Designing an Evolving Internet TV Program Guide

Patrick Baudisch
Institute for Integrated Information and Publication Systems IPSI
German National Research Center for Information Technology GMD
64293 Darmstadt, Germany
Tel: +49-6151-869-854
E–mail: baudisch@gmd.de

ABSTRACT

The system described in this article helps users in compiling personal TV schedules. The system's goal is to generate the schedules automatically based on user profiles. To tide over the profile's training period, the system provides the users with additional tools that help in finding the desired information manually or semi-automatically while generating training input for the profile at the same time. This approach results in an evolving system. While the system behaves like a printed TV program guide in the beginning, it develops into a browser/ query tool, offers persistent settings and bookmarks in the next stage and finally allows the users to compile the desired profile.

KEYWORDS

TV program guide, user profile, user modeling, adaptivity, adaptability, information seeking, information filtering, social filtering, user interfaces

INTRODUCTION

The goal of a TV program guide is to assist users in compiling TV schedules. The set of available tools a system can offer to accomplish that task depends on the medium. When using a printed TV program guide, users can typically choose between a complete table and several category-based listings. In an electronic, e.g. Internet TV program guide, additional tools are available, such as query tools and profile-based automatic filters. A profile-based filter is the most powerful tool a system can provide. Once users have specified their personal likes and dislikes, the system performs most tasks automatically, freeing the user from redundant work (see for example [Baclace91]).

The main problem is that a useful profile cannot be generated until the system has learned enough about the user. Therefore profile-based tools require a training phase. There are several possible training methods: The Ringo system [Maes94, Shardanand94] and its sequel Firefly [Firefly] use distinct phases for training and actual usage (see figure 1). When users log into the system for the first time they have to subscribe and train their profile for a while before they can get any recommendations. Since the users do not see an immediate reason for the subscription and the training mechanism, these requirements will scare away many potential users. See [Grudin94, p.97] for a discussion on the disparity in work and benefit.

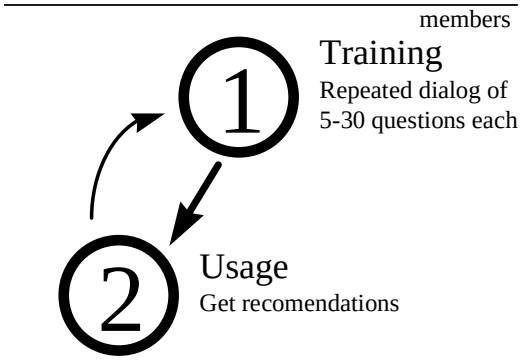

Figure 1: Firefly uses a distinct training period. The user can return to the training tool anytime to provide more input.

The approach presented in this article is to provide users with a set of tools instead of one single profile-based tool. While the profile based-tool is the most powerful of them, the other tools are less automated yet require no distinct training input. Therefore these simpler tools can be applied right from the beginning – before any profile has been built. The usage of each of these tools provides the information that allows the users to move on to the tool of the next higher automation level and finally allows the generation of a profile for the automatic filtering.

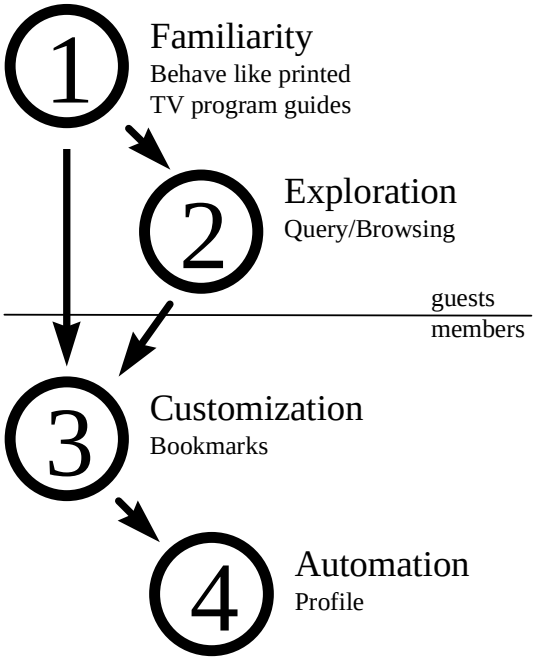

Figure 2: Each stage provides input for the following stage. No distinct training period is needed.

The proposed approach is split into four stages providing three different tools. Whenever enough input has been provided by the user, the user is led to the next stage. Since the functionality of all previous stages stays available, the system gains functionality over time. The four stages are shown in figure 2 and will be described in the following sections.

