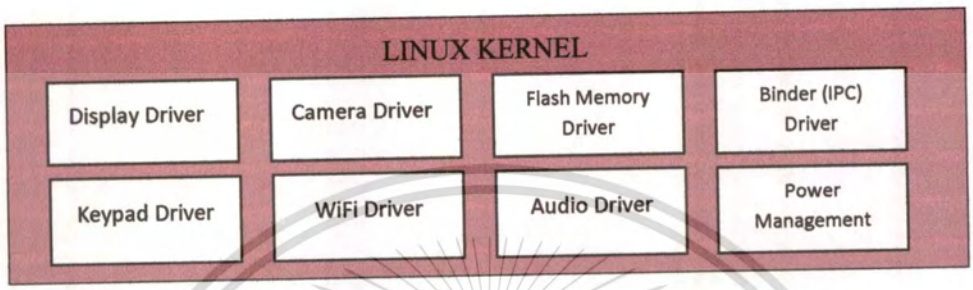


Figure 2.9 *Linux Kernel Architecture*

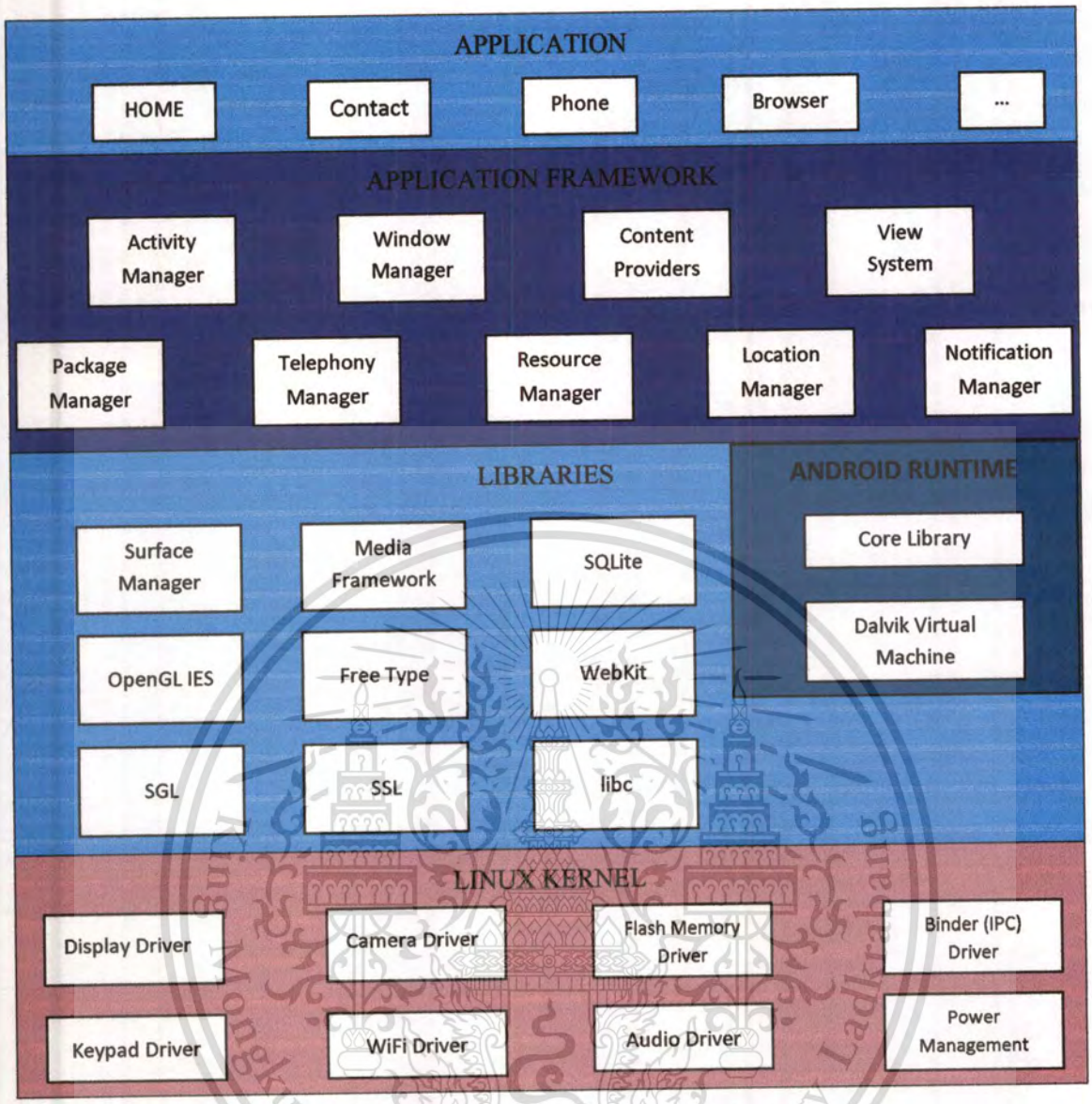


Figure 2.10 Android Architecture

2.8.4 Application Component

Advantage of android is the application can be a component of others. So it is make the developing easy and fast.

Application Component separated into 4 parts:

2.8.4.1 Activity

Activity or commonly known as *User Interface* is the page that communicated with users. Each of application can have more than 1 page or 1 activity and each of activity will collect the using status of each part. Examples:

- Developers can assign how to display the menu list, with picture or description, etc.
- For the Send Message application, there is activity that show list of message communication. Another one is activity for select the communicator. The other will be the page for older message, etc.

2.8.4.2 Service

Service is the *Background Process* that support user to use application. Examples:

- Service which played the music during user used other part of application or other application.
- Connect to Network.
- Calculated values of each activity, etc.

2.8.4.3 Broadcast and Intent Receiver

These respond to requests for service from another application. A *Broadcast Receiver* responds to a system-wide announcement of an event. These announcements can come from devices such as battery low, or from any program running on the system. An Activity or Service provides other applications with access to its functionality by executing an *Intent Receiver*, a small piece of executable code that responds to requests for data or services from other activities.

2.8.4.4 Content provider

This part is the part of data for each of application. It is created to share data with other activities or services that can be a files system or Database. Example:

- Google can access to share data with user in the application which required user's data, etc.

An application doesn't have to use all of the Android components, but a well-written application will make use of the mechanisms provided, rather than reinventing functionality or hard-coding references to other applications.

2.8.5 Android Activity Lifecycle

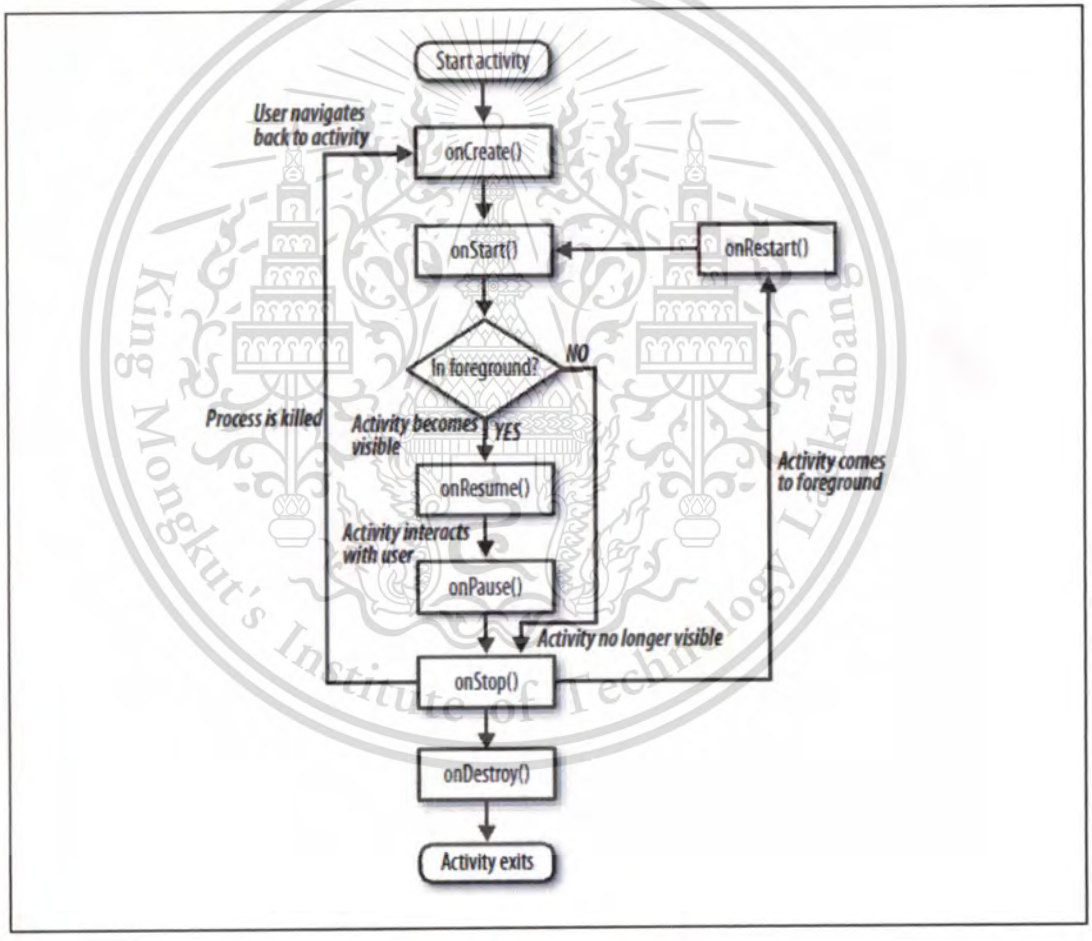


Figure 2.11 *Activity Life Cycle*

Android is designed around the unique requirements of mobile applications. In particular, Android recognizes that resources such as memory and battery, are limited on most mobile devices, and provides mechanisms to conserve those resources. The

mechanisms are clear in the Android Activity Lifecycle, which defines the states or events that an activity goes through from the time it is created until it finishes running. The lifecycle is shown diagrammatically in Figure 2.9.

Application's activity monitors and reacts to these events by example methods that override the Activity class methods for each event:

2.8.5.1 onCreate

Called when the activity is first created. This is the place developer normally create views, open any persistent datafiles their activity needs to use, and in general initialize the activity. When calling **onCreate**, the Android framework is passed a **Bundle** object that contains any activity state saved from when the activity ran before.

2.8.5.2 onStart

Called just before your activity becomes visible on the screen. Once **onStart** completes, if your activity can become the foreground activity on the screen, control will transfer to **onResume**. If the activity cannot become the foreground activity for some reason, control transfers to the **onStop** method.

2.8.5.3 onResume

Called right after **onStart** if your activity is the foreground activity on the screen. At this point your activity is running and interacting with the user. You are receiving keyboard and touch inputs, and the screen is displaying your user interface. **onResume** is also called if your activity loses the foreground to another activity, and that activity eventually exits, popping your activity back to the foreground. This is where your activity would start (or resume) doing things

that are needed to update the user interface (receiving location updates or running an animation, for example).

2.8.5.4 onPause

Called when Android is just about to resume a different activity, giving that Activity the foreground. At this point the activity will no longer have access to the screen, so developer should stop doing things that consume battery and CPU cycles unnecessarily. If they are running an animation, no one is going to be able to see it, so they might as well suspend it until they get the screen back. Their activity needs to take advantage of this method to store any state that they will need in case their activity gains the foreground again—and it is not guaranteed that the activity will resume. If the mobile device they are running on runs out of memory, there is no virtual memory on disk to use for expansion, so the activity may have to make way for a system process that needs memory. Once should exit this method, Android may kill the activity at any time without returning control to them.

2.8.5.5 onStop

Called when the activity is no longer visible, either because another activity has taken the foreground or because the activity is being destroyed.

2.8.5.6 onDestroy

The last chance for the activity to do any processing before it is destroyed. Normally developer'd get to this point because the activity is done and the framework called its finish method. But as mentioned earlier, the method might be called because Android has decided it needs the resources the activity is consuming.

2.8.6 Android Service Lifecycle

The lifecycle for a service is similar to that for an activity, but different in a few important details:

2.8.6.1 onCreate and onStart differences

Services can be started when a client calls the **Context.startService(Intent)** method. If the service isn't already running, Android starts it and calls its **onCreate** method followed by the **onStart** method. If the service is already running, its **onStart** method is invoked again with the new intent. So it's quite possible and normal for a service's **onStart** method to be called repeatedly in a single run of the service.

2.8.6.2 onResume, onPause, and onStop are not needed

Recall that a service generally has no user interface, so there isn't any need for the **onPause**, **onResume**, or **onStop** methods. Whenever a service is running, it is always in the background.

2.8.6.3 onBind

If a client needs a persistent connection to a service, it can call the **Context.bind Service** method. This creates the service if it is not running, and calls **onCreate** but not **onStart**. Instead, the **onBind** method is called with the client's intent, and it returns an **IBind** object that the client can use to make further calls to the service. It's quite normal for a service to have clients starting it and clients bound to it at the same time.

2.8.6.4 onDestroy

As with an activity, the **onDestroy** method is called when the service is about to be terminated. Android will terminate a service when there are no more

clients starting or bound to it. As with activities, Android may also terminate a service when memory is getting low. If that happens, Android will attempt to restart the service when the memory pressure passes, so if your service needs to store persistent information for that restart, it's best to do so in the **onStart** method.

2.9 World Wide Web

The **World Wide Web** abbreviated as **WWW** and commonly known as **the Web** is a system of interlinked hypertext documents accessed via the Internet. With a web browser, one can view web pages that may contain text, images, videos, and other multimedia and navigate between them via hyperlinks.

2.9.1 History

In the June 1970 issue of *Popular Science* magazine Arthur C. Clarke was reported to have predicted that satellites would one day "bring the accumulated knowledge of the world to your fingertips" using a console that would combine the functionality of the Xerox, telephone, television and a small computer, allowing data transfer and video conferencing around the globe.

In March 1989, Tim Berners-Lee wrote a proposal that referenced ENQUIRE, a database and software project he had built in 1980, and described a more elaborate information management system. With help from Robert Cailliau, he published a more formal proposal (on November 12, 1990) to build a "Hypertext project" called "WorldWideWeb" as a "web" of "hypertext documents" to be viewed by "browsers" using a client–server architecture. This proposal estimated that a read-only web would be developed within three months and that it would take six months to achieve "the creation of new links and new material by readers, authorship becomes universal" as well as "the

automatic notification of a reader when new material of interest to him/her has become available." While the read-only goal was met, accessible authorship of web content took longer to mature, with the wiki concept, blogs, Web 2.0 and RSS/Atom.

The proposal was modeled after the Dynatext SGML reader by Electronic Book Technology, a spin-off from the Institute for Research in Information and Scholarship at Brown University. The Dynatext system, licensed by CERN, was technically advanced and was a key player in the extension of SGML ISO 8879:1986 to Hypermedia within HyTime, but it was considered too expensive and had an inappropriate licensing policy for use in the general high energy physics community, namely a fee for each document and each document alteration.

A NeXT Computer was used by Berners-Lee as the world's first web server and also to write the first web browser, WorldWideWeb, in 1990. By Christmas 1990, Berners-Lee had built all the tools necessary for a working Web: the first web browser ; the first web server; and the first web pages, which described the project itself. On August 6, 1991, he posted a short summary of the World Wide Web project on the alt.hypertextnewsgroup. This date also marked the debut of the Web as a publicly available service on the Internet. The first photo on the web was uploaded by Berners-Lee in 1992, an image of the CERN house band Les Horribles Cernettes.

The first server outside Europe was set up at SLAC to host the SPIRES-HEP database. Accounts differ substantially as to the date of this event. The World Wide Web Consortium says December 1992, whereas SLAC itself claims 1991. This is supported by a W3C document entitled *A Little History of the World Wide Web*.

The crucial underlying concept of hypertext originated with older projects from the 1960s, such as the Hypertext Editing System (HES) at Brown University, Ted Nelson's Project Xanadu, and Douglas Engelbart's oN-Line System (NLS). Both Nelson and Engelbart were in turn inspired by Vannevar Bush's microfilm-based "memex", which was described in the 1945 essay "As We May Think".

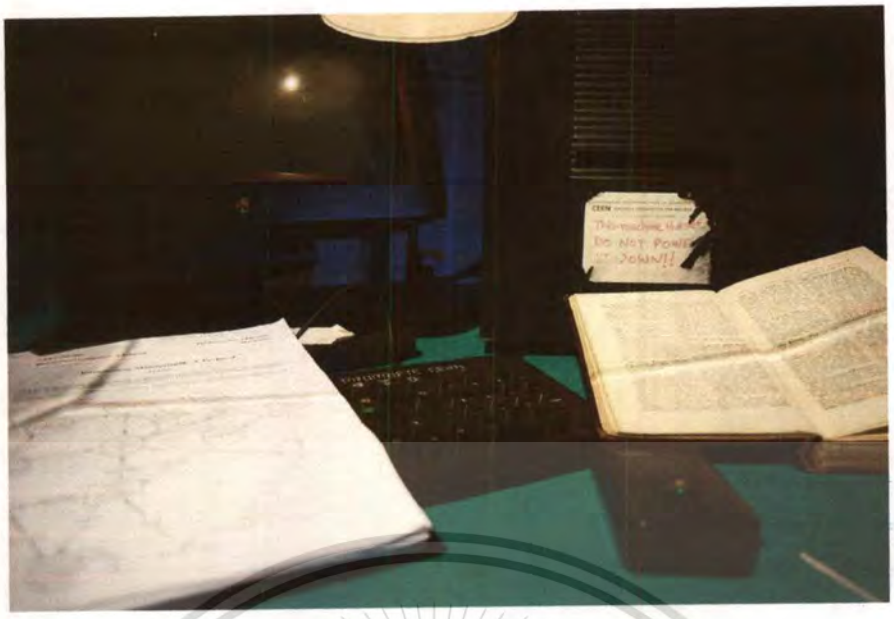


***Figure 2.12** NeXT Computer*

Berners-Lee's breakthrough was to marry hypertext to the Internet. In his book *Weaving The Web*, he explains that he had repeatedly suggested that a marriage between the two technologies was possible to members of *both* technical communities, but when no one took up his invitation, he finally tackled the project himself. In the process, he developed three essential technologies:

1. A system of globally unique identifiers for resources on the Web and elsewhere, the Universal Document Identifier (UDI), later known as Uniform Resource Locator (URL) and Uniform Resource Identifier (URI)
2. The publishing language HyperText Markup Language (HTML)
3. The HyperText Transfer Protocol (HTTP).

