MeWare for Sale: Developer's Approaches to Serious Mobile Music Games

Charlotte Pierce
The University of Melbourne
Grattan Street,
Parkville 3010,
Victoria, Australia
charlotte.pierce@unimelb.edu.au

Clinton J. Woodward, Anthony Bartel
Swinburne University of Technology,
John Street,
Hawthorn 3122,
Victoria, Australia
cwoodward@swin.edu.au, abartel@swin.edu.au

ABSTRACT
Music students face a highly complex task with a significant cognitive load. Serious games are one teaching tool used to manage this complexity, as they have been found to increase student's engagement and foster self-regulated, independent learning behaviours.

In this paper we examine serious music games on the mobile iOS platform. We particularly focus on how developers approach the creation and publication of these games. To this end, we look at the design and development of serious music games, including their development histories, revenue models, user management models, and data management models. We then frame these characteristics in terms of the type of software the games represent, which indicates how and if users were considered during development.

Our findings provide valuable insight into the field of serious music games, in understanding how the current state of the field came to be and how it might evolve in the future.

Keywords
serious games, educational games, music education

INTRODUCTION
Learning musical skills is difficult, both from the student and educator's perspective (Kwalwasser, 1955). Students are asked to take on a significant cognitive load, and educators are asked to guide students through the complex tasks of learning musical notation, musical concepts, and the physical skills of playing an instrument (Hein, 2014).

Serious games are one tool that have emerged to help manage this complexity. These are games which "merge a non-entertaining purpose with a video game structure"

Proceedings of DiGRA 2020

(Djaouti et al., 2011). That is, they are video games which are specifically designed to teach players skills that apply outside of the game's context.

Serious games can offer some key benefits over other teaching tools. For example, because games are, by definition, designed to engage and motivate players, players are more likely to spend time with a game and enter a state of flow. This is a positive influence on skill and knowledge acquisition (Elliot and Dweck, 2013; Graesser et al., 2009; Kiili et al., 2012; Ritterfeld et al., 2009), so in an educational sense is likely to lead to higher quality learning outcomes. Games also offer the ability to provide immediate automated feedback, encourage independent learning behaviours (Gee, 2003), and provide context for more formal types of learning (Cassidy and Paisley, 2013; Vygotsky, 1978; Wood et al., 1976).

In this paper we examine serious music games on the iOS platform. Mobile games (a.k.a., 'apps') were selected as the focus for two reasons. First, tablet applications and devices are becoming increasingly popular as teaching aids due to their mobility, ease of use, and variety of interaction mechanisms. Second, many of the available desktop and web applications have mobile counterparts which have either feature parity or more functionality. Within the mobile space, the iOS platform was chosen as it features the largest library of serious music games compared to other platforms such as Android.

The particular focus of this study is to investigate how developers are making and presenting serious music games. This includes how the games are categorised and described, the revenue models they use, their development history, and how player data is managed. These characteristics provide evidence for the type of software these games represent. For this purpose we consider the three types of software defined by Eric Sink (2006):

1. **MeWare:** The developer creates software. The developer uses it. Nobody else does.
2. **ThemWare:** The developer creates software. Other people use it. The developer does not.
3. **UsWare:** The developer creates software Other people use it. The developer uses it too.

The reason for using these categories is that they provide valuable in- sight into how and if users were considered during the creation of a software product. This is a key factor in determining the potential audience and educational value of a serious game. For example, 'MeWare' software is unlikely to be designed for purposes beyond the developer's personal wants and needs. Such software tends to be very specific in design and functionality, and may not contain features that would be considered standard in general consumer software products. 'ThemWare', on the other hand, is likely to be a more well-rounded and polished product. This is software that is made for profit. However, 'ThemWare' is often more expensive, may come across as generic, and not include features that seem obvious to users. 'UsWare' is arguably the best type of software (Jeff Atwood, 2008), as developers who are also users can create a more ideal product. However, 'UsWare' is a rare phenomenon.

By looking at the described characteristics and the most likely type of software each game represents, we can examine how the developers see their work as contributing to the field of serious music games. We also gain some indication of the quality of these games as pieces of software, as games, and as educational resources.

This paper is structured as follows. The next section presents the method used, including the game selection process and the data collected. Following this, the results

of executing this method are presented. Finally, the last section provides a discussion of these results and potential future work that may be undertaken to expand on this study.

METHOD

App Selection

As this review focused on applications available on the iOS platform, the iTunes app store[1], maintained by Apple Inc., was used to select a representative sample. First, the following search phrases were defined:

- music theory
- music theory tutor
- music tutor
- sight reading
- music sight reading

A recursive process was then applied in order to expand the results of searching these phrases in the app store to a larger list of apps to consider for inclusion in the sample. Given the limited number of results the iTunes app store returns for individual search queries this process heavily relies on Apple's algorithm for identifying related apps.

