EIT ITC Labs Activity 13131, Apps for Your Car

Task A1308, Deliverable TD1302

An Exploration Interface for Choosing In-Car Applications

Catalin Barbu, Anthony Jameson, and Adrian Spirescu
Deutsches Forschungszentrum für Künstliche Intelligenz GmbH, Saarbrücken

Contents

Abstract

This report discusses some of the challenges involved in searching within an App Store for in-car applications. It then explains how the paradigm of *parallel exploration* was applied in extended to create a testable demonstrator which addresses these challenges. The benefits of the new functionality and interface features offered by this demonstrator are illustrated with concrete examples. A nonparallel variant of the demonstrator brings the advantages of the parallel-exploration-based demonstrator (as well as some limitations) into focus.

1 Background and Purpose of the Demonstrator

A general question that arises in connection with applications for cars is that of how drivers can select an in-car application for a particular purpose. This problem can be seen in part as a specific instance of the more general problem of app selection: As is well known, app stores typically feature hundreds of thousands of apps. Even if we restrict attention to apps which are especially well suited for in-car use, we typically have thousands of apps. Here are two challenges faced by a user who aims to select an app for his car from an app store using a conventional web-based user interface:

- A great variety of such apps exist, ranging from those that can be used to find a gas station in a particular area to an app that allows the user to control advanced functions in his car. Because of the speed with which this type of app is developing, there exist no standard categories or taxonomies, and users may be unsure even of what keywords to use when searching for such apps. Selecting in-car apps in an app store is therefore more a matter of exploration than of well-informed search.

- Trial and error is a frequently used strategy by users who are searching for new in-car apps: They try out the app—perhaps in a demo version—and then decide whether to use it or try another app with similar functionality. This strategy is more problematic with in-car apps than it is with other types, because in many cases the usefulness and usability of an in-car app can be determined only during driving itself (and not with a couple of minutes of exploration of the app outside of a realistic context).

  Consequently, support is needed for users to examine the experience of people who already own the app and have used it, and it should be possible to compare these experiences so as to maximize from the start one's chances of finding the most suitable app.

2 Parallel Exploration as a Starting Point

In view of these considerations, a group at DFKI comprising two researchers (Anthony Jameson and Catalin Barbu) and three student assistants / master's students (Adrian Spirescu, Tanja Schneeberger, and Edit Kapcari) applied the DFKI paradigm of *parallel exploration* to the problem of selecting apps in an app store, developing new functionality intended to address the challenges just mentioned. Parallel exploration had already been well received by the research community and industry on the basis of prototypes developed in carrier projects, one of which, for example, was demonstrated for four days at the 2013 Conference on Human Factors in Computing Systems (CHI 2013) in Paris (Buschbeck et al., 2013; Jameson, Spirescu, Schneeberger, Kapcari, & Buschbeck, 2013).

Parallel exploration can be viewed as a significant generalization of the familiar paradigm of faceted search/browsing (see, e.g., Tunkelang, 2009), in which a user starts with a large set of items in a repository (e.g., the apps in an app store) and successively applies filters (e.g., names of categories or price ranges) to zoom in on specific subsets. (A variant of the APPS FOR YOUR CAR demonstrator that supports direct comparison with this type of faceted browsing is discussed below.) One key innovation of parallel exploration is that a user is enabled to formulate a number of interrelated queries in parallel and compare their results, which are shown in a tree-like hierarchy on the screen.

This basic principle in itself addresses one of the main challenges in search for apps: the absence of familiar, well-defined categories. If a user doesn't know which category or tag is