The World Wide Web had a number of differences from other hypertext systems that were then available. The Web required only unidirectional links rather than bidirectional ones. This made it possible for someone to link to another resource without action by the owner of that resource. It also significantly reduced the difficulty of implementing web servers and browsers (in comparison to earlier systems), but in turn presented the chronic problem of *link rot*. Unlike predecessors such as HyperCard, the World Wide Web was non-proprietary, making it possible to develop servers and clients

independently and to add extensions without licensing restrictions. On April 30, 1993, CERN announced that the World Wide Web would be free to anyone, with no fees due.[18] Coming two months after the announcement that the server implementation of the Gopher protocol was no longer free to use, this produced a rapid shift away from Gopher and towards the Web. An early popular web browser was Viola WWW for Unix and the X Windowing System.

Scholars generally agree that a turning point for the World Wide Web began with the introduction of the Mosaic web browser in 1993. A graphical browser developed by a team at the National Center for Supercomputing Applications at the University of Illinois at Urbana-Champaign (NCSA-UIUC), led by Marc Andreessen, funding for Mosaic came from the U.S. *High-Performance Computing and Communications Initiative* and the *High Performance Computing and Communication Act of 1991*, one of several computing developments initiated by U.S. Senator Al Gore. Prior to the release of Mosaic, graphics were not commonly mixed with text in web pages and the Web's popularity was less than older protocols in use over the Internet, such as Gopher and Wide Area Information Servers (WAIS). Mosaic's graphical user interface allowed the Web to become, by far, the most popular Internet protocol.

The World Wide Web Consortium (W3C) was founded by Tim Berners-Lee after he left the European Organization for Nuclear Research (CERN) in October 1994. It was founded at the Massachusetts Institute of Technology Laboratory for Computer Science (MIT/LCS) with support from the Defense Advanced Research Projects Agency (DARPA), which had pioneered the Internet; a year later, a second site was founded at INRIA with support from the European Commission DG InfSo; and in 1996, a third continental site was created in Japan at Keio University. By the end of 1994, while the total number of websites was still minute compared to present standards, quite a number of notable websites were already active, many of which are the precursors or inspiration for today's most popular services.

Connected by the existing Internet, other websites were created around the world, adding international standards for domain names and HTML. Since then, Berners-Lee has played an active role in guiding the development of web standards (such as the markup languages in which web pages are composed), and in recent years has advocated his vision of a Semantic Web. The World Wide Web enabled the spread of information over the Internet through an easy-to-use and flexible format. It thus played an important role in popularizing use of the Internet. Although the two terms are sometimes conflated inpopularuse, *WorldWideWeb* is not synonymous with *Internet*. The Web is a collection of documents and both client and server software using Internet protocols such as TCP/IP and HTTP.

2.9.2 Function

The terms Internet and World Wide Web are often used in every-day speech without much distinction. However, the Internet and the World Wide Web are not one and the same. The Internet is a global system of interconnected computer networks. In contrast, the Web is one of the services that runs on the Internet. It is a collection of textual documents and other resources, linked by hyperlinks and URLs, transmitted by web browsers and web servers. In short, the Web can be thought of as an application "running" on the Internet.

Viewing a web page on the World Wide Web normally begins either by typing the URL of the page into a web browser or by following a hyperlink to that page or resource. The web browser then initiates a series of communication messages, behind the scenes, in order to fetch and display it. As an example, consider the Google page with the URL

http://www.google.co.th/

The web browser parses the HTML, interpreting the markup that surrounds the words in order to draw that text on the screen.

Many web pages consist of more elaborate HTML which references the URLs of other resources such as images, other embedded media, scripts that affect page behavior, and Cascading Style Sheets that affect page layout. A browser that handles complex HTML will make additional HTTP requests to the web server for these other Internet media types. As it receives their content from the web server, the browser progressively renders the page onto the screen as specified by its HTML and these additional resources.

2.10 C#

C# is a multi-paradigm programming language encompassing strong typing, imperative, declarative, functional, generic, object-oriented ,and component-oriented programming discipline. It was developed by Microsoft within its .NET initiative and later approved as a standard by Ecma (ECMA-334) and ISO (ISO/IEC 23270). C# is one of the programming languages designed for the Common Language Infrastructure.

C# is intended to be a simple, modern, general-purpose, object-oriented programming language. Its development team is led by Anders Hejlsberg. The most recent version is C# 4.0, which was released on April 12, 2010.

2.10.1 History

During the development of the .NET Framework, the class libraries were originally written using a managed code compiler system called *Simple Managed C* (SMC). In January 1999, Anders Hejlsberg formed a team to build a new language at the time called Cool, which stood for "C-like Object Oriented Language". Microsoft had considered keeping the name "Cool" as the final name of the language, but chose not to do so for trademark reasons. By the time the .NET project was publicly announced at the July 2000 Professional Developers Conference, the language had been renamed C#, and the class libraries and ASP.NET runtime had been ported to C#.

C#'s principal designer and lead architect at Microsoft is Anders Hejlsberg, who was previously involved with the design of Turbo Pascal, Embarcadero Delphi (formerly CodeGear Delphi and Borland Delphi), and Visual J++. In interviews and technical papers he has stated that flaws in most major programming languages such as C++, Java, Delphi, and Smalltalk, drove the fundamentals of the Common Language Runtime (CLR), which, in turn, drove the design of the C# language itself.

James Gosling, who created the Java programming language in 1994, and Bill Joy, a co-founder of Sun Microsystems, the originator of Java, called C# an "imitation" of Java; Gosling further claimed that

> *"C# is sort of Java with reliability, productivity and security deleted."*

Klaus Kreft and Angelika Langer stated in a blog post that

> *"Java and C# are almost identical programming languages. Boring repetition that lacks innovation,"*
>
> *"Hardly anybody will claim that Java or C# is revolutionary programming languages that changed the way we write programs,"*

and

> *"C# borrowed a lot from Java - and vice versa. Now that C# supports boxing and unboxing, we'll have a very similar feature in Java."*

Anders Hejlsberg has argued that C# is "not a Java clone" and is "much closer to C++" in its design.

Since the release of C# 2.0 in November of 2005, the C# and Java languages have evolved on increasingly divergent trajectories, becoming somewhat less similar. One of the first major departures came with the addition of generics to both languages, with vastly different implementations. C# makes use of reification to provide "first-class" generic objects that can be used like any other class, with code generation performed at class-load time. By contrast, Java's generics are essentially a language syntax feature, and

they do not affect the generated byte code, because the compiler performs type erasure on the generic type information after it has verified its correctness.

Furthermore, C# has added several major features to accommodate functional-style programming, culminating in their LINQ extensions released with C# 3.0 and its supporting framework of lambda expressions, extension methods, and anonymous classes. These features enable C# programmers to use functional programming techniques, such as closures, when it is advantageous to their application. The LINQ extensions and the functional imports help developers reduce the amount of "boilerplate" code that is included in common tasks like querying a database, parsing an xml file, or searching through a data structure, shifting the emphasis onto the actual program logic to help improve readability and maintainability.

C# used to have a mascot called Andy (named after Anders Hejlsberg). It was retired on 29 Jan 2004.

2.10.2 Versions

In the course of its development, the C# language has gone through several versions:

Table 2.1 *Versions of C#*

Version	Language specification			Date	.NET Framework	Visual Studio
	ECMA	ISO/IEC	Microsoft			
C# 1.0	December 2002	April 2003	January 2002	January 2002	.Net Framework 1.0	Visual Studio .NET 2002
C# 1.2			October 2003	April 2003	.Net Framework 1.1	Visual Studio .NET 2003
C# 2.0	June 2006	September 2006	September 2005	November 2005	.Net Framework 2.0	Visual Studio 2005
C# 3.0	None		August 2007	November 2007	.Net Framework 2.0 (Except LINQ/Query Extensions) .Net Framework 3.0 (Except LINQ/Query Extensions) .Net Framework 3.5	Visual Studio 2008 Visual Studio 2010
C# 4.0			April 2010	April 2010	.Net Framework 4.0	Visual Studio 2010

Table 2.2 *Summary of Versions*

	C# 2.0	C# 3.0	C# 4.0	C# 5.0 (planned)
Features added	• Generics • Partial types • Anonymous methods • Iterators • Nullable types • Private setters (properties) • Method group conversions (delegates)	• Implicitly typed local variables • Object and collection initializers • Auto-Implemented properties • Anonymous types • Extension methods • Query expressions. • Lambda expressions • Expression trees	• Dynamic binding • Named and optional arguments • Generic co- and contravariance	• Asynchronous methods • Compiler as a service

2.11 .NET Framework

The .NET Framework is a software framework that runs primarily on Microsoft Windows. It includes a large library and supports several programming languages which allow each language can use code written in other languages). Programs written for the .NET Framework execute in a software environment, known as the Common Language Runtime (CLR), an application virtual machine that provides important services such as security, memory management, and exception handling. The class library and the CLR together constitute the .NET Framework.

The .NET Framework's Base Class Library provides user interface, data access, database connectivity, cryptography, web application development, numeric algorithms, and network communications. Programmers produce software by combining their own source code with the .NET Framework and other libraries. The .NET Framework is intended to be used by most new applications created for the Windows platform. Microsoft also produces a popular integrated development environment largely for .NET software called Visual Studio.

2.11.1 History

Microsoft started development on the .NET Framework in the late 1990s originally under the name of Next Generation Windows Services (NGWS). By late 2000 the first beta versions of .NET 1.0 were released.

Version 3.0 of the .NET Framework is included with Windows Server 2008 and Windows Vista. Version 3.5 is included with Windows 7, and can also be installed on Windows XP and the Windows Server 2003 family of operating systems. On April 12, 2010, .NET Framework 4 was released alongside Visual Studio 2010.

The .NET Framework family also includes two versions for mobile or embedded device use. A reduced version of the framework, the .NET Compact Framework, is available on Windows CE platforms, including Windows

Mobile devices such as smart phones. Additionally, the .NET Micro Framework is targeted at severely resource-constrained devices.

***Table 2.3** .NET Framework Versions*

Version	Version Number	Release Date	Visual Studio	Default in Windows
1.0	1.0.3705.0	2002-02-13	Visual Studio .NET	Windows XP Tablet and Media Center Editions
1.1	1.1.4322.573	2003-04-24	Visual Studio .NET 2003	Windows Server 2003
2.0	2.0.50727.42	2005-11-07	Visual Studio .NET 2005	Windows Server 2003 R2
3.0	3.0.4506.30	2006-11-06		Windows Vista, Windows Server 2008
3.5	3.5.21022.8	2007-11-19	Visual Studio .NET 2008	Windows 7, Windows Server 2008 R2
4.0	4.0.30319.1	2010-04-12	Visual Studio .NET 2010	
4.5	4.5.40805	2011-09-13 (Developer Preview)	Visual Studio .NET '11'	Windows 8, Windows Server 8

2.11.2 Architecture

2.11.2.1 Common Language Infrastructure (CLI)

The purpose of the Common Language Infrastructure (CLI) is to provide a language-neutral platform for application development and execution, including functions for Exception handling, Garbage Collection, security, and interoperability. By implementing the core aspects of the .NET Framework within the scope of the CLI, this functionality will not be tied to a single language but will be available across the many languages supported by the framework. Microsoft's implementation of the CLI is called the Common Language Runtime, or CLR.

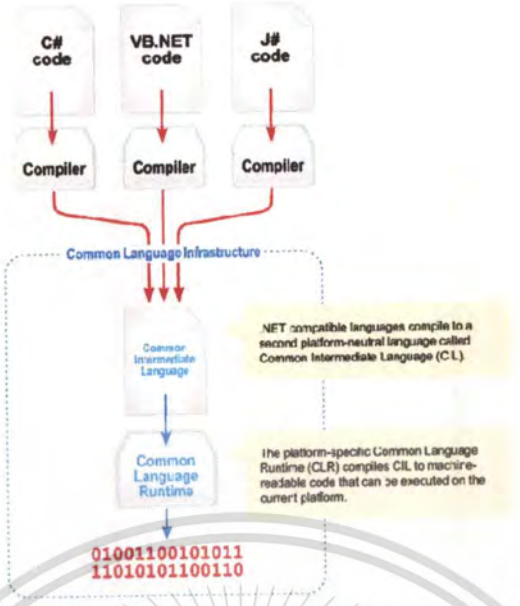


Figure 2.13 Overview of the Common Language Infrastructure

2.11.2.2 Security

.NET has its own security mechanism with two general features: Code Access Security (CAS), and validation and verification. Code Access Security is based on evidence that is associated with a specific assembly. Typically the evidence is the source of the assembly. Code Access Security uses evidence to determine the permissions granted to the code. Other code can demand that calling code is granted a specified permission. The demand causes the CLR to perform a call stack walk: every assembly of each method in the call stack is checked for the required permission; if any assembly is not granted the permission a security exception is thrown.

2.11.2.3 Class library

The .NET Framework includes a set of standard class libraries. The class library is organized in a hierarchy of namespaces. Most of the built in APIs are part of either System or Microsoft namespaces. These class libraries implement a large number of common functions, such as file reading and writing,

graphic rendering, database interaction, and XML document manipulation, among others. The .NET class libraries are available to all CLI compliant languages. The .NET Framework class library is divided into two parts: the Base Class Library and the Framework Class Library.

The Base Class Library (BCL) includes a small subset of the entire class library and is the core set of classes that serve as the basic API of the Common Language Runtime. The classes in mscorlib.dll and some of the classes in System.dll and System.core.dll are considered to be a part of the BCL. The BCL classes are available in both .NET Framework as well as its alternative implementations including .NET Compact Framework, Microsoft Silverlight and Mono.

The Framework Class Library (FCL) is a superset of the BCL classes and refers to the entire class library that ship with .NET Framework. It includes an expanded set of libraries, including Windows Forms, ADO.NET, ASP.NET, Language Integrated Query, Windows Presentation Foundation, Windows Communication Foundation among others. The FCL is much larger in scope than standard libraries for languages like C++, and comparable in scope to the standard libraries of Java.

***Table 2.4** Namespaces in the BCL.*

Namespaces in the BCL.
System
System. CodeDom
System. Collections
System. Diagnostics
System. Globalization
System. IO
System. Resources
System. Text
System. Text.RegularExpressions

2.12 XML

Extensible Markup Language (XML) is a set of rules for encoding documents in machine-readable form. It is defined in the XML 1.0 Specification produced by the W3C, and several other related specifications, all gratis open standards.

The design goals of XML emphasize simplicity, generality, and usability over the Internet. It is a textual data format with strong support via Unicode for the languages of the world. Although the design of XML focuses on documents, it is widely used for the representation of arbitrary data structures, for example in web services.

Many application programming interfaces (APIs) have been developed that software developers use to process XML data, and several schema systems exist to aid in the definition of XML-based languages.

As of 2009, hundreds of XML-based languages have been developed, including RSS, Atom, SOAP, and XHTML.XML-based formats have become the default for many office-productivity tools, including Microsoft Office (Office Open XML), OpenOffice.org (OpenDocument), and Apple's iWork.

2.12.1 History

XML is an application profile of SGML (ISO 8879).

The versatility of SGML for dynamic information display was understood by early digital media publishers in the late 1980s prior to the rise of the Internet. By the mid-1990s some practitioners of SGML had gained experience with the then-new World Wide Web, and believed that SGML offered solutions to some of the problems the Web was likely to face as it grew. Dan Connolly added SGML to the list of W3C's activities when he joined the staff in 1995; work began in mid-1996 when Sun Microsystems engineer Jon Bosak developed a charter and recruited collaborators. Bosak was well connected in the small community of people who had experience both in SGML and the Web.

XML was compiled by a working group of eleven members, supported by an (approximately) 150-member Interest Group. Technical debate took place on the Interest Group mailing list and issues were resolved by consensus or, when that failed, majority vote of the Working Group. A record of design decisions and their rationales was compiled by Michael Sperberg-McQueen on December 4, 1997. James Clark served as Technical Lead of the Working Group, notably contributing the empty-element "<empty />" syntax and the name "XML". Other names that had been put forward for consideration included "MAGMA" (Minimal Architecture for Generalized Markup Applications), "SLIM" (Structured Language for Internet Markup) and "MGML" (Minimal Generalized Markup Language). The co-editors of the specification were originally Tim Bray and Michael Sperberg-McQueen. Halfway through the project Bray accepted a consulting engagement with Netscape, provoking vociferous protests from Microsoft. Bray was temporarily asked to resign the editorship. This led to intense dispute in the Working Group, eventually solved by the appointment of Microsoft's Jean Paoli as a third co-editor.

The XML Working Group never met face-to-face; the design was accomplished using a combination of email and weekly teleconferences. The major design decisions were reached in a short burst of intense work between August and November 1996, when the first Working Draft of an XML specification was published. Further design work continued through 1997, and XML 1.0 became a W3CRecommendation on February 10, 1998.

2.12.1.1 Sources

XML is a profile of an ISO standard SGML, and most of XML comes from SGML unchanged. From SGML comes the separation of logical and physical structures (elements and entities), the availability of grammar-based validation (DTDs), the separation of data and metadata (elements and attributes),

mixed content, the separation of processing from representation (processing instructions), and the default angle-bracket syntax. Removed were the SGML Declaration (XML has a fixed delimiter set and adopts Unicode as the document character set).

Other sources of technology for XML were the Text Encoding Initiative (TEI), which defined a profile of SGML for use as a "transfer syntax"; and HTML, in which elements were synchronous with their resource, document character sets were separate from resource encoding, the xml:lang attribute was invented, and (like HTTP) metadata accompanied the resource rather than being needed at the declaration of a link. The Extended Reference Concrete Syntax (ERCS) project of the SPREAD (Standardization Project Regarding East Asian Documents) project of the ISO-related China/Japan/Korea Document Processing expert group was the basis of XML 1.0's naming rules; SPREAD also introduced hexadecimal numeric character references and the concept of references to make available all Unicode characters. To support ERCS, XML and HTML better, the SGML standard IS 8879 was revised in 1996 and 1998 with WebSGML Adaptations. The XML header followed that of ISO HyTime.

