6-9-2000

Personal Radio

John C. Artz Jr
Dartmouth College

Dartmouth College Computer Science Technical Report
PCS-TR2000-372

Personal Radio

by
John C. Artz, Jr.

Honors Thesis

Advisors: Dave Kotz, Daniela Rus

Department of Computer Science
Dartmouth College
Hanover, New Hampshire

June 9, 2000

Department of Computer Science

Dartmouth College

Personal Radio

John C. Artz, Jr.

Honors Thesis

ABSTRACT

With the development of new technologies that allow the broadcast of digital data over radio signals, there are many possibilities for improving upon the traditional radio station model for content delivery. The idea of Personal Radio is a system that tailors content to meet the needs of each individual. Using Global Positioning System (GPS) technology to play location specific content, the listening history to play content an appropriate number of times, and user feedback to learn personal preferences, the Personal Radio provides the listener with the content that is the most useful/interesting to them. This paper will examine the general design of such a system and present solutions developed in the implementation of several pieces of the design.

Table of Contents

ii List of Figures

iii Preface

This preface introduces the reader to the history of the project and gives acknowledgement to the people who have assisted in the development of this project.

iii.1 Project History

When we began investigating what applications of wireless technologies could be used in cars to provide an improvement over the experience users currently have while in their cars, we came up with a concept of personal assistant that could provide information access within the user's car. It would be able to help users find restaurants, gas stations or a specific product. It would then give them driving directions to this location and then help them find an empty parking space. We conducted a survey to gauge user interest. Though many people were interested in this system, many people were worried about the safety of such a system. Additionally, they felt that many of the services were already available over the radio.

In considering the implementation of such a system, we came to the realization that there was not much that would be realistic or interesting to implement. Designing a product/service location system would only require the creation of an accurate map and databases that represented the products and services available at specific locations. Though driving directions are an interesting and challenging problem, many companies are already developing products to handle this task and it seemed futile and pointless for me to attempt to duplicate their work with this project. Finally, the application of locating empty parking spaces, while quite interesting and extremely useful seemed to be more of a sensing problem than a computer science problem. The task would be to develop an accurate and cheap sensing network. After the map of empty and not empty spaces was created, it would be fairly simple to develop a system that would direct users to these parking spaces.

So after exploring the first route for several weeks, we decided to abandon that path for our current project. Ironically enough, the current project was similar to an idea that I had last summer, save that we are using it globally, rather than on the internet as I had imagined.

iii.2 Acknowledgements

First I would like to thank Professor Dave Kotz and Professor Daniela Rus, my advisors, for all of their assistance. Our weekly meetings were always a source of inspiration and invigoration for me. Without their dedication I would have given up on this project long ago.

I would like to thank Stan Williams for listening to our many ideas and helping us find direction.

Additionally, I would like to thank Bob Gray, and Karen Nakamura at GPSY.com for their help with the GPS devices.

Ned Holbrook, as always, was an invaluable help with programming on the Mac.

Many thanks to Mark Mounts who helped me with my library research.

And finally, I would like to thank all of my friends and family who supported me during this process. Laura for keeping me sane, Ian for keeping me in school, and everyone else for listening to me talk about my thesis.

1 Introduction

Our vision is to create a system that allows users to access digital audio content anytime, anywhere. Currently access to audio information is done via the radio, or a recorded medium such as tape, CD, mini disc or Mp3. While these methods of audio access allow users to hear audio content, it is within a rigid framework. Radio stations control exactly what you hear, and with recorded media, you know exactly what will be played before you play it. We envision a world in which users can, at any time, listen to the “radio” — request news, weather forecasts, stock quotes, traffic reports or other information and the most recent, location-relevant and user-appropriate content will be played.

In this paper, we will consider the design of such a system, analyze how current companies are planning to deliver digital audio content, offer several algorithms that will allow us to construct this system and present the basic implementation of Personal Radio.

2 A Vision of Personal Radio

Imagine the following... *About 6:05 p.m. everyday I leave my office and get into my car. I plug my Personal Radio into my car stereo and turn it on. I like to hear the traffic report first thing so that I can avoid any potential snarls on my way home. I push the traffic button on my Personal Radio and it begins reporting some of the spots where traffic is bad. Unlike traditional radio, it doesn't begin with traffic reports about the worst spots in the city. It begins with traffic reports that are close to me, and in the direction I am heading. The second report I hear describes an accident on my route home, so I choose to take another route. After this report, the next accident is far from my route and so I can stop listening to the traffic report and catch up on the current events of the day. As with the traffic report, the news is customized to my personal tastes.*

Eventually, the system plays a commercial. Potentially, the user might be able to pay a monthly fee and avoid commercials, but in general commercials are necessary and sometimes even useful. The problem with commercials on traditional radio is that they don't usually pertain to your situation and you hear them many times. Commercials can be useful — if I am looking to buy a new car, I would want to hear about a sale at a car dealership. Personal Radio would use personalization strategies to provide targeted advertising.

As I drive home, I hear a commercial for cheap gasoline at a gas station that is on the way home. The combination of my hearing the advertisement and seeing the gas station just ahead, cause me to stop and take advantage of the sale. Targeted advertising is good for advertisers as it means greater returns on their ad expenses, and also helps consumers as it allows them to take advantage of various special offers.

I decide to listen to some music. As with the other types of content, my music selection is tailored to my personal tastes. As I listen to songs, I have the capability to skip over a song I don't want to hear, rate how much I like a song, or just listen. If I rate a song, Personal Radio will adjust how frequently it plays that song, as it has learned more about my personal preferences.

While this example is by no means comprehensive, it serves to give a general idea of how a user might take advantage of Personal Radio.

3 Motivation

Currently, everyone receives audio content in two ways. It can be chosen for you and you can tune in and be presented with the content, or you can seek out various content pieces. For example, under the radio and television models, you turn them on and they present you with programming that has been chosen for you by someone else. Alternately, if you rent a movie to watch or buy a CD to listen to, you are picking your own content. Though both of these methods of content exposure have similar results, the motivations behind choosing one of the methods are quite different. Currently no system caters to both of these motivations. By exploring the reasons that people choose these different methods, perhaps one system can be developed that satisfies the needs of both situations.

There are advantages and disadvantages to the two ways to receive content. If your content is chosen for you, you will hear content that you might not have known about. Also, you can tune in just to see what is on. There are times when you might be looking to relax and this type of content presentation is exactly what you want. However, this model does not provide you with any choice. You are unable to access the content that has not been chosen to be presented to you. When you pick your own content the opposite is true. While you have unlimited access to various content pieces, it is more difficult to simply be entertained by this model. If you want to zone out for a little while, it will be difficult if you must constantly be picking what you want to hear next. There are times when each of us wants to be entertained and times when we each want to hear a specific piece of content.

Rather than restrict users to one of these situations, we want to create a device that can satisfy both their desire to be entertained, and their desire to hear specific pieces of content. Users will not have to directly pick the content that they are exposed to, but they will have the capability to do so. Additionally, the system should use feedback about the content it plays so that it can learn users' preferences over time. If the user wants to be entertained, the system will happily choose content for the user. If the user wants to hear a specific piece or type of content the system should be able to respond to this request and produce the desired content.