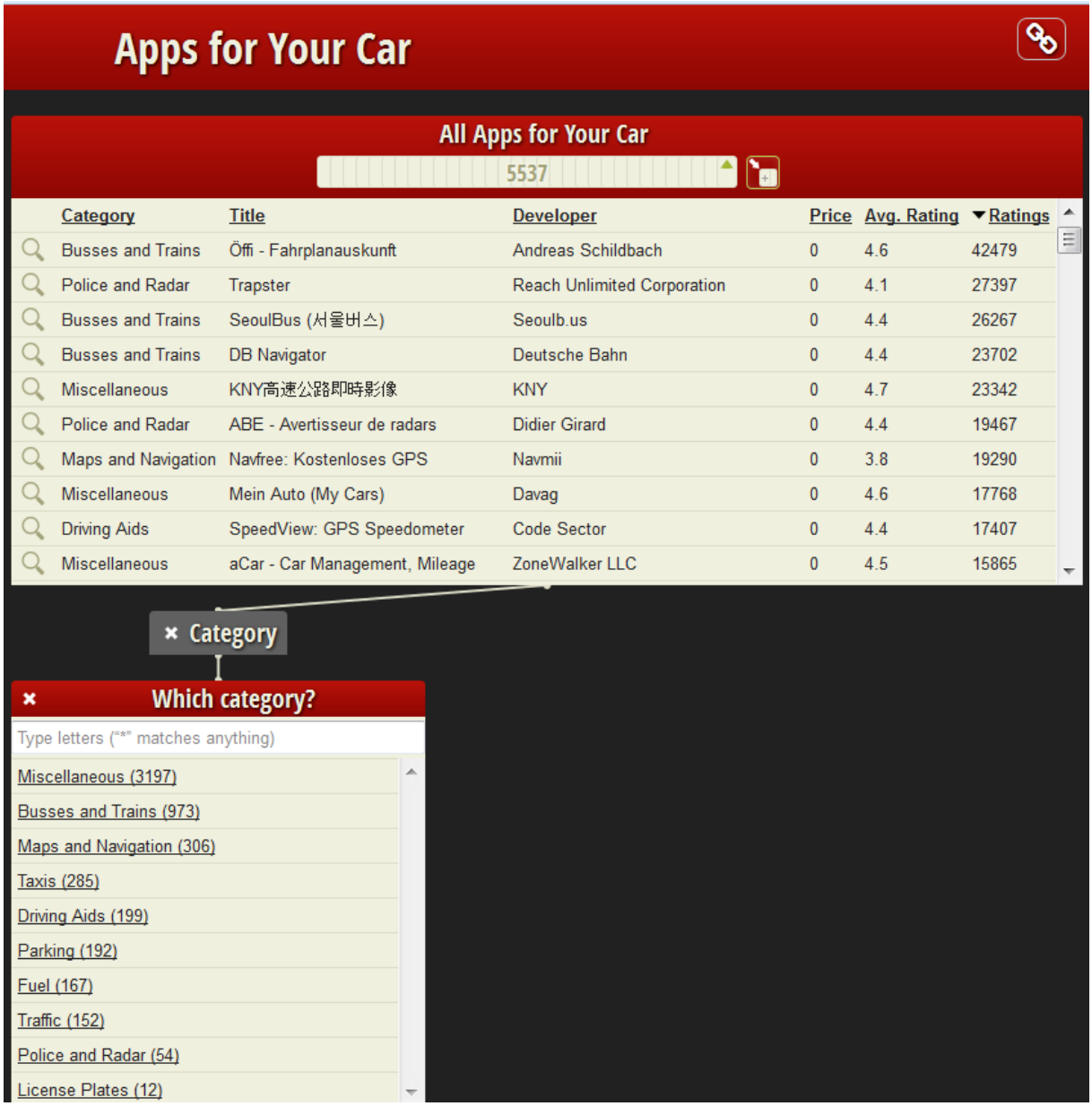

Figure 1: A user begins his exploration by selecting a category with which to filter the complete set of apps in the database.

most relevant to his need, he can specify two or more in parallel, compare the results directly with each other, and then decide which (if any) of the results to explore further.

In the following,we illustrate the features of the demonstrator using examples and screen-shots. Interested readers are invited to try the demonstrator out themselves at the following URL:

http://apps-for-your-car.com.

For example, Figure 1 shows how the user has begun to filter the total set of available apps in the demonstrator using the predefined categories. Even though these categories were defined by the designers of the demonstrator (in terms of keywords that appear in the titles and descriptions of the apps) with a view to maximizing clarity and relevance, a user may not see which of the available categories is most relevant to his search need—say, finding an app that will help him

