

ChordBook: ギターコード譜の協調的作成支援システム

タンチタウンタアワオン チャクリット^{†1} ヨンピサンポップ パボン^{†2}
トインタヌナム パタナモン^{†1} 大平雅雄^{†2}
ルンサワン アノン^{†1} 松本健一^{†2}

ユーザ同士による協調的なコード譜の作成を支援するシステム ChordBook を提案する。ChordBook は、クラウドソーシングの概念に基づいて設計し、ソフトウェア開発で利用されている分散型バージョン管理システムを基盤技術として実装したシステムである。iPad 上で動作するクライアント側アプリケーションと、分散環境下でのファイル管理をおこなうためのサーバーからなる。本稿では、ChordBook の設計と実装について詳述するとともに、ChordBook の利用シナリオについて述べる。

ChordBook: A System that Supports Collaborative Guitar Chord Sharing

CHAKKRIT TANTITHAMTHAVORN,^{†1}
PAPON YONGPISANPOP,^{†2} PATANAMON THONGTANUNAM,^{†1}
MASAO OHIRA,^{†2} ARNON RUNGSAWANG^{†1}
and KENICHI MATSUMOTO^{†2}

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.

^{†1} カセサート大学 (タイ王国バンコク市)
Kasetsart University, Bangkok, Thailand
^{†2} 奈良先端科学技術大学院大学 (奈良県生駒市)

1. はじめに

近年のインターネット技術の発展により、ギター演奏の際に用いるギター用のコード譜 (図 1: ギターのコード進行を歌詞に対応させた楽譜) を、インターネット上のコードアーカイブからダウンロードし利用する形態が一般的になっている²⁾。コードアーカイブを提供するサイトは通常、コード譜をダウンロードし利用するだけでなく、ユーザ自らがアレンジを加え作成したコード譜を公開する機能を備えているため、ギター愛好家によるオンラインコミュニティとして人気を集めているものが多い。例えば、Ultimate Guitar Archive^{*1}には 30 万曲以上のコード譜が保存・共有されており、150 万人以上の登録ユーザがそれらを利用している。

一方、ユーザ数が増加するにつれて、ユーザがアレンジを加えた異なるコード進行を持つコード譜が同じ楽曲に対して作成され登録されることが珍しくなくなった (図 2)。結果的に、ユーザがコード譜をダウンロードする際に、自分の好みや演奏スタイルにあったコード譜を選ぶことが難しくなってきた。また、ユーザ自らが作成したコード譜を登録・編集することはできても、他のユーザが作成したコード譜を編集することは通常許可されていないため、Wikipedia のようなマスコラボレーションは生じず、特定の活発なユーザがコード譜を多数作成し登録しているというのが現状である。

本研究の最終的なゴールは、現在のオンライン上のコードアーカイブの仕組みをさらに一歩押し進め、ギター愛好者同士のマスコラボレーションを可能にするシステムを構築することである。そのためにはまず、以下のような課題をクリアする必要がある。

- 複数のユーザが同一のコード譜を編集する際に生じるアイデアの不一致を解消 (あるい

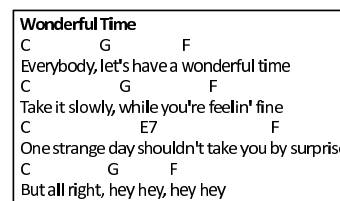


図 1 コード譜の例



図 2 登録楽曲の重複

Nara Institute of Science and Technology, Ikoma, Nara, Japan

*1 <http://www.ultimate-guitar.com/>

は回避)する手段を提供すること

- 複数の派生バージョンが存在するコード譜の中からコミュニティの代表となるバージョンを決定する手段を提供すること
- コミュニティ内での代表バージョンとは別に、各ユーザが自分の好みのバージョンを保持し続けるための手段を提供すること

マスコラボレーションの代表例として取り上げられる Wikipedia であっても、上記の課題は十分に解決していない。例えば、ある記事に対して見解の異なる複数のユーザが記事を頻繁に書き換えるといった争いが見受けられる。