After searching a phrase, each app in the results list was considered in turn. If the app was not already in the list of candidates, it was searched. Searching involved adding the app to the list of candidates, opening the list of apps related to it, then searching each of the unseen apps from that list. This process iteratively expanded the space of connected apps, meaning a large number of candidates could be identified from only a small number of search phrases. Once a list of candidates was found, it was filtered according to the criteria illustrated in Figure 1. The first criterion was that the game must be available on the Australian iOS app store, as this is the only version of the store accessible from Australia. If this was true, the game's description and screenshots were perused in an attempt to identify whether the app was skill- based (defined by Cherner et al. (2014) as being applications which aim to help a learner build basic abilities and fundamental knowledge in a subject area). Once past this stage, the game's rating and reviews (both positive and negative) were briefly considered. This was done in an effort to identify any obvious issues (e.g., frequent crashing, catastrophic bugs) that would prevent the game from being properly assessed. In cases where a game clearly had such issues it was omitted. Finally, the game was checked for compatibility with the review hardware.

This process of selection and filtering resulted in a final sample of 175 out of 313 iOS applications being selected.

Data Collection

Background and design data

As suggested by Childress and Lee (1999) and Lee and Cherner (2015), the background and design data we collected focuses on general information about the game and its developer, and which devices are supported.

Data collected included the name of the developer, the category the game is listed under in the app store, and the version of the game that was assessed. We also noted the

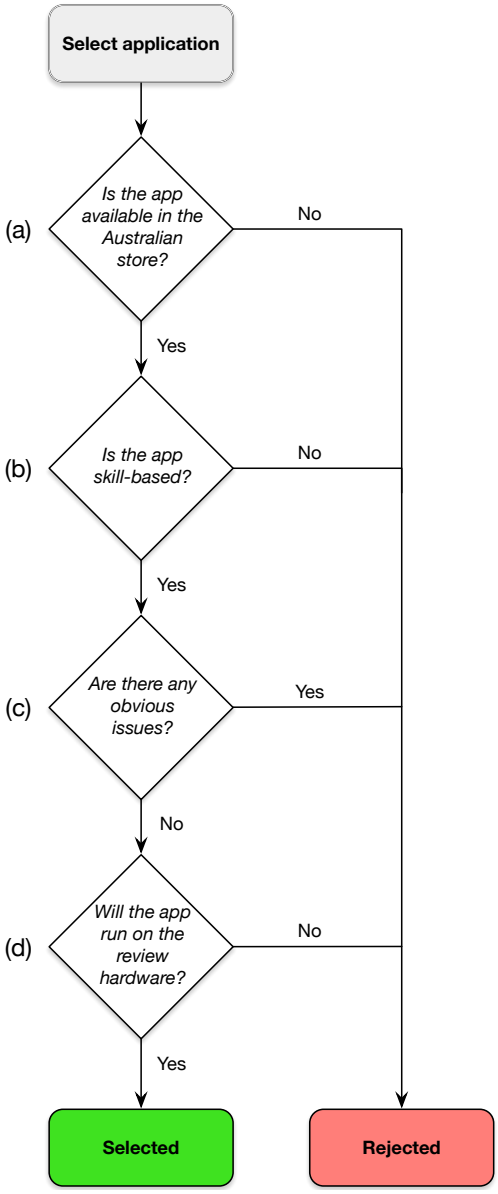

Figure 1: The process used to filter candidate iOS applications for inclusion in the analysis, using four key criteria

content rating of the game, its update history, and whether it runs on the iPad or iPhone (or both). Finally, the development status of each game was identified as being 'active' or 'inactive', where games which had received no updates in six months were classified as 'inactive'.

The update history was taken from the website AppShopper[2]. AppShopper records version and pricing changes of apps on the iOS app store, noting the date and relevant details (e.g., version number, new price) of these events.

Identifying revenue models

Each game was identified as using one of five revenue models:

1. **Free:** The game has no costs associated with it, and does not show advertisements to players. Essentially, no revenue model is used.

2. **Ad-supported:** The game does not have any upfront or in-game monetary costs, but it shows advertisements to players. The developer gets paid by the advertising companies.
3. **In-app purchases:** The game does not have any upfront costs, but content or unlockable features can be paid for within the game.
4. **Paid:** The game requires players to pay a one-time upfront cost.
5. **Subscription:** The game requires players to pay a recurring fee to continue playing the game. For example, players might be asked to pay $5 a month.

For those games which did require some monetary payment (i.e., those using the '*in-app purchases*', '*paid*', or '*subscription*' revenue models), we noted the current cost of the game and any price changes that have occurred in the past. As with the update history, the pricing history of each game was taken from the AppShopper tracking website.

User Management Data

The user management data consists of three categories:

1. Profile management
2. Device management
3. Score management

The 'profile management' category indicates whether multiple players can independently track their progress on a single installation of a game and, if so, if that progress can be transferred between instances of the game. Instances of a game refer to unique installations of a game on the same device. So, if a player uninstalled and reinstalled a game, they would be creating a new instance of that game. Three values are possible:

1. **Single:** The game supports only a single player on any one device. Progress data is stored locally and deleted with the game data if the player uninstalls the game.
2. **Local:** The game supports multiple players, but player profiles are stored locally. Profiles which are stored locally are deleted with the game data if the player uninstalls the game. This means that the profiles and progress data will not transfer to a new instance of the game.
3. **Online:** The game supports multiple players and player data is stored online. Profiles which are stored online are not deleted with the game data if the player uninstalls the game. This means the profiles and progress data can be transferred to a new instance of the game.