Ideas that developed during discussion which were novel in XML included the algorithm for encoding detection and the encoding header, the processing instruction target, the xml:space attribute, and the new close delimiter for empty-element tags. The notion of well-formedness as opposed to validity (which enables parsing without a schema) was first formalized in XML, although it had been implemented successfully in the Electronic Book Technology "Dynatext" software; the software from the University of Waterloo New Oxford English Dictionary Project; the RISP LISP SGML text processor at Uniscope, Tokyo; the US Army Missile Command IADS hypertext system; Mentor Graphics Context; Interleaf and Xerox Publishing System.

2.12.1.2 Versions

There are two current versions of XML. The first (*XML 1.0*) was initially defined in 1998. It has undergone minor revisions since then, without being given a new version number, and is currently in its fifth edition, as published on November 26, 2008. It is widely implemented and still recommended for general use.

The second (*XML 1.1*) was initially published on February 4, 2004, the same day as XML 1.0 Third Edition, and is currently in its second edition, as published on August 16, 2006. It contains features (some contentious) that are intended to make XML easier to use in certain cases. The main changes are to enable the use of line-ending characters used on EBCDIC platforms, and the use of scripts and characters absent from Unicode 3.2. XML 1.1 is not very widely implemented and is recommended for use only by those who need its unique features.

Prior to its fifth edition release, XML 1.0 differed from XML 1.1 in having stricter requirements for characters available for use in element and attribute names and unique identifiers: in the first four editions of XML 1.0 the characters were exclusively enumerated using a specific version of the Unicode standard (Unicode 2.0 to Unicode 3.2.) The fifth edition substitutes the mechanism of XML 1.1, which is more future-proof but reduces redundancy. The approach taken in the fifth edition of XML 1.0 and in all editions of XML 1.1 is that only certain characters are forbidden in names, and everything else is allowed, in order to accommodate the use of suitable name characters in future versions of Unicode. In the fifth edition, XML names may contain characters in the Balinese, Cham, or Phoenician scripts among many others which have been added to Unicode since Unicode 3.2.

Almost any Unicode code point can be used in the character data and attribute values of an XML 1.0 or 1.1 document, even if the character corresponding to the code point is not defined in the current version of Unicode. In character data and attribute values, XML 1.1 allows the use of more control characters than XML 1.0, but, for "robustness", most of the control characters introduced in XML 1.1 must be expressed as numeric character references. Among the supported control characters in XML 1.1 are two line break codes that must be treated as whitespace. Whitespace characters are the only control codes that can be written directly.

There has been discussion of an XML 2.0, although no organization has announced plans for work on such a project. XML-SW (SW for skunkworks), written by one of the original developers of XML, contains some proposals for what an XML 2.0 might look like: elimination of DTDs from syntax, integration of namespaces, XML Base and XML Information Set (*infoset*) into the base standard.

The World Wide Web Consortium also has an XML Binary Characterization Working Group doing preliminary research into use cases and properties for a binary encoding of the XML infoset. The working group is not chartered to produce any official standards. Since XML is by definition text-based, ITU-T and ISO are using the name *Fast Infoset* for their own binary infoset to avoid confusion (see ITU-T Rec. X.891 | ISO/IEC 24824-1).

2.13 jQuery

jQuery is a cross-browser JavaScript library designed to simplify the client-side scripting of HTML. It was released in January 2006 at BarCamp NYC by John Resig. Used by over 49% of the 10,000 most visited websites, jQuery is the most popular JavaScript library in use today.

jQuery is free, open source software, dual-licensed under the MIT License or the GNU General Public License, Version 2. jQuery's syntax is designed to make it easier to navigate a document, select DOM elements, create animations, handle events, and develop Ajax applications. jQuery also provides capabilities for developers to create plug-ins on top of the JavaScript library. This enables developers to create abstractions for low-level interaction and animation, advanced effects and high-level, theme-able widgets. The modular approach to the jQuery framework allows the creation of powerful and dynamic web pages and web applications.

Microsoft and Nokia have announced plans to bundle jQuery on their platforms, Microsoft is adopting it initially within Visual Studio for use within Microsoft's ASP.NET AJAX framework and ASP.NET MVC Framework while Nokia has integrated it into their Web Run-Time widget development platform. jQuery has also been used in MediaWiki since version 1.16.

2.13.1 Features

jQuery contains the following features:

- DOM element selections using the cross-browser open source selector engine Sizzle, a spin-off out of the jQuery project
- DOM traversal and modification (including support for CSS 1-3)
- Events
- CSS manipulation
- Effects and animations
- Ajax
- Extensibility through plug-ins
- Utilities - such as user agent information, feature detection
- Compatibility methods that are natively available in modern browsers but need fallbacks for older ones - For example the inArray() and each() functions.
- Cross-browser support

2.14 Web service

A Web service is a method of communication between two electronic devices over the web.

The W3C defines a "Web service" as "a software system designed to support interoperable machine-to-machine interaction over a network". It has an interface described in a machine-processable format (specifically Web Services Description Language, known by the acronym WSDL). Other systems interact with the Web service in a manner prescribed by its description using SOAP messages, typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards."

The W3C also states, "We can identify two major classes of Web services, REST-compliant Web services, in which the primary purpose of the service is to manipulate XML representations of Web resources using a uniform set of "stateless" operations; and arbitrary Web services, in which the service may expose an arbitrary set of operations."

2.13.1 Big Web services

"Big Web services" use Extensible Markup Language (XML) messages that follow the SOAP standard and have been popular with traditional enterprises. In such systems, there is often a machine-readable description of the operations offered by the service written in the Web Services Description Language (WSDL). The latter is not a requirement of a SOAP *endpoint*, but it is a prerequisite for automated client-side code generation in many Java and .NET SOAP frameworks (frameworks such as Apache Axis2, Apache CXF, and Spring being notable exceptions). Some industry organizations, such as the WS-I, mandate both SOAP and WSDL in their definition of a Web service.

2.14.2 Web API

Web API is a development in Web services (in a movement called Web 2.0) where emphasis has been moving away from SOAPbased services towards representational state transfer (REST) based communications. REST services do not require XML, SOAP, or WSDL service-API definitions.

Web APIs allow the combination of multiple Web services into new applications known as mashups.

When used in the context of Web development, Web API is typically a defined set of Hypertext Transfer Protocol (HTTP) request messages along with a definition of the structure of response messages, usually expressed in an Extensible Markup Language (XML) or JavaScript Object Notation (JSON) format.

When running composite Web services, each sub service can be considered autonomous. The user has no control over these services. Also the Web services themselves are not reliable; the service provider may remove, change or update their services without giving notice to users. The reliability and fault tolerance is not well supported; faults may happen during the execution. Exception handling in the context of Web services is still an open research issue, although this can still be handled by responding with an error object to the clients.

2.14.3 Styles of use

Web services are a set of tools that can be used in a number of ways. The three most common styles of use are RPC, SOA and REST.

2.14.3.1 Remote procedure calls

RPC Web services present a distributed function (or method) call interface that is familiar to many developers. Typically, the basic unit of RPC Web services is the WSDL operation.

The first *Web services* tools were focused on RPC, and as a result this style is widely deployed and supported. However, it is sometimes criticized for not beingloosely coupled, because it was often implemented by mapping services directly to language-specific functions or method calls. Many vendors felt this approach to be a dead end, and pushed for RPC to be disallowed in the WS-I Basic Profile.

Other approaches with nearly the same functionality as RPC are: Object Management Group's (OMG) Common Object Request Broker Architecture (CORBA),Open Software Foundation's (OSF) DCE/RPC, Microsoft's RPC (a part of DCOM) or .NET Remoting and Sun Microsystems's Java/Remote Method Invocation (RMI).

Figure 2.14 Architectural elements involved in the XML-RPC.

2.14.3.2 Service-oriented architecture

Web services can also be used to implement an architecture according to service-oriented architecture (SOA) concepts, where the basic unit of communication is a message, rather than an operation. This is often referred to as "message-oriented" services.

SOA Web services are supported by most major software vendors and industry analysts. Unlike RPC Web services, loose coupling is more likely, because the focus is on the "contract" that WSDL provides, rather than the underlying implementation details.

Middleware analysts use enterprise service buses (ESBs) that combine message-oriented processing and Web services to create an event-driven SOA. Examples of open-source ESB are Mule and Open ESB. An alternate approach is the Hub-and-Spoke model, which is server based. An example of this is Enterprise Message Service (EMS, an implementation of JMS).

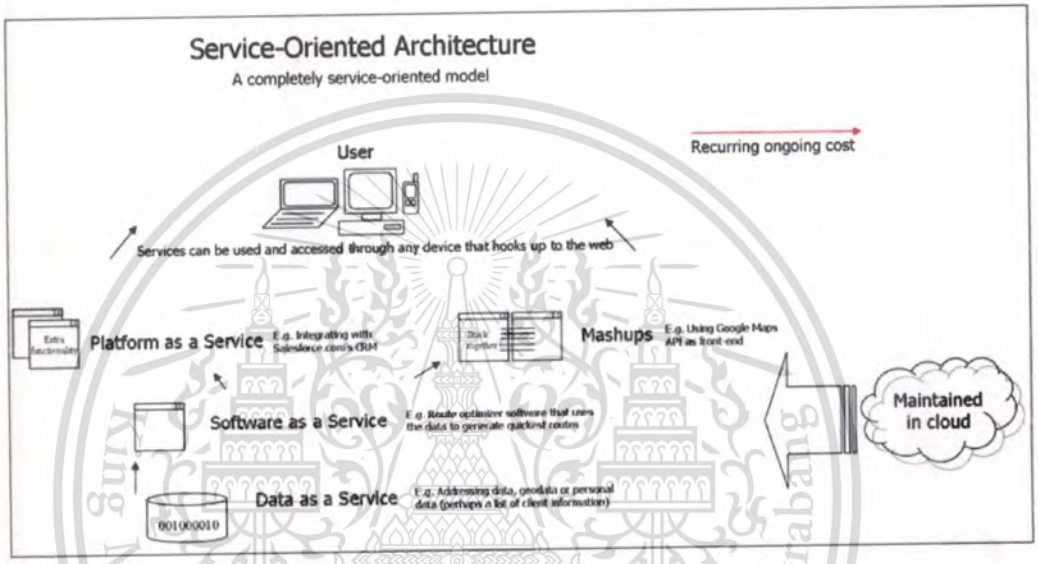


Figure 2.15 Web services in a service-oriented architecture.

2.14.3.3 Representational state transfer (REST)

REST attempts to describe architectures that use HTTP or similar protocols by constraining the interface to a set of well-known, standard operations (like GET, POST, PUT, DELETE for HTTP). Here, the focus is on interacting with stateful resources, rather than messages or operations. Clean URLs are tightly associated with the REST concept.

An architecture based on REST (one that is 'RESTful') can use WSDL to describe SOAP messaging over HTTP, can be implemented as an abstraction purely on top of SOAP (e.g., WS-Transfer), or can be created without using SOAP at all.

WSDL version 2.0 offers support for binding to all the HTTP request methods (not only GET and POST as in version 1.1) so it enables a better implementation of RESTful web services. However, support for this specification is still poor in software development kits, which often offer tools only for WSDL 1.1.

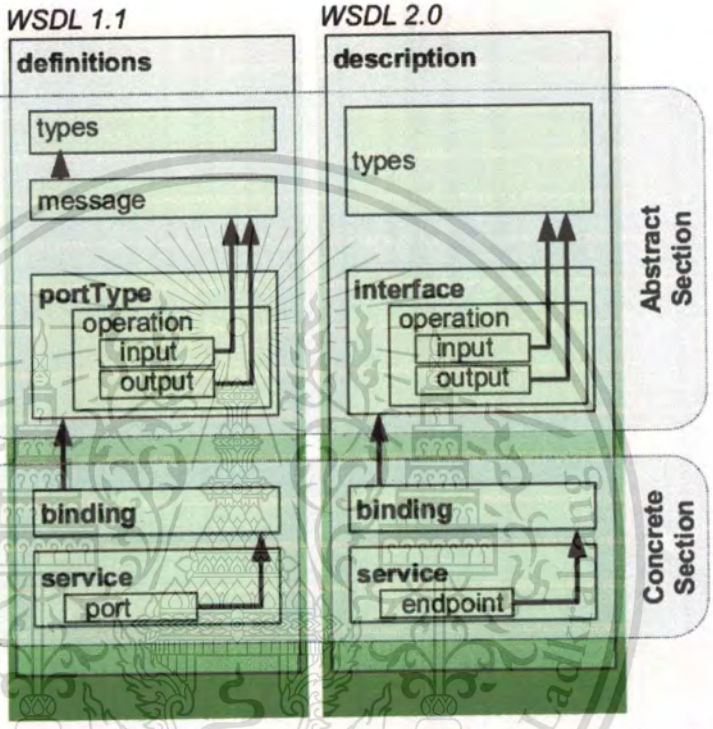


Figure 2.16 Representation of concepts defined by WSDL 1.1 and WSDL 2.0 documents.

2.15 Setting Up Development Environment

This chapter included the solutions of setting tools that need to use for the development follow:

1. Installation of Sun's Java Development Kit (JDK)
2. Installation of The Eclipse IDE
3. Installation of The Android Software Developer's Kit (SDK)
4. Installation of The Android Developer Tool (ADT)
5. Creation of Android Visual Device (AVD)

2.15.1 Installation of JDK

First of all download the 'setup' from oracle website

http://www.oracle.com/technetwork/java/javase/downloads

Click the Java button then select the platform and language that need to download. After download 'setup' will be the installation:

1. Double click JDK setup.
2. Click *Next* and select the location to install
3. Wait until the installation complete then click Finish