本稿では、ユーザ同士による協調的なコード譜の作成を支援するシステム ChordBook を提案する。ChordBook は、クラウドソーシング^{1),4),5)}の概念に基づいて設計し、ソフトウェア開発で利用されている分散型バージョン管理システムを基盤技術として実装したシステムである。iPad 上で動作するクライアント側アプリケーションと、分散環境下でのファイル管理をおこなうためのサーバーからなる。次章ではまず、クラウドソーシングと分散型バージョン管理システムの概念と用語について説明し、3章において ChordBook の設計と実装について詳述する。4章では、ChordBook の利用シナリオについて紹介し、5章においてまとめと今後の課題を述べる。

2. 概念と用語

2.1 クラウドソーシング

「集合知 (wisdom of crowds)」は、少人数の専門家よりも一般の人々の意見や判断を集約した場合の方が時として正しいことがあるとして、Surowiecki が広く一般に紹介した概念⁷⁾である。その後、集合知の考え方をコラボレーションの側面を協調して発展させたものとして、「クラウドソーシング (crowdsourcing)」^{1),4),5)}という概念も広まりつつある。クラウドソーシングとは、従来の方法では多大なコストや時間のかかる大がかりな作業を、多数の人間による明示的あるいは暗黙的な協働によって解決する枠組みである。インターネット技術の進歩によりクラウドソーシングによって様々な種類のタスクや問題解決が行えるようになってきている³⁾。例えば Apache や Mozilla などのオープンソースプロジェクトは、クラウドソーシングの先駆けといえる存在であり、世界中の多数のボランティア開発者のコラボレーションを通じて、商用プロダクトに引けを取らない品質を備えたソフトウェアの無料利用を可能にしている。Wikipedia も、オンライン百科事典をユーザ同士のマスコラボレーションによって実現している点においては、クラウドソーシングの一種といえる。

きる。その他にも、ユーザに少額の謝金を支払うが数量を必要とするタスク (市場調査など)をおこなうための Amazon Mechanical Turk^{*1)}のようなシステム⁶⁾もクラウドソーシングの考え方に基づいたものである。

2.2 分散化型バージョン管理システム

ソフトウェア開発では従来より、CVS^{*2)}や Subversion^{*3)}をはじめとするバージョン管理システムが広く利用されている。ソフトウェア開発の過程で作成される膨大な数のファイルとその変更履歴を管理する必要があるためである。これら従来のバージョン管理システムは、中央サーバーのリポジトリから常に最新のバージョンをローカルマシン上にコピーし、かつ、開発者全員が最新バージョンに更新した状態で作業をおこなうことを前提として設計されているため、オフライン環境下で作業を開始したり、ローカルマシン上に複数のバージョンを作成して試行錯誤するといったことを個々の開発者が独断でおこなうことは困難であった。

そこで、従来のバージョン管理システムの制約を克服するために、Github^{*4)}や Mercurial^{*5)}、Bazaar^{*6)}などの分散型のバージョン管理システムが考案され、特にオープンソース開発者間で近年高い人気を集めている^{*7)}。一般に公開するための公開 (中央) サーバは存在するものの、中央集権型のバージョン管理ではなく、個々のユーザがローカルリポジトリを持ち作業をおこなう。任意のユーザ間で更新されたファイルを同期したり、特定のユーザのローカルリポジトリから公開サーバーのリポジトリに優れたバージョンのみを取り入れるといったことができる。分散型バージョン管理システムを用いることで、開発者は以下のような利点を得ることができる。

- すべてのユーザがローカルマシンに sandbox を持つことができる。すなわち、ファイルの更新やロールバックを各自のローカルマシンで自由におこなえる。
- オフラインでも動作し、変更を共有する際にのみオンラインで作業をすればよい。公開サーバーが停電などで停止していても個々の開発者はそれを気にせずいつでも作業がおこなえる。

*1 Amazon Mechanical Turk: <https://www.mturk.com/mturk/>

*2 CVS - Concurrent Versions System: <http://www.nongnu.org/cvs/>

*3 Apache Subversion: <http://subversion.apache.org/>

*4 Github - Social Coding: <https://github.com/>

*5 Mercurial SCM: <http://mercurial.selenic.com/>

*6 Bazaar version control system: <http://bazaar.canonical.com/>

*7 <http://betterexplained.com/articles/intro-to-distributed-version-control-illustrated/>に、従来のバージョン管理システムと分散型バージョン管理システムの違いが分かりやすい図解がある。

