Who Needs Interaction Anyway? Exploring Mobile Playlist Creation from Manual to Automatic

Dominikus Baur[1], Bernhard Hering[2], Sebastian Boring[1], Andreas Butz[1]
Human-Computer Interaction Group, University of Munich (LMU), Munich, Germany
[1] {dominikus.baur, sebastian.boring, andreas.butz}@ifi.lmu.de, [2] heringb@cip.ifi.lmu.de

ABSTRACT
Currently available user interfaces for playlist generation allow creating playlists in various ways, within a spectrum from fully automatic to fully manual. However, it is not entirely clear how users interact with such systems in the field and whether different situations actually demand different interfaces. In this paper we describe *Rush 2*, a music interface for mobile touch-screen devices that incorporates three interaction modes with varying degrees of automation: Adding songs manually, in quick succession using the *rush* interaction technique or filling the playlist automatically. For all techniques various filters can be set. In a two-week diary study (with in-depth interaction logging) we gained insight into how people interact with music in their everyday lives and how much automation and interactivity are really necessary.

Author Keywords
Music, mobile, recommendation, automation, interaction.

ACM Classification Keywords
H5.m. Information interfaces and presentation (e.g., HCI): Miscellaneous.

General Terms
Design, Human Factors.

INTRODUCTION
Personal media collections on mobile devices have grown substantially in recent years, which makes finding relevant items a daunting task. In addition, mobile devices still have limited input and output capabilities further complicating this task. Recommender systems can support users with accessing their music collections by suggesting suitable items. At the moment, however, it is unclear how to best integrate recommendations into mobile interfaces: Ideas range from purely automatic (e.g., Apple's iTunes Genius playlists), via semi-automatic methods (e.g., repeated recommendations in rush [1] or constraint-based generation in SatisFly [7]) to direct selection (e.g., on map-based representations of the whole collection [4]). Unfortunately, existing evaluations are mostly lab studies and only cover an artificial subsection of the music's place in everyday use (cf. [9]). Furthermore, it remains unclear whether one type of interaction is sufficient for all situations or if different situations demand different ways of accessing one's music.

In this paper we present *Rush 2* (see Figure 1), a mobile interface for music consumption that – in contrast to existing systems that restrict the user to a single style of access – lets users freely configure their style of interaction from automatic to manual. We conducted a two-week diary study with extensive interaction logging and observed how music was consumed in real life situations. The results give insight into what aspects might be important for an intended usage scenario and inform future music interface designs.

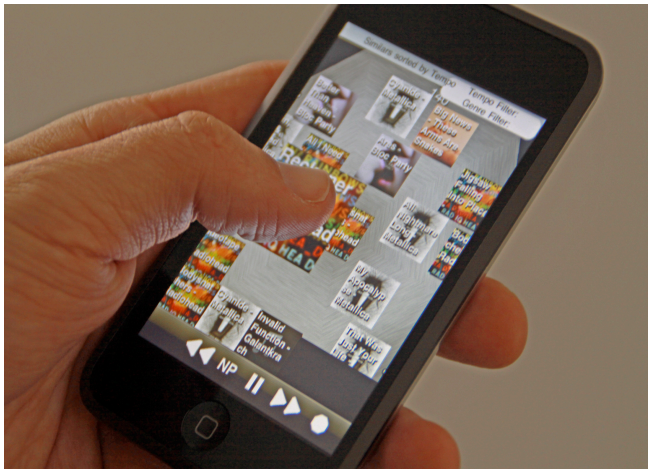

Figure 1 – Creating playlists with *Rush 2* on a mobile device.

RELATED WORK
Most research on recommender systems focuses on the so-called "single shot" approach – recommending the one item that fits best. Suggestions for more than one item (e.g., complete playlists) are rare ([5,8] and cf. [2]). It is more common in commercial music services such as *Last.fm*, *MOG* or *Play.me*, that provide web radio based on a given seed artist, tag or a user's listening history. Non-automatic music selection is either done in lists or on abstract, similarity-based maps (e.g., [4]), thus with either too little automatic support by the system (alphabetic lists) or too much abstraction (no direct access to songs). Commercial mobile systems are mostly list-based, but some also feature

IUI 2011, February 13–16, 2011, Palo Alto, California, USA.

