ChordBook: A Portable Guitar Chord Song Book Using Crowdsourcing Techniques

Chakkrit Tantithamthavorn[1], Papon Yongpisanpop[2], Masao Ohira[2],
Arnon Rungsawang[1] and Kenichi Matsumoto[2]

[1]*Department of Computer Engineering,*
Faculty of Engineering,
Kasetsart University, Bangkok, Thailand
{b5105256, fenganr } @ ku.ac.th

[2]*Graduated School of Information Science,*
Nara Institute of Science and Technology,
Nara, Japan
{ papon-y, masao, matumoto } @ is.naist.jp

Abstract

Recently guitar players routinely rely on Internet guitar chord archives to find songs that they want to play. However, a wide range of use of the guitar chord archive which is collecting a piece of guitar chord song sheets sometimes results in creating overlapping same songs and increasing derivative songs with many versions. For instance, UltimateGuitar.com, which is the largest guitar chords and tabs archives, still confronts with this kind of problems. In this paper, we propose ChordBook that is a system to allow users to collaboratively create/edit/share guitar chords. ChordBook applies crowdsourcing techniques including a distributed version control mechanism to solve the problem of repeatedly created different versions of the same song. In the example scenario section of the paper, we show the usage of ChordBook to present how the system resolves the problem by using crowdsourcing techniques and distributed version control mechanism.

Key Words: chord archives, crowdsourcing, collaborative music creation, distributed version control

1. Introduction

As a common practice of today's guitarists, they use a guitar chord songbook, which is a small collection of song sheets in a book, to play a guitar. Figure 1 shows a song sheet which is composed of lyrics and guitar chords to identify the tone of music. Since the popularization of the Internet, online music communities have exponentially increased and then made enormous resources of volunteered music information open to public [4]. In particular, due to the major advances in technology development together with the emergence of Web 2.0, it is now possible for common users to build a large online collection of song sheets which called Chord Archives.

There is a wide range of technologies being used in creating online chord archives. The most famous website is the Ultimate Guitar Archives [10]. Currently, they have over 300,000 song sheets with various kinds of music instruments. With the increased number of registered users and song sheets, however, overlapped (same) songs and many derivative versions are sometimes registered to the archives as shown in Figure 2. As a result, users are facing with the difficulty in choosing better one for them. Moreover, if some parts in a guitar chord are incorrect, guitarists cannot continue to play the guitar chord. Since the current guitar chord archives do not provide the permission to edit an archived chord by anonymous users, the online music communities confront with a severe shortage of contributors and lack of collaboration among guitarists.

Creating online chord archives with explicit collaboration among guitarists presents several challenges: *how can they avoid conflicts of their musical ideas when they try to edit one chord by several guitarists (users)?, how can they choose the*

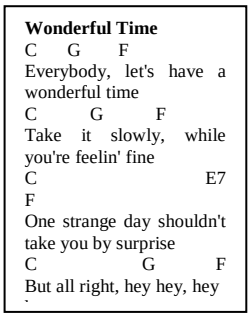
Wonderful Time
C G F
Everybody, let's have a wonderful time
C G F
Take it slowly, while you're feelin' fine
C E7
F
One strange day shouldn't take you by surprise
C G F
But all right, hey hey, hey

Figure 1.
An example of a song sheet with guitar chords

Figure 2.
Overlapping song titles in an online chord archives

better chord from a variety of derivative versions as a representative chord in the online music community?, How can each guitarist preserve a guitar chord with own music style in the community? And so forth.

In this paper, we present ChordBook, a portable guitar chord song book running on iPad. One of the primary advantages of an iPad based application with Internet connectivity is the ability to access to chord archives anytime and anywhere. ChordBook addresses the above challenges in creating online chord archives, based on the two ideas as follows. First, we use a crowdsourcing model to create chord archives by the online music community as a nature of collaboration. ChordBook help users collaboratively compose a chord without a single composer. Second, we apply the distributed version control concept to allow users to control versions of songs inside ChordBook, which means that the system supports the interoperability of their music ideas.