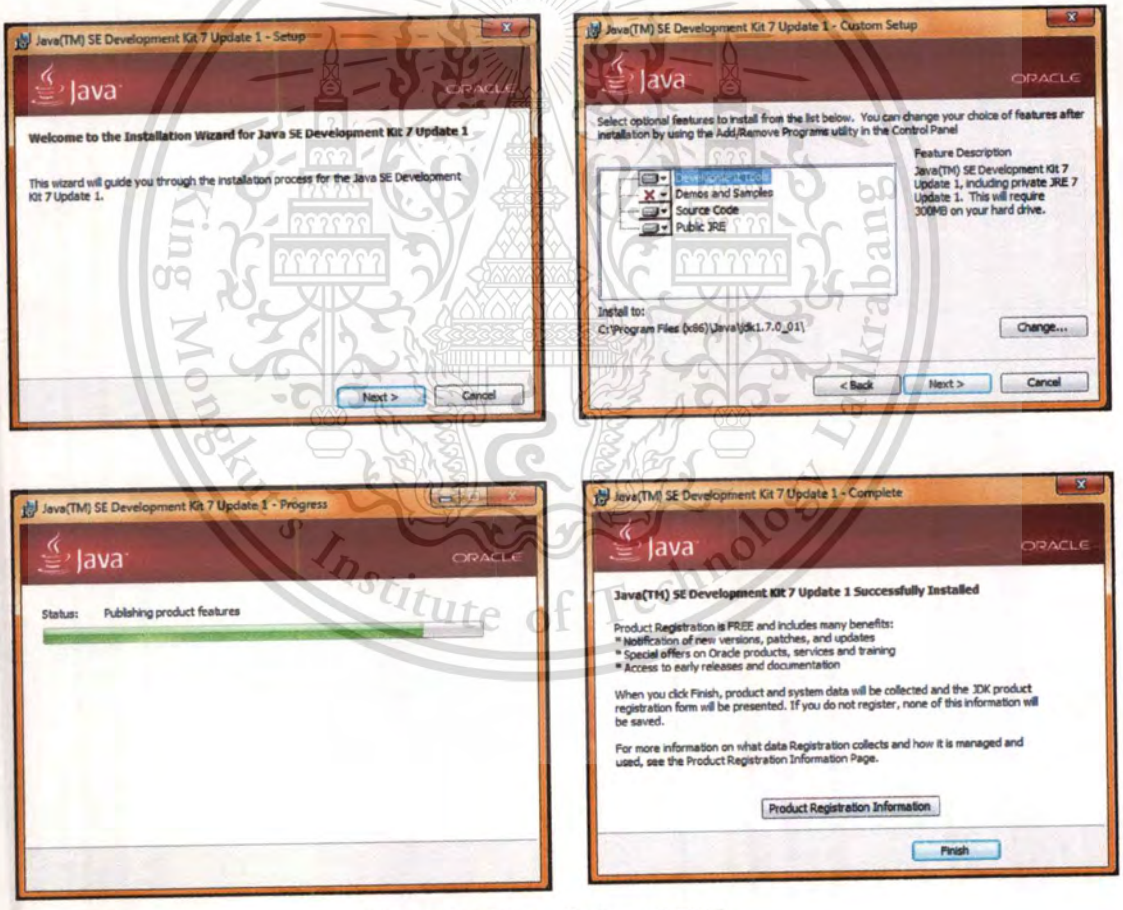


***Figure 2.17** Installation of JDK*

2.15.2 Installation of Eclipse

- Download 'Eclipse' from *http://www.eclipse.org/downloads/*

- Choose *'Eclipse IDE for Java Developer'*

- Unzip file to the directory *C:\eclipse*

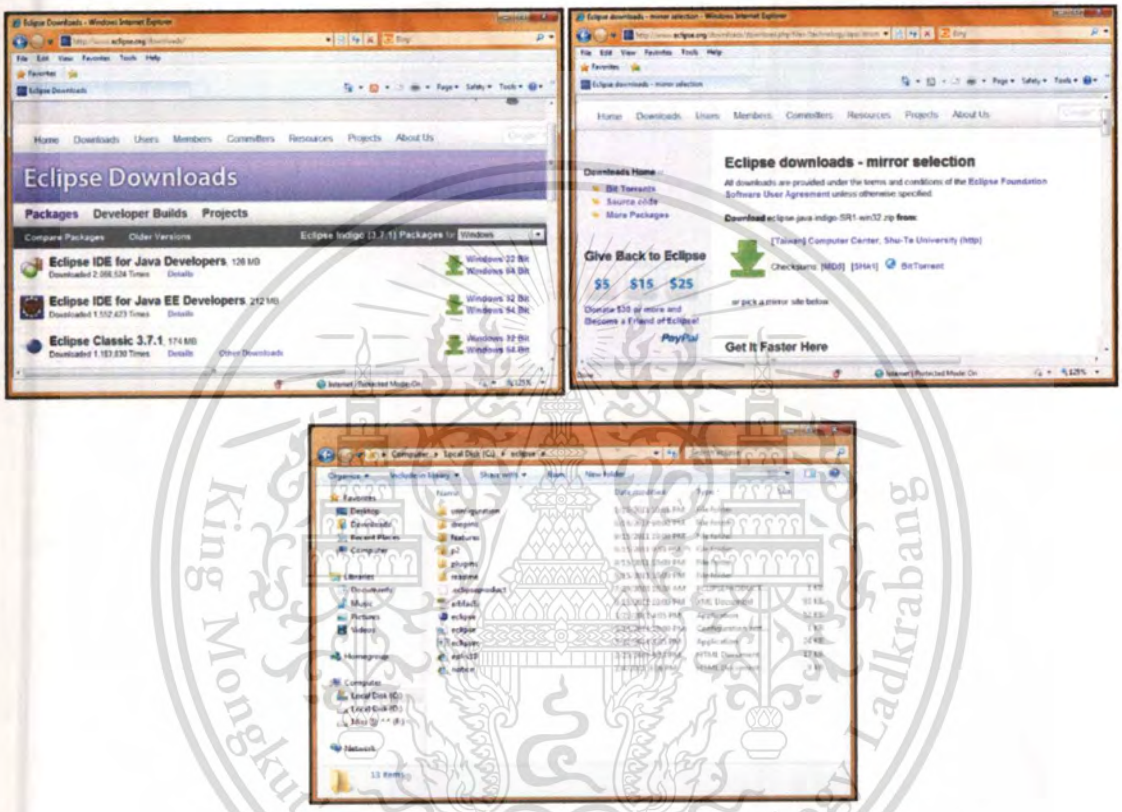


***Figure 2.18** Installation of Eclipse*

2.15.3 Installation of Android SDK

- Download *'Android SDK'* from *http://developer.android.com/sdk/index.html*

- Unzip file to the directory *C:\android-sdk-windows*

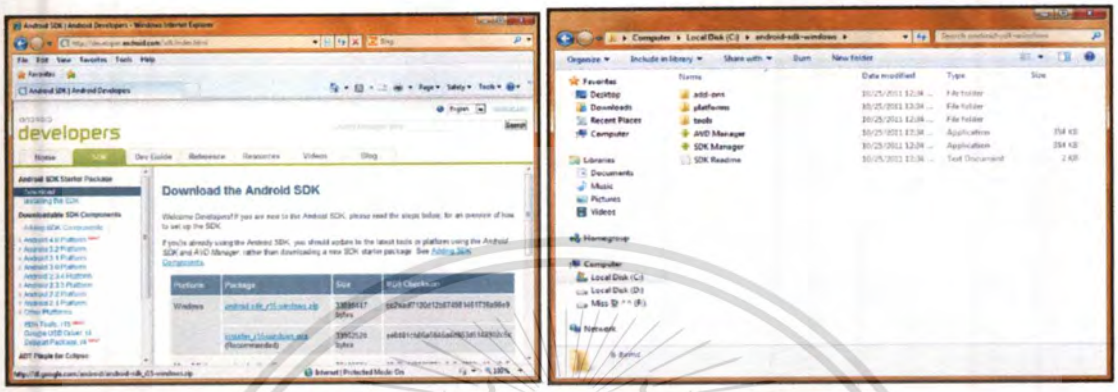


***Figure 2.19** Installation of Android SDK*

2.15.4 Installation of ADT

ADT (Android Development Tool) is Plug-in of Eclipse that support the process which is necessary for *Application Development*

- Open program *'Eclipse'*

- On menu bar, click *Help* > *Install New Software...*

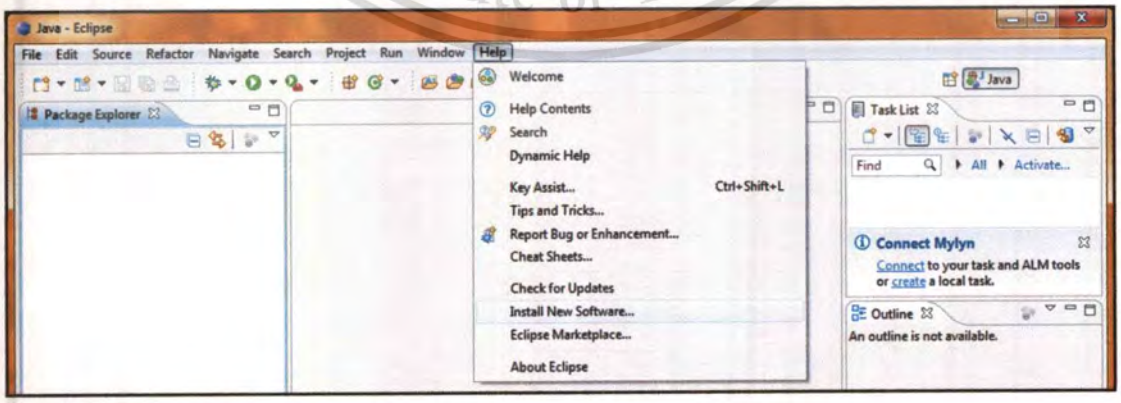


***Figure 2.20** Installation of ADT 1*

- *Install* dialogue box will appear then click *Add...*

- Fill in the *Repository* that you need (in this case use name 'Android').

- Fill the location in part of *Location* (in this case use *http://dl-ssl.google.com/android/eclipse/*) then click OK

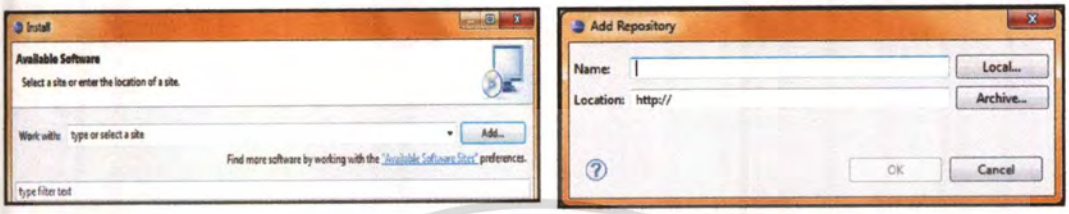


Figure 2.21 *Installation of ADT 2*

- System will display the list for choose.

- Click *Select All* then click *Next >*

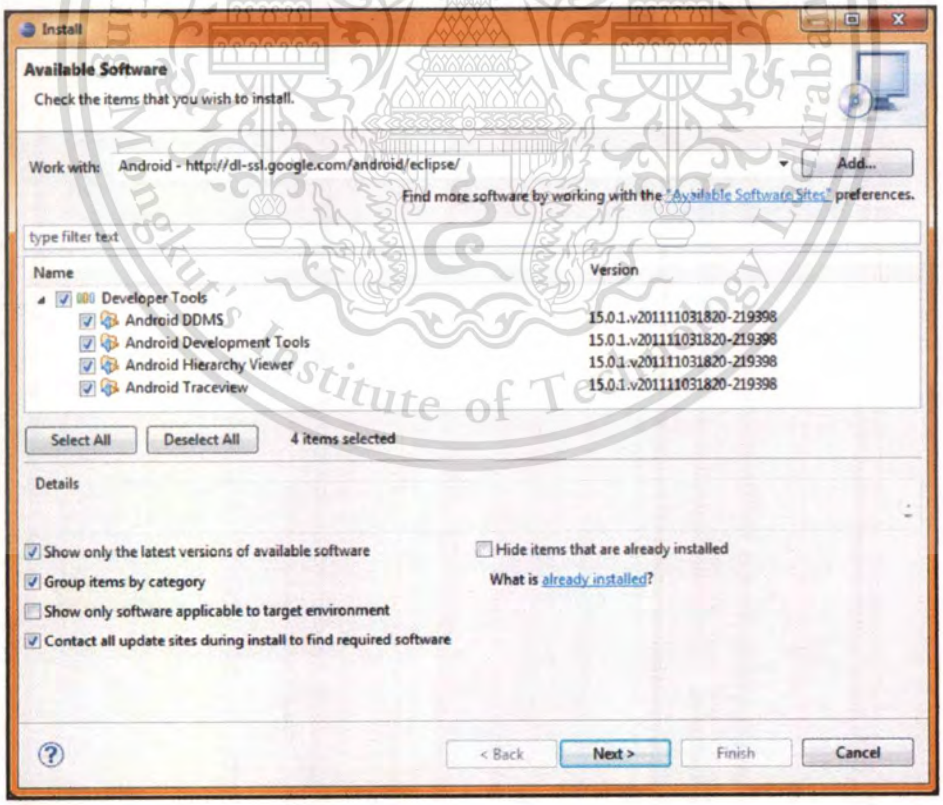


Figure 2.22 *Installation of ADT 3*

- Choose *I accept the terms of license agreements* then click *Finish*

- Wait until the installation complete after that click *Restart Now*, program will restart

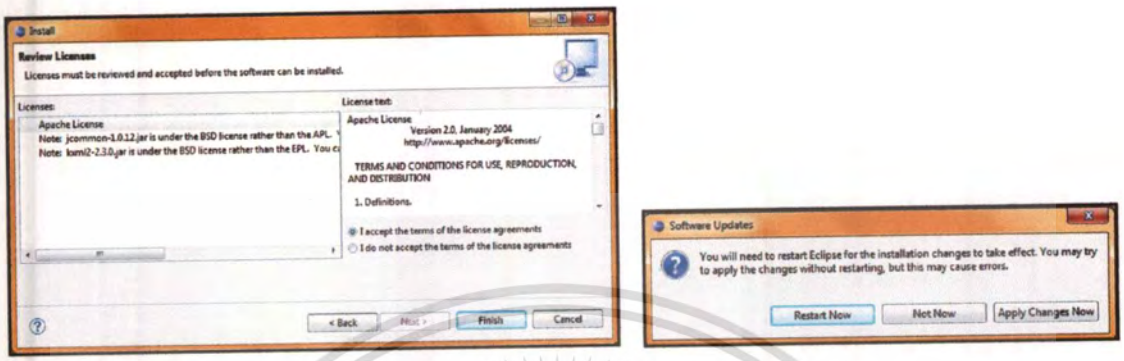


***Figure 2.23** Installation of ADT 4*

Chapter 3

Multicriteria and Multidimension technique

This chapter is refer to the technique that appropriate with Recommender System. This is developed under Multicriteria and Multidimension technique. To search a Neighbor those have a common affection.

3.1 Studying for new Music Recommender technique

To develop the Music Recommender System more expertly, we use a new technique that appropriate with Recommender System by use Multicriteria Rating and Multidimention. Steps are as :

1. Study current Recommender System in the market to understand a good point and weak point of each Recommender Systems.

2. Gather information of current Recommender system Application and website in the market for study a Recommender system at present. In website and application that use a Recommender System, function can be summarized as Table 3.1.

***Table 3.1** Show Attribute of Recommender website and application in Thailand and foreign country*

Attribute	Youtube	Amazon	Twitter	Exteen
Website member	No	No	Yes	Yes
Recommender Systems	Yes	Yes	Yes	Yes
Searching option	Yer	Yes	Yes	Yes
Data's Rating	Yes	Yes	No	Yes
Data's Review	Yes	Yes	No	Yes
Detail of data	Yes	Yes	Yes	Yes
Link with data's website	No	No	Yes	Yes
New data	Yes	Yes	Yes	Yes
Recommend data for everyone	Yes	Yes	No	No

3. Analyze information from learning current Recommender systems. Information for Multicriteria Rating and Multidimension that found in research is show a difference process.

From Chapter2 that show detail about each Recommender System. We decide to use Multicriteria Rating and Multidimention technique to use with Music Recommender systems.

3.2 Process of Music Recommender systems that use a new technique can separate in step :

1. Create a Profile of each Music by separate data follow Multicriteria, so we get 8 type and 70 fields and 1 type of Multidimension :

1.1 Pop songs

1.1.1 We found love

1.1.2 without you

1.1.3 Sexy and I know it

1.1.4 Stereo hearts

1.1.5 The one that got away

1.1.6 Move like jagger

1.1.7 Marry you

1.1.8 Someone like you

1.1.9 Good feeling

1.1.10 You make me feel

1.2 Dance songs

1.2.1 I like how it feels

1.2.2 Countdown

1.2.3Brand new bitch

1.2.4 Buy my love

1.2.5 Levels

1.2.6 Too much in love

1.2.7 Lat Drag

1.2.8 Party people

1.2.9 Love you like a love song

1.2.10 We're all no one

1.3 R&B songs

1.3.1 Lotus Flower Bomb

1.3.2 Dance

1.3.3 She will

1.3.4 Headlines

1.3.5 That way

1.3.6 Make me proud

1.3.7 5O'clock

1.3.8 You the boss

1.3.9 Body 2 Bosy

1.3.10 Work out

1.4 Rock songs

1.4.1 The sound of winter

1.4.2 Lovely boy

1.4.3 Walk

1.4.4 Paradise

1.4.5 Face to the floor

1.4.6 Tonight

1.4.7 These Days

1.4.8 The adventures of Rain Dance Maggie

1.4.9 Bottoms up

1.4.10 Cough Syrup

1.5 Alternative songs

1.5.1 Aberdeen

1.5.2 After Midnight

1.5.3 Monarchy of Roses

1.5.4 Helena Beat

1.5.5 Colours

1.5.6 Shake it out

1.5.7 What you want

1.5.8 Not your fault

1.5.9 Punching in a dream

1.5.10 The Righteous Women

1.6 Country songs

1.6.1 We owned the night

1.6.2 Tattoos on this town

1.6.3 Keep me in mind

1.6.4 Baggage Claim

1.6.5 Country must be country wide

1.6.6 God gave me you

1.6.7 Let it rain

1.6.8 Easy

1.6.9 I got you

1.6.10 Drink in my hand

1.7 Jazz songs

1.7.1 Boom town

1.7.2 Marrakesh

1.7.3 Easy come easy go

1.7.4 The lady in my life

1.7.5 Hot sauce

1.7.6 Sweet sea

1.7.7 Red suede shoes

1.7.8 All my life

1.7.9 Eyes for you

1.7.10 Slam dunk

1.8 Gender of Artist

1.8.1 Male

1.8.2 Female

1.9 Multidimention can separate in 1 dimension:

1.9.1 Genre

2. Create User profile from analyze both Multidimentional and Multicriteria together. Record User profile by correct rating of each user in 2 parts:

2.1 Work follow Part of Multicriteria

2.2 Work follow Part of Multidimension

3. Normalize to User Profile in range 0-1 after keep rating score of user finished.By separate in 2 step follow: 1. Normalize by using Multicriteria 2. Normalize by using Multidimension

4. Calculate weight of Multicriteria and Multidimension

5. After calculate weight of Multicritiria and Multidimension, use value of weight and multiply with total values of criteria and dimension for weight User Profile.

6. Calculate a distance of satisfaction between Active User and user B. By adjust value in User profile of user B with step1-5 to make it straight with Active User. After that find a difference of each position between Active User and user B then find average of all difference value

7. After calculate all difference of Active User and User B. Use that values calculate by use Multicriteria to find a similarity between Active User and User B

8. Use Sc and Sd multiply by weight that can separate in Multicriteria and Multidimension.

9. Find a neighbor maximum 5 user that most similar to Active User, calculate affection rating that Recommender system keep for each song and send it to Active User.

10. Calculate of $AVG_{a,\text{we found love}}$

Chapter 4

System Development

For *Multicriteria Rating* and Multidimension, prototype system, called Music Recommender, was created. This chapter including procedures of application, database design, and user interfaces.

4.1 Music Recommender procedures

This application uses Multicriteria Rating and Multidimension to calculate distance value of satisfaction value of each user. So details of this application are:

- **Procedures of the system to recommend a song for user**

1. User rated for songs in the system at least 10 songs.
2. Create *User Profiles* with satisfaction rate of each user by record in 2 part
   - Multicriteria
   - Multidimension
3. Crate new *User Profiles* for active user to weight values.
4. Calculate similarity value between *Active User* and other users in the system. Select 5 neighbors who have the most nearby satisfaction with active user.
5. Calculate satisfaction value of system for *Active User*. There are 2 parts, first, weight average satisfaction values of neighbors – *weight* is the similarity values between *Active User* and *neighbors*. Second part is satisfaction values of genre from users in the system.
6. Calculate average value from *Song Profile* of each song that neighbors ever rated for the *Multidimension.* Then use weight value of user profile of active user in part of multidimension to be *Average weight* value.

7. System selected 6 songs, which have most satisfaction value, for recommend to *Active User*.

4.2 Database Design

For test music recommender system for each user, there are entities for record *User Profile* and *Song Profile* including the entities for record data from user interface. Database can separate by

1. Group of entities for record data of user and data for calculation of recommender system.
2. Group of entities for record data of songs.
3. Group of entities for user interface.

4.2.1 Group of entities for record data of user and data for calculation of recommender system

- Entity *tbl_user* is the entity for record basic information of users who registered to the system.

Field	Type	Key	Description
userid	Int(4)	Primary Key	Id of each user
email	Varchar(80)		Email of each user
psw	Varchar(80)		Password of each user

- Entity *tbl_user_vote* is the entity for record *Rating* value of each song from each user. System uses this value for calculate the recommended content for active user.

Field	Type	Key	Description
id	Int(11)	Primary Key	Id of song rating
uid	Int(11)		Id of user
songid	Int(11)		Id of song
r1	Int(11)		Satisfaction value to song
r2	Int(11)		Satisfaction value to genre
r3	Int(11)		Satisfaction value of the artist

- Entity *tbl_score* is the entity for record score of user from user's rating to each song.

Field	Type	Key	Description
id	Int(11)	Primary Key	Id of song rating
uid	Int(11)		Id of user
songid	Int(11)		Id of song
r1	Int(11)		Satisfaction value to song
r2	Int(11)		Satisfaction value to genre
r3	Int(11)		Satisfaction value of the artist

- Entity *tbl_score_weight* is the entity for record score of user from user's rating to each song. Score, which recorded in this entity, are weighted by times that user rated the song.