4 Scientific Goals

Our goal, when we began this project, was to develop a prototype system to play personalized, location dependant content, continually playing content based on the learned preferences of the user and also responding to user input. This means that the client must be able to both 'push' content to the user and 'pull' content at the user's request. These two modes will allow the user to both be entertained and also hear what they want, when they want it. In order to create this system there are many technical issues that need to be addressed.

- How does the server decide what content should be distributed?
- How does the content get delivered to the users?
- What should the protocol be for user's requesting specific content?
- What algorithms should be used to determine what to play next?
- How can the system use Global Positioning System (GPS) data to determine how relevant location based content would be to our current situation?
- How do we introduce randomness into the system in order to keep the content selection interesting?
- How does the system learn user preferences?
- What does the interface for such a system look like?
- What do users want?

5 System Goals

In every system, certain things must happen in order for the system to "succeed." The first system goals are essential to making the system work. The system must:

- Generate the next content to play before the currently playing content has finished playing.
- Cache new content to play.
- Whenever skip is pressed, a piece of content must be available to play. The number of items in the cache must be greater than the number of times users want to skip the current content.
- System must play content.

The following goals, while not essential to the actual execution of the program are necessary in order to qualify the system as successful. The system should:

- Play an appropriate amount of new content. To determine this, ask users if they hear too much "new" content. Record how many content items are new, and how many they have heard previously. If users complain of too much new content, adjust the content selection algorithm until it changes the ratio of new to old so that the users are pleased.
- Learn user's preferences in a reasonable time. This is extremely hard to measure because reasonable is such a subjective word. A user study might suggest how long people are willing to wait for the system to learn their preferences. A status bar about how well the system thought it knew the user's preferences would probably allow the system to take longer to get the preferences right. When users can perceive progress, they are willing to wait far more patiently.
- Produce a correct play list. To measure the correctness of a play list, all content should be played proportionately to its weight.

- Choose content that is pleasing to the users. This measures the overall success of the system. If the system cannot please the users, it will fail even if everything else is perfect. To measure this, a user study must be done. While running the user study, have users report periodically about how well they like the content that they are hearing. At the same time, record how often the user likes or dislikes the content that has been chosen by recording how often they rate content. If the system must constantly change the rating of content then it is probably not playing content that the users want to hear. A second measure of how successful the system is could use how frequently the user presses the skip button. For example, the system is not successful if the user listens to 10 seconds of 12 different songs and then turns the system off.
- Present an intuitive and useful interface. To determine if this is the case, ask the users which pieces are most important to them and what things they wish they could do that they cannot currently do. At the same time, record how many times each button is pressed. The buttons that are both not important and pressed very infrequently should be reconsidered.

6 Basic Architectural Elements

Before we explore the details of the system, we will examine other work that can assist in the design of the system. However, a basic understanding of the main architectural blocks must be achieved in order to understand how the related work compares to the needs of our system and where it will give guidance in the design process.

The system can be broken down into four basic pieces. Figure 1 diagrams the relationship between those pieces.

- There must be some way for new content to be distributed to the device, thus the Content Distributor.
- Once the content has been delivered to each device, the device must maintain a cache of content to play, so that it can have sufficient content from which to choose items to play.
- The main piece of this system is the Content Decider. It is responsible for choosing which content to play. It then is responsible for telling the playing device what to play.
- Finally, there must be some controls for the system which the user can use to make their desires known. The input should include the following:
  - Category Selection
  - Content Rating
  - Content Control (Run, Stop, Skip)
  - Preferences

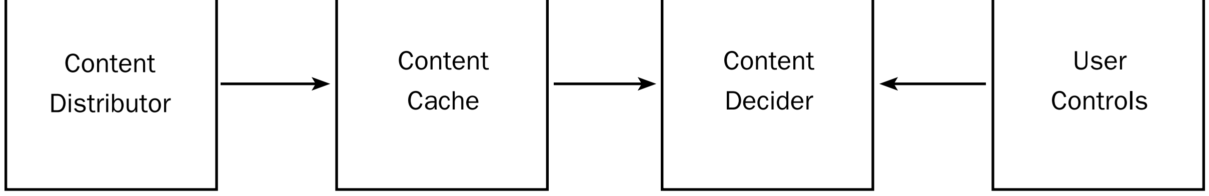

Figure 1.

The four basic blocks of Personal Radio.

7 Related Work

For each of these basic blocks, there are some related areas of Computer Science can give guidance in the design of these components. In this section, each will be presented and the pieces that can be used and the parts that are missing will be identified. The areas include:

- **Broadcasting** - Content Distribution
- **Caching** - Content Cache
- **Scheduling** - Content Decider
- **Priority Queues** - Content Decider

Additionally, companies doing similar tasks will be analyzed.

7.1 Broadcasting

To distribute the content, a broadcast system of some kind will be the most appropriate. This way, when there is new content to be distributed, all devices can listen to the content and choose to save it for later playback or choose to ignore it. Broadcasting is more appropriate than point-to-point connections because all content has the potential of being interesting to all devices. Rather than sending the same content down many individual links over time, just send it once to everyone.

Imielinski and Viswnanthan in [4], present a broadcast publishing system that relates directly to the Personal Radio design. Rather than broadcasting radio content, they focus on the publication of data such as stock quotes, files and other such data items. However the strategies they discuss can be employed to broadcast the radio content.

> Publishing is a spontaneous and periodic broadcasting (or multicasting) of data on the wireless channel by the MSS to a specific group of clients… It involves client initiated filtering of the published data stream which arrives on the downlink channel. (Imielinski, 301)

This concept of "publishing" describes precisely the model that our system should employ in order to broadcast the data. All the data gets broadcast repeatedly, and each device listens to what is being broadcast. If a device wishes to request a specific piece of data, they connect to the broadcast server and request the content. This content is then fit into the broadcast stream. Much of [4] is spent analyzing this process and how it impacts user wait times.

While [4] nicely outlines many of the broadcasting techniques that would be useful to broadcasting the content, the ideas of broadcast publishing come from an earlier system design. In [5] Gifford et al. describe a system that would use radio waves to allow information access via personal computers. This system, though it uses text based content, has many similar goals to the system that we design. This work was done in the mid 1980s, but his analysis of the benefits of the broadcast approach still apply directly to our model.

> The approach of sending information to the user's location and processing it there has a number of advantages. First, the central site can support any number of broadcast service users. Second, locating processing power with the user allows for a high-quality user interface. Third, local processing and storage can be used to assist the user in managing a larger volume of available information... (Gifford, 458)

7.2 Caching

Once the content has been delivered to the client, it is up to the client to decide what to do with it. As both space and time for playing are limited, we want to keep only those items that have a chance of being played. To determine what to keep, the system should decide how likely it is that it will play this content and keep only the most likely content. In addition to simply deciding which content to keep on disk, we must choose which content to have in memory.

Traditional caching theory proposes three main alternatives for how to load items into the cache. Direct Mapped cache access maps each location on disk to one specific location in the cache. So if we want an item and it is not in the cache, we must replace a specific piece of the cache with the new item. Fully Associative cache access eliminates the mapping requirement. Each item can be put in any location in the cache. Finally there is Set Associative cache access which is a hybrid of the other two cache access types. It provides buckets for different chunks of memory. This allows one item to be in a small number of locations within the cache.