Figure 2 – Similar songs (a), similar songs filtered by genre (b), tempo (c), other songs from the same artist (d), setting filters (e).

automatic, artist- or tag-based playlists (Pandora[1], Grooveshark[2]). Slice[3] is one example that provides a graphical representation of related artists for creating (non-interactive) radio. All of the mentioned systems, however, only provide one way of accessing songs and feature no ways to switch between different strategies.

SIMILARITY-BASED MUSIC INTERACTION

When designing *Rush 2* we had two goals in mind: first, the system should support an arbitrarily sized music collection and still keep it manageable. Second, the user should be flexible enough to create various types of playlists. Therefore, we considered user tasks that lay on the spectrum between fully automatic and fully manual: With fully automatic solutions, choices are restricted to picking a seed song and the length of the playlist (e.g., Apple Genius). This simply leaves users with the resulting playlist and little ways of adapting it. On the other end of the spectrum, list-based interfaces that often make use of the inherent musical hierarchy of songs, albums, artists and genres leave users with little more than their intuition and a lot of tapping when trying to create an interesting playlist.

To bridge this gap, we decided to equip the interface with various filters and other tools for accessing one's music, but including similarity measures and falling back to automatic playlist creation whenever possible. Also, we wanted all of this to happen on the level of songs, so users had maximum control over the results.

Interaction and display

The first step of interacting with *Rush 2* is choosing a seed song. As succeeding songs should fit together style-wise, all interaction is based on receiving follow-up suggestions for the last item. The resulting interface is based on so-called 'attribute wheels' (see Figure 2): The current seed item is displayed in the middle of the wheel, while similar songs group around it. Each wheel represents a certain musical category (e.g., tempo) and its spokes stand for different values (e.g., 80-100 beats/minute) and contain corresponding songs. A song is represented by its album cover art with artist and title name overlaid. Four wheels are available: Similar songs without categorization, songs categorized by genre, by tempo and other songs from the seed song's artist by albums. Choosing a wheel as the central display metaphor lets us easily represent different categories and display more items at once than in a (non-scrollable) list, further allowing direct comparison between items.

Figure 3 – The playlist (left) and the controls (right) in *Rush 2*.

As *Rush 2* is targeted at mobile touch-screen devices, most user interaction is based on tapping and dragging with one finger. Controls for listening to music are shown at the bottom of the interface (see Figure 3 (right)). Dragging a song has two meanings: first, releasing it at the bottom of the screen adds it to the playlist (see Figure 3 (left)). Second, dropping it in the center of the wheel turns it into the new seed song. The user can further activate filters for genre and tempo on all wheels, to combine multiple attributes. Active filters are shown in the upper right corner of the interface and can be adjusted by tapping on the button (see Figure 2e). A tempo filter can also be set by shaking the device in the corresponding rhythm (e.g., while jogging) (see [5] for a more elaborate solution).

To focus on a single spoke with more detail, users can perform a double-tap on it (see Figure 4). The application zooms in and displays up to ten songs. Furthermore, we employed the *rush* interaction technique [1] for quickly adding relevant songs to the current playlist. By touching and holding on an empty area of the screen, the application zooms in. When the finger is moved, the song canvas moves in the opposite direction. To select a song, users cross it with their fingers. New songs appear beyond the wheels borders. To end the interaction, users simply lift their fingers.

The important aspect of *Rush 2* is combining automation with manually set filters. All displayed songs are similar to the current seed item. To prevent user frustration with too

[1] http://www.pandora.com
[2] http://www.grooveshark.com
[3] http://www.slicestation.com

similar songs (cf. [1]), we decided to take random similar songs instead of only showing the most similar ones. When the system reaches the end of a playlist (i.e., the last item has been played or skipped), additional songs are taken automatically from the active wheel/filter combination. Thus, users can simply choose a seed item (and optionally a wheel according to their current mood) to be able to listen to suitable music with minimal interaction.

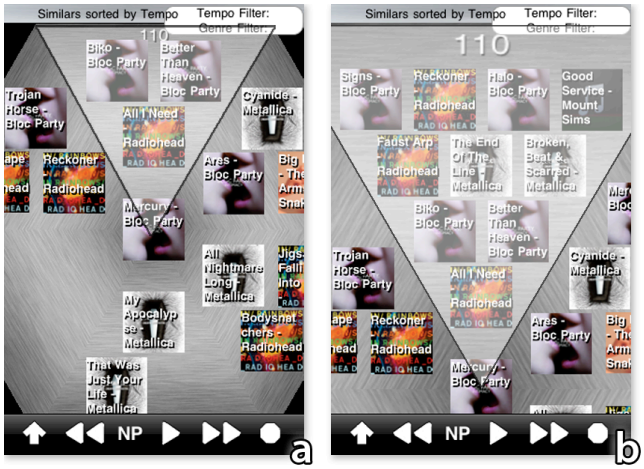

Figure 4. Users can perform a double-tap on a spoke of interest (a) in order to get more detail (b). The shaded area in (a) denotes the enlarged region shown in (b).