Field	Type	Key	Description
id	Int(11)	Primary Key	Id of song rating
uid	Int(11)		Id of user
songid	Int(11)		Id of song
r1	Int(11)		Satisfaction value to song
r2	Int(11)		Satisfaction value to genre
r3	Int(11)		Satisfaction value of the artist

- Entity *tbl_distance* is the entity for record the distance value of satisfaction value between active user and other users in the system.

Field	Type	Key	Description
disid	Int(11)	Primary Key	Id of distance value record
act_uid	Int(11)		Id of active user
other_uid	Int(11)		Id of other users
dxm1	float(15,5)		Distance value of satisfaction value between active user and other users in the system
dxm2	float(15,5)		Similarity value of multicriteria satisfaction value between active user and other users in system
dxm3	float(15,5)		Similarity value of multidimension satisfaction value between active user and other users in the system

- Entity *tbl_simcheck* is the entity for record similarity values of each user multiply by song which user rated.

Field	Type	Key	Description
id	Int(11)	Primary Key	Id of similarity value record
uid	Int(11)		Id of user
songid	Int(11)		Id of song
SR	float(15,5)		Value of multiplication of similarity of other users to each song
sr1	float(15,5)		Similarity value between active user and other users in the system

- Entity *tbl_pd* is the entity for record similarity values of each user multiply by song which user rated.

Field	Type	Key	Description
pid	Int(11)	Primary Key	Id of prediction value record
songid	Int(11)	Primary Key	Id of song
p1	Int(11)		Prediction value of each song

- Entity *tbl_show* is the entity for record similarity values of each user multiply by song which user rated.

Field	Type	Key	Description
showid	Int(11)	Primary Key	Id of similarity value record
uid	Int(11)		Id of other user in the system
songid	Int(11)		Id of song
SR	float(15,5)		Value of multiplication of similarity of other users to each song
sr1	float(15,5)		Similarity value between active user and other users in the system

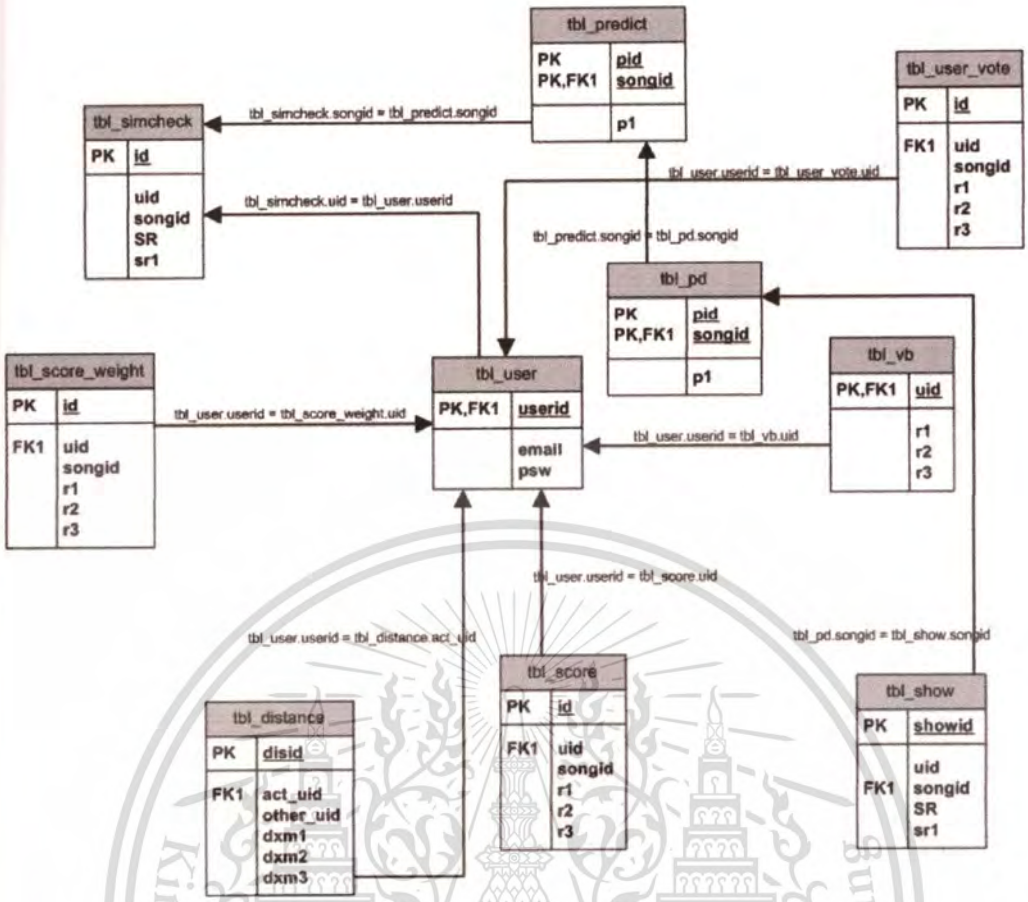


***Figure 4.1** ER-Diagram of Group of entities for record data of user and data for calculation of recommender system*

4.2.2 Group of entities for record data of songs

- Entity *tbl_song* is the entity for record basic information of song in the system.

Field	Type	Key	Description
sid	Int(11)	Primary Key	Id of song recording
songid	Int(11)		Id of song
songname	varchar(80)		Name of song

- Entity *tbl_genre* is the entity for record genre of song in the system.

Field	Type	Key	Description
genreid	Int(11)	Primary Key	Id of genre
genrename	varchar(80)		Name of genre

- Entity *tbl_songAndgenre* is the relative entity between song and genre.

Field	Type	Key	Description
sgid	Int(11)	Primary Key	Id of data recording
genreid	Int(11)		Id of genre
songid	Int(11)	Primary Key	Id of song

- Entity *tbl_artist* is the entity for record artist of each song in the system.

Field	Type	Key	Description
aid	Int(11)	Primary Key	Id of artist
aname	varchar(80)		Name of artist

- Entity *tbl_songAndartist* is the relative entity between song and genre.

Field	Type	Key	Description
said	Int(11)	Primary Key	Id of data recording
aid	Int(11)		Id of artist
songid	Int(11)	Primary Key	Id of song

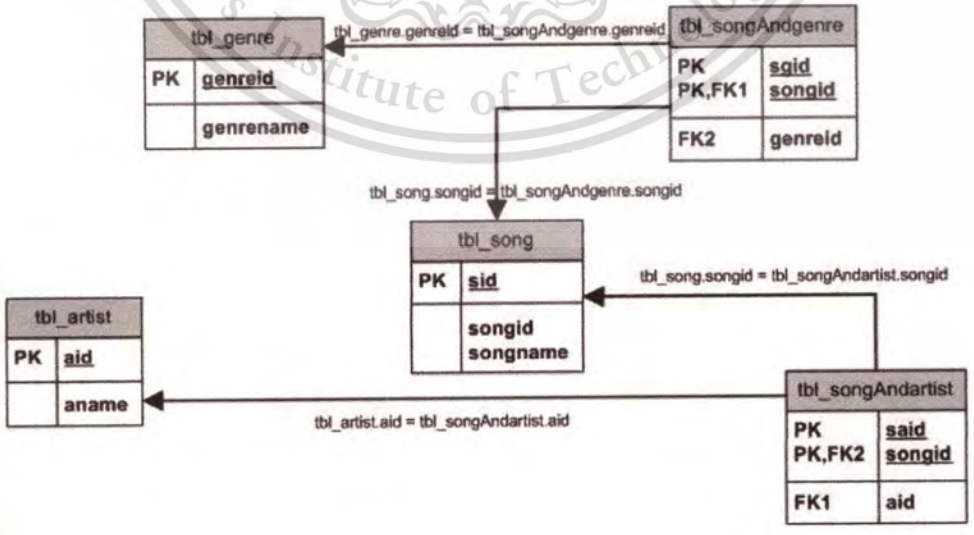


***Figure 4.2** ER-Diagram of Group of entities for record data of songs*

4.2.3 Group of entities for user interface

User interface of this project is android application for test the recommender system. So this group of entities show in ER-Diagram below

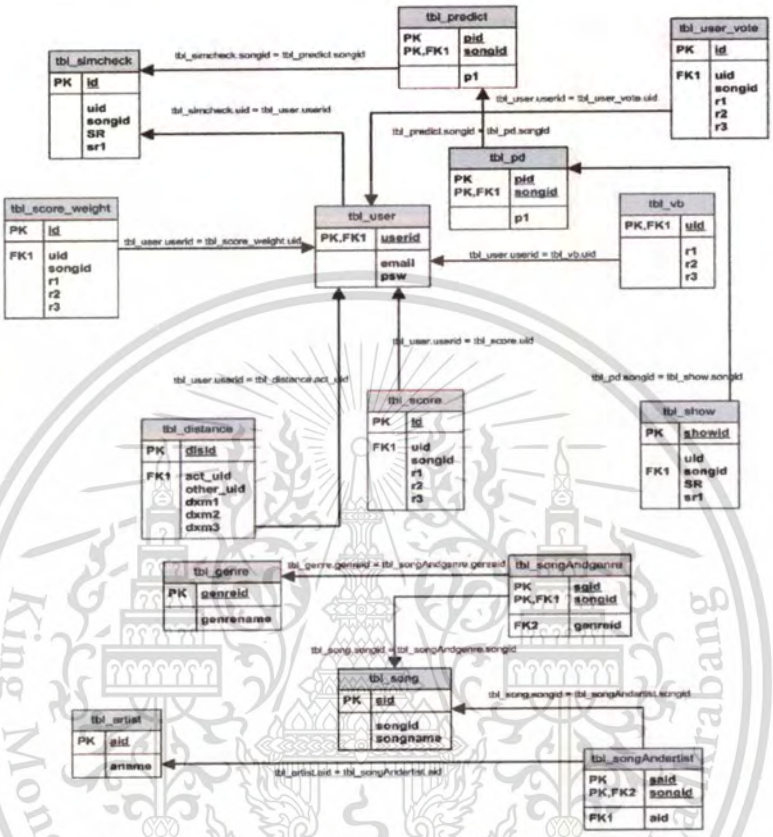


Figure 4.3 Group of entities for user interface

4.3 Connect to database

This application uses php file to connect with database. There are 4 php files:

- config.php: define details and accessibility of database.

```
<?php

define("DB_HOST", "localhost");
define("DB_USER", "root");
define("DB_PASSWORD", "aroiomlet");
define("DB_DATABASE", "android_api");
?>

```

***Figure 4.4** config.php*

- DB_Connect.php: contains the SQL to connect to database with details in *config.php*

```php
<?php
class DB_Connect {

    function __construct() {
    }

    function __destruct() {
    }

    // Connecting to database
    public function connect() {
        require_once 'include/config.php';
        // connecting to mysql
        $con = mysql_connect(DB_HOST, DB_USER, DB_PASSWORD);
        // selecting database
        mysql_select_db(DB_DATABASE);

        return $con;
    }

    // Closing database connection
    public function close() {
        mysql_close();
    }

}

?>

```

Figure 4.5 *DB_Connect.php*

- DB_Functions.php: contains SQL that use with database

```php
<?php

class DB_Functions {

    private $db;

    function __construct() {
        require_once 'DB_Connect.php';
        $this->db = new DB_Connect();
        $this->db->connect();
    }

    function __destruct() {
    }

    public function storeUser($sname, $semail, $spassword) {
        $uuid = uniqid('', true);
        $hash = $this->hashSSHA($spassword);
        $encrypted_password = $hash["encrypted"];
        $salt = $hash["salt"];
        $result = mysql_query("INSERT INTO users(unique_id, name, email, encrypted_password, salt, created_at) VALUES('$uuid', '$sname', '$em
ail', '$encrypted_password', '$salt', NOW())");
        if ($result) {
            $uid = mysql_insert_id();
            $result = mysql_query("SELECT * FROM users WHERE uid = $uid");
            return mysql_fetch_array($result);
        } else {
            return false;
        }
    }

    public function addComment($comment) {
        $result = mysql_query("UPDATE users SET comment_at = NOW(), comment = '$comment' WHERE uid = '$uid'");
        if($result){
            $uid = mysql_insert_id();
            $result = mysql_query("SELECT * FROM users WHERE uid = $uid");
            return mysql_fetch_array($result);
            } else{return false;}
    }

    public function getUserByEmailAndPassword($semail, $spassword) {
        $result = mysql_query("SELECT * FROM users WHERE email = '$semail'") or die(mysql_error());
        $no_of_rows = mysql_num_rows($result);
        if ($no_of_rows > 0) {
            $result = mysql_fetch_array($result);
            $salt = $result['salt'];
            $encrypted_password = $result['encrypted_password'];
            $hash = $this->checkhashSSHA($salt, $spassword);
            if ($encrypted_password == $hash) {
                return $result;
            }
        } else {
            return false;
        }
    }

    public function isUserExisted($semail) {
        $result = mysql_query("SELECT email from users WHERE email = '$semail'");
        $no_of_rows = mysql_num_rows($result);
        if ($no_of_rows > 0) {
            return true;
        } else {
            return false;
        }
    }

    public function hashSSHA($spassword) {
        $salt = sha1(rand());
        $salt = substr($salt, 0, 10);
        $encrypted = base64_encode(sha1($spassword . $salt, true) . $salt);
        $hash = array("salt" => $salt, "encrypted" => $encrypted);
        return $hash;
    }

    public function checkhashSSHA($salt, $spassword) {
        $hash = base64_encode(sha1($spassword . $salt, true) . $salt);
        return $hash;
    }
}
?>
```

***Figure 4.6** DB_Functions.php*

- index.php: define the variables from application with variables in database

```php
<?php

if (isset($_POST['tag']) && $_POST['tag'] != '') {
    $tag = $_POST['tag'];

    require_once 'include/DB_Functions.php';
    $db = new DB_Functions();

    $response = array("tag" => $tag, "success" => 0, "error" => 0);

    if ($tag == 'login') {

        $email = $_POST['email'];
        $password = $_POST['password'];

        $user = $db->getUserByEmailAndPassword($email, $password);
        if ($user != false) {
            $response["success"] = 1;
            $response["uid"] = $user["unique_id"];
            $response["user"]["name"] = $user["name"];
            $response["user"]["email"] = $user["email"];
            $response["user"]["created_at"] = $user["created_at"];
            $response["user"]["updated_at"] = $user["updated_at"];
            echo json_encode($response);
        } else {
            $response["error"] = 1;
            $response["error_msg"] = "Incorrect email or password!";
            echo json_encode($response);
        }
    } else if($tag == "comment"){
        $comment = $_POST['comment'];

        $user = $db->addComment($comment);
        if($user != false){
            $response["success"] = 1;
            $response["user"]["comment"] = $user["comment"];
            $response["user"]["comment_at"] = $user["comment_at"];
            $response["user"]["comment_id"] = $user["comment_uid"];
            }
        } else if (tag == 'register') {
        $name = $_POST['name'];
        $email = $_POST['email'];
        $password = $_POST['password'];

        if ($db->isUserExisted($email)) {
            $response["error"] = 2;
            $response["error_msg"] = "User already existed";
            echo json_encode($response);
        } else {
            $user = $db->storeUser($name, $email, $password);
            if ($user) {
                $response["success"] = 1;
                $response["uid"] = $user["unique_id"];
                $response["user"]["name"] = $user["name"];
                $response["user"]["email"] = $user["email"];
                $response["user"]["created_at"] = $user["created_at"];
                $response["user"]["updated_at"] = $user["updated_at"];
                echo json_encode($response);
            } else {
                $response["error"] = 1;
                $response["error_msg"] = "Error occured in Registation";
                echo json_encode($response);
            }
        }
    } else {
        echo "Invalid Request";
    }
} else {
    echo "Access Denied";
}
?>
```

Figure 4.7 index.php

4.4 User Interface Design

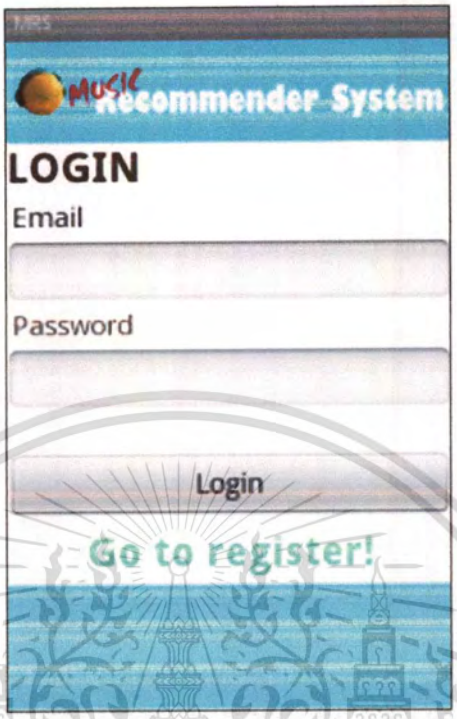


Figure 4.8 Login Screen

***Figure 4.9** Register Screen*

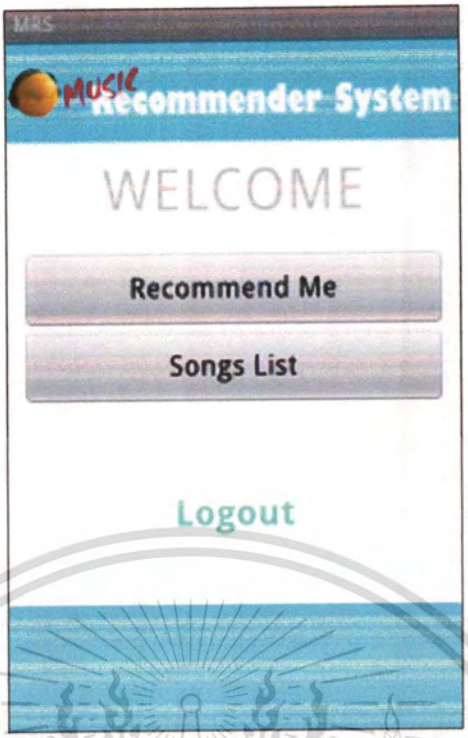


Figure 4.10 Main Screen

Figure 4.11 Menu Screen

Figure 4.12 Song List Screen

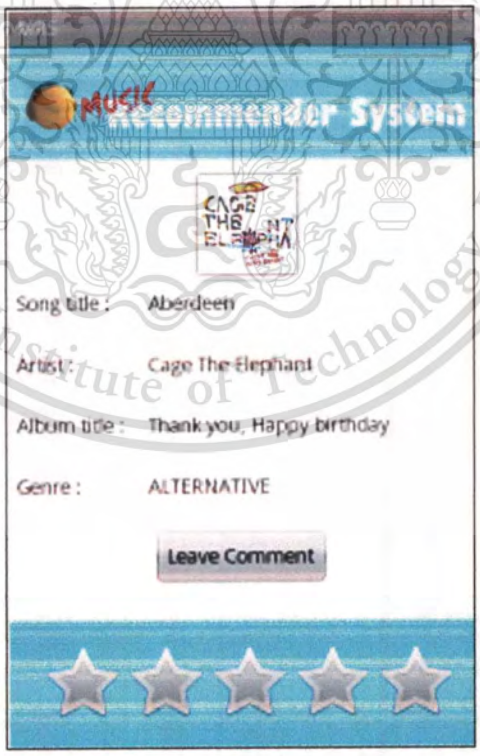


***Figure 4.13** Content Screen*

Figure 4.14 Comment Screen

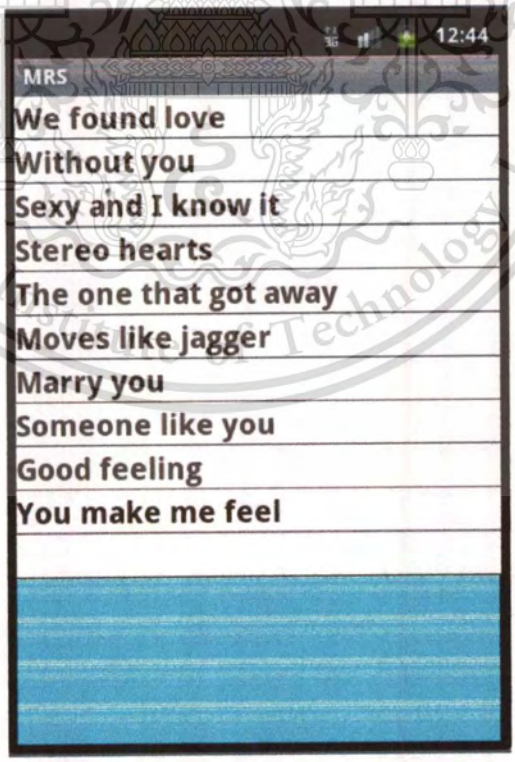


Figure 4.15Recommeder List Screen

4.5 Hardware and Software for develop system

For develop the recommender system, android device application was created and use MySQL to create databases.