The 'device management' category pairs with this, indicating whether a game supports syncing or transferring player data between separate physical devices. Two values are possible:

1. **Single-device:** Player data is stored locally and can not be transferred to another device.
2. **Multi-device:** Player data is stored either locally or online and can be transferred to another device.

The 'score management' category indicates whether players can reset the scores within a game and, if so, whether the scores can be reset across the entire game, for individual portions, or both. There are four potential approaches to score management:

1. **None:** No score management mechanisms are offered. The only way to erase progress data is to uninstall the game.
2. **Reset-single:** Progress data can be reset for individual activities or areas of the game.
3. **Reset-all:** Progress data can only be reset on a game-wide basis.
4. **Both:** The game offers both the *reset-single* and *reset-all* score management strategies.

User Reviews Data

All applications on the iTunes store are open to receiving publicly published written reviews, each of which is accompanied by a whole-number score between 1 and 5 (inclusive). Once 5 reviews have been submitted for an application the associated scores are combined into an aggregate. We noted the aggregate score (if available) for each of the games. We also collected the number of reviews each game has received in total, and for the latest version. This helps us to understand how the games have been received by players.

RESULTS

Background and Design

Unsurprisingly, all of the games have a content rating of '4+', meaning they contain no offensive or age-restricted content (Apple, Inc., 2019). A large majority of the games (i.e., 144) are available on both the iPad and iPhone, meaning players are generally not limited in their choice of device. These characteristics mean there is a large potential audience for these games.

Across the collection of games, there are 74 unique developers. The large majority of these developers have no other published work on the app store, and a small minority have published a comparatively large number of games. For example, *AtPlayMusic* have published 23 games in their 'PlayAlong' series, for various instruments ranging from the acoustic guitar to the oboe. Similarly, *Rising Software* has published 15 games in their 'Auralia' and 'Musition' series.

There is almost an even split in the development status of the games. Just over half (i.e., 94, or approx. 54%) are classified as 'inactive' – that is, they have received no updates in the past six months. The remainder are still being actively developed, but are still rarely updated. Overall, this indicates a general lack of interest by the developers in maintaining their work after it is initially published.

This is supported by examining the game's development histories, which show that most have received few to no updates after their initial publication. Figure 2 shows that half of the games have 5 or fewer version changes, and some have never been updated. A small number of the games have greater than 20 updates. Three games – *Piano Melody Pro* (Learn To Master Ltd, 2017), *Rhythm Sight Reading Trainer* (Rolfs Apps, 2016), and *Piano Maestro* (JoyTunes, 2016) – have each received 40 updates, the highest number seen.

To place these update counts in context, we compared them to a similar sample of other games available on the app store. This second group of games is similar in size, price, release window, and store ranking. Figure 3 shows the results of this comparison. It can be seen that whilst there are several games of both types which have few updates, on the whole the non-music games have received many more updates over their lifetime.

Figure 4 shows the distribution of primary genres the developers have nominated for their work in the iOS app store. An application's primary genre represents a top-level

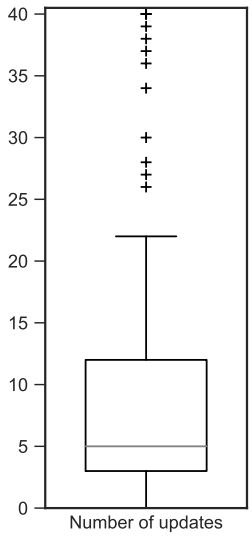

Figure 2: Distribution of number of updates for the sample of games

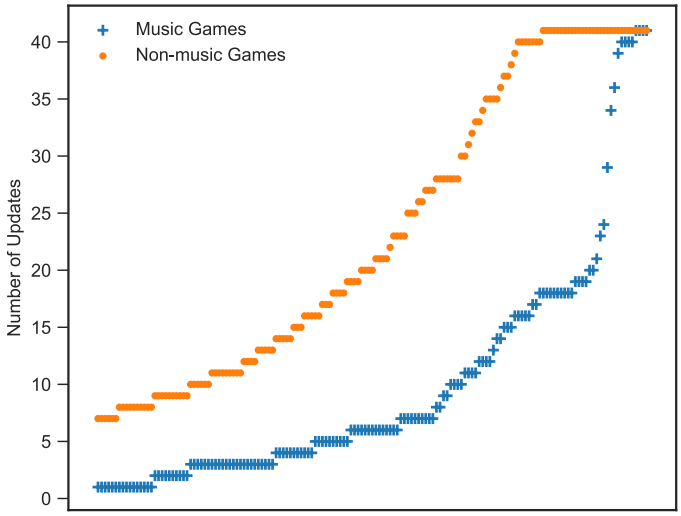

Figure 3: Number of updates for the sample of serious music games and a similar sample of other, non-music games on the app store

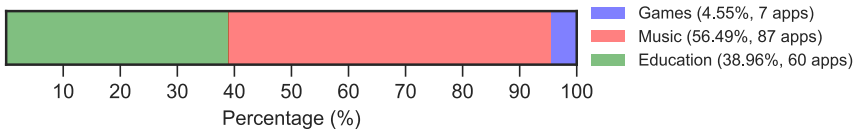

Figure 4: Primary genres of the sampled games

categorisation, affecting where it is contained in the app store, the charts it appears in, and its prominence in search results (Apple Inc., 2017). It also indicates how a developer sees their work, and how they wish it to be seen by others. A primary genre can be further specified with secondary, tertiary, and quaternary genres. For example, *Note Tutor* (NoteTutor, 2014) has a primary genre of 'Music', secondary genre of 'Family', tertiary genre of 'Educational', and finally a quaternary genre of 'Games'. Not all of the sampled games have four levels of categorisation, but all have at least primary and secondary genres.