Implementation

Rush 2 was implemented in iOS 4 for the Apple iPhone and iPod Touch. The underlying graphics are based on OpenGL ES and music, cover art and metadata (i.e., artist, title and genre) are taken directly from the device's iTunes library. To find similar items for all songs we use the Last.fm web-service[4], which has a library of millions of songs. The lists of similar items are created by (1) filtering the library for the currently set wheel/filters and (2) finding the most similar songs based on the Last.fm data. If this process does not produce enough relevant items, we fill the display with songs that are similar regarding other, non-filter criteria (genre, tempo). Automatically extracting the tempo from a piece of audio content is non-trivial, so we take this information from The Echo Nest web-service[5].

EVALUATION

Existing music interface evaluations focus on single-strategy applications and are mostly lab studies. Work from sociomusicology highlights various contexts in which music is used (e.g., [9]), but naturally ignores the relevant interface requirements. Therefore, the goal of our evaluation was to see how people really interact with an interface as flexible as *Rush 2* in their everyday lives and what features are relevant for what usage context.

[4] http://www.last.fm/api
[5] http://developer.echonest.com

Setup

To learn about *Rush 2* we conducted a two-week diary study. Participants were asked to use their own devices and music collections to gather realistic results. We asked our participants to fill out a short questionnaire every day they used the system and tell us how satisfied they were and in what contexts they had used it. This usage context was very important, as it showed which parts of the interface were helpful for a given task. In addition, the application logged all relevant interactions (e.g., adding a song to the playlist or switching the wheel) and sent it back via email. After two weeks, participants were asked to fill out a more extensive questionnaire and could keep the application on their mobile devices in exchange. The high overhead required for the study only allowed us to recruit 4 participants (all male, avg. age: 23.5 years, avg. library size: 1263 songs).

Results

Despite the few participants, the in-depth data gave us deep insight into the participants' interaction with music and our application. The participants used the application in 15 listening sessions with an average length of 65.9 minutes and listened to a total of 303 songs. We extracted four classes of usage scenarios from the results:

1. *Background music* (4 sessions by 2 users): In this scenario, the user's intentions for listening to music were either for entertainment (e.g., while playing a board game) or for setting a certain mood (e.g., playing comfy music), but not as a primary occupation. This scenario was characterized by a low involvement on the user's side and a resulting heavy use of automatic filling of the playlist (95% versus 5% for drag-and-drop). Typical sessions had the user choose a seed song, keeping the *similar* wheel and letting the music play without interaction (one participant set a tempo filter for choosing more mellow music).
2. *Active listening* (5 sessions by 3 users): Here, the user was much more involved in interacting with the application and used it for exploring his or her collection while creating a playlist. Interactive techniques like drag-and-drop and *rush* slightly outweighed autofill (54.2% to 45.8%). The default *similar* wheel was the preferred one (35.9%), while the use of the other wheels was equally distributed.
3. *Exercising* (1 session by 1 user): We only had a single participant using *Rush 2* while exercising. As expected, he made exclusive use of the tempo filter and autofill.
4. *Other/Non-categorized* (5 sessions by 2 users): For these sessions, we were unable to find a category as participants had forgotten to mention them in their diary entries (we collected the diaries after the study and could not expect them to still remember their intentions). Based on their characteristics, they are very similar to *background music* (90.2% autofill) which might explain why participants forgot about them.

Table 1 summarizes all scenarios, the number of songs added with different techniques and the wheels. We found that automatic suggestions played an important role in any scenario (79% of all songs were added automatically), but users also relied on choosing the right wheel and filters beforehand: Wheels were switched on average 4.5 times per session, filters set 1.4 times. Also, skipping songs seemed to depend more on whether users were actively listening to music than whether they liked a song: active listeners skipped 25% of all songs (in addition to having selected manually more than half of them), while background listeners skipped only 17% of the almost exclusively automatically added songs. This is relevant for recommender systems that rely on skipping a song being an expression of disliking it (e.g., [6]). One thing to keep in mind is that users sometimes switched between usage scenarios in one session (which might also explain the non-categorized sessions). In one example, a user added three songs and then relied on autofill while driving. After getting into a traffic jam, he again started to interact with the app.