4.5.1 Software

1. Eclipse for create application for android device
2. SQLite Database Browser for create and arrange database
3. Microsoft Office Excel 2007 for record data for developed and test system
4. Appserve

Development and testing of music recommender system can separate into 2 parts

1. Part of calculation of distance value of similarity value of each user to find neighbors.
2. Part of music recommend for each user calculated by similarity value of neighbor.
3. Part of *User Profile* and *Song Profile* which is recorded by SQLite Database Browser

Chapter 5

Discussion

5.1 Solution

Assume 5 users to rate 10 songs per user from 70 songs in the system. After user required recommended list from system, system recommended 5 songs for user (rating score is 1-5 if user rated for song equal or greater than 4 system will record that user satisfy in that song, on the other hand, if user rated for song equal or less than 3 system decide that user does not satisfy in that song) to record data that how user satisfy in recommender system.

Each user rated 5 songs from recommended list. Mean that there are 25 songs rated back to system.

5.2 Expect Result

From the rating of 5 songs, which system recommended to user, there should be at least 3 songs from recommended list that satisfied by user. So if the result according to the hypothesis that mean there are 15 songs satisfied from recommended list from system.

So the expected percentage of system is as follow:

Number of song 25 songs Number of satisfied song 15 songs

Accurate percentage $\frac{15\times100}{25} = 60\%$

The expected accurate percentage result of system is 60%.

Chapter 6

Conclusion and Problem

6.1 Conclusion

This project uses the solution of *Multicriteria* and *Multidimension* to create recommender system. System created *User Profile* to 2 parts. In part of *Multicriteria*, system considered the properties of each song. In part of *Multidimension* , system analyzed genre of song that user select. For the prediction part, system does not only consider on neighbor but system also considered average of satisfaction value of all user, who rate for that song, in system too.

6.2 Problems

1. Collected sample data for estimate quality of system need much of data.
2. Application for test system on android device is not successful.
3. Application for android cannot record data to database.
4. Application for android cannot select data from database.

6.3 Suggestion

- Use database on server instead put data direct into application.
- Try to test application on many different devices for test application.

Chapter 7

Application Manual

At first, users have to login to the system for rate song in the system then receive recommended list from system.

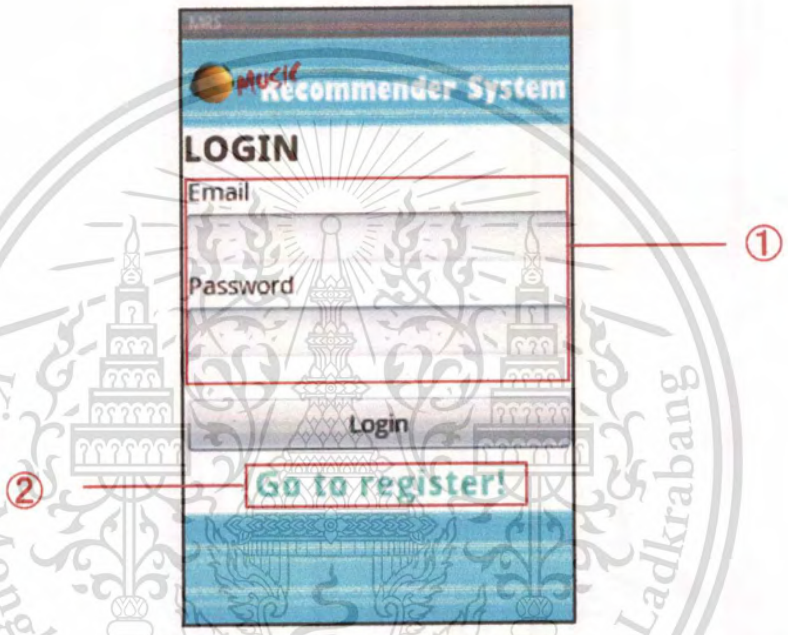


Figure 7.1 Login Screen

From figure 7.1, user should login on ① to access the system by insert e-mail and password. In case of user does not have account of application, user should register by clicking on ② then application will link to Register screen.

Figure 7.2 Register Screen

Register page require user to insert profile information. After complete select ② to login. And ③ if user already have an account.

After login, users can see main screen that provide user to choose recommender list screen or song list. Choose ③ if user need to logout.

Figure 7.3 Main Screen

If user choose button "Recommend me" , system link to Recommender list screen . User choose a song from list and system link to content of song follow Figure 7.7

Figure 7.4 Recommeder List Screen

If user choose ② from figure 7.3 application link to Menu screen and provide user choose genre of song follow ① in Figure 7.5

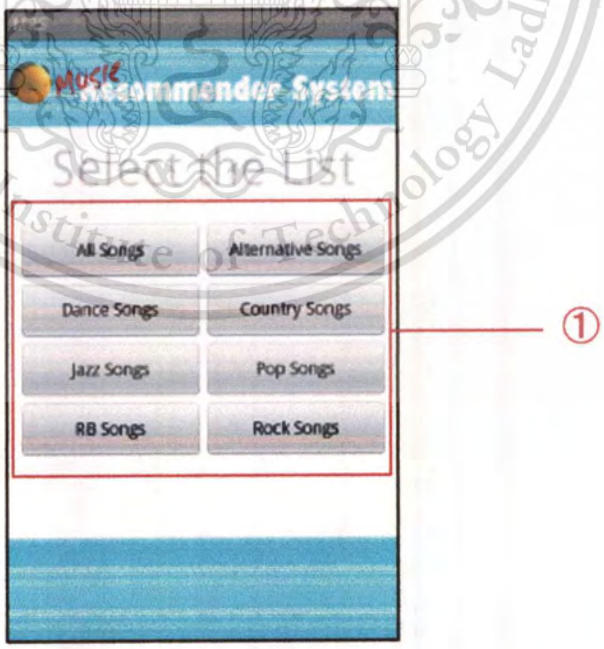


Figure 7.5 Menu Screen

Each genre of song will show list of song in nest page follow figure 7.6

Figure 7.6 Song List Screen

After user choose a song from list it will show a content screen

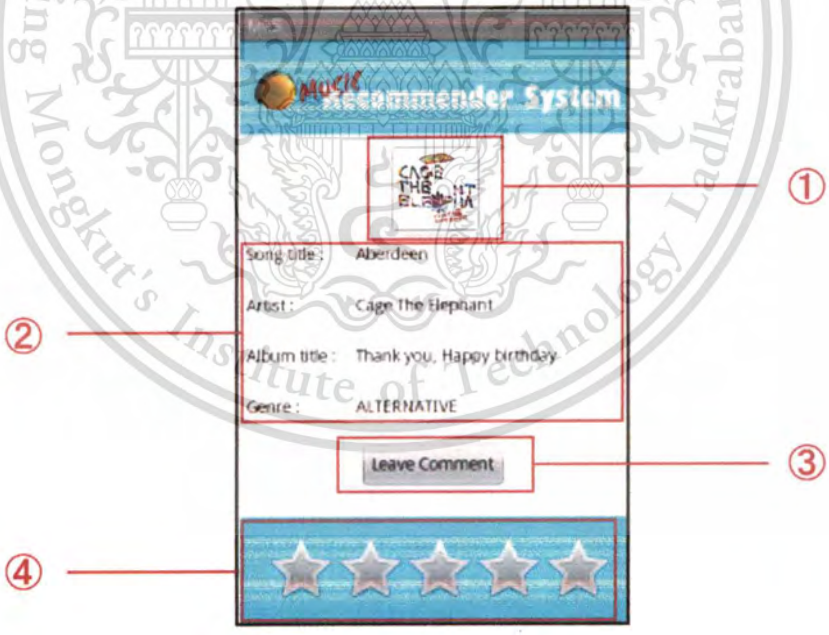


Figure 7.7 Content Screen

Part ①: Show image of song's album

Part ②: Show details of song (title, artist, album title, genre)

Part ③: 'Leave Comment' button for presses to leave a comment to song (figure 7.8)

Part ④: Rating bar for users to rate that song

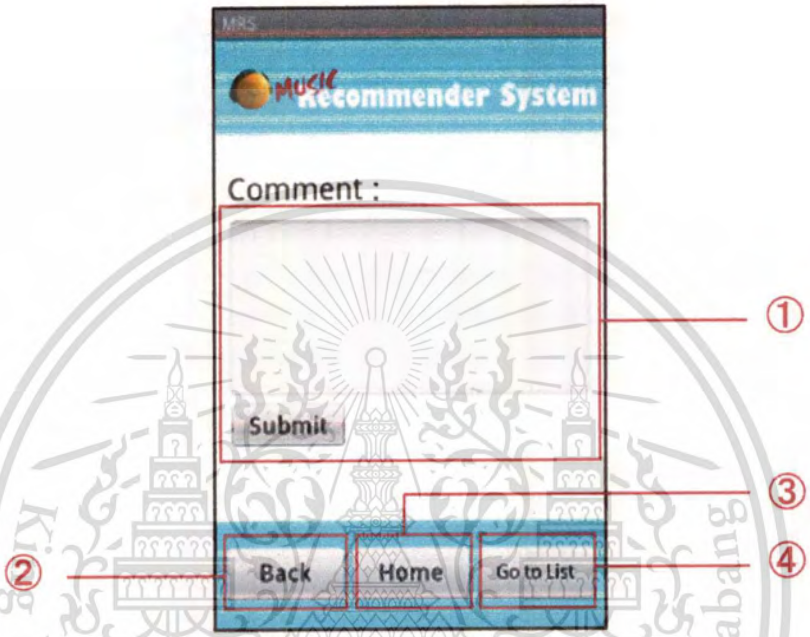


***Figure 7.8** Comment Screen*

User can leave comment in ① and click submit after finish. And at footer of page provide different menu.

Part ②: Go back to content screen

Part ③: Go to Home screen follow Figure 7.3

Part ④: Go to list of Song list screen follow Figure 7.6

Appendices

Appendix A

Android Application Installing Instruction

Appendix A. Install Application for Android

"Eclipse" is a software development environment and extensible plug-in, it is written mostly in Java. Develop packages for the software is available to any version of android, user can extend its abilities by installing plug-ins written for the Eclipse Platform, such as development toolkits for other programming languages, and can write and contribute their own plug-in modules.

Eclipse permit user to export application on android mobile, download appropriated plug-in for android version and connect android mobile with computer. Synchronize mobile and computer export application on android mobile follow this step :

1.) Choose android Tool > Export Signed Application Package…For create file.apk

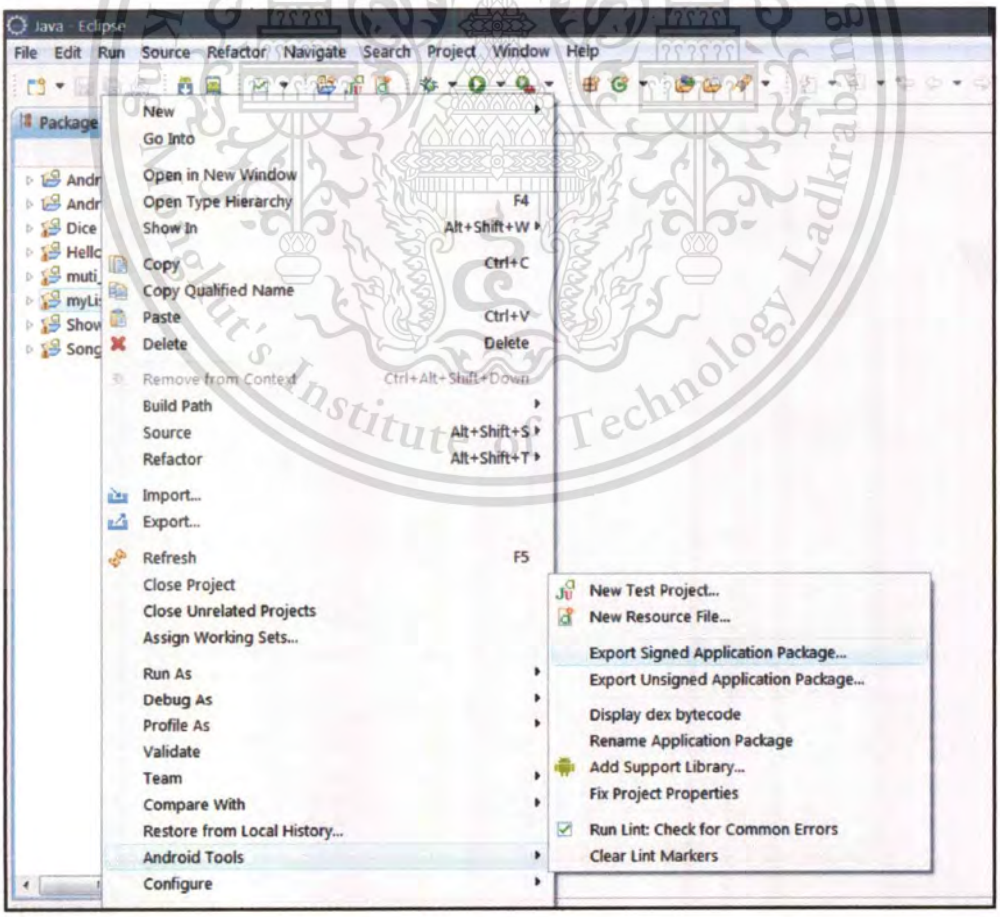


***Figure A.1** Android Tools*

2.) After "Export Android Application" dialog is appearing, browse for your project.

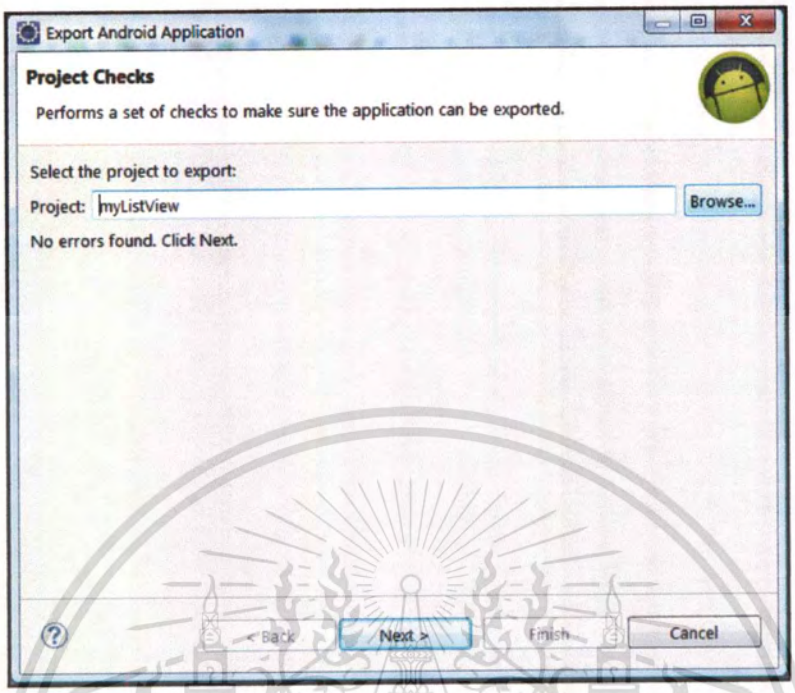


***Figure A.2** Export Android Application*

3.) choose Next > Create new keystore and choose directory for keep a keystore.