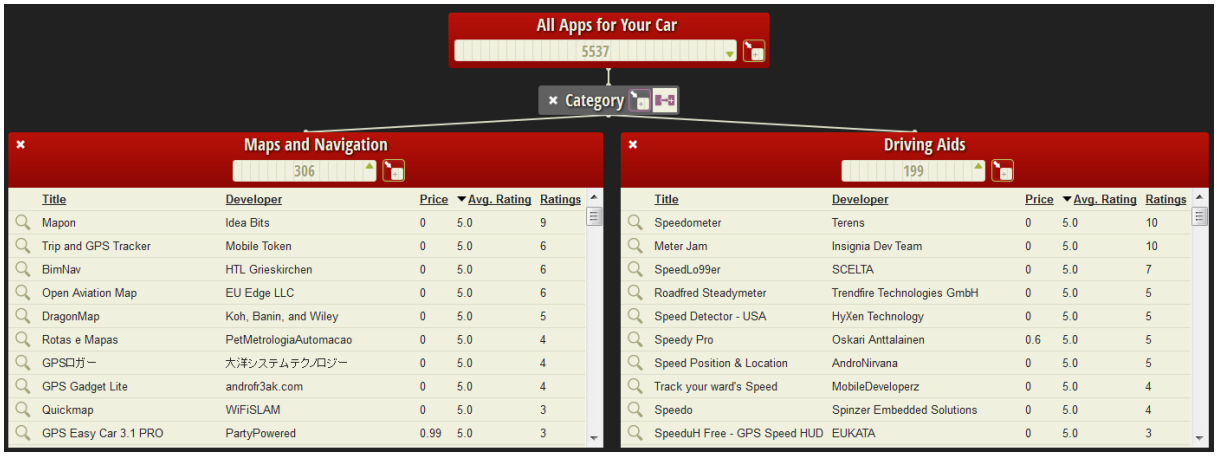

Figure 2: The user can compare the apps in two categories whose meaning is not immediately clear.

notice when he is exceeding the speed limit. Should he look under “Maps and Navigation” or “Driving Aids” (not to mention “Miscellaneous”, which includes apps that are hard to classify automatically)?

As is shown in Figure 2, parallel exploration allows the user to select two or more categories at once and compare the apps contained in them. Within each category of results, the user can sort the results by any of the attributes shown in the column headings, and he can inspect individual applications by clicking on the magnifying glass icons on the left (see below for examples). In this case, he may notice the relevant-looking app called “Speed Detector – USA” in the category “Driving Aids”, which suggests that additional relevant apps can be found in this category (though the user can continue to attend to the other category he wishes to).

3 Extensions to Parallel Exploration Introduced for This Demonstrator

In Figure 3, the user has decided to focus on “Driving Aids”, and he wants to filter the 199 apps in this category further to focus on those most likely to fulfill his need. There exist no predefined subcategories, but he can use flexible keyword search, including wildcards, to filter in an ad hoc way—for example, looking for all apps that have the word “detector” somewhere in the title (see Figure 4). The ability to filter in this way is not normally found in faceted browsing systems; it was introduced in the present demonstrator to address exactly the problem encountered in this example, which is typical of novel domains such as that of in-car applications. It appears to have considerable potential for many other application domains.

Of course the user can also not be sure which keyword-based query is most appropriate; but here again the parallelism of parallel exploration comes to the rescue: The user can try the alternative search for the keyword “watcher” without losing the results of his search for “detector” (see Figure 5). The user sees that both sets of search results include promising apps with average user ratings greater than 4.0.

The question for the user now becomes that of which of the top-rated apps is the most promising one. The user would like to compare “Speed Detector EVO” with both versions of “Speed Watcher”—in the latter case, to see exactly what the advantages of the “Pro” version

Figure 3: The user can drill down further by specifying a keyword-based query.

are over the "Free" version. As can be seen in Figure 6, the user can have these three apps displayed side by side, so that he can directly compare not only their objective features but also their ratings and reviews. Scrolling down as in Figure 7, the user can see not only the fundamental differences between the user interfaces of the first and second apps but also the subtle difference between the Free and the Pro versions, which is crucial for the choice between those two versions.