The remainder of this paper is structured as follows. Section 2 gives background knowledge on a crowdsourcing system and a distributed version control system. After establishing the conceptual foundation of our approach, we present the key design principles behind ChordBook in Section 3. We then illustrate an example usage scenario in Section 4. Section 5 discusses further challenges in the future and we conclude the paper in Section 6.

2. Background Knowledge and Concept

There are three fundamental knowledges and concepts behind the design of ChordBook; Crowdsourcing, Distributed Version Control and Guitar Chord Data Format.

2.1 Crowdsourcing

The idea of using crowds of people for dedicated purposes has been discussed by Surowiecki [9]. The term of crowdsourcing has been employed by Howe [7] later in 2006.

Crowdsourcing [6,7,8] is a new paradigm for utilizing the power of "crowds" of people to facilitate large scale tasks that are costly or time consuming with traditional methods. It can externalize the risk of failures and it only pays for products or services that meet its expectations. In recent years, systems with the crowdsourcing techniques have been significantly increasing. The open source software projects such as Apache and Mozilla which have been creating quality software products through collaboration among volunteered developers over the world is regarded as the first real crowdsourced Internet project that gained a great success. Another example includes Wikipedia which continues to be one of the biggest online encyclopedias through the mass collaboration of users.

2.2 Distributed Version Control System

A distributed version control system (e.g., Github, Mercurial and Bazaar) is widely used to keep track of software revisions in software development. It allows geographically distributed developers to work on a given project together without necessarily being connected to a common network. Developers have their own local repository, which can be edited without any permission and change to the project's central repository.

Distributed version control system focuses on sharing changes to software artifacts. Every change is managed by having a unique id. Using a distributed version control system in software development brings several advantages to developers as follows:

- **Everyone has a local sandbox.** You can make changes and roll back on your local machine.
- **It works offline.** You only need to be online to share changes. Otherwise, you can stay on your local machine, checking in and undoing, no matter if the "server" is down or you are on anywhere.
- **It is fast.** Every commit and revert are done locally. There is no shaky network or server to ask for old revisions from a year ago.
- **It handles changes well.** A distributed version control system is built to share changes. Every change has a guide which makes it easy to track.
- **Branching and merging are easy.** Because every developer can have their own branches (derivative versions), every shared change in the central server repository is treated as reverse integration. To make this easy, the system has features to automatically combine changes and avoid duplicates.
- **Less management.** The system allows developers to feel free to use it; there is no "always-running"

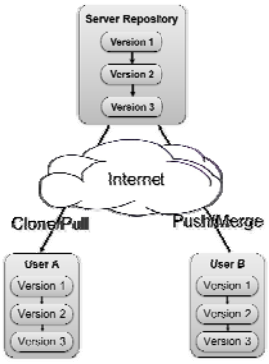

Figure 3.
Workflow of a distributed version control system

server software to be installed. Also, it may not require you to "add" new users; you can just pick URLs to pull from. This management style leads to avoid political headaches in large projects.

Figure 3 illustrates a workflow of the distributed version control system. Users have a central server repository to collect a latest version called master version. To interact with the sever repository, the system has useful features including:

- **Clone** downloads source code from the server repository to your local repository.
- **Pull** gets updates local repository to the latest version of the server repository.
- **Push** uploads or commits a source code from local repository to the server repository.
- **Merge** resolves a conflict changes from multiple branches into single version.

We apply the concept and mechanism of the distributed version control to online chord archives in order to support collaboration among guitarists.

2.3 Guitar Chord Data Format

To store a guitar chord in a database, we use a simple ASCII format for transcribing songs with guitar chords and lyrics as shown in figure 4. Although this format is legible as it is, there are many popular programs for displaying, transposing and printing.