- ローカルマシン上で動作するため軽快である。従来のバージョン管理では常に最新のバージョンを中央サーバからダウンロードする必要がありネットワーク帯域によっては作業開始までに時間がかかったが、ネットワーク帯域を気にせずにすぐに作業が行える。
- 各ユーザの各変更ユニークな ID が付加されてバージョンが管理されるため、変更点の追跡（誰が開発したどのバージョンのどのファイルかなど）が容易になった。
- 公開サーバのリポジトリは各ユーザのブランチを逆統合するために、変更を統合したり複製を防止する機能が備わっているため、各ユーザが自身のブランチ（派生バージョン）を自由に持つことができる。
- システムはユーザが自由に作業できるようにし中央管理はおこなわない（特定のバージョンを利用することをすべてのユーザに強制しない）。厳正なユーザ管理（アクセス権の管理）もおこなわないため、大規模プロジェクトで生じがちなユーザ同士の揉め事も減らすことができる。

本研究は、このような利点をもった分散型バージョン管理システムを技術基盤として応用することで、前述した多数のオンラインユーザによる協調的なコード譜作成支援のための課題をクリアできるのではないかと考えた。

3. ChordBook の設計と実装

本章では、ユーザ同士による協調的なコード譜の作成を支援するシステム ChordBook を構築するにあたっての設計コンセプトと実装について説明する。

3.1 設計コンセプト

ChordBook は、前章で述べたクラウドソーシングの概念に基づいて、世界中のギター愛好家らが協調的にコード譜を作成・編集することを支援し、(従来の中央集権的管理ではなく) コミュニティ主導型のコードアーカイブ構築を促すことを目指している。図 3 は、ChordBook の設計コンセプトを図示したものである。iPad 上で動作するクライアント側アプリケーションを用いることで、ユーザは屋内外の利用環境を問わず、(1) コードアーカイブからコード譜をダウンロードし演奏に利用、(2) 自身が作成したコード譜をコードアーカイブにアップロード、(3) 他のユーザのコード譜を編集できるようにすることを想定している。また、これらの要件に加え、各ユーザが独自に変更を加えたコード譜を保持し続けることを可能にするために、オープンソース開発で近年急速に普及が進んでいる分散型のバージョン管理システムを技術基盤として応用する。さらに、ユーザを獲得する仕組みとして、特定の楽曲や音楽ジャンルなどで趣味趣向の合致するユーザ同士によるサブコミュニティ形成を支援する。

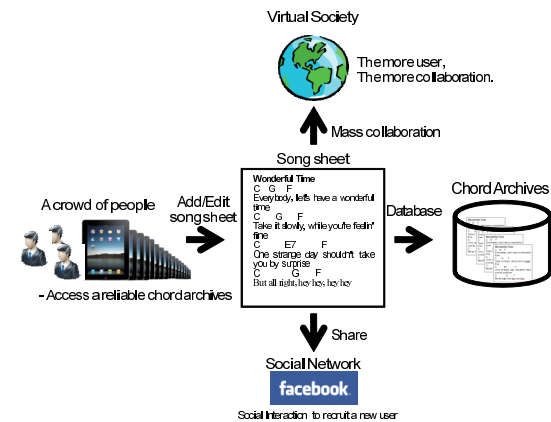


図 3 ChordBook の設計コンセプト

具体的には、Facebook などのソーシャルネットワークサービスとの連携をおこなう予定である。

3.2 クライアント

オープンソース開発で利用されている分散型バージョン管理システムは、Github のように Web アプリケーションとして実装され Web ブラウザ上で動作するものも多いが、ChordBook では、モバイル端末で動作する Web アプリケーションよりもリッチなユーザ体験を提供するために、クライアント側のアプリケーションは、iPad 上で動作するよう実装している。本クライアント側アプリケーションは、Git^{*1}を参考にした分散型バージョン管理システムをベースに実装されており、主に以下の 9 つの機能を備えている。

バージョン管理機能 iPad のクライアント側で各ユーザが保持するコード譜のバージョン管理をおこなう ChordBook の中心的機能である。一般的な分散型バージョン管理システムと同様に、各クライアントはローカルのリポジトリ（データベース）が実装されており、中央サーバ（コミュニティの代表バージョンを保存しておくためのコードアーカイブ）と同期をとりつつ、独自に作成・編集したコード譜を他ユーザからの干渉を受けずに保持し続けることができる。ローカルリポジトリには SQLite を利用し、ASCII 形式で記述されたコード譜のバージョン管理をおこなう。