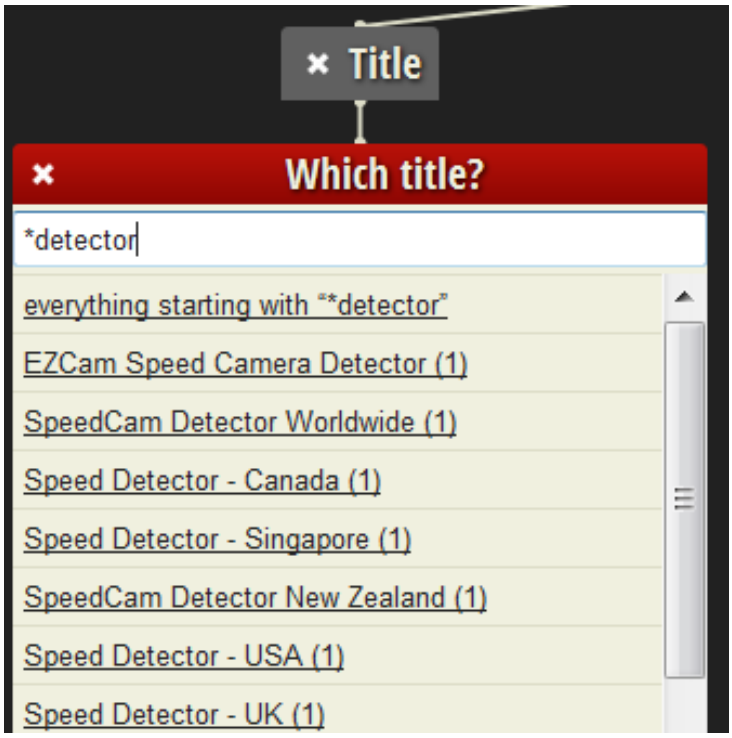

Figure 4: A query that includes wildcards can serve as a proxy for a category of interest that has not been defined within the system.

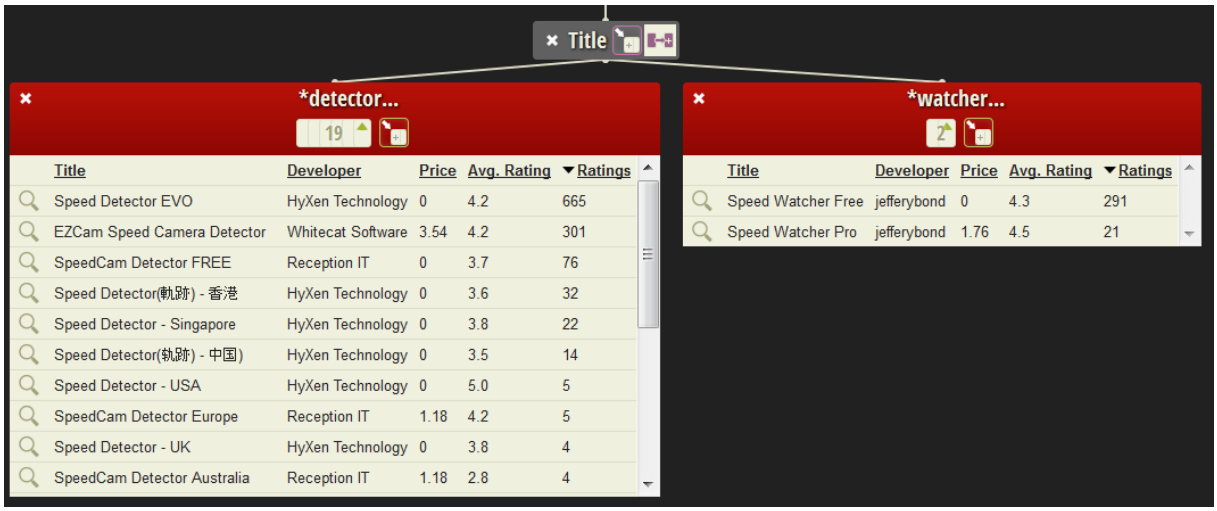

Figure 5: Parallelism is helpful when two or more keyword searches seem equally promising.

Figure 6: In particular, comparing reviews carefully is helpful because of the impracticality of trying each app out separately in the car.

4 Comparison With Normal Faceted Browsing

The benefits of the parallelism offered by parallel exploration can perhaps be seen most clearly if we compare this type of interaction with the most similar type of interaction that can be achieved using currently widespread interaction styles. For this purpose, we developed a variant of the app exploration demonstrator that sticks closely to the familiar and widely used paradigm of faceted search/browsing (see, e.g., Tunkelang, 2009; Hearst, 2009). This "faceted browsing" interface (Figure 8) includes all of the useful features of the main interface that are not necessarily tied to the idea of parallel exploration—such as the ability to filter by keyword search using wildcards—so as to ensure that all differences in the usability and usefulness of the two interfaces are due as far as possible to the presence or absence of parallelism. This alternative interface can accessed at the following URL:

Figure 7: Being able to examine screenshots of apps in parallel is helpful both when the apps are very different and when they are very similar.

http://apps-for-your-car.com/fb

Figure 8 shows the state of the faceted browsing interface after the user has filtered to "Driving Aids" and titles containing the word "watcher". Note that, to get to this point, the user may have had to examine other categories and keyword searches one at a time, keeping the results in his mind while backtracking so as to remember which direction seemed most promising. At the same time, it must be granted that the traditional interface has a more orderly, less cluttered appearance, and it can show the filtering criteria continuously on the left-hand side of the screen. Both of these advantages are due to the fact that the interface does not need to display multiple interrelated queries and search result sets.