Each of these caching strategies could be employed effectively in our system. For example, there could be one location for each of the content categories. This would function like a Direct Mapped Cache. This would allow quick switching between categories. Alternately, we could employ a caching algorithm and just put content at any location in the cache. Finally, the Set Associative cache access might be most useful because it would allow us to have buckets for each category, achieving the benefits of both types.

Unlike a traditional cache, the items that have been used most frequently are not likely to be used again in the near future. So rather than employing a Least Recently Used replacement algorithm, we might want to use a Most Recently Used Replacement algorithm. (Hennessy, 376) Thus we would replace the items that had just been used, with items that might be used soon.

In [7], Young describes an algorithm called the Landlord Algorithm where each item in the cache is provided with a certain amount of credit. When an item is needed, if it is in the cache, it acquires additional credit, but if it is not in the cache, the system charges "rent" to all other cache items and then removes items who have run out of credit. This algorithm can act like a least recently used paging strategy, or a first-in-first-out paging strategy depending upon the amount of credit given to a re-used item. (Young, 3)

The Landlord algorithm, while it generalizes quite nicely traditional caching mechanisms, has difficulty meeting the different needs of a system such as Personal Radio. As with traditional caches, we want the items that we are going to use to already be in memory when we want to use them. The only difference is that the items that we have just used are likely not be used again for some time. At the same time though, all items that we haven't used don't have equal chance that they will be used sometime soon. The items that are in the cache should be chosen based on their chance of being played in the near future. Perhaps an algorithm that replaces the items that are least likely to be used, would be more appropriate than an algorithm based on the time it was last used. Since the chance that it would be likely to be used (played) would be based upon the time it was last used, the caching algorithm would just utilize *additional* information to decide what to have in memory.

7.3 Scheduling

Once we have the content, it becomes the goal to decide which content should be played. To decide what to play next we can employ a strategy similar to those used when scheduling processes within an operating system. There are many algorithms used to do operating system scheduling. They can be preemptive or non-preemptive, meaning that one process can be interrupted for a more important process. (Tanenbaum, 63) Since a user might want to interrupt the content stream for an emergency, and they defiantly want to interrupt when they choose to skip a content piece the system should employ a selectively preemptive algorithm. Under ordinary circumstances, the system runs in a non-preemptive mode with each content piece running to completion. If a content item arrives with a priority set above a certain threshold then the current content piece will be interrupted and the new content piece will be played instead. Additionally, if the user tells the system to skip the current content, the system will cease the playing of the current content and begin with new content.

Additionally, Operating System scheduling considers the order in which to execute tasks. Though there are many algorithms designed to give each task a fair time allotment, priority scheduling is the algorithm we want. Rather than giving each content equal play time, we want to play the content that the user wants to hear more frequently. Tanenbaum describes this process “each process is assigned a priority and the runnable process with the highest priority is allowed to run.” (65) Though he discusses ways to decide priorities none are really appropriate to the content decision task. The key to making the content decision task work is to choose the priorities correctly. This paper shall explore that in section 9.

Tanenbaum does mention one piece of priority assignment that applies to our paradigm. He discusses the idea of starvation — that low priority tasks will never be able to run because the high priority tasks might run indefinitely. The solution he describes is that the priority of high priority processes should decrease over time so as to allow lower priority processes a chance to run. (Tanenbaum, 65) This concept will be key for our priority selection algorithm.

7.4 Priority Queues

In addition to the actual scheduling of the content, we need a data structure that will easily allow the selection of each content piece. The obvious choice would appear to be a priority queue. Cormen, Leiserson and Rivest (CLR) describe a priority queue as follows: "A priority queue is a data structure for maintaining a set S of elements, each with an associated valued called a key. A priority queue supports the following operations :[Insert, Max(), Extract_Max()]." (CLR, 149) There are many ways to implement a priority queue. Most use some sort of list or tree. One implementation in CLR uses a heap. While a heap would allow quick removal of the elements with the highest priority, the strategy that we employ will introduce a slight degree of randomness, counteracting the benefit of being able to extract the max element quickly. In [10] Brown analyzes the sorted list implementation of a priority queue. "Both of the linear list schemes are easy to implement and are quite efficient when the queue size is small." (Brown, 6) For the purposes of this project, a sorted linked list structure seems to be the simplest and most efficient structure.

7.5 Competitive Analysis

To determine the controls that the user should have available, a variety of Internet Radio systems were examined. Most had controls similar to a CD Player: Play, Pause, Stop, Skip. In addition a few had the capability to rate the current song. On some of the players though, the rating buttons were under a separate menu and multiple buttons had to be pressed in order to rate a song. To simplify this process, options that will be frequently used, such as the rating system, must be easily available.

There were two obvious ways that these radios earned revenue. Either they had banner ads displayed in the player, or they played commercials in between songs. One player seemed to play commercials after a certain number of songs. Of course, when a user skipped three songs, they would then be subjected to a commercial. If they then skipped another three songs, they would then get another commercial. It was not possible to skip the commercials so the user might wind up listening to all commercials (if the device doesn't choose good content to play). Rather than choosing when to play a commercial based on the number of songs chosen, perhaps the algorithm should allow a certain amount of time between commercials. This way a user could skip songs for that time before being subjected to commercials.

For the remainder of this section, various companies who are acting in the digital content arena shall be examined. Their strengths and weaknesses shall be identified. This analysis shall show that the potential competitors in the personalized radio space all are missing key elements that keep them from meeting the goals we have outlined for the system.

7.5.1 Digital Radio

Companies such as USA Digital Radio ('USADR'), Lucent, and Digital Radio Express ('DRE') are working on developing and deploying Digital Audio Broadcasting ('DAB'). This system would simply replace the terrestrial analog abrogating with a digital broadcasting method. USADR has developed an in-band, on-channel ('IBOC') DAB allowing current radio stations to broadcast both analog and digital content over the same channel (iDAB). According to their web site their technology "provides for enhanced sound fidelity, improved reception, and new data services." [1] These services will be available soon. In [2] the FCC outlines many of the issues concerning the development of these technologies.

This technology should revolutionize the quality of the sound that we receive on our radios. However, it does not alter the way in which radio stations operate. Though this is beneficial to the radio stations, as it is easy to adopt, it does not provide consumers with the benefits capable due to the power of portable computing. As will be shown by the remainder of the companies that shall be examined, none of these broad market areas have put together all the pieces. While the digital radio companies are doing a good job improving the broadcast quality, they are ignoring improvements that can be made to the radio players.

7.5.2 Satellite Radio

In addition to the traditional radio station, two companies are developing Satellite Radio systems. XMRadio and Sirrus Satellite Radio (formerly CDRadio) are rushing to launch satellites, develop reception devices and deploy their systems. These systems work similarly to satellite television. Each user would pay a monthly fee and in return receive 100 crisp digital radio stations. The quality of this sound will be excellent and they intend to offer 100 distinct stations each catering to a different taste. One station might play opera and another might play traditional Celtic folk tunes. They intend to offer a wide variety of content that is not usually offered in most markets.