*1 Git - Fast Version Control System: <http://git-scm.com/>

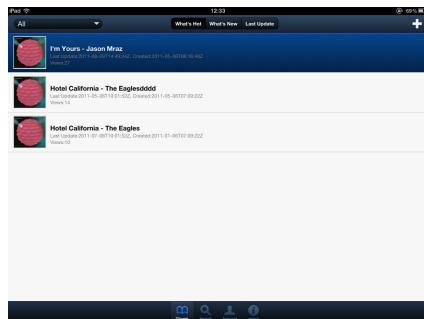


図 4 ランキング機能

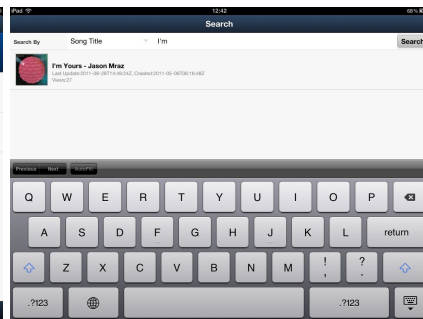


図 5 検索機能

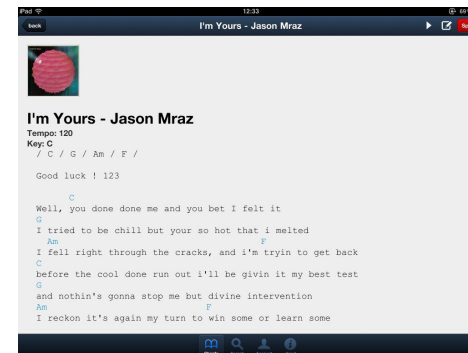


図 6 自動スクロール機能：指定されたテンポに応じてコード譜が自動スクロール表示される。画面右上のボタンはこのコード譜を編集したり情報を付加する編集モードを切り替えるためのものである。同様に、画面右上の赤いボタン (Spam) は悪意あるユーザを報告するためのものである。

オフラインアクセス機能 ユーザがオフライン時に更新した情報を 24 時間ごとにキャッシュし、ネットワークに接続された際に自動的に中央サーバに反映（さらに各クライアントに反映）する機能。ユーザは ChordBook を利用する際に常にオンラインに接続する必要はなく、場所を気にせずコード譜の作成・編集に集中することができる。

評価機能 中央サーバに登録されている個々のコード譜に対して、感想やアイデアを投稿し共同作者らとさらに品質を高めるための議論をおこなうことができる。悪意のあるユーザからの投稿をスパムとしてコード譜作成者に報告する機能も有する。

更新通知機能 中央サーバあるいは任意のユーザのコード譜が変更されたことをユーザに通知する機能。ウォッチリスト、e-mail、RSS フィード、IRC チャンネルを通じた自動通知をおこなうことができ、ユーザは好みの通知方法を選択することができる。

ランキング機能 中央サーバに登録されているコード譜あるいは各ユーザがローカルに保持しているコード譜に対しての統計情報（ダウンロード数など）を集計しユーザに提供する（図 4）。

検索機能 コード譜の検索結果を、曲名、アーティスト名、アルバム名、楽曲のジャンル毎に返す（図 5）。

自動スクロール機能 ギター演奏中のユーザにコード譜を自動スクロールしながら表示する。コード譜中で指定されたテンポに従い自動スクロールの速度を調節する機能も備えており、自動スクロールされるコード譜を見ながらユーザはギター演奏に集中することができる。

ソーシャルネットワーク機能 Facebook や Twitter などのソーシャルネットワークサービ

```
{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}
```

図 7 データフォーマット

スを介して、ユーザが気に入ったコード譜を友人知人と共有したり推薦する機能である。ソーシャルネットワーク機能によって、新規のユーザ獲得と趣味趣向の合うユーザ同士のサブコミュニティの形成につなげることを狙っている。