Similarly, when the user zooms in on a particular app in order to read its reviews, the reviews are displayed using a large proportion of the screen real estate—which has the benefit of making the reviews easier to scan. The drawback, of course, is that the user will have to click back and forth between the apps of interest if he wants to compare their reviews.

In sum, the parallel exploration interface offers a number of apparent advantages that address the challenges that are typical of search for in-car apps, but it is quite possible that at least in some situations users will do better with a conventional faceted browsing interface. A follow-up user study that is being conducted in 2014 using the APPS FOR YOUR CAR demonstrator

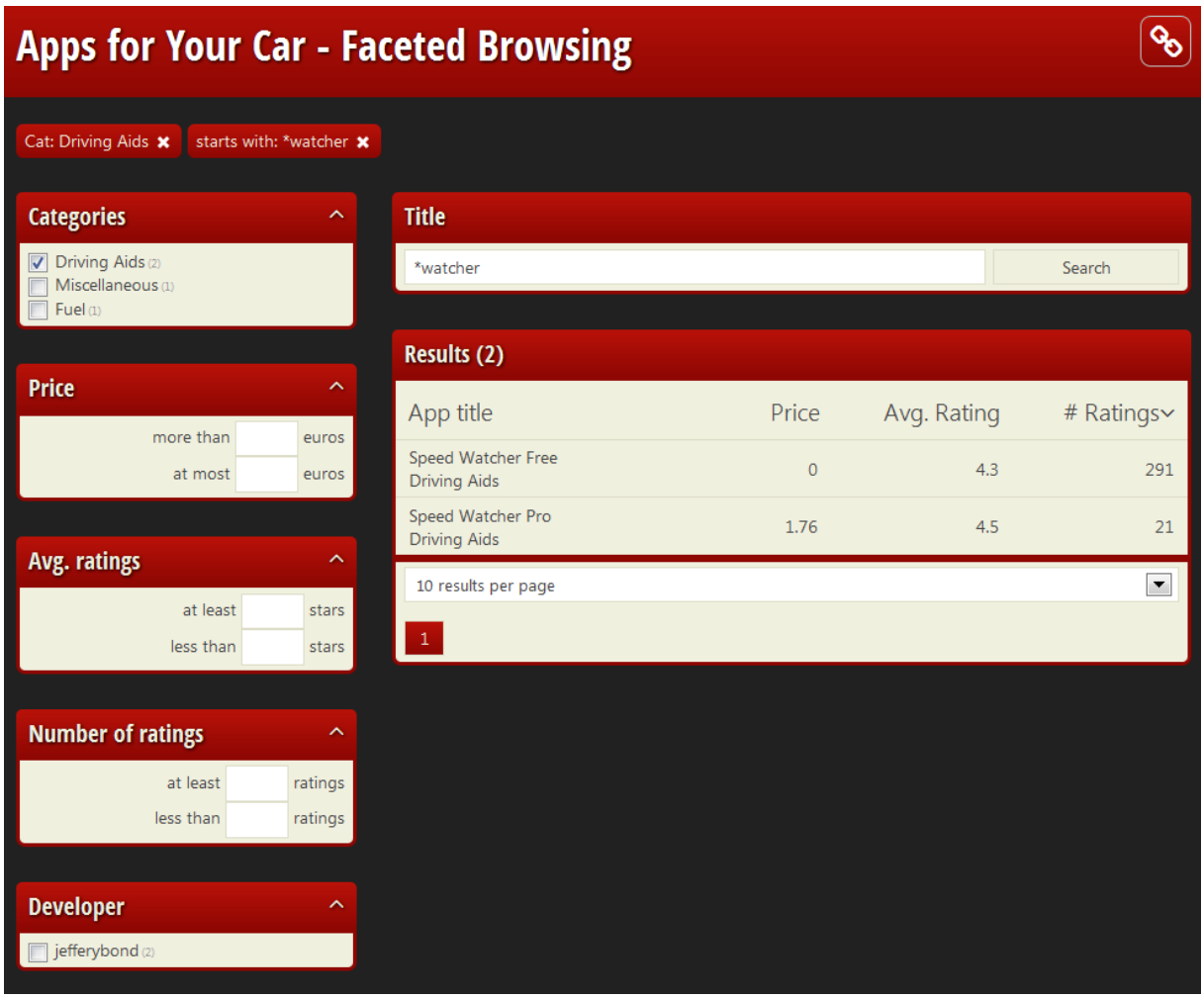

Figure 8: The nonparallel variant of the demonstrator realizes the familiar interface style of faceted browsing.

aims to shed light on this question. A possible ultimate solution is to offer both interface styles and allow users to switch back and forth between them or stick to a single style if they have a strong general preference for it.