Once again, the satellite radio companies are not revolutionizing the radio industry. Though they will provide many additional channels, a user must channel surf to find what they are looking for.

7.5.3 Internet Radio

This group of companies has gotten the playing mechanism right. Companies such as Sonicnet.com, MyCaster.com, ClickRadio.com and Launch.com provide users with a personalized radio service. Users typically download a player and spend some time customizing the type of music that they want to hear. Then the player begins to play music. Most of the players use a streaming format to play the music and therefore the quality is unfortunately low. They also allow users to skip songs and rate songs. The radio should, over time begin to play songs that the user prefers, based upon how they rate the songs. However, it seems to take quite a while before the player begins to play content that is tailored to the user. In non-scientific tests, the time taken to begin to play "good" music was longer than the user was willing to wait.

Figure 2

LaunchCast Player

Though the personalization is an excellent idea, until it is better implemented and the quality of the sound improves this idea will not replace traditional radio. Additionally, these products require users to be tethered to their internet connection.

7.5.4 Mp3 Players

As is shown by the huge popularity of sites such as napster, Mp3s are quickly becoming the accepted standard for digital audio. Many products are available that will play these near CD quality sound files. It is quite simple to download an Mp3 player for any computer. Recently, there has been a growth of portable Mp3 devices. These work just like a walkman or portable CD player, except that they don't skip and always sound good due to their digital sound format. In addition to the portable players, recently companies like eMpeg and DelphiAuto have been introducing devices designed to play Mp3s in your car.

While both the portable Mp3 players and the Mp3 in-your-car players provide excellent sound quality, in order to load them with songs, you must hook them up to your computer. Finally, they play the content as a traditional CD player might, either randomizing or simply iterating through the content. These systems need a better way to download content to the device and the playing mechanisms need to take advantage of personalization techniques.

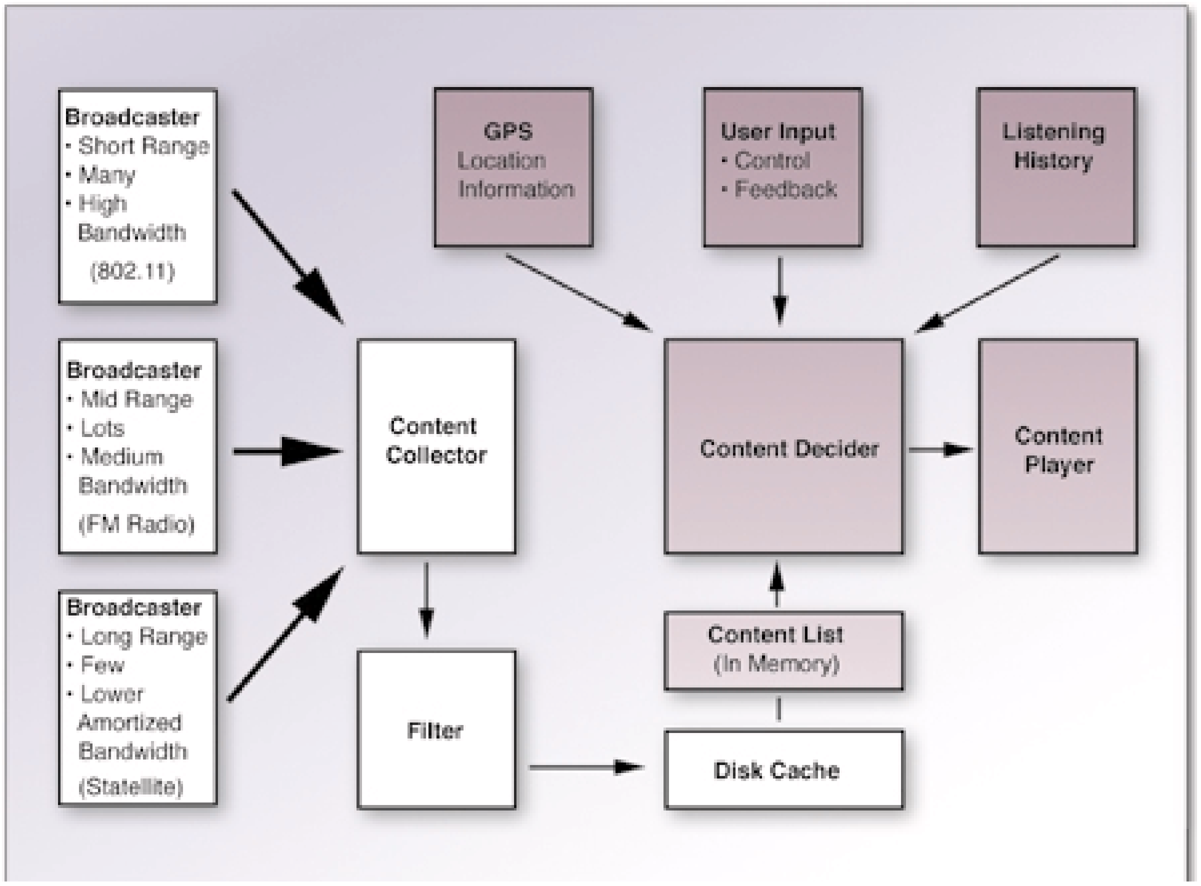

Figure 3.

The system Architecture

This is the architecture of Personal Radio. Content is broadcast from a variety of sources. Each source has a different application based on two attributes: range, and bandwidth. Long Range devices, such as satellites, would be used to broadcast information that would be useful to all devices, national news, advertising, music etc. Though it has high bandwidth, the fact that it has to push so much content effectively lowers its amortized bandwidth. To fill in all of the local information there might be shorter range, medium bandwidth broadcasters, such as local radio stations. These would broadcast local news, traffic, weather and advertising. Finally, there might be a short range really high bandwidth source, such as 802.11, at a gas station or in your home, where you could connect and quickly get all the recent content. This multi-level broadcast approach would allow many different sources to be providing content to users. Some content doesn't change very much. For example, a music or ad piece of content never needs to be updated. Therefore, these content types might never need to be broadcast over the lower bandwidth broadcasters. Devices could load many of these content pieces while near a high bandwidth connection and then not need to load any more until it once again comes within range of a high bandwidth connection. The correct models for the relationship between all of the different broadcasters must be explored more fully.

Regardless of how the content is broadcast, it all comes into the device through the content collector. The content collector just listens and buffers all incoming content. Each content packet is made up of two pieces: The info file and the content file. By receiving the info file first, the content collector is able to determine if it should listen and store the content that is being broadcast. Since this works almost like an index, such a system could potentially employ a strategy such as described in [4] to go to sleep, or go listen somewhere else for the duration of the next file to be broadcast, knowing that we don't care about that particular file.

After a file is stored in the memory, it is now up to the system to decide if and when to play it. The content decider uses 3 pieces to determine is fact.

- Location Information - GPS
- Time Interval - The time since this element was last played.
- Popularity - How popular a content piece is

The content decider takes in these inputs and produces a song as an output. It passes this on to the content player, and the content player then plays the content. While the broadcasting and caching pieces of the system are essential to the overall success of the system, we decided not to implement them. Since we would not be able to explore the actual broadcasting of the content, we decided that rather than simulate a weaker version of the broadcasting, to simply leave it out. The shaded regions of the diagram show the sections that we focused on to implement.