Chords are delimited by square brackets. If the chord name follows a known system a program can properly interpret it. Non-standard names are allowed though. Curly braces delimit directives. They are used to indicate meta-data and to mark sections of a song. Lyrics are any text that is not otherwise differentiated.

```
{title:Wonderful Time}
{subtitle:Words and Music (C) 2007 Adam Monsen}
[C]Every[G]body, [F]let's have a wonderful time
[C]Take it [G]slowly, [F]while you're feelin' fine
[C]One strange [E7]day shouldn't [F]take you by surprise
[C]But all right, [G]hey hey, [F]hey hey
{start_of_chorus}
[C]Ahhh [Am]ahhh [G]ahhh
[C]Ahhh [Am]ahhh [G]ahhh
{end_of_chorus}
```

Figure 4. Guitar chord data format

3. ChordBook Design Principles

In this section, we present the design of ChordBook which is using the crowdsourcing techniques and the distributed version control concept to support collaborative creation of guitar chords among guitarists.

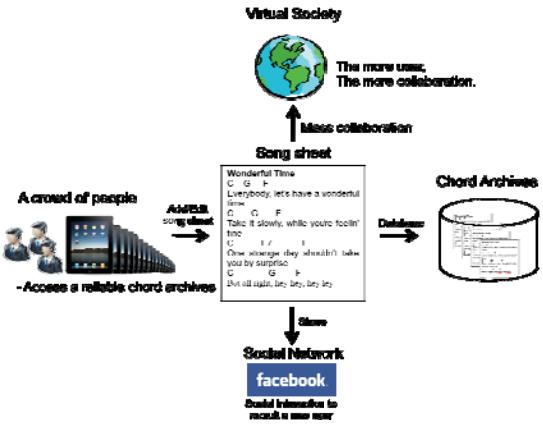

Figure 5. ChordBook Overview

3.1 ChordBook Design Overview

Figure 5 shows an overview of the design concept of ChordBook which is applied the Crowdsourcing technique. Users add and edit a song sheet from an iPad application (client) with a user friendly interface. Additionally, they can share to his friend via social network such as facebook or twitter in order to recruit a new contributor. Finally, all of song sheets are collected in a database that can later be a huge chord archives for guitarist. ChordBook utilizes client-server architecture by using iPad as a mobile client to access song sheets from the server.

3.1.1 Client Architecture

We use iPad to install our application, which will be able to handle more user experiences than web application on a mobile platform [2]. We can decompose client architecture into nine roles.

Revision: Due to a distributed version control system, each developer has his/her own repository. We apply this concept to the ChordBook client by exploiting SQLite as a local database on iPad to keep track of versions of song sheets as an ASCII text based format as we describe in section 2. A song sheet might be composed by volunteered guitarists (a system's user) around the world without one single composer. If others read a song sheet and notice something is wrong or if they have more information on the song, they are able to edit the song or provide more in-depth explanations. Moreover, they can add a note about the song on the corresponding discussion area. In this way the information is continuously verified and enriched by different users who share the same interest. On the other hand, because of this feature, the song sheets are also exposed to vandalism by malicious users. We consider this issue to be resolved based on the