The APPS FOR YOUR CAR demonstrator is serving during the year 2014 as a starting point for the implementation of the "Exploration Showcase App" within the high-impact EIT ICT Labs activity 3CIXTY. Consequently, all of the advances and lessons learned from the current demonstrator will be exploited and expanded upon, and a great deal of new user feedback will be acquired.

5 Practical Aspects of the Demonstrator

5.1 Acquiring Information About Apps

The main challenge in initializing the system was to find a sufficiently comprehensive store with in-car apps that allowed all of the information about apps to be extracted for use in the APPS FOR YOUR CAR demonstrator. The main Android store does not allow crawling, but the

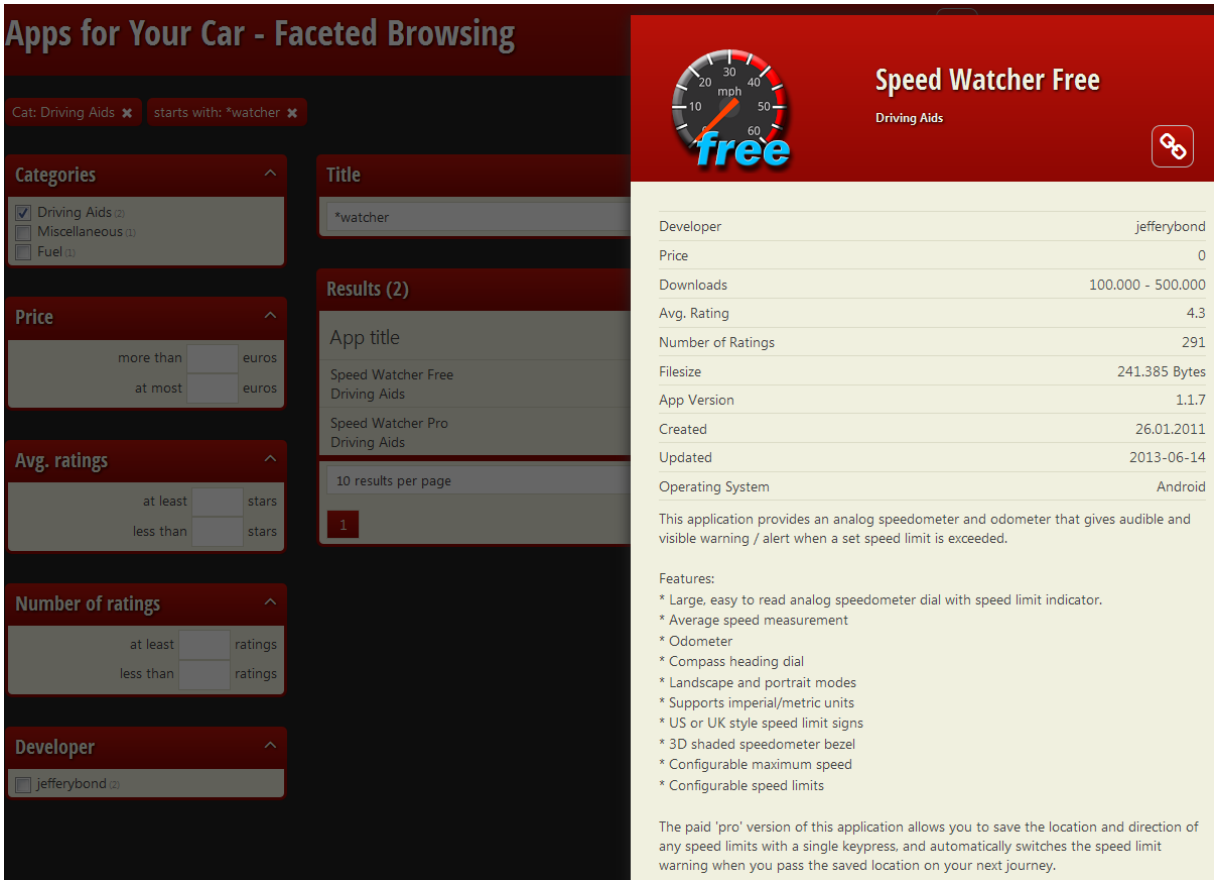

Figure 9: In the nonparallel interface, details concerning an individual app can be displayed with more screen space than in the parallel version.

website AndroidPIT[1] does.