9 Content Decider Analysis

First we must get a high level picture of how the system organizes the information about the various content items. When the system starts up, it loads into memory all of the info files. These are sorted into separate lists by category. There is one list for each category. Each category is then sorted by weight. Since the popularity of each content item will not change very frequently, this is the dominating factor in the ordering of the content items. In the example below the weights are listed for the advertisements. To choose which content item to play, a random number between 0 and the total weight of that row is chosen.

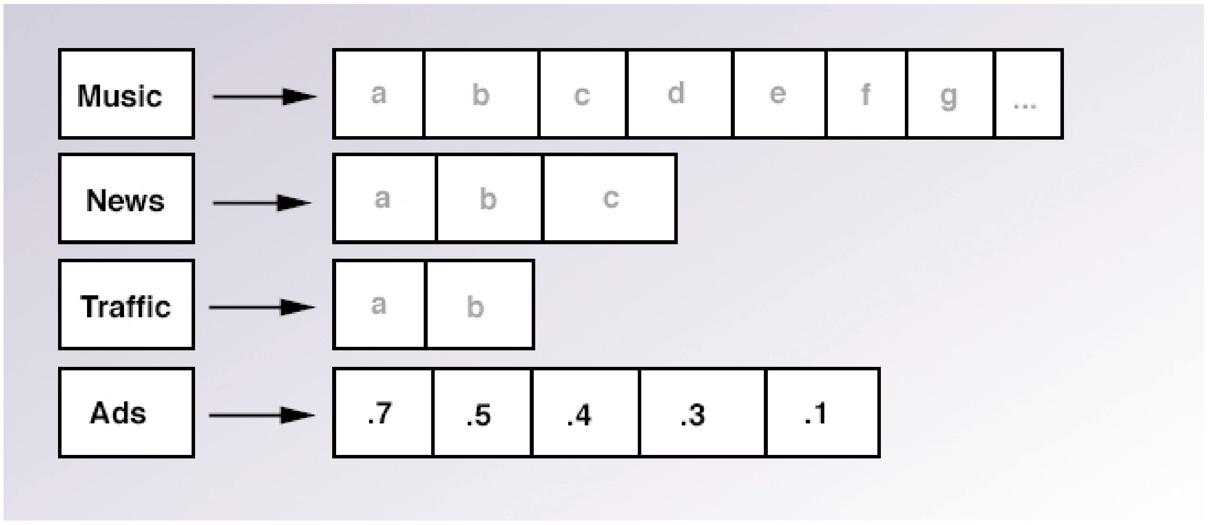

Figure 4.

The internal structure of the content items

For example, the random number 1.2 is chosen. The system then moves down the list summing the weights of the various content items. When the sum gets to be greater than the random number, that is the content item that will be played. The weight of that content item will then drop because it has been played recently. Because of the ordering of the list, the most popular items will have a tendency to be played more frequently than the less popular items at the end of the list. Rather than re-sorting the list, we leave all the items where they are because over time that content item will creep back up to the same location it was in before it was played and the list will be sorted once again. As users rate the content items, the list may become unsorted. Periodically, the list will need to be resorted as items get rated.

So the important question becomes, how do we compute the weight of each item? We compute the three contributions based on the factors outlined above: Location, Time and Popularity. We then use those in the overall formula to compute the weight of a single item. Each weight is a value between 0 and 1. Due to the method described above to choose the content to be played next, those items with a higher weight value are more likely to be played.

9.1 Overall Weight Computation

We compute the Popularity contribution, the Time contribution and the Location contribution. Each is then multiplied by a proportionality constant (Cp, Ct, and Cl respectively). These constants allow the system to adjust how much each factor affects the weight of a particular content item. They can each be independently set but their values range from 0 to 1.

$$\text{Weight} = \frac{(Cp*Popularity + Ct*Time + Cl * Location)}{(Cp + Ct + Cl)}$$

`Cp, Ct, Cl allow for relative weights amongst factors`

Example: Popularity is high (1), Time is high (1) and Location is average (.5). If the system determines that Time and Location plus a little bit of Popularity should define this particular weight computation, then constants Cp = .2, ,Ct = 1, and Cl = .9 might be defined. Thus the weight would be computed to be:

$$\text{Weight} = .79 = (.2\ (1) + 1\ (1) + ,9\ (.5))\ /\ (.2 + 1 + .9\ )$$

Since the Time and Weight portions were high, the result is driven mostly by their values. If you set Cp, Ct and Cl all to the same value, then the weight is just the average of the Popularity, Time and Location contributions.

This strategy is a good way to choose the weight because it allows for flexibility. For each type of content, different pieces are important. In the situation where a content item is not location based, Cl can be set to 0 and the weight will be computed based on the other two contributions.

9.2 Popularity Contribution

There are two values that go into the popularity contribution. Each content item has an aggregated popularity (Global Rating), either defined by a chart (such as the Billboard chart) or by compiling users' personal ratings and taking an average. This piece is important to rank new unheard content, and to help decrease the popularity as the content item goes out of style. The second piece that is important is the user's personal rating (Personal Rating). To compute the Popularity contribution we use the following formula:

```
Popularity = C (Global Rating) + (1-C) (Personal Rating)
```

The C value determines how fast the system switched from the global rating to the personal rating. One way to define the C value is as follows:

```
C = 1 / Ratings*
Ratings is how many times we have rated this item.
```

With this definition, as we rate the song more times, the amount that we will be using the Global Rating will approach 0. Therefore, the more that we rate this item, the more that our Rating will play a role in the overall value. This is a nice simplistic way to determine C, but we would like the number of times we have heard this content item (Impressions) to play a role in the computation of C. An alternate definition of C is as follows:

```
if( Impressions < Learning Constant ) > C = MIN_C - I/L
if( Impressions ≥ Learning Constant ) > C = MIN_C
Learning Constant how soon the system uses Personal
Rating
MIN_C bounds the minimum contribution of Global Rating
```

While we have heard the content item a small number of times, the amount we use the personal rating is very small. When we have heard the content item more times the amount that we will use the Global Rating will approach the Minimum defined value.

* when ratings is 0, C is 1

9.3 Time Contribution

The time contribution controls how quickly a content item returns to a normal weight after it has been played. The first piece that must be computed is the Repeat Time. This value will be used to keep songs from being play too frequently. This value is based upon the traditional radio station playing technique. Each song at WDCR has a certain number of plays per week. For example, the most popular songs get played about 27 times per week. The least popular songs will get played once every two weeks. So the repeat time for these would about 4 hours and two weeks respectively.