3.3 中央サーバ

中央サーバは、膨大な数のコード譜が保存でき、かつ、バージョン管理をおこなえる必要がある。ChordBook の中央サーバは、Web アプリケーションフレームワークとして Ruby on Rails を用い、Acts As Versioned Plug-in^{*1} を利用することでコード譜のバージョン管理をおこなう。これにより、ユーザはいつでも任意のバージョンに戻ることができるた

*1 http://agilewebdevelopment.com/plugins/acts_as_versioned_association

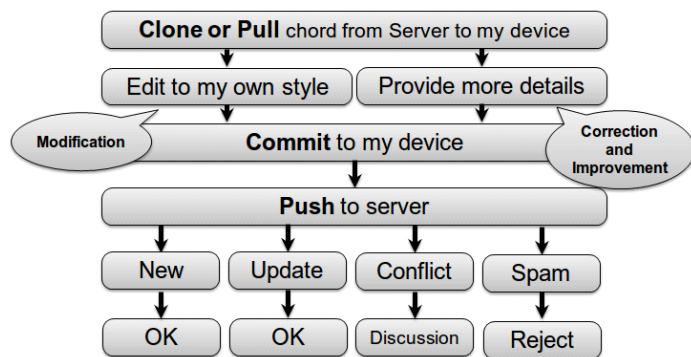


図 8 ワークフロー

め、新規のコード譜の作成が自由におこなえる。また、ユーザ権限の管理と認証には Devise Plug-in^{*1} を用い、中央サーバに登録されるコード譜の管理・編集権限を持つユーザを設定することができる。中央サーバと iPad 間の通信は、JSON 形式のデータを HTTP プロトコルで GET/POST メソッドを用いておこなう。

3.4 データフォーマット

コード譜は、図 7 に示すようなギターのコード進行と歌詞からなる楽曲を記述した ASCII 形式のテキストデータとしてデータベースに保存される。大括弧で括られた記号がコードを表しており、一般的なコードであればシステムが正しく解釈できる。中括弧の中には曲名、作詞作曲者名、曲の区切りなどを示すメタデータを記述する。それ以外のテキストが歌詞を示している。本フォーマットは、人間にとって可読性の高いシンプルなものであるが、ディスプレイ上でコード譜を表示したり、自動移調するためのプログラムが数多くあり、計算機でコード譜を扱うための比較的一般的な形式である。

3.5 ワークフロー

図 8 は、ChordBook 利用時のワークフローを示したものである。ユーザはまず、中央サーバから iPad へコード譜をダウンロードするために clone（既存リポジトリの複製をローカルに作製する）するか、あるいは、他のユーザのバージョンあるいは中央サーバのマスターバージョンを pull（他のリポジトリの変更点をローカルリポジトリにマージする）する。

ローカルリポジトリに反映されたコード譜に、何か問題があったり追加の情報を加えたい場合は、ユーザが直接編集をおこない他のユーザと議論をおこなうことができる。その際、ユーザは自身のローカルリポジトリに commit（変更を反映する）しバージョンを記録させることができる。もし、そのコード譜が満足のものであれば、公開リポジトリ（中央サーバ）に push（自身のローカルリポジトリの内容を送信する）することができる。中央サーバに関連する曲名がすでに存在する場合、中央サーバはその結果をユーザに通知することで、重複あるいは類似する曲名が中央サーバに繰り返して登録されないようにする。

コード譜が全く新しい物の場合、中央サーバはユーザのリクエスト（push）を自動で受け付けるが、更新されたバージョンである場合、そのコード譜が最新のものであるかどうかをチェックし、最新バージョンであれば受け付ける。一方、コード譜が最新のものではない（古いバージョンを再度、ユーザが公開リポジトリに反映しようとしている）場合は、ユーザ同士で意見が異なるものとみなし、ユーザ間での議論を促す。

また、スパムや悪意のあるユーザを制御するために、ChordBook には二種類の対策が機能として用意されている。1つは、ユーザがスパムの存在を管理権限のあるユーザに報告する機能である。もう1つは、数分以内に頻繁に変更されたり更新されたりするコード譜を検知し、その楽曲を編集したり更新したりすることを一定時間（現在の設定では 24 時間）制限する機能である。

4. 利用シナリオ

図 9 は、ユーザ（ギター演奏者）の ChordBook 利用シナリオを図示したものである。以下にその流れを順に説明する。

Step1 Alice は iPad の ChordBook クライアントを用いて、“Hotel California” のコード譜を新規に作成しようと思い立った。