Over 5500 Android applications have been preloaded into the demonstrator's database, complete with descriptions, price information, and user reviews and ratings.

Any app that seems especially relevant for car drivers is included, even if it is not specifically designed for cars (e.g., apps for searching for restaurants). The reason for this inclusiveness is that the system offers users enough possibilities to narrow their focus by themselves, so it does not have to be narrowed for them in advance.

5.2 Implementation Notes

The APPS FOR YOUR CAR demonstrator consists of the five components shown in Figure 11. This section describes the purpose and functionality of each component and explains briefly how the components were implemented.

5.2.1 Parallel Exploration User Interface

This interface is accessible via the following link: http://apps-for-your-car.com.

It is an HTML-based website that was implemented with JAVASCRIPT, using the framework JQUERY. It was tested primarily with the Firefox browser, but it supports all major browsers.

[1] `www.androidpit.de`

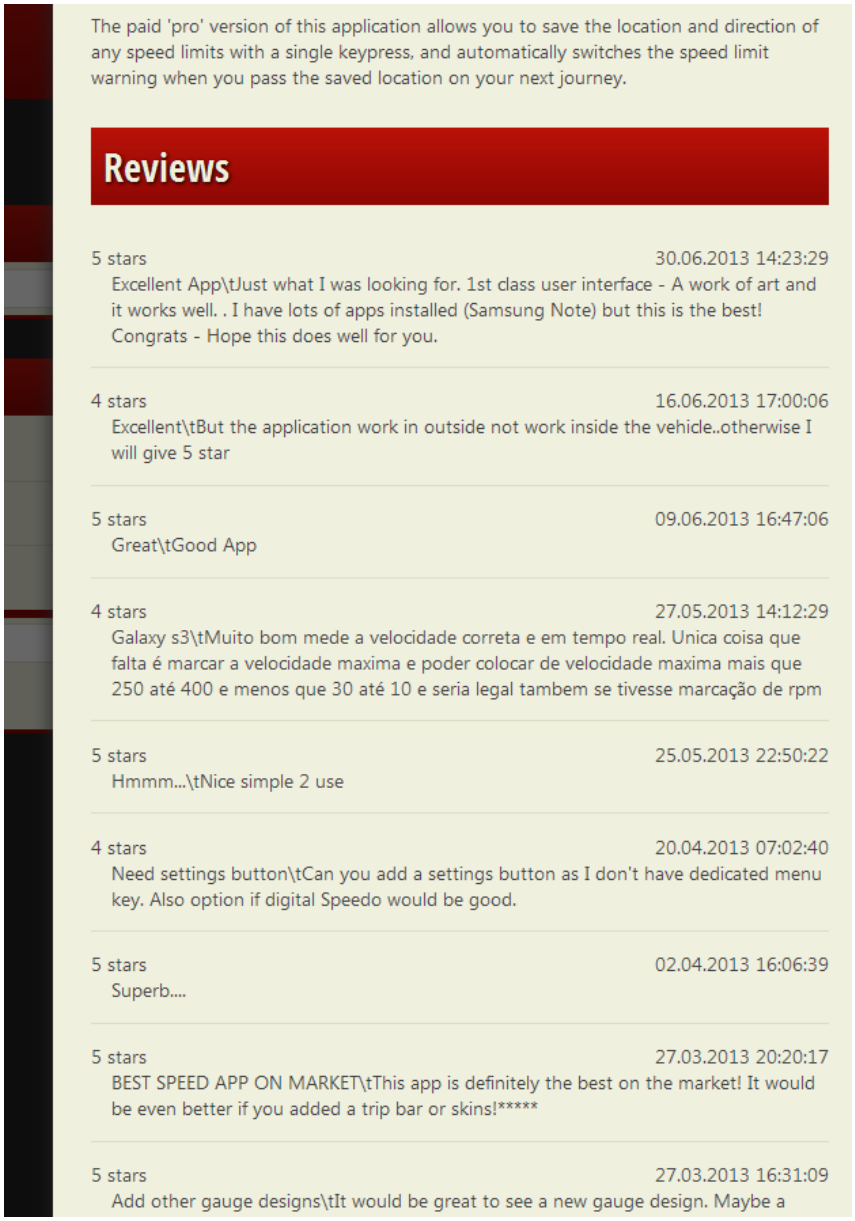

Figure 10: The additional screen space may be especially helpful when long textual reviews are available, though it comes at the price of an inability to make direct comparisons between sets of reviews.