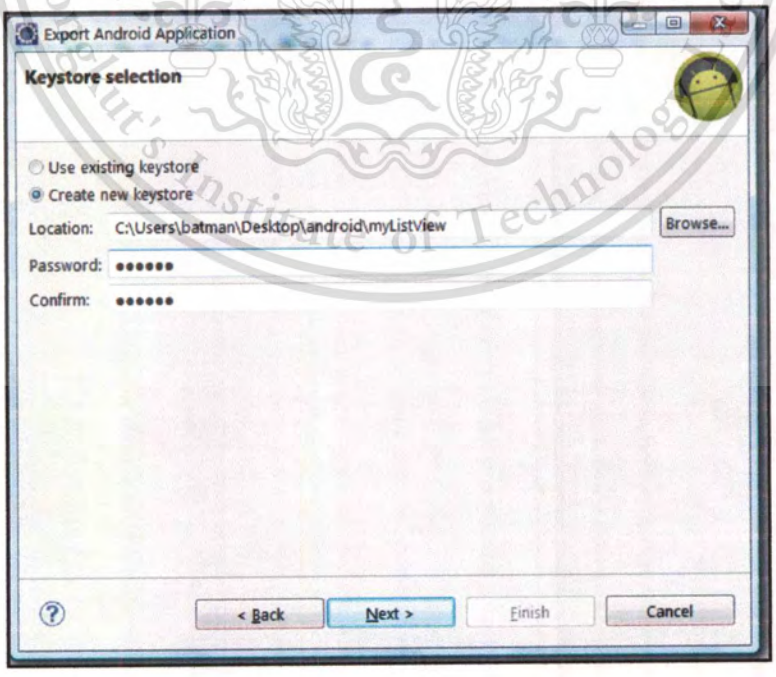


***Figure A.3** Information in Export Android Application*

3.) Fill information for Application detail

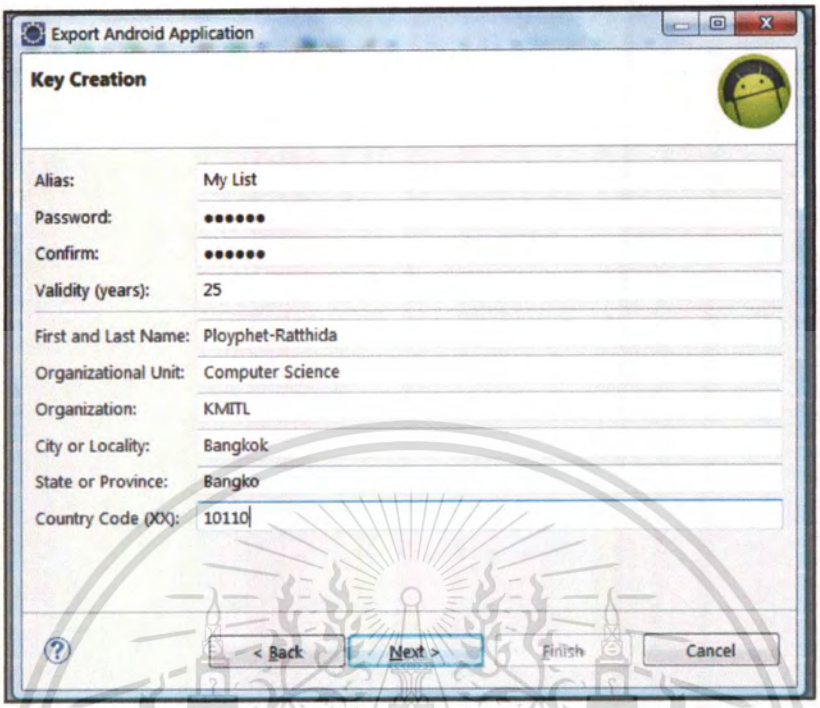


Figure A.4 Key Creation on Export Android Application

4.) choose location for .apk file.

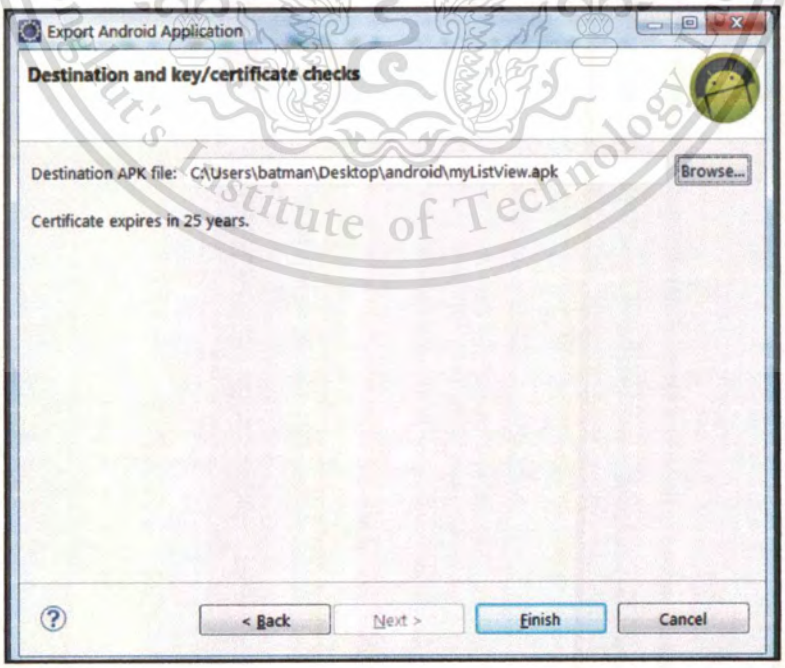


***Figure A.5** Destination APK file*

5.) After choose finish, .apk file appear in destination. Then synchronize with android mobile and Lunch program.

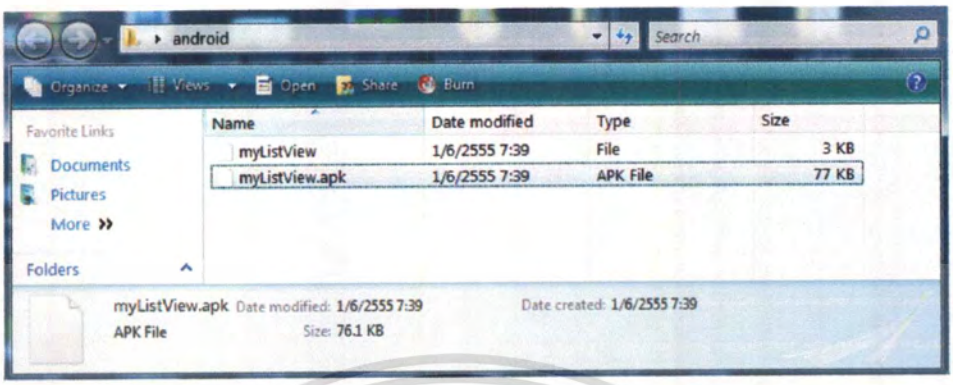


***Figure A.6** Destination Folder*

Appendix B

Multicriteria and Multidimension technique

Multicriteria and Multidimension technique

1. Create a Profile of each type by separate data follow Multicriteria and Multidimension.

2. Create User profile from analyze both Multidimentional and Multicriteria together. Record User profile by correct rating of each user in 2 parts

2.1 Part of Multicriteria, correct rating when user give a overall Rating. System let user give a rating by number of star (1-5 arrange in order from least to greatest) if user give total of rating score more than 4 system will record data in User Profile for one point follow Multicriteria that define in Song profile. But if total rating score is least than 4 system will not record any score in User profile, follow figure B.1 that show 3 criteria in User Profile that users is not rate data yet. Figure 3.2 show User profile after user give total rating score equal 3 for "We found love" song than system record a score in criteria: 1.Pop Genre in section1 2.Gender of Artist

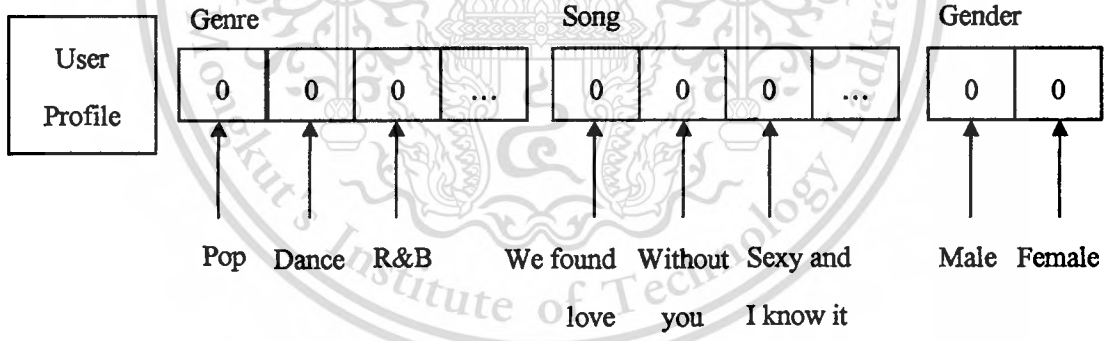


Figure B.1 A starter profile without any rating by using Multicriteria

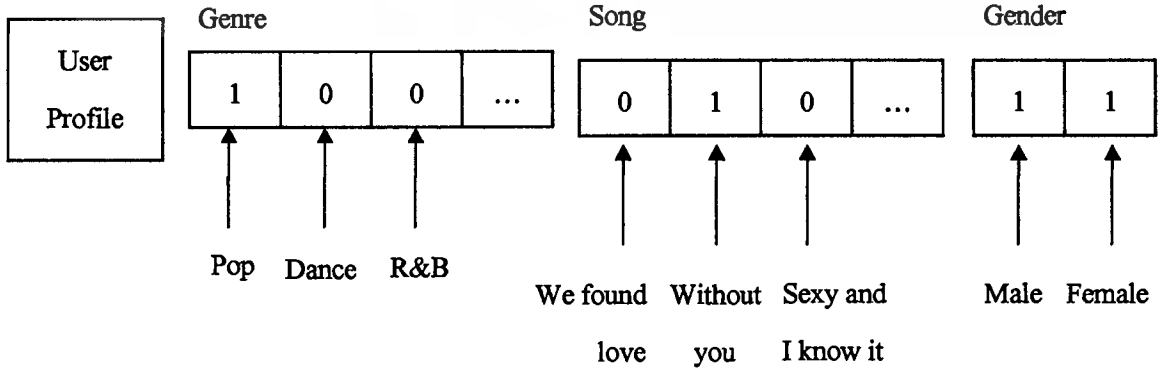


Figure B.2 How to store score after user rate some data by using Multicritiria

2.2 Part of Multidimension correct rating when user gives a rating follow dimension of each song. Each dimension give 1-5 arrange in order from least to greatest so it have only one case that system will correct rating score is user give all dimension in 5 star. But if user gives 1-4 star system will record a score equal to 0. Figure B.3 show a starter User profile that user is not give a rating yet. And in Figure B.4 Show a score rating that user give for genre.

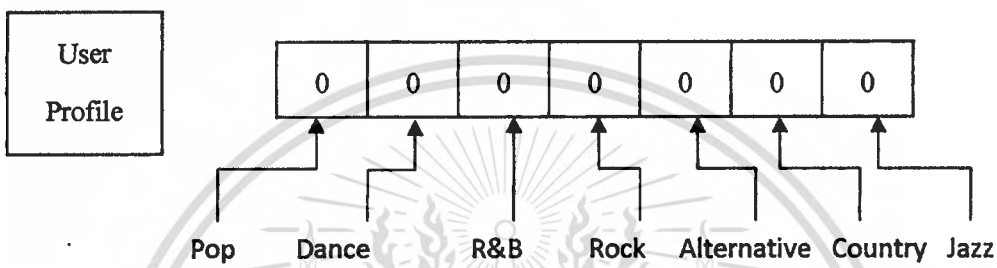


Figure B.3 A Starter profile without any rating by using Multidimension

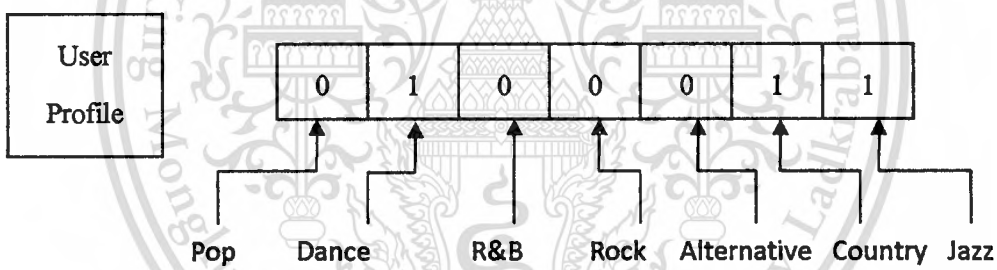


***Figure B.4** How to store score to user profile after rate some data by using Multidimension*

3. Normalize to User Profile in range 0-1 after keep rating score of user finished.By separate in 2 step follow: 1. Normalize by using Multicriteria 2. Normalize by using Multidimension

3.1 Normalinze by using Multicriteria is collect all score that user rate and divide by number of times that user have rate to User profile. For example, user rate 15 times so every field of criteria is divide by 15 follow Figure B.5

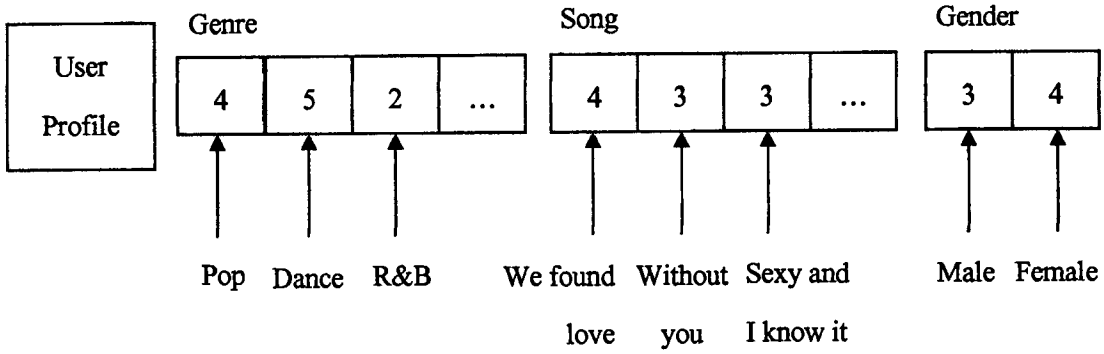


Figure B.5 Show when user give rating 15 times by using Multicriteria

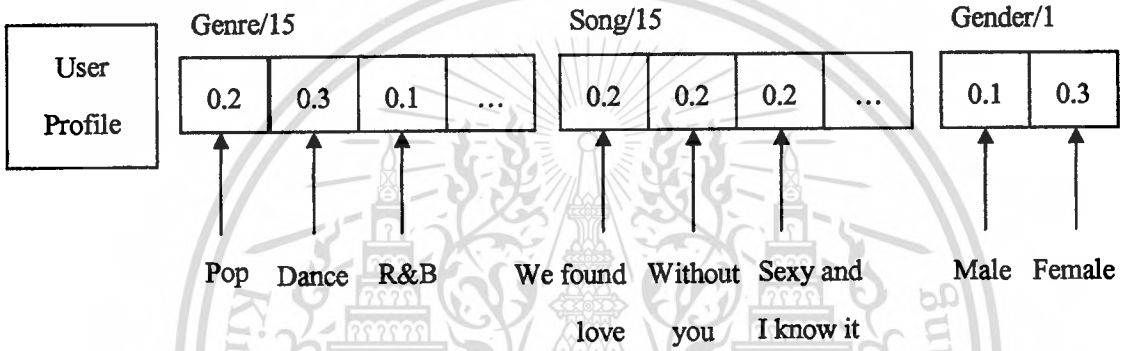


Figure B.6 using a figure 3.5 data to normalize by using Multicriteria

3.2 Normalize by using Multidimension, use same process with Normalize by using Multicriteria. For example total rating is follow figure B.7 and result after divide is figure B.8

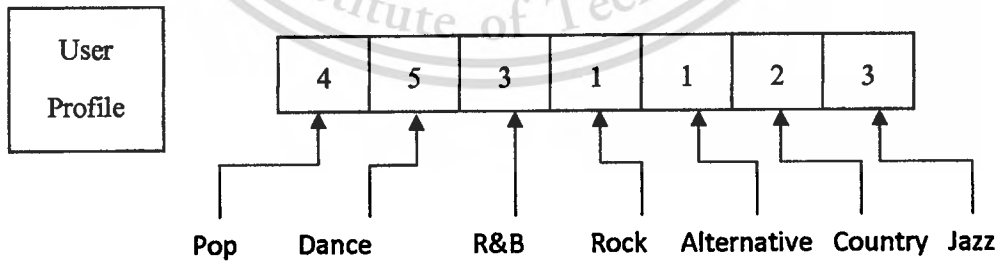


Figure B.7 Show when user give rating 15 times by using Multidimension

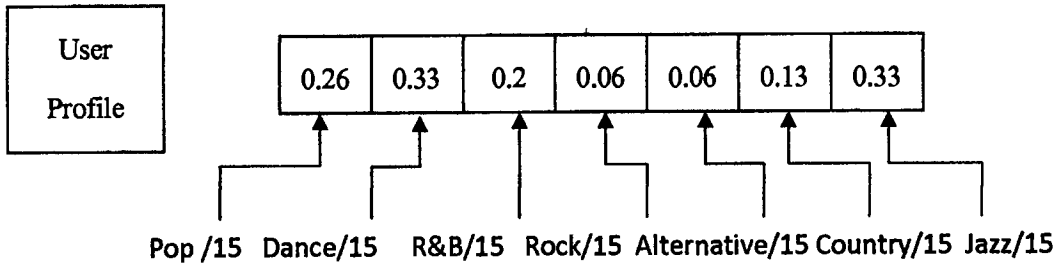


***Figure B.8** using a figure B.5 data to normalize by using Multidimension*

4. Weight of Multicriteria and Multidimension

4.1 Calculate weight of each criteria by consider from a highest value in each criteria follow figure B.9 And then calculate weight by compare with a position that have a highest value in criteria follow equation

$$\text{Weight for Genre} = \frac{0.33}{0.26+0.30+0.13} = 0.43$$

$$\text{Weight for Song} = \frac{0.26}{0.26+0.20+0.20} = 0.39$$

$$\text{Weight for Gender} = \frac{0.33}{0.13+0.33} = 0.71$$

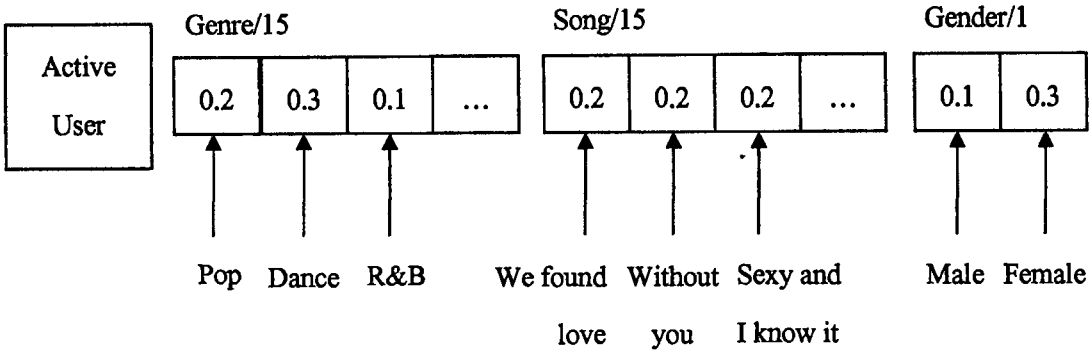


***Figure B.9** Show that highest of each criteria in Multicriteria*

4.2 Calculate weight of each dimension by use values from figure B.8 divide by value of another dimension follow equation

$$\text{Weight for Pop genre} = \frac{0.26}{0.26+0.33+0.2+0.06+0.13+0.33} = 0.19$$

$$\text{Weight for Dance genre} = \frac{0.33}{0.26+0.33+0.2+0.06+0.13+0.33} = 0.25$$

$$\text{Weight for R\&B genre} = \frac{0.2}{0.26+0.33+0.2+0.06+0.13+0.33} = 0.64$$

$$\text{Weight for Rock genre} = \frac{0.06}{0.26+0.33+0.2+0.06+0.13+0.33} = 0.04$$

$$\text{Weight for Alternative genre} = \frac{0.06}{0.26+0.33+0.2+0.06+0.13+0.33} = 0.04$$

$$\text{Weight for Country genre} = \frac{0.13}{0.26+0.33+0.2+0.06+0.13+0.33} = 0.09$$

$$\text{Weight for Jazz genre} = \frac{0.33}{0.26+0.33+0.2+0.06+0.13+0.33} = 0.25$$