	Auto	Drag	Rush	S	G	T	A	Skip
BGD	117	6	0	10	0	1	3	21
ACT	44	39	13	14	9	7	9	24
EXC	23	1	0	1	0	0	0	1
OTH	55	6	0	8	1	1	4	22

Table 1 - How were songs added, which wheels were used (S = Similar, G = Genre, T = Tempo, A = Artist), and how many songs were skipped in the four usage scenarios.

With autofill being the main way of adding songs to a playlist, the quality of the recommended items was an important factor. The post-study questionnaire showed that the participants had different opinions regarding this point: While two agreed that *Rush 2* suggested songs they would expect, two disagreed. The free-form textual feedback showed that recommendations unsurprisingly failed for uncommon or classical music, where the underlying, collaborative filtering-based data from Last.fm is narrow (we had a recall of 85.6% on average). The lack of data was worse regarding the tempo, as The Echo Nest's database did only contain on average 56.7% of the users' songs that often left the tempo wheel scarcely populated (we decided against uploading songs for analysis as it would have strained the network connections of the participants' devices too much).

Regarding *Rush 2*'s usability, the post-study questionnaire was quite revealing. We used a modified version of IBM's usability satisfaction questionnaire [3] and the participants' answers were mostly neutral only slightly tending towards positive. While they were satisfied with the number of items displayed and the operation speed and were confident that they could become productive using *Rush 2*, the rest of the questions showed that participants had problems with the general ease of use. A subsequent formative evaluation could improve the application in this regard.

CONCLUSION AND FUTURE WORK

In this paper we presented *Rush 2*, an application for mobile devices that allows music access in several ways with different levels of user involvement. With our two-week diary study we identified some typical usage scenarios as well as their influence on the interaction. We also confirmed the value of being able to select a set of songs either through explicit filters or by setting an attribute wheel. The central result of the study is that both automation and manual interaction have their places: if users just want to listen to music they are glad to have a reliable recommender that produces reasonable results (and does not distract them from what they are actually doing by, e.g., forcing them to skip a song). When they want to actively explore their collections or look for specific music they use all tools available. They might also switch between the two cases in mid-session, so music-centric applications should support all these tasks and not restrict themselves to one of them. To learn more about the connections between usage scenarios and interaction we plan to make a version of *Rush 2* available online to attract a larger number of users. This would also give us a platform for subsequent improvement of interaction and automation techniques.

REFERENCES

1. Baur, D., Boring, S., Butz, A. Rush: Repeated Recommendations on Mobile Devices. *Proc. IUI '10*, ACM (2010), 91-100.
2. Hansen, D.L., Golbeck, J. Mixing It Up: Recommending Collections of Items. *Proc. CHI '09*, ACM (2009), 1217-1226.
3. Lewis, J. IBM computer usability satisfaction questionnaires: psychometric evaluation and instructions for use. *Intl. Journal of Human Computer Interaction 7*, 1 (1995).
4. Neumayer, R., Dittenbach, M., Rauber, A. Playsom and pocketsomplayer, alternative interfaces to large music collections, *Proc. ISMIR '05*, (2005), 618-623.
5. Oliver, N., Flores-Mangas, F. MPTrain: A Mobile, Music and Physiology-Based Personal Trainer. *Proc. MobileHCI '06*, ACM (2006), 21-28.
6. Pampalk, E., Pohle, T., Widmer, G. Dynamic playlist generation based on skipping behaviour. *Proc. ISMIR '05*, (2005), 634-637.
7. Pauws, S., Wijdeven, S.V. User evaluation of a new interactive playlist generation concept. *Proc. ISMIR '05*, (2005), 638-643.
8. Ragno, R., Burges, C.J.C., Herley, C. Inferring Similarity Between Music Objects with Application to Playlist Generation. *Proc. MIR '05*, ACM (2005), 73-80.
9. Rentfrow, P.J., Gosling, S.D. The do re mi's of everyday life: The structure and personality correlates of music preferences. *Journal of Personality and Social Psychology 84*, 6 (2003), 1236-1256