5.2.2 Faceted Browsing User Interface

This traditional interface is accessible via the following link: http://apps-for-your-car.com/fb . In terms of implementation, the same techniques as for the “Parallel Exploration User Interface” were used for this interface, and it is connected to the same backend.

5.2.3 Backend

The backend was implemented in PHP. It accesses a MYSQL database, which stores both the data about the domain entities (in this case, the applications) and data about users’ actions, which are provided by the logging system. It returns, on the basis of the AJAX POST-Requests

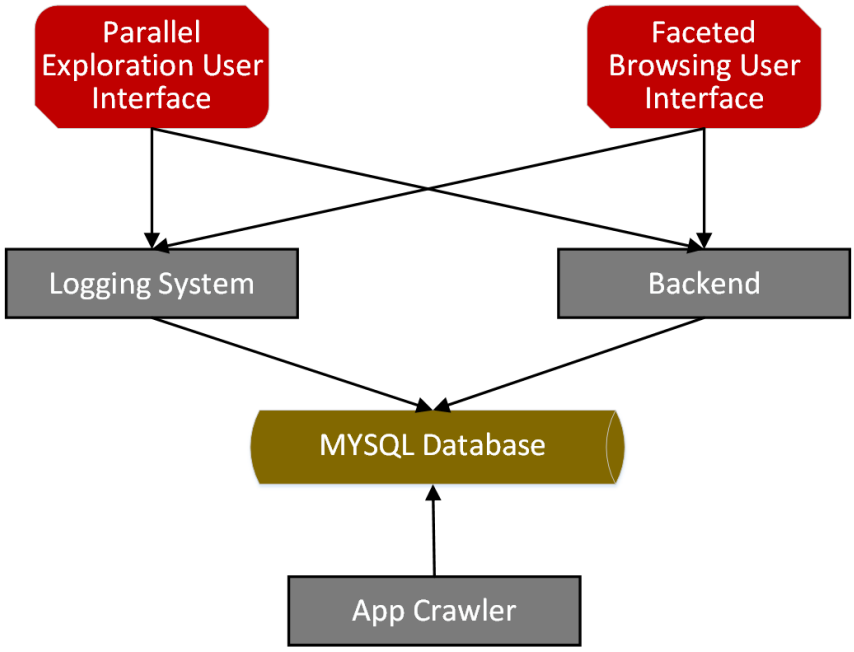

Figure 11: Components of the app exploration demonstrator. (An arrow from *A* to *B* means that data flows from *A* to *B*.)

from the two user interfaces, JSON-ENCODED data that is interpreted by the user interfaces.

5.2.4 App Crawler

This component crawls the data from ANDROIDPIT (http://www.androidpit.com) and stores the results in the MYSQL database. This component is also implemented in PHP.

5.2.5 Logging System

Every action taken by a user of one of the user interfaces (e.g., selecting an option from a menu) is sent to this component as a record with a number of fields such as User ID, time stamp, type of action, objects involved in the action. The component simply stores each record as it comes in. A user can query the MYSQL database interactively using MYSQL queries. A program can send queries to the component and use the answers to compute statistics or draw graphics. In a combination of these two cases, the user interface to this component offers a menu of standard queries which can be selected by a human user.

This component helps the programmers to test the functionality of the user interfaces, and it helps the interaction designers to understand the usage of the implemented functions by the users and to detect usability problems, including missing or not-needed functionality. More generally, it makes it possible to conduct user studies that involve fine-grained logging of users' actions.

References

Buschbeck, S., Jameson, A., Spirescu, A., Schneeberger, T., Troncy, R., Khrouf, H., … Hyvönen, E. (2013). Parallel faceted browsing. In *Extended Abstracts of CHI 2013,*

the Conference on Human Factors in Computing Systems (Interactivity Track).

Hearst, M. A. (2009). *Search user interfaces.* Cambridge, UK: Cambridge University Press.

Jameson, A., Spirescu, A., Schneeberger, T., Kapcari, E., & Buschbeck, S. (2013). Learnability and perceived benefits of parallel faceted browsing: Two user studies. In *Proceedings of the hypertext 2013 workshop on intelligent exploration of semantic data.* Paris.

Tunkelang, D. (2009). *Faceted search.* Palo Alto, CA: Morgan & Claypool.