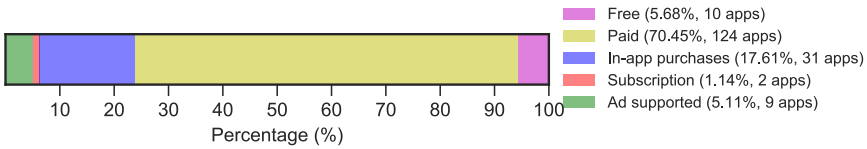

Figure 5: Distribution of revenue models used by the sample of serious music games

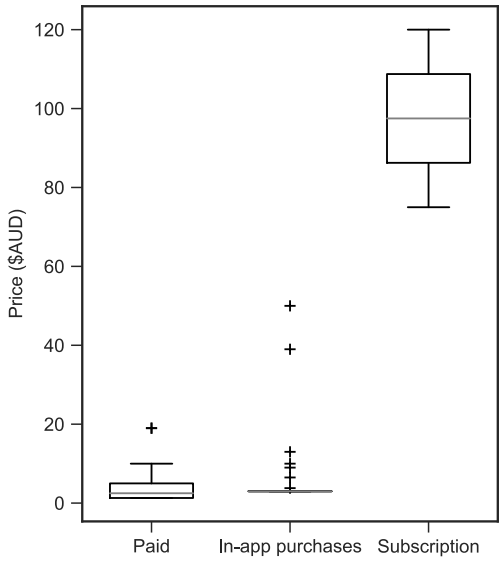

Figure 6: Distribution of game prices by revenue model

Genre	**Description**
Music	Apps that are for discovering, listening to, recording, performing, or composing music, and that are interactive in nature. For example: music creation, radio, education, sound editing, music discovery, composition, lyric writing, band and recording artists, music videos and concerts, concert ticketing.
Education	Apps that provide an interactive learning experience on a specific skill or subject. For example: arithmetic, alphabet, writing, early learning and special education, solar system, vocabulary, colors, language learning, standardized test prep, geography, school portals, pet training, astronomy, crafts.
Games	Apps that provide single or multiplayer interactive activities for entertainment purposes. For example: action, adventure, arcade, board, card, family, music, puzzle, racing, role playing, simulation, sports, strategy.

Table 1: Application store genre descriptions provided by Apple Inc. (2017)

Across the entire sample, three different primary categories were nominated: ‘Music’, ‘Education’, and ‘Games’. The definitions of these categories are provided in Table 1. ‘Music’ is most commonly nominated, closely followed by ‘Education’. Only a small number of developers have chosen to assign their work to the ‘Games’ category. This core set of genres is repeated at all four levels of categorisation. Other genres such as ‘Reference’, ‘Entertainment’, and ‘Family’ also appear, but less frequently.

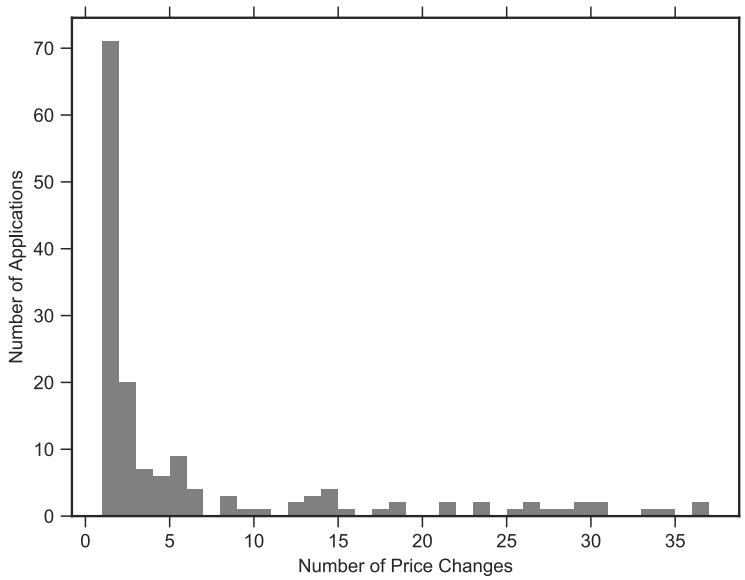

Figure 7: Distribution of number of price changes for the sample of serious music games. Note that the initial price setting is counted as one change.

Revenue Models

Figure 5 shows that the large majority of the games use the *Paid* revenue model, requiring some upfront payment from players. However, as can be seen in Figure 6, these costs represent a relatively affordable price range, with a median price of $2.49 AUD. The next most popular revenue model is *In-app purchases*. Games using this model are also relatively affordable, with a median price of $2.99 AUD. Very few games are entirely free. That is, very few of the games receive no money from players or advertisers. Interestingly, the games which ask for recurring payments (i.e., use the *Subscription* model) are also those with the highest prices. One potential reason for the popularity of non-subscription revenue models is that they are easier to implement in software. They also require less maintenance over time.