It would be nice if the repeat time could be computed as a function of the popularity rather than having to define some number of plays per week for each song. Such a function would appear as follows:

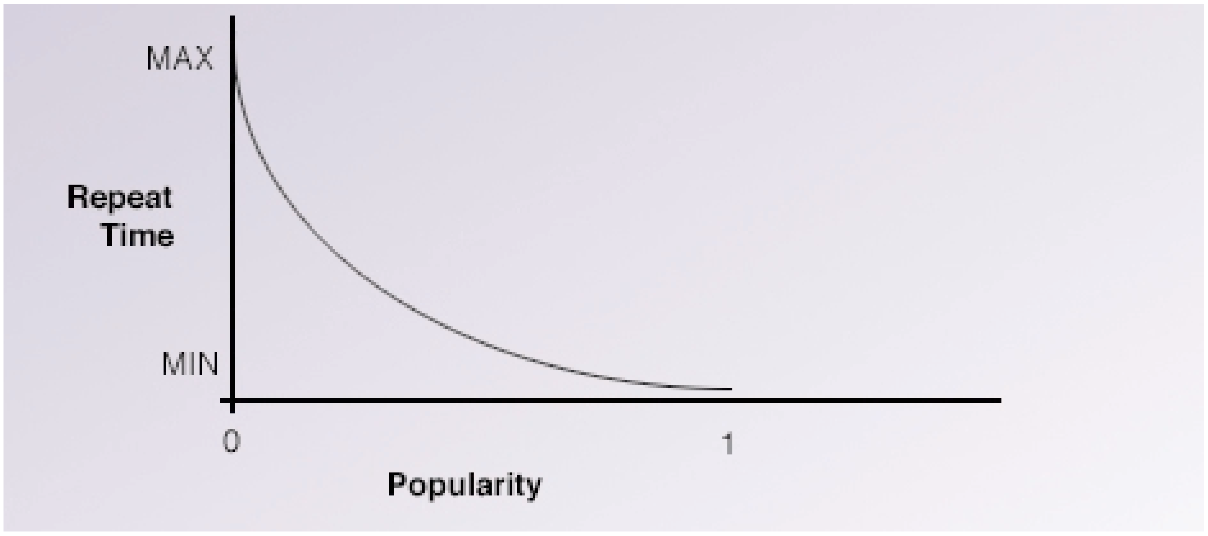

Figure 5.

Repeat Time Graph

As the popularity increases, the Repeat Time decreases to some minimum. As the popularity decreases, the Repeat Time rises to some maximum.

After we have computed the repeat time, we use this to determine the Time Contribution. We want the content to not be played for a length of time equal to the repeat time. Then over the next repeat time block the Time Contribution rises back to the normal.

This graph shows this relationship:

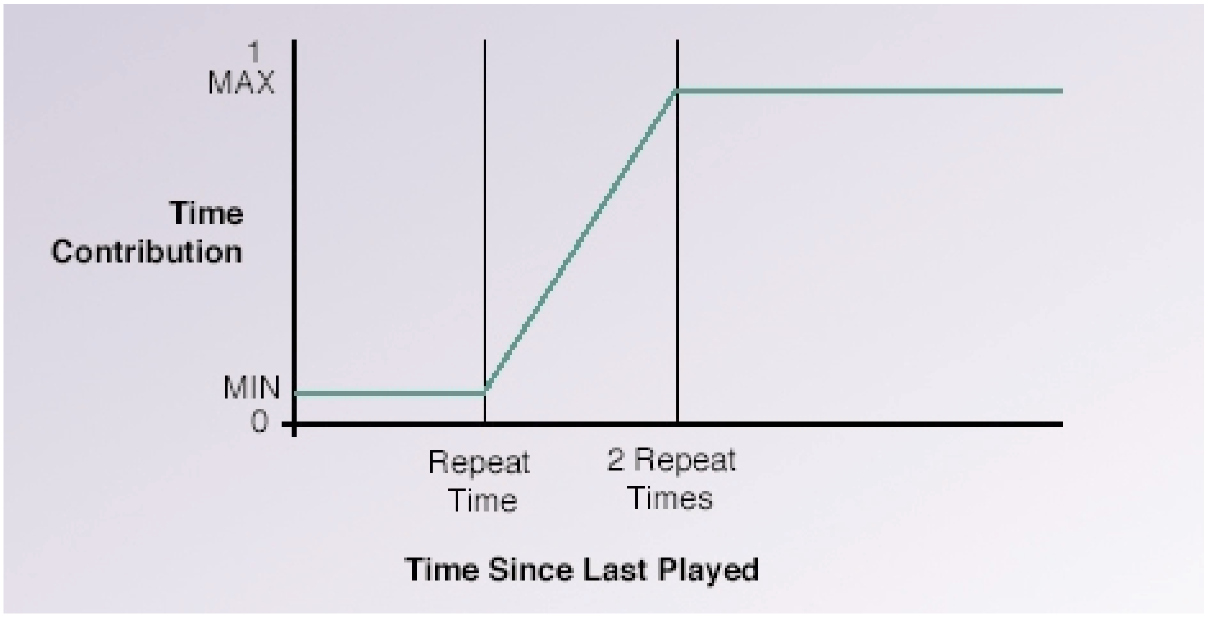

Figure 6.

Time Contribution Graph

Once again, the Maximum defines how much of a role the Time Contribution can play in the overall weight and the Minimum defines the minimum Contribution time can make. The Minimum must be greater than 0, or potentially no content could be played if all the content had already been played.

9.4 Location Contribution

The location contribution is the most interesting of the three contributions. We want to use all of the information available to us from the GPS device: Location, Heading and Speed. Additionally, we want to adjust which items are “closest” to us based not on their distance from us but on their distance from us and where we will soon be. The most obvious solution to this problem is to use a repeating group of ellipses. The first focus point of the ellipse would be the location of the device. The second focus point would be computed in the direction of our heading. The distance separating the two points might be the (speed * average play time). This way, (assuming inertia) we would be at the second focus point by the time the content ended.

This image shows two scenarios and demonstrates how the ellipse changes based on speed.