Step2 Alice は “Hotel California” の歌詞とそれに付随するギターコードの編集を試みるが、スキル不足から十分満足のいくようには完成させることが出来なかった。

Step3 そこで Alice は、ひとまず初版のマスターバージョンとして中央サーバにアップロード（push）することにした。

Step4 Alice がコード譜を登録したのを知って、Alice の友人 Bob が残りの部分を完成させることを買って出る。Bob はまず、自分の iPad 上の ChordBook クライアントに Alice が作成したコード譜を中央サーバからダウンロード（clone）する。

Step5 Bob はダウンロードしたコード譜を編集し、情報を追加してコード譜を完成させる

*1 <https://github.com/plataformatec/devise>

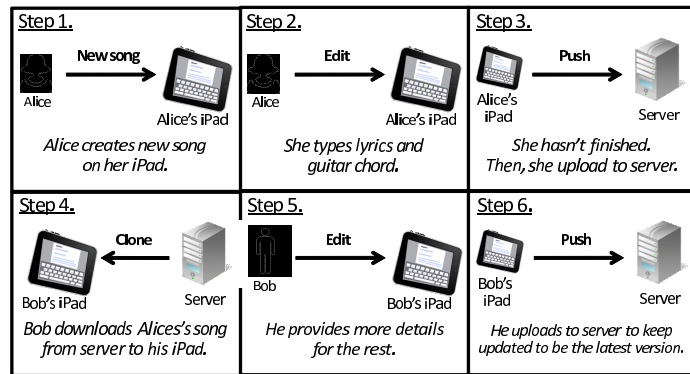


図 9 ChordBook 利用シナリオ

ことができた (図 10)。

Step6 その後 Bob は、最新バージョンを反映させるために中央サーバにアップロード (push) し、完成した“Hotel California”のコード譜を Alice をはじめ ChordBook ユーザ全員が利用できるようにした。

5. まとめと今後の課題

本稿では、クラウドソーシングの概念に基づいて設計し技術基盤として分散型バージョン管理システムを応用し実装した、ユーザ同士による協調的なコード譜の作成を支援するシステム ChordBook を提案し、ChordBook の利用シナリオを紹介した。今後の課題は、Apple App Store*1にて本システムを一般に公開し、実運用を通じたシステムの評価をおこなうことである。また、本稿ではギター用のコード譜の作成支援を対象としたが、適用対象を他の楽器へ広げることも検討している

謝辞 本研究の一部は、文部科学省「次世代 IT 基盤構築のための研究開発」の委託に基づいて行われた。また、本研究の一部は、文部科学省科学研究補助費 (基盤 B: 課題番号 23300009, 若手 B: 課題番号 22700033) による助成を受けた。



図 10 完成したコード譜の例

参考文献

- 1) Brabham, D.C.: Crowdsourcing as a Model for Problem Solving: An Introduction and Cases, *The International Journal of Research into New Media Technologies*, Vol.14, No.1, pp.75–90 (2008).
- 2) Calvin, K. M.L. and Tan, B. C.Y.: The Internet is changing the music industry, *Communications of the ACM*, Vol.44, No.8, pp.62–68 (2001).
- 3) Doan, A., Ramakrishnan, R. and Halevy, A.Y.: Crowdsourcing systems on the World-Wide Web, *Communications of the ACM*, Vol.54, pp.86–96 (2011).
- 4) Howe, J.: The Rise of Crowdsourcing (2006). <http://www.wired.com/wired/archive/14.06/crowds.html>.
- 5) Howe, J.: *Crowdsourcing: Why the Power of the Crowd Is Driving the Future of Business*, Crown Business, NY (2008).
- 6) Kittur, A., Chi, E.H. and Suh, B.: Crowdsourcing user studies with Mechanical Turk, *Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems (CHI'08)*, pp.453–456 (2008).
- 7) Surowiecki, J.: *The Wisdom of Crowds: Why the Many Are Smarter Than the Few and How Collective Wisdom Shapes Business, Economies, Societies and Nations*, Doubleday, NY (2004).

*1 <http://www.apple.com/jp/ipad/from-the-app-store/>