There are some outliers within the price distributions. For example, *Guitar Lessons - Rock Prodigy* (The Way of H, 2015) is the most expensive game to use in-app purchases, with one course costing $49.99 AUD, two courses costing $79.99 AUD, and all four courses costing $99.99. *Piano Dust Buster* (JoyTunes, 2015) is also an outlier, offering a $38.99 AUD in-app purchase to unlock additional songs. The most expensive *Paid* applications – *Better Ears* (Appsolute GmbH, 2015) and *Piano Ultimate* (By Better Day Wireless, 2015) – are significantly cheaper, requiring only a one-off cost of $18.99 AUD each.

Although the prices of the games are relatively low, their typically narrow focus, as found by Pierce (2019), means players might need to purchase multiple games to meet their needs.

Figure 7 shows the distribution of the number of price changes for the serious music games. The majority of games tend to alter their prices fewer than 5 times in total, and very few games have over 10 price changes. Many of the games have never changed price. *Glow Piano Lessons* (Apps For Hunger Inc., 2011) most frequently adjusts its cost, having done so on 37 occasions.

Figure 8 shows in more detail how the prices of the games have changed over time. As with Figure 6, it can be seen that aside from a small number of outliers most of the games have relatively low prices. These prices can be further examined in Figure 9,

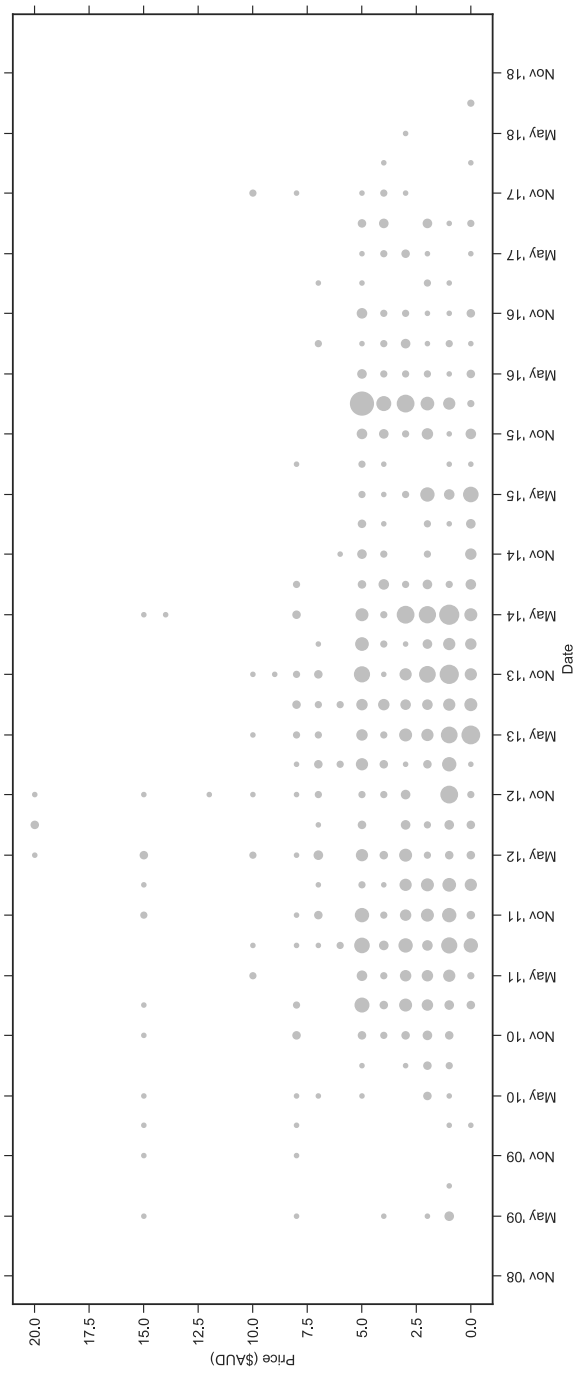

Figure 8: Price changes made to the sample of games over time. The bubble sizes indicate the number of applications at each price point, with the smallest bubble representing 1.

which shows the price changes of the games up to and including $20AUD. Interestingly, the prices set by developers have formed a grid, indicating that a small number of common price points are favoured. Across the entire sample, in the sub-$20.00AUD price range only 14 prices are ever selected: $0.99 to $9.99 in $1 increments, then $11.99, $13.99, $14.99, and $19.99. From these options, prices less than $5AUD are particularly popular.