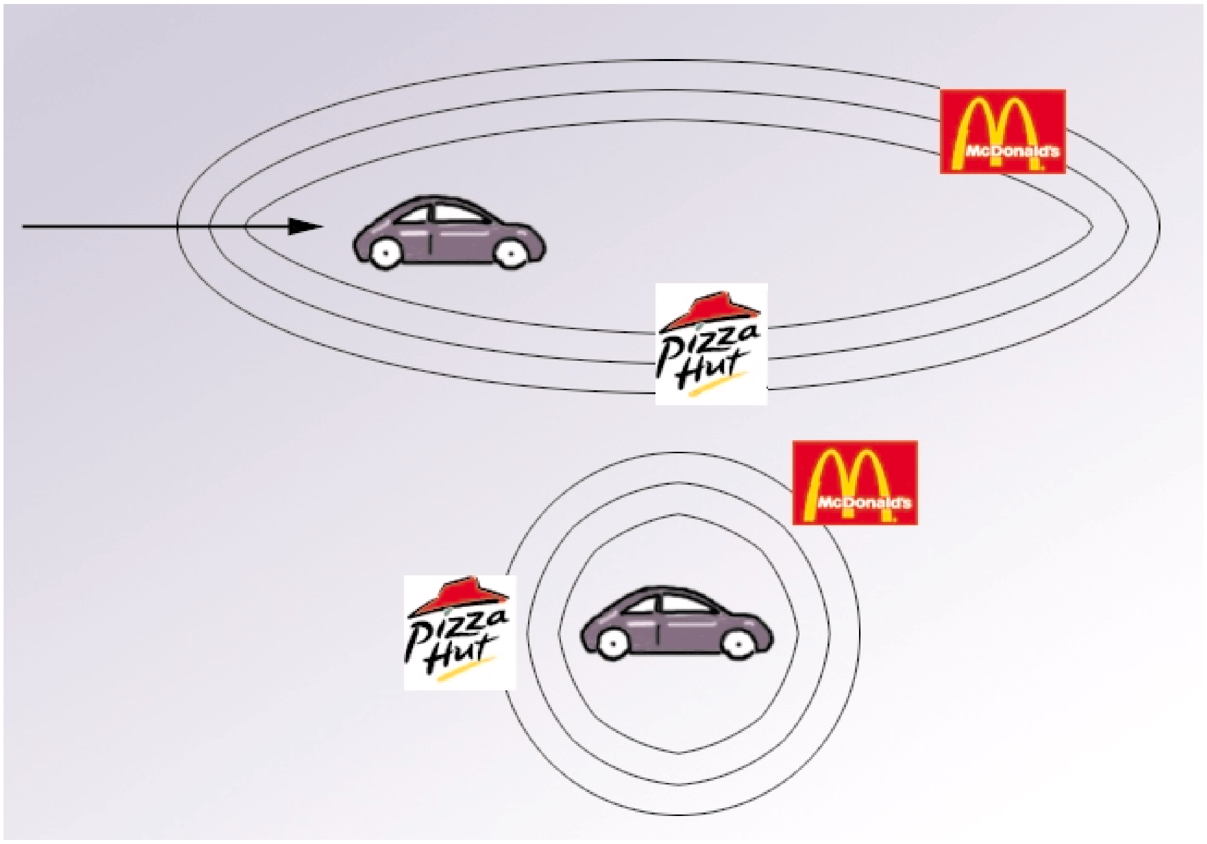

Figure 7.

Relative Location Graph

In the top graphic, because the Pizza Hut is on the outer ellipse and the McDonalds is on the inner ellipse, the McDonalds would be "closer" even though in driving distance it might be farther away.

Because every point on the perimeter of an ellipse is the same distance away from the two foci we can compute the sum of the distance from each of the foci as the "distance" from our current path. Using this distance, we can then use a graph similar to the one we used to compute the repeat time to compute the Location Contribution.

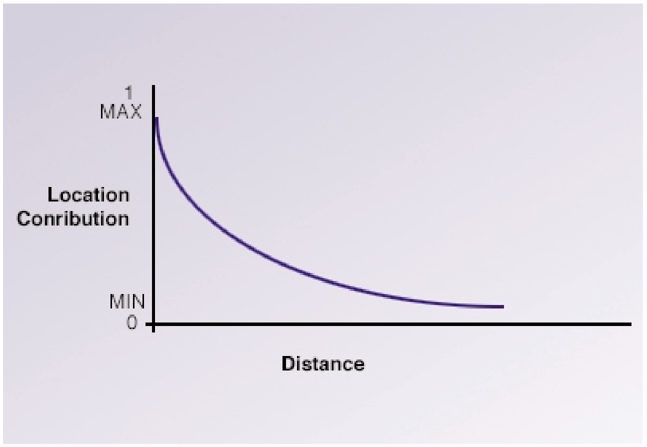

Figure 8.

Location Contribution Graph

The diagram to the right shows how the distance computation is done.

```
A = My Location

B = Content Location

C = Projected Location
```

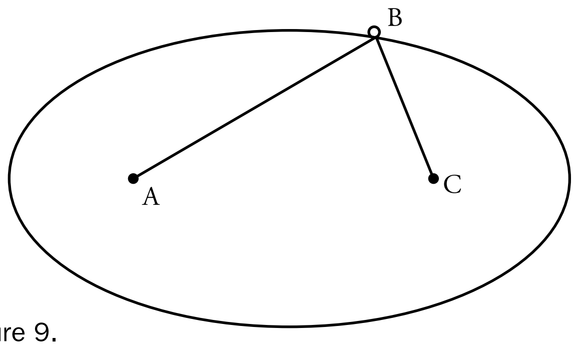

Figure 9.

Ellipse diagram.

The formula to compute the distance is:

```
Distance = Distance ( a -> c)  ) + Distance ( c -> b )
```

This system employs a very simple learning algorithm. When new content arrives, the personal rating is set to the global rating. When you rate a content piece, your personal rating is averaged with the global rating. The formula that does this averaging is:

$$\text{Personal} = \frac{Cg\ (\text{Global}) + Cp\ (\text{Personal}) + Cn\ (\text{New})}{Cg + Cp + Cn}$$

```
Personal = Personal Rating
Global = Global Rating
New = User Entered Rating
Cg, Cp, Cn how much each factor influences result
```

By using all of the information available, we are able to remove outlying values and respond accurately to trends in the user's preferences. If we set Cg, Cp and Cn to 1:

Time	Global	Personal	New	Result
0	.70	-	-	.70
1	.70	.70	.17	.52
2	.70	.52	.50	.57
3	.70	.57	.83	.70
4	.70	.70	1.0	.80
5	.70	.80	1.0	.83
6	.70	.83	.33	.62
7	.10	.62	.17	.29

In this example, the global rating stays the same and the user first grows to like the content, and then begins to dislike the content. Finally, the global rating changes because lots of people don't like this content; this causes the rating to change drastically.

In order to more accurately tune this learning algorithm, user studies should be conducted using different values of Cg, Cp, and Cn. User feedback, in addition to statistical analysis of how many times users needed to rate various content pieces would help discover the proper values for Cg, Cp and Cn.

Figure 4. shows the Interface developed for this project. The buttons along the left allow the user to switch between various content categories. The buttons along the right allow the user to rate the current content. There are two modes for the system. If the system is running, it will continually choose content to play. If they system is stopped, then the user can choose to play an