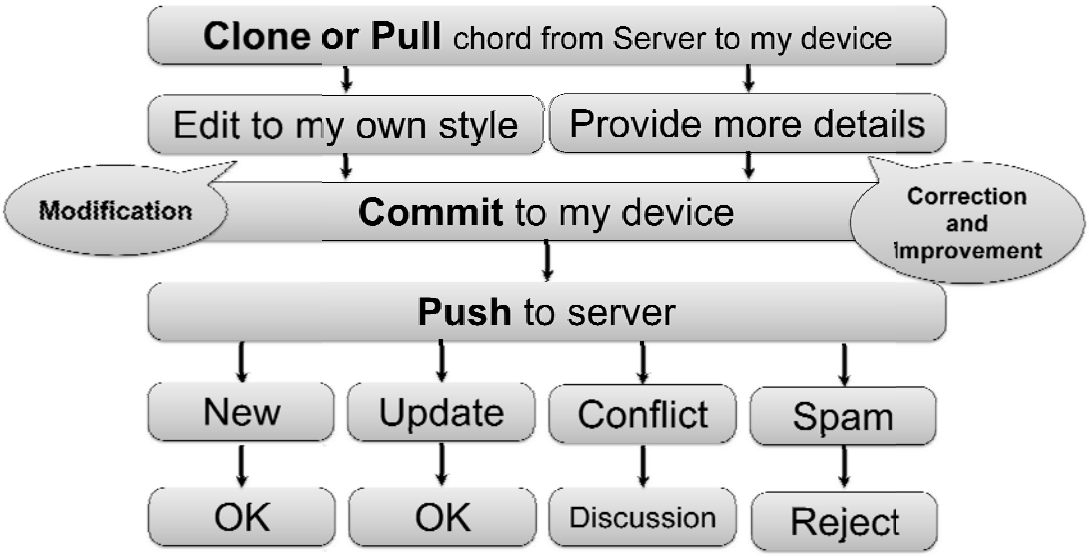

Figure 6. Workflow of ChordBook Application

crowdsoursing mechanism, because all song sheets are peer reviewed (often very quickly such as Wikipedia) and anyone can report malicious users who unnecessarily remove a song sheet and/or arbitrarily edit it. The system also provides a feature to easily revert to the last proper version of that song sheet.

Searching: ChordBook can suggest a sample search result from a user query and sort by title, artist, album names, and genre.

Transposing or Key Changing: You can transpose a song from its original key in order to keep its highest and lowest notes within a comfortable range on your instrument, or possibly to accommodate the vocal range of a singer.

Automatic Scroll: ChordBook provides you an automatic scroll for guitarists to play the song without manually scrolling down the song sheet.

Social Network: You can share or recommend a particular song sheet to your friends via social network such as Facebook, Twitter and etc.

User Evaluation: From a song sheet that you download or clone from server, you can see a discussion area to see further information and idea. Moreover, you can report some comments as a spam to send a remove comment request to a moderator.

Updated: There are multiple ways to inform to users when a song sheet is changed: it can be placed on a special watch list, an e-mail with the changes can be sent or RSS-feeds and IRC-channels can be used, to just name a few.

Statistics: A moderator of each chord archive can see the basic statistics such as top views.

Online and Offline access: Once access a song sheet, the ChordBook application will cache a song sheet every 24 hours into a local database (SQLite) in the particular format as we described earlier.

3.1.2 Server Architecture

We use Ruby on Rails as a web framework to develop a server side to store a huge collection of song sheets. We use “acts_as_versioned plug-in” [1] to manage different versions of guitar chord song sheets so that users can revert to a previous version or create new ones. We use “device plug-in” [5] for authentication solution. The central server communicates with an iPad client through a GET/POST request from HTTP Protocol with the JSON data format.

3.2 ChordBook Workflow

Figure 6 illustrates a workflow of the ChordBook application. Firstly, a user starts with “Clone” to download song sheet from the server to iPad. The user also can “Pull” a song sheet from local and/or master versions. After the user prompts “clone”, if s/he notices that something is wrong or if s/he would like to have more information on the song, then the user can edit the song, provide more in-depth explanations or discuss it among users. During the mediation, you can “commit” to your device to keep track of versions in local repository in your device. If the user prefers to distribute or publish his guitar chord to the Internet, you can “push” the guitar chord to the server. If the server side finds a related song title name, the server will return a set of the result to the user. The user has to decide which one is a better title for the song. We believe that the human decision can solve the problem of same/similar song titles

repeatedly created. If a chord is completely new one, the server will accept user's request automatically. For an updated version, the server will check it first that the song version is the latest or not. If it is the latest, the server will also accept user's request. If not, this version will be conflict and users will be guided to have discussions among users. ChordBook also have two ways to handle a spam and malicious users. First, the system allows users to report a spam and then will send a request to a moderator. Second, if the system detects some frequently changed or updated guitar chords within minutes, it will lock the song to be edited automatically and stop the update from users for 24 hours.

3.3 Advantages of Using Crowdsourcing

Since crowdsourcing practices have numerous characteristics, the advantages will be largely dependent on the type of crowdsourcing under consideration. We can differentiate the advantages of crowdsourcing in several aspects as below.

Low Cost: ChordBook uses the crowdsourcing techniques especially regarding the building and sharing a guitar chord song sheet by voluntary work among users. This would result in collaborative chord creation at relatively low cost.