User Management

Figure 10 shows the distribution of approaches taken by the games towards user profile management. The most common approach – used by 128 of the games – is to implement

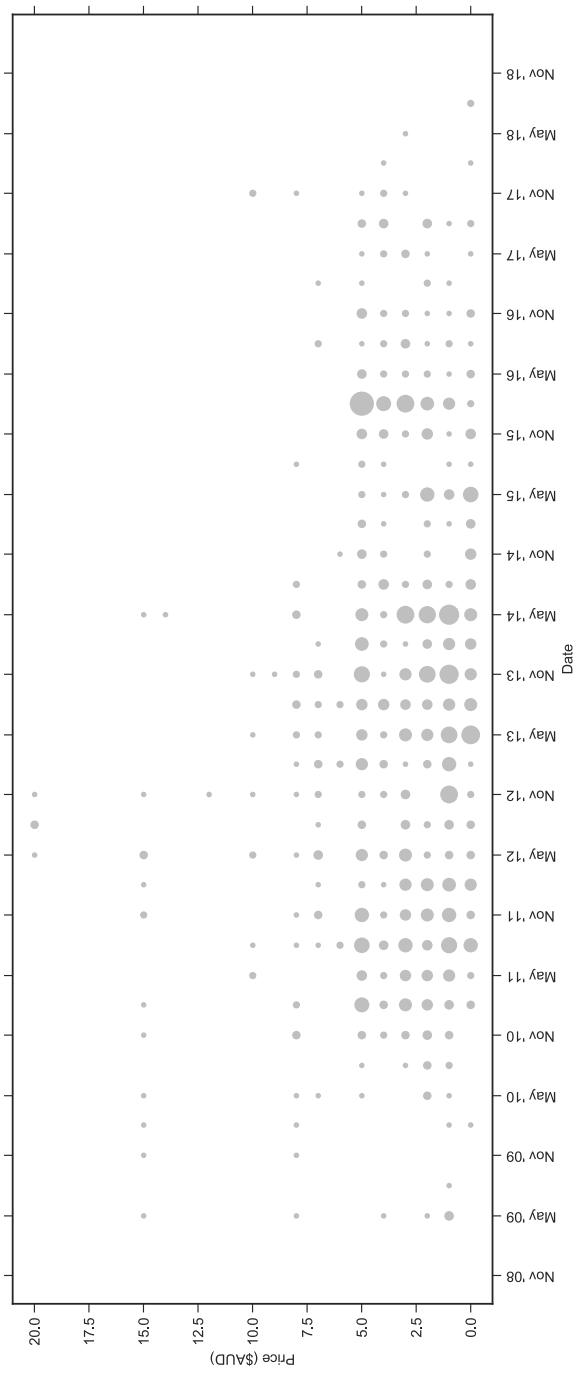

Figure 9: Price changes made to the sample of games costing less than $20AUD over time. Each bubble represents one price change.

a *Single* profile, where the game's data management extends to only one player on one device. *Local* profiles, used by 23 games, are the next most frequently used approach. This approach allows players to identify themselves when the game starts (e.g., by logging in), then have individual progress and performance data stored for each unique player. Both of these approaches are relatively easy to implement, given that they are largely defined by a lack of functionality. A small number of the games offer online profiles, and approximately 20 use no profiles at all. These 20 games are those which have no concept of progress data, thus no need for profiles.

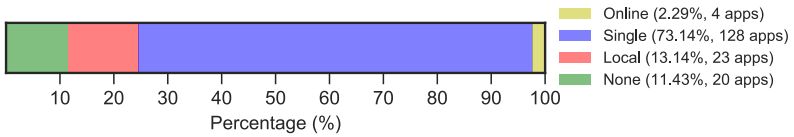

Figure 10: Distribution of approaches to profile management by the sample of games

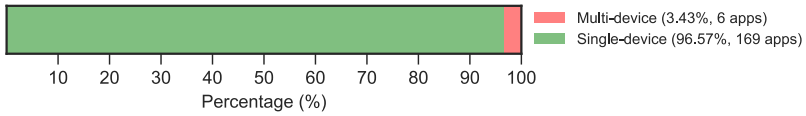

Figure 11: Distribution of approaches to device management by the sample of games

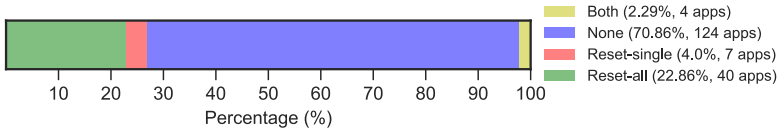

Figure 12: Distribution of approaches to score management by the sample of music games.

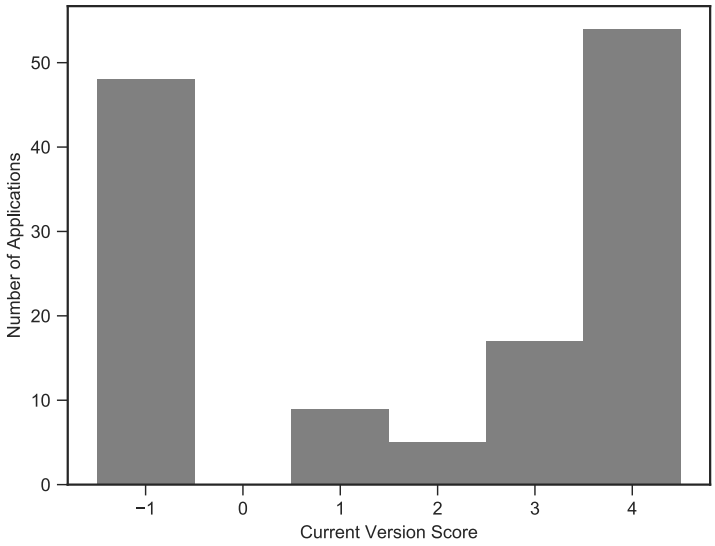

Figure 13: Distribution of aggregate review scores for the games in the sample. A score of -1 indicates that a game had an insufficient number of reviews to calculate an aggregate.