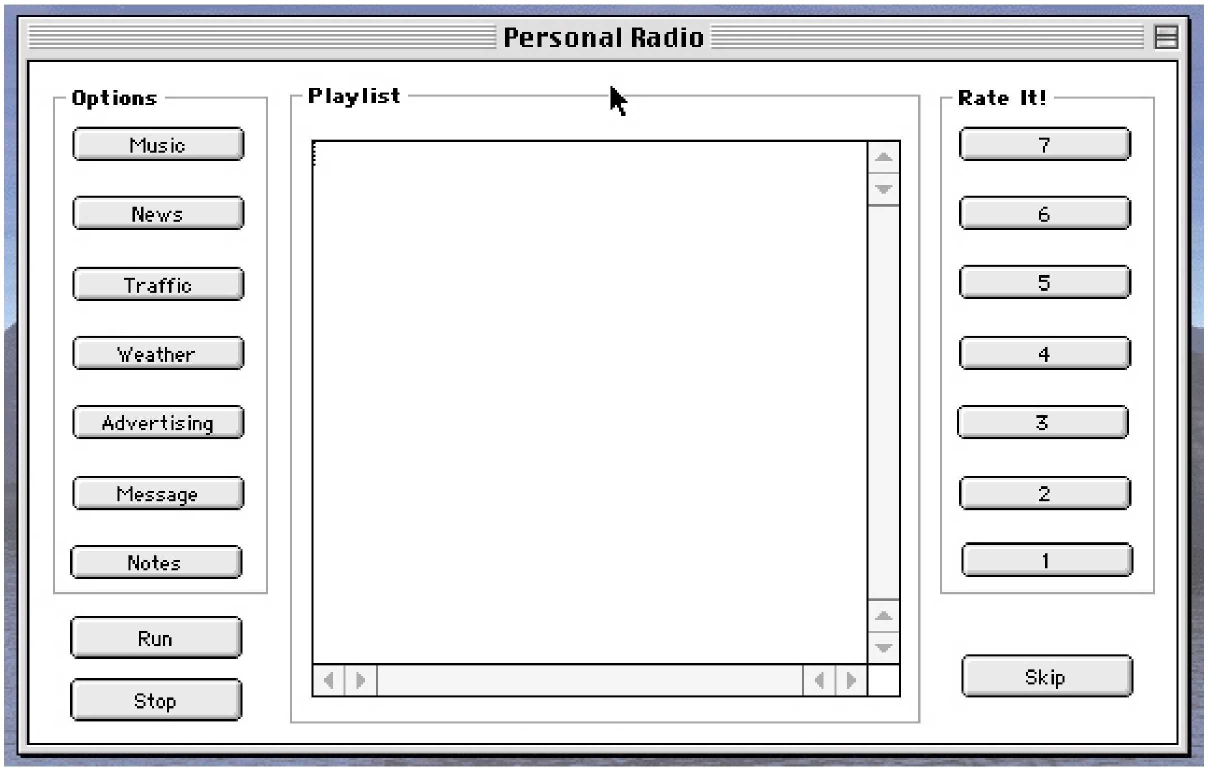

Figure 10.

The Personal Radio Interface

individual content piece by pushing one of the category buttons on the left side. Finally, the user may end the current content piece and move on to the next content piece by pushing the skip button. This interface provides the basic operations necessary to use this system.

The following pieces of the system are implemented:

- GUI interface - allows for basic input and output.
- GPS input - the system reads GPS coordinates from a GPS Device
- Content Decision - the system will choose items to play based on the algorithms outlined in Section 9.
- Basic Learning - The system learns user input using the algorithm described in Section 10.
- Output - The system produces the apple scripts necessary to control the player.
- Content Circulation - The system automatically circulates between the various content types.
- User Preferences - the system loads the user preferences from a file.

12 Future Work

The work completed so far in this project is but the tip of the iceberg. Throughout the development of this system, we discussed many ideas that might be interesting to pursue. I have tried to document them, with the ideas that we had about them so that they won't be lost.

- **Broadcasting** - *The specific method for broadcasting, and for choosing what to broadcast must be determined.* Every channel that could be used for broadcasting, including satellite, digital radio, or 802.11 would choose what to broadcast based on a few factors — the time sensitivity, the global rating, how long since the last broadcast, and the number of requests received for this content. However the type of content located on each channel would be quite different. Time sensitive, universally interesting content would be broadcast from the wide satellite channel. Local content, such as news, ads and messaging would be broadcast using the medium digital radio channel. Static content, such a global ads and music would be available at the short range. Since this content is not going to change very much, we can acquire a lot of it at once and keep it cached. Of course, as user preferences change, we might have to tune in to a wide area server that broadcast this static content to supplement what we already had.
- **User Requests** - *The methods whereby users can request specific content need to be decided upon.* Perhaps the user could use a more expensive medium (such as a cell-phone) to create a point-to-point connection with the server. This way they could request specific content. The content could then be returned in one of two ways, either over the point-to-point connection or over the satellite broadcast. The point-to-point return method would be simpler and quicker, but also more costly to the user. The broadcast method would probably be preferable, because if one user is requesting specific content, probably another user wants that content too. By broadcasting, we service both requests without requiring a request from the second desiring user.

- **Content Receipt** - *The system must be designed to check for new content.* Due to the lack of broadcasting, the system does not dynamically add new content into the play lists. This functionality would be necessary for the final implementation.
- **Caching** - *The system must be extended to deal with limited storage space.* Currently, the system does not deal with the issues of space or memory. In order for the system to go live, it would have to deal with these issues.
- **Proof** - *We must prove that the algorithms described within this paper are good, and if they are not, we must improve upon them.* Though I have outlined many algorithms that make logical sense, I have no proof as to how well they work in reality. Throughout this paper I have suggested ways to examine the validity of these algorithms. These and perhaps other methods need to be used to determine if the algorithms work as they are intended.
- **Content Playing** - *The interface between the system and the player must be fixed.* Though much of the code is in place to play the content, difficulties in obscure system level calls prevented the system from actually playing the content. The plan was to use AppleScript to control SoundAppPPC. The current implementation simply create the AppleScript. The code necessary to execute this AppleScript require further debugging.
- **Device-ification** - *The system needs to be developed in handheld hardware*. In order for this system to be truly successful, the device need to be portable so as to allow anywhere, anytime access. The device could have a smaller cache on board but when you hook into the stereo, or the car, it would then be able to use a larger storage device.
- **Voice Interaction** - *The system should be voice controlled.*

12 Conclusions

Personal Radio is a logical next step for the digital broadcasting world. In this project, I have considered many of the pieces necessary to build this system. My implementation proves that such a system is not only possible, but realizable. Though there are many more questions to be answered before Personal Radio is finished, this project has begun to explore what might be.

I hope to be able to continue to improve upon the system that I have developed, perform some of the many studies I suggested, and implement the additional pieces necessary to make Personal Radio a reality.

References

[1] USA Digital Radio ('USADR'). May 31, 2000. <http://www.usadr.com/technology.html>.

[2] USA Digital Radio ('USADR'). May 31, 2000. <http://www.nab.org/SciTech/Dab/ibocsubmission.asp>

[3] Stewart, James. *Calculus*. Pacific Grove, CA: Brooks/Cole Publishing Company, 1985.

[4] T. Imielinski and S. Viswnanthan. "Wireless Publishing: Issues and Solutions." In Tomasz Imielinski and Henry F. Korth, editors, *Mobile Computing*, chapter 11, pages 299-329. Kluwer Academic Publishers, 1996.

[5] David Gifford et. al., "The Application of Digital Broadcast Communications to Large Scale Information Systems," *IEEE Journal on selected areas in communications*, Vol 3, No. 3, pages 457-467, May 1985.

[6] Hennessy, John and Patterson, David. *Computer Architecture a Quantitative Approach.* San Francisco, CA: Morgan Kaufmann Publishers, Inc., 1996.

[7] Young, Neal. *On-line file caching*. Dartmouth College Department of Computer Science Technical Report PCS-TR97-320.

[8] Tanenbaum, Andrew. *Modern Operating Systems*. Upper Saddle River, NJ: Prentice Hall, 1992.

[9] T. Cormen, C. Leiserson, and R. Rivest. *Introduction to Algorithms*. Cambridge: The MIT Press, 1997.

[10] Brown, Mark. *The Analysis of a Practical and Nearly Optimal Priority Queue*. New York, NY: Garland Publishing, 1980.

The source code for my implementation is available at http://www.cs.dartmouth.edu/~jca/radio.