STAGE 1: FAMILIARITY

When users encounter a new system they usually have some expectations about the functionality and the usability of the system. In many cases these expectations will be based on their experiences with printed TV program guides. To help the users getting started, the system should try to do what the users expect. If a printed TV program guide provides its users with functions such as an exhaustive table of today's program, the electronic system should offer that too, even if it could offer much cleverer solutions for the same problem. By adding a printing function to the system users can convert the electronic guide to a printed guide if they like. No subscription is needed in this stage.

STAGE 2: QUERY AND BROWSING

After having used the system as a substitute for a printed guide for a while, some users will explore the system and try out other functions. Since the print-medium-inspired functions always worked on the whole set of broadcasts of a given time interval, the next step is to reduce the number of broadcasts users have to check.

The query tool we are developing consists of different menus that allow users to specify queries. The query parameter most relevant to personal taste is the category. Typical high-level categories are 'action', 'comedy', 'sports' or 'soap'. See [Steinm96] for a discussion on genres/categories. While some other parameters are of a more formal nature (i.e. date, time, channel) the category is a broadcast attribute that contains information about the broadcast's content. Therefore the category is especially relevant for profile generation. To consider this, the category menu offers a higher level of detail compared to the other query menus. It is implemented as a tree style menu that displays a hierarchical set of categories. Beginning with the root node that represents all broadcasts, each category can be expanded into sub-categories.

The browsing mechanism for queries gives users an impression of the category space. Because all other categories on the level of an executed category are shown as well, users do not only see what they get but also what they excluded from the query. To encourage users to browse the category structure, category expansion is fast.

The category structure in an electronic service can be more powerful than the one in a printed TV program guide: It does not need to map to a linear structure, and since it can be expanded step by step its overall size can be much bigger. The category structure in an electronic service can be implemented as a deeply structured directed acyclic graph (DAG) to organize categories, while printed program guides are usually restricted to shallow trees. A DAG structure has a major advantage over a tree: Since it allows broadcasts and sub-categories to be part of more than one parent-category, users can find the same broadcasts and categories via different paths in the DAG. Thus the DAG structure supports different traversal strategies on the categories. The fact that a broadcast cannot always be assigned to exactly one genre leads to major problems in tree structured categories, but not in the DAG, where a user can find a broadcast in *all* related categories. The hierarchical structure of the categories defines different abstraction levels. On each level users have the choice to explore the next level or to execute the query defined by the current level already. This way they can adapt the level of detail to their personal level of interest in that part of the category structure.

STAGE 3: BOOKMARKS

After using the query tool for a while, users will execute some queries repeatedly. Since traversing the category structure and entering query parameters can be time consuming, the next step is to reduce interaction redundancy by allowing interesting queries to be saved. After having saved a 'bookmark', the user can recall it any time and execute it in the current context, i.e. in the current time and date. An implementation of the bookmark idea can be found in the Internet TV program guide 'TV-Movie Online' [TVMovie].

To provide additional functionality on the 'bookmarks' like the profile described below, bookmarks have to be saved on the server side. Therefore this mechanism requires a user account, i.e. a subscription. As long as users do not want any active services like daily emails an anonymous account is sufficient, thus granting discretion. The account can be used to provide the user with additional customizations like channel and viewing time settings.

STAGE 4: PROFILE AND AUTOMATION

After some more sessions the users have 'bookmarked' a set of their favorite queries. Some of these depend on specific moods so they execute them only once in a while. They execute other queries every time they log into the system. These queries do obviously not depend on a mood – they represent the mood-independent base of their taste. To speed up the traversal of this set of bookmarks during every session, the next step is to combine these bookmarks to form a single complex bookmark. When users have built such a 'taste-bookmark' or 'profile' they can execute it instead of all the bookmarks it is made of.

Since the result of the profile tends to be rather lengthy, it is especially important to provide it with a relevance order. The relevance order allows users to check the most relevant broadcasts first or even to ignore the non-relevant ones completely. To calculate a relevance order for the profile, the system needs a relevance order on all involved bookmarks plus the

information on how to combine them. Relevance orders for simple queries can be won from approaches like social filtering, see [Maes94].

One possibility to compile the single relevance orders into the relevance order of the profile, is to apply the formula: relevanceInProfile = relevance * height + indentation. In this formula 'indentation' determines the relative relevance of a bookmark while 'height' determines the ratio by which the results of two similarly rated queries are mixed.