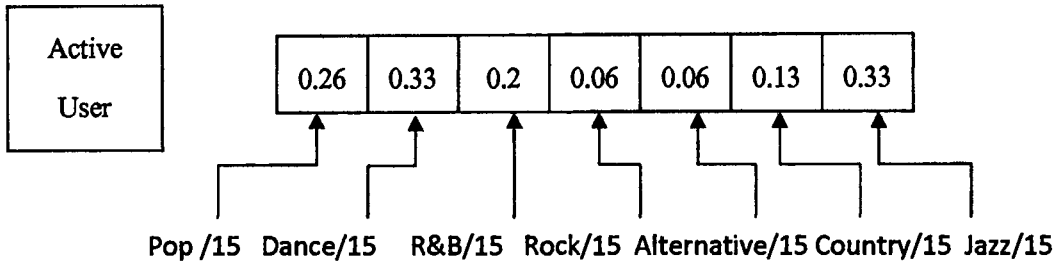


***Figure B.10** Show values that use for finding weight by use Multidimension*

5. After calculate weight of Multicritiria and Multidimension, use value of weight and multiply with total values of criteria and dimension for weight User Profile that can separate in 2 ways :

5.1 Multicriteria multiply, use a weight that calculates in step 4.1 multiply by every criteria. For example weight of Genre is 0.43 use this value multiply by every field of genre follow figure B.11 use weight of Genre multiply by Pop Genre that have value equal 0.26(Figure B.9) and after multiply values is become 0.1118 ,Weight of Genre 0.43 multiply by Dance Genre 0.33 result is 0.1419 For R&B Genre multiply by weight of genre result is 0.0559, etc.

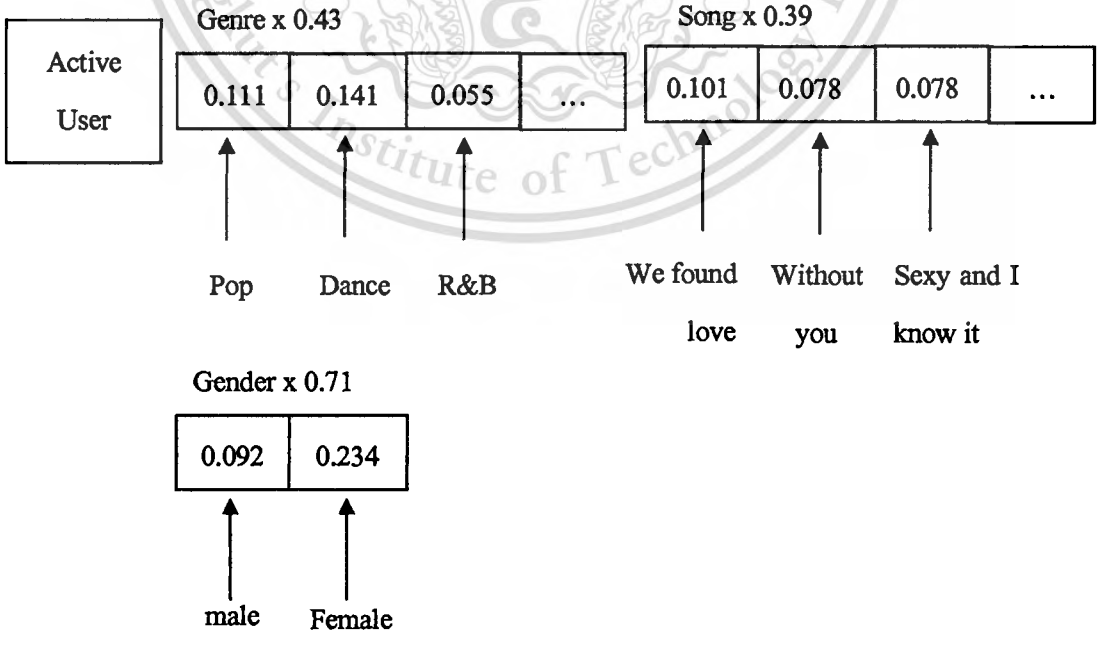


***Figure B.11** Show step that use weight of each criteria from step 4 multiply by every field of criteria by using Multicriteria*

5.2 Multidimension multiply, use a weight from step 4.2 multiply by every dimension. For example,

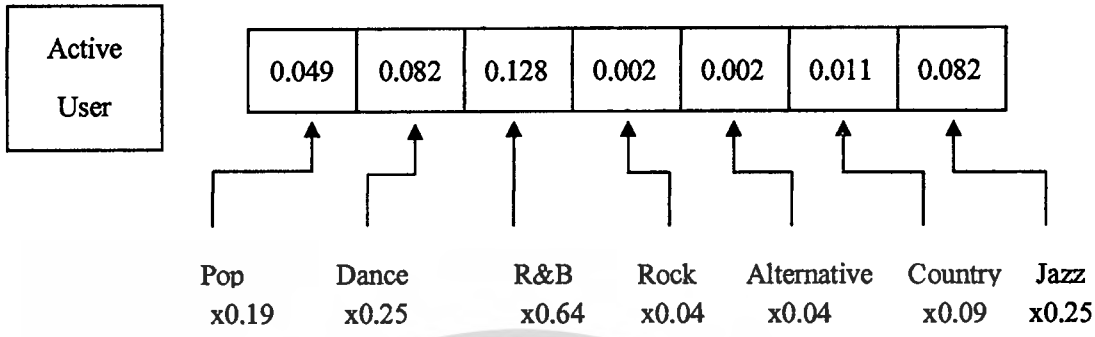
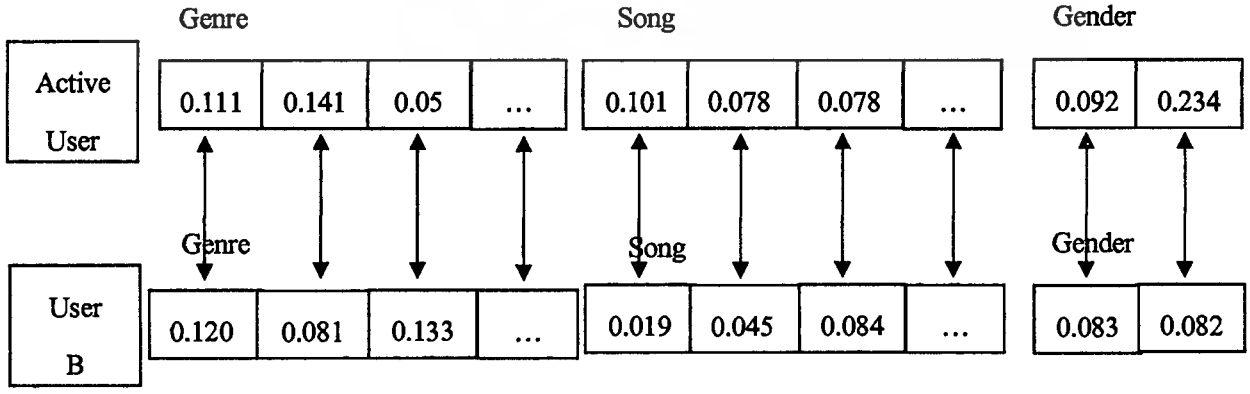


Figure B.12 Show how to calculate weight of each dimension from step 4 to multiply by dimension by using Multidimension

6. Calculate a distance of satisfaction between Active User and user B. By adjust value in User profile of user B with step1-5 to make it straight with Active User. After that find a difference of each position between Active User and user B then find average of all difference value that called "Euclidean Distance". Euclidean Distance is values that show a distance of satisfaction between Active User and user B. For Multicriteria can calculate follow this equation $Dc = \frac{\sum_{i=0}^{n} |ai - ai|}{n}$ whereas Dc is a difference of Multicriteria between Active User and another user in system. n is number of field in Multidriteria. ai is field of position i of Active User and ai' is position of user. Follow figue 3.13



Figue B.13 Calculate a distance of satisfaction between Active User and user B

To calculate value of Euclidean Distance of Multidimension is $Dd = \frac{\sum_{i=0}^{m} |ri - ri'|}{m}$ whereas Dd is a difference value of Multidimension between Active User and another user in system, m is number of total field of Multidimension, ri is field in osition I of Active User, ri' is field position I of another user follow figure B.14

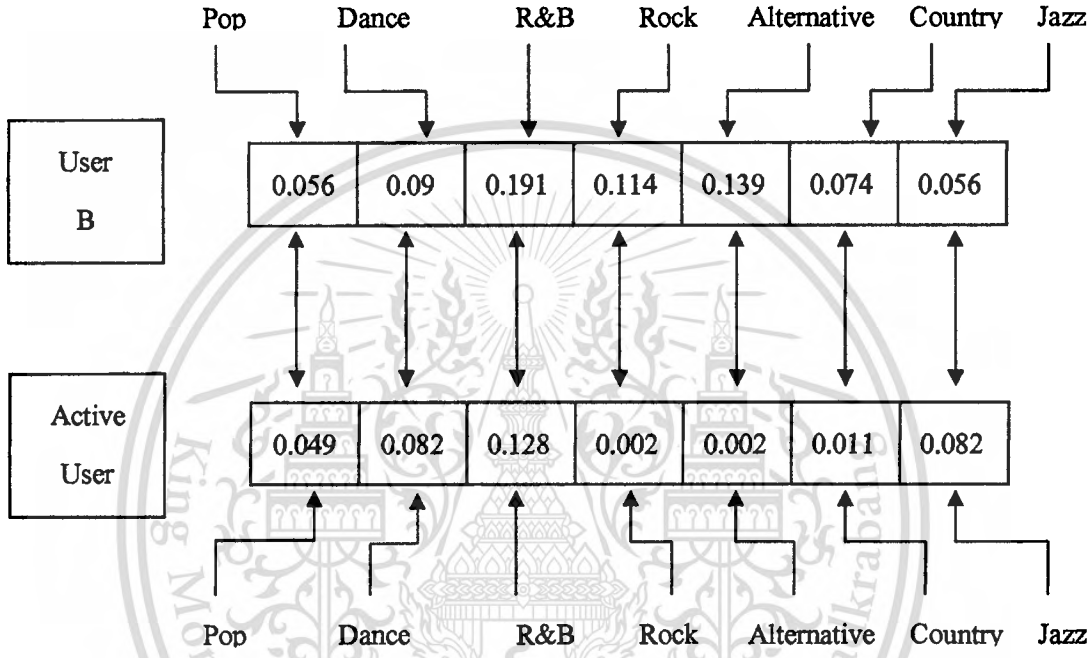


***Figure B.14** Calculate satisfaction between Active User and user B by using Multidimension*

7. After calculate Euclidean Distance of Active User and User B. Use that values calculate follow equation3.1 that use Multicriteria to find a similarity between Active User and User B

$$Sc = \frac{1}{1+Dc}$$ Equation B.1

Where Sc is a similarity between Active User and User B in Multicriteria and Dc is a Euclidean Distance result in Multicriteria. And equation 3.2 is calculate in Multidimension to find similarity between Active User and User B

$$Sd = \frac{1}{1+Dd}$$ Equation B.2

Whereas Sd is a similarity values between Active User and User B in Multidimension and Dd is a Euclidean Distance result in Multidimension

8. Use Sc and Sd multiply by weight that can separate in Multicriteria and Multidimension :

8.1 Multicriteria: Sc multiply by 0.3

8.2 Multidimension : Sd multiply by 0.7

And result of similarity (Stotal) is Stotal = 0.3Sc + 0.7Sd and sort Stotal to choose a neighbor that have Stotal similar to Active User maximum 5 user.

9. After find a neighbor maximum 5 user that most similar to Active User, calculate affection rating that Recommender system keep for each song and send it to Active User. Rating is separate in 2 parts; in first part is use affection rating that neighbor rate for each song to calculate weight Average,Weight is a similarity between Active User and neighbor . Second part is genre of song in database: pop, dance, R&B, rock, alternative, country, jazz that show in equation 3.3

$$\text{Satisfaction values from system}_{a,j} = \frac{\sum_{i=1}^{N}(Sa,iRi,j)}{\sum_{i=1}^{N}(Sa,i)} + \text{AVG}_{a,j}$$ equationB.3

Sa,i is a similarity between Active User and Neighbor i , $R_{i,j}$ is a total score that neighbor i rate to song j. For example if need to know a rating for "We found love" song the Satisfaction values from

$\text{system}_{a,\text{we found love}}$ so $S_{a,1}$ is a similarity between Active User and neighbor 1 $R_{1,\text{we found love}}$ is a total rating that neighbor 1 rate for "we found love" song. Sa,2 is a similarity between Active User and neighbor 2 $R_{2,\text{we found love}}$ is a total rating that neighbor2 rate for "we found love". Then calculate for

AVG by consider Item Profile in Multidimension and satisfaction of all user with dimension of "we found love" after that will get average values of satisfaction from every user in system to "web found love" song

This Recommender is recommend for Active User so use weight of User profile in Multidimension of Active User to weight a average, show in equation

$$AVG=(W_{a,pop}\times avg_{pop,j})+(W_{a,dance}\times avg_{dance,j})+(W_{a,R\&B}\times avg_{R\&B,j})+(W_{a,rock}\times avg_{rock,j})+$$
$$(W_{a,alternative}\times avg_{alternative,j})+(W_{a,country}\times avg_{country,j})+(W_{a,jazz}\times avg_{jazz,j})$$

$W_{a,\,pop}$ is weight from User profile in Multidimension and pop genre avgpop is a average of total rating that users give for j song and for wa,another dimension and avgof dimension,etc. have the same appearance.

In calculate $AVG_{a,we\ found\ love}$ show in figure B.15 by vector in top of figure is average that users give to each dimension of "we found love". Another vector in figure is weight of Multidimension of User Profile for Active User

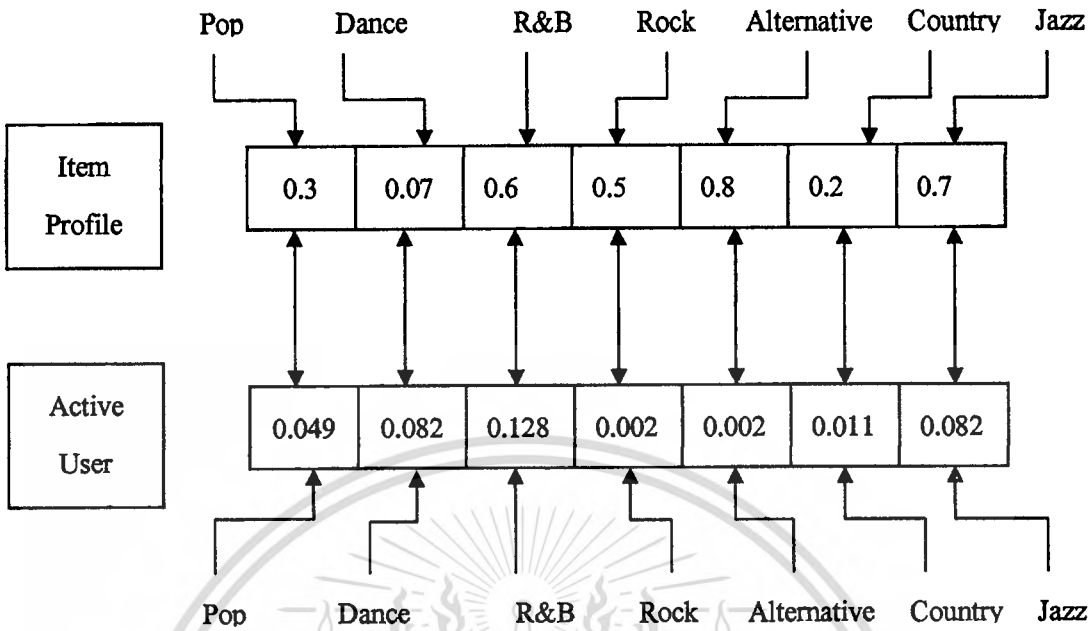


***Figure B.15** Calculate satisfaction by using Multidimension*

So, calculate of $AVG_{a,we\ found\ love}$ is

$AVG_{a,we\ found\ love} = (0.3x0.049)+(0.7x0.082)+(0.6x0.128)+(0.5x0.002)+(0.8x0.002)+(0.7x0.082)$