High Quality: Addressing a mass of skilled individuals through an open call is a proven approach to problem solving, as illustrated by the numerous tasks from Amazon Mechanical Turk [3]. In this case, the quality of guitar chord song sheet refers to the originality and enables us to profit from individual ideas around the world.

High Motivations: According to the need of the new type of a guitar chord song book, we propose ChordBook which is much more conveniences to guitarists. Through explicit collaboration among users, they might feel that they owns a part of the ChordBook, thus they are enforced to contribute to ChordBook, as open source projects have succeed to do so. ChordBook also provides the ways to establish, measure, and show user's reputation by ranking his effort and popularity of his song sheets. This might further lead to the dedicated collaboration among users.

4. Example Usage Scenario

In this section, we illustrate our application usage scenario [Figure 7] apply our approaches by showing our collaborative function, which is really useful to guitarist.

Step 1: Alice creates new song "Hotel California" using ChordBook application in her own iPad.

Step 2: She edits the song by typing lyrics and follow by guitar chords.

Step 3: She could not finish the rest of the song because she lack of skill. Then, she push or simply called upload her song sheet to server to initiate the first master version.

Step 4: Bob is a volunteer to continue her song. He starts by clone the song sheet from server to his iPad.

Step 5: He provides more details for the rest of the song.

Step 6: After he finished, he need to push or simply called upload to server to keep updated to be the latest version. Figure 8 shows the last result from iPad application.

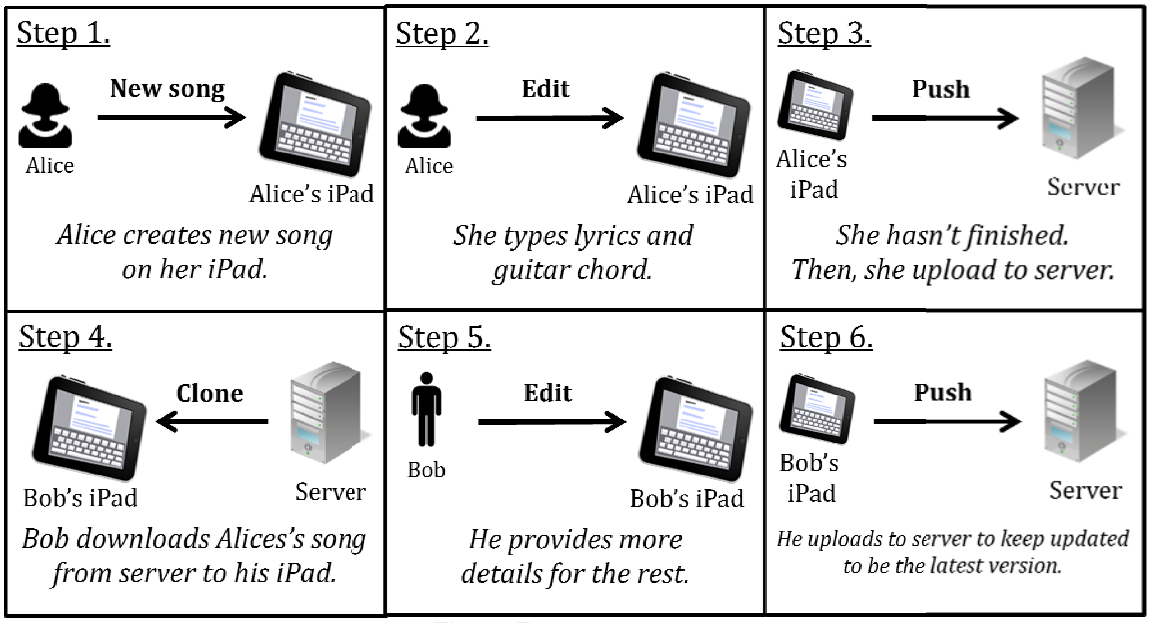

Figure 7. Usage Scenario

By using these steps, Guitarists will be able to collaborate on the song and avoid the duplication of the song.