Figure 3 shows a screen shot of the profile editor we are developing. It allows the user to specify the two parameters 'height' and 'indentation' interactively. The editor is implemented as an extended mode of the simple bookmark list. In the bookmark list each bookmark is displayed as the set of its query parameters. In the profile mode each bookmark is displayed with an additional rectangle having a surface proportional to the query's 'volume', i.e. the average number of broadcasts per week returned by this query. The horizontal position of a bookmark determines 'indentation', i.e. the bookmark's relevance. The height of a bookmark's rectangle determines the factor 'height'. The editor allows one to shift rectangles horizontally to modify their bookmarks' relevance and to deform rectangles to modify their 'height' keeping the rectangle's surface constant.

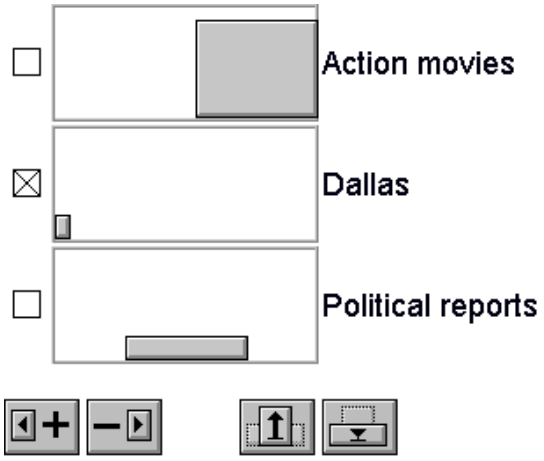

Figure 3: The profile editor. The category 'action movies' has the highest volume as visualized by its surface. Since relevance is ordered from left to right 'Dallas' has the highest relevance. When executing this profile 'Dallas' is output first, then the best 50% of 'political reports', then the remaining political reports mixed with the most relevant action movies and finally the remaining action movies. To modify the checked bookmark's relevance, rectangles can be shifted by the user by clicking the '+' and '-' arrows. The aspect ratio of rectangles can be altered by clicking the deformation buttons.

TOOLS AND MOTIVES

Each of the three presented tools, category browser, bookmark and profile editor is especially useful for a different task. Which tool a user finds most adequate for generating the personal schedule depends on why that person wants to watch TV. Among others three basic motives play a role: 1) information seeking 2) emotional need and 3) pastime. These motives are not mutually exclusive. Virtually all broadcasts can be viewed as pastime. Most broadcasts will satisfy some emotional need. Many broadcasts deliver some sort of information, even if the information passed often has aspects of entertainment, e.g. results of sport events.

If users looks for specific information in the first place they might want to use a query/ browsing tool. Assuming that their current informational needs can be satisfied once and for all, a profile is not of much use. Emotional needs occur repeatedly. Therefore they are best supported by bookmarks which can be executed separately. In case users are looking for an entertainment viewing schedule, well-trained user profiles return the best results for the lowest effort.

CONCLUSION

The proposed system helps users in compiling their personal TV guide by building a user profile that can finally generate the schedule automatically. Since profiles require training before they can return useful results, the proposed system provides the users with two additional tools: category browser and bookmarks. These tools accomplish two tasks: 1) They provide the user with additional support for the information seeking and emotional-needs motives and 2) they provide the information necessary to get to the tool of the next higher automation level. The result is an evolving system.

The system suggests the usage of the next tool whenever the user shows interaction redundancy that can be reduced by introducing that tool. If the user executes a query that results in a reasonable number of broadcasts the system will suggest creating an account and bookmarking the query. If the user uses some bookmarks repeatedly the system will suggest converting them into a profile.

The fact that users can cause the system to evolve does not mean that all users will reach every stage: Any of the stages described might satisfy some users

who will not push the system any further. This way the system becomes just as powerful and complex as the user wants it to. After all a profile is not very useful for sporadic users.

The main advantage of the proposed system is that it does not require a distinct training phase: Users can profit from the system right from the beginning.

BIBLIOGRAPHY

1. [TVMovie] http://www.tvmovie.de
2. [Firefly] http://www.firefly.com
3. [Maes94] U. Shardanand, P. Maes: *Social Information Filtering: Algorithms for Automating "Word of Mouth"*, CHI'95 Human Factors in Computing Systems, S.210-217, 1994
4. [Shardanand94] U. Shardanand: *Social Information Filtering for Music Recommendations*, 1994
5. [Baclace91] Paul E. Baclace: *Information Intake Filtering*, Draft, Bellcore Information, Filtering Workshop, November 8,1991
6. [Grudin94] Jonathan Grudin: *Eight Challenges for developers*, Communications of the ACM, January 94, Vol. 37, No. 1
7. [Steinm96] R. Steinmetz et al; *The Personal Electronic Program Guide - Towards the Preselection of Individual TV Programs*, 1996