The ‘device management’ characteristic reveals how this player data is managed across different physical devices. Figure 11 shows that overwhelmingly the games offer only

a *single-device* approach. This means that game data is almost always stored locally, resulting in that data being permanently deleted if the game is uninstalled. Again, this approach is likely popular because it is easier to implement than the alternative. The few games which support *multi-device* device management allow players to transfer their progress between physical devices.

For players wishing to manage progress data, Figure 12 shows that most of the games do not offer any mechanisms for doing so. As with the most commonly used profile

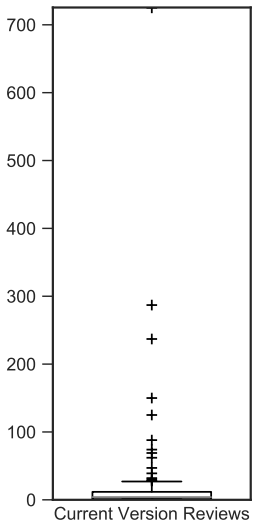

Figure 14: Distribution of the number of reviews for the latest version of each game in the sample

and device management approaches, this is again the easiest option to implement. Games which do tend to offer score management functionality usually only provide a

reset-all approach, meaning all progress history is erased. Very few of the games offer the ability to reset progress on a more granular level.

User Reviews

Figure 13 shows the distribution of aggregate scores for the sample of serious music games. None have received the maximum aggregate score of 5. However, the next highest aggregate rating (i.e., 4) is the most common, indicating that many of the games have been positively received. A small number have lower aggregate ratings, but most of the remaining games simply have not received enough reviews to calculate an aggregate score.

The number of games which do not have enough reviews for an aggregate score is not surprising when examining the distribution of review counts across the sample. This is shown in Figure 14. As can be seen, most of the games have received very few reviews. Most have received fewer than 10 reviews, and the median number of reviews is 4. Only a small number of the games have received more than 20 reviews, and even fewer still cross the line into triple digit counts. This can be examined further in Figure 15, which shows the same distribution with outliers removed.

Piano Maestro (JoyTunes, 2016) has, by far, the largest number of reviews, with 725 at the time of writing. *Piano Dust Buster* (JoyTunes, 2015) is a distant second, with 291 reviews. Both of these games have received mostly positive reviews, with aggregate ratings of 4 stars. They also both utilise the *in-app purchases* revenue model. Three other applications – *Glow Piano Lessons* (Apps For Hunger Inc., 2011), *Piano Tutor for iPad* (SmileyApps, 2015), and *Music Theory and Practice by Musicopoulos* (SpartanApps, 2013) – have also received over 100 reviews each. These are all published by different developers, all follow the *paid* revenue model, and have all received aggregate ratings of 4 stars.

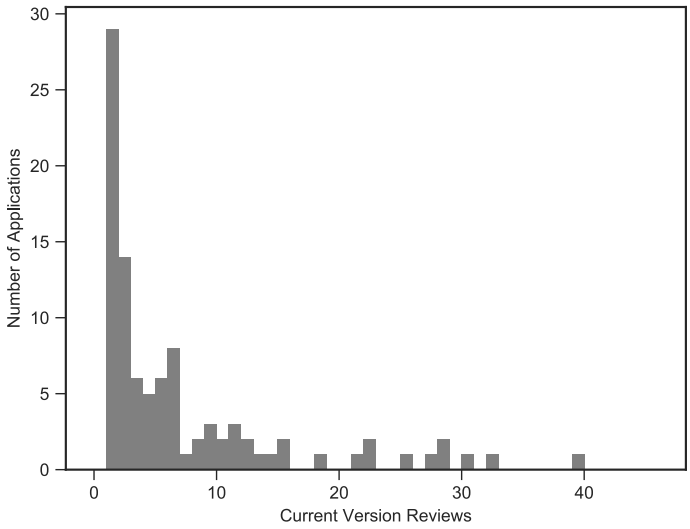

Figure 15: Distribution of the number of reviews for the latest version of each game in the sample, with outliers removed

DISCUSSION AND FUTURE WORK

These results indicate that the field of serious mobile music games, whilst well populated, contains a selection of games which would most accurately be described as 'MeWare'. Many of the games are characterised by limited to no development activity, low pricing, and designs which prioritise minimising development time and complexity rather than focusing on player experience and learning outcomes. They are typically developed by people working on their own, who have no other published software. This suggests that the developers are not specifically seeking to gain income from their work. Many of the game's store descriptions also indicate that they were made for the developer's personal use or for a friend or child. That is, the developers had some other purpose for making a serious music game (e.g., self-study) and published it as an afterthought. The fact that the games also tend to lack player profiles and robust data management mechanisms further supports the notion that the developers did not consider how their work might be used by a larger audience.

In contrast to this, a small selection of the games could be described as 'ThemWare'. These games tend to be priced higher, use subscription revenue models, and are more frequently updated with bug fixes and new functionality. Because of this they have received more attention from players, as evidenced by their higher number of reviews on the app store. These games are more often developed by groups or companies who have other published software, which further indicates that they are seen by their developers as real sources of income.