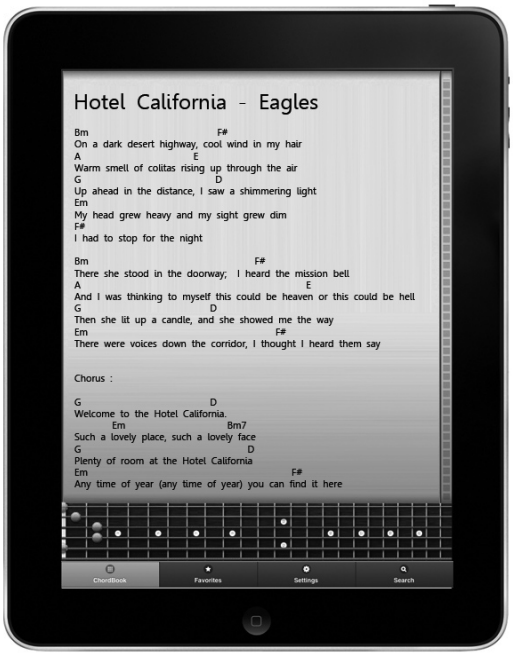

Figure 8. ChordBook Interface on iPad

5. Discussion

In this section, we discuss about how to avoid the duplication of songs title when there are more than one guitarists want to create the song and commit it into chord archives database. As we have mentioned before in section 3.1.1, our ChordBook is built on top of Distributed Version Control mechanism, which will be able to control version of songs inside the server. The same song can not be created inside the chord archives but it can have many type of the song as shown in figure 9. For example, Hotel California will be able to create just once but ChordBook allows Hotel California to have many types of itself (eg. acoustic, classic and etc.) and inside the version control of Hotel California, branch can represent type of the song. ChordBook also allows client to work with his/her own version inside iPad without Push it to the chord archives. Most of the guitar chord archives allow users freely create many version of the same song and upload it to the server. But with the Distributed Version Control concept, before users create a song, system will check whether that song has already submitted. If there is one, the system will force users to clone that song instead of allow users to create a new submission.

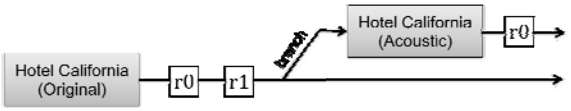

Figure 9. ChordBook Interface on iPad

6. Conclusion

We propose a new approach using crowdsourcing techniques and distributed version control to improve chord archives as shown in usage scenario in section 4. Moreover, ChordBook also provides more collaboration of guitarist to online music community. In the future work, we are also interested to apply other kind of music instruments into our system.

7. Acknowledgement

This research was supported by Nara Instituted of Science and Technology (NAIST), Japan and Kasetsart University, Thailand. Thanks to all researchers in Software Engineering Laboratory at NAIST for valuable comments. Thanks to anonymous reviewers for their comments and feedback.

8. References

[1] Acts as versioned association (http://agilewebdevelopment.com/plugins/acts_as_versioned_association)
[2] Charland, A. and Leroux, B. Mobile application development: web vs. native. Commun. ACM 54, 5 (May 2011), 49-53. 2011
[3] Kittur, A., Ed, H, Chi. and Suh, B., Crowdsourcing user studies with Mechanical Turk. In Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing *systems* (CHI '08). ACM, New York, NY, USA, 453-456, 2008
[4] K. M. L. Calvin and B. C. Y. Tan. The Internet is changing the music industry. Communications of the ACM, 44(8):68–68, August 2001.
[5] Devise (https://github.com/plataformatec/devise)
[6] Brabham, D., Crowdsourcing as a model for problem solving: An introduction and cases. The Journal of Research into New Media Technologies. 14(1), pp 75-90, 2008
[7] Howe, J., 2006. The rise of Crowdsourcing. Wired http://www.wired.com/wired/archive/14.06/crowds.html, June , 2006
[8] Howe, J., Crowdsourcing: Why the power of the crowd is driving the future of business. Random House Publishers, 2008
[9] Surowiecki, J., 2004. The wisdom of crowds. Anchor, 2005.
[10]Ultimate Guitar Archives (http://www.ultimate-guitar.com)