Future work in this area could focus on serious music games available in other environments, such as the Android, web, and desktop platforms. This work would seek to discover whether there are similar patterns of development and design decisions as we have found on iOS. It would also be valuable to interview the developers of these games to learn how they see their work, and their thought processes when making design and development decisions.

These additional studies would provide further insight into the field of serious music games, particularly on mobile platforms. It would help us to more completely understand how the currently available games came to be made, thus how the state of the field might continue to advance and evolve in the future.

BIBLIOGRAPHY

Apple Inc. (2017). Choosing a category.

Apple, Inc. (2019). App store connect help: My app – app ratings. https://help.apple.com/app-store-connect/#/dev269f11291.

Apps For Hunger Inc. (2011, September). Glow Piano Lessons. https://itunes.apple.com/au/app/glow-piano-lessons/id441970188. Version 1.3.1.

Appsolute GmbH (2015, October). Better Ears - Music and Ear Training. https://itunes.apple.com/au/app/better-ears-music-ear-training/id284444548. Version 3.0.1.

By Better Day Wireless (2015, March). Piano: Ultimate. https://itunes.apple.com/us/app/piano-ultimate/id530911981. Version 1.1.

Cassidy, G. G. and A. M. Paisley (2013). Music-games: A case study of their impact. Research Studies in Music Education 35 (1), 119–138.

Cherner, T., J. Dix, and C. Lee (2014). Cleaning up that mess: A framework for classifying educational apps. Contemporary Issues in Technology and Teacher Education 14(2), 158–193.

Childress, M. and G.-L. Lee (1999). Reviewing software as a means of enhancing instruction. Information Technology in Childhood Education 255, 261.

Djaouti, D., J. Alvarez, and J.-P. Jessel (2011). Classifying serious games: the g/p/s model. In Handbook of research on improving learning and motivation through educational games: Multidisciplinary approaches, pp. 118– 136. IGI Global.

Elliot, A. J. and C. S. Dweck (2013). Handbook of competence and motivation. Guilford Publications.

Eric Sink (2006). Yours, Mine and Ours. https://ericsink.com/articles/Yours_Mine_Ours.html.

Gee, J. P. (2003). What video games have to teach us about learning and literacy. Computers in Entertainment (CIE) 1 (1), 20–20.

Graesser, A., P. Chipman, and F. Leeming (2009). Deep learning and emotion in serious games. In Serious games, pp. 105–124. Routledge.

Hein, E. (2014). Music games in education. In Learning, Education and Games, pp. 93–108. ETC Press.

Jeff Atwood (2008). UsWare vs. ThemWare. https://blog.codinghorror.com/usware-vs-themware/.

JoyTunes (2015, September). Piano Dust Buster by Joy- Tunes. https://itunes.apple.com/au/app/piano-dust-buster-by-joytunes/id502356539. Version 3.0.2.

JoyTunes (2016, October). Piano Maestro by JoyTunes - Piano practice. https://itunes.apple.com/au/app/piano-maestro-by-joytunes-piano-practice/id604699751. Version 4.1.1.

Kiili, K., S. De Freitas, S. Arnab, and T. Lainema (2012). The design principles for flow experience in educational games. Procedia Computer Science 15, 78–91.

Kwalwasser, J. (1955). Exploring the musical mind. Coleman-Ross Co.

Learn To Master Ltd (2017, May). Piano Melody Pro - Learn Songs and Play by Ear. https://itunes.apple.com/us/app/ piano-melody-pro-learn-songs-and-play-by-ear/id437498350. Version 11.20.

Lee, C.-Y. and T. Cherner (2015). A comprehensive evaluation rubric for assessing instructional apps. Journal of Information Technology Education: Research 14, 21–53.

NoteTutor (2014, December). Note Tutor. https://itunes.apple.com/us/app/note-tutor/id315099430. Version 1.3.1.

Pierce, C. (2019). Intelligent Tools for Deliberate Music Practice: Evolving Targeted Sight Reading Exercises. Ph. D. thesis, Swinburne University of Technology.

Ritterfeld, U., M. Cody, and P. Vorderer (2009). Serious games: Mechanisms and effects. Routledge.

Rolfs Apps (2016, December). Rhythm Sight Reading Trainer.

https://itunes.apple.com/au/app/rhythm-sight-reading-trainer/ id396302174. Version 8.7.0.

SmileyApps, L. (2015, September). Piano Tutor for iPad. https://itunes.apple.com/au/app/piano-tutor-for-ipad/id364898961. Version 7.2.

SpartanApps (2013, October). Music Theory and Practice by Musicopoulos. https://itunes.apple.com/us/app/music-theory-practice-by-musicopoulos/id319290362. Version 2.6.1.

The Way of H (2015, March). Guitar Lessons: Rock Prodigy. https://itunes.apple.com/au/app/guitar-lessons-rock-prodigy/id407303228. Version 2.2.1.

Vygotsky, L. (1978). Interaction between learning and development. Readings on the development of children 23 (3), 34–41.

Wood, D., J. S. Bruner, and G. Ross (1976). The role of tutoring in problem solving. Journal of child psychology and psychiatry 17 (2), 89–100.

ENDNOTES

[1] http://www.apple.com/au/itunes/
[2] http://appshopper.com/