MediaConnector: A Gestalt Media Sharing System

By

Surjit Savji Patel

B.S.c (Hons) Computer Science
Manchester University
Manchester, United Kingdom
1992

Submitted to the Program in Media Arts and Sciences,
School of Architecture and Planning,
In partial fulfillment of the requirements for the degree of

Master of Science in Media Arts and Sciences
at the
Massachusetts Institute of Technology

September, 2002

***Author*- Surjit Savji Patel**
Program in Media Arts and Sciences
Aug 20, 2002

***Certified by* – V. Michael Bove Jr.**
Principal Research Scientist, Object Based Media Group
Thesis Advisor

***Accepted by* – Andrew B. Lippman**
Chairperson
Departmental Committee on Graduate Studies
Program in Media Arts and Sciences

MediaConnector: A Gestalt Media Sharing System

By

Surjit Savji Patel

Submitted to the Program in Media Arts and Sciences,
School of Architecture and Planning,
on August 20th, 2002
In partial fulfillment of the requirements for the degree of

Master of Science in Media Arts and Sciences
at the
Massachusetts Institute of Technology

Abstract

Our desire to have common experiences with other people leads us to share media such as photographs and music. With computer networks as the media delivery system we create new opportunities for recording media utilization and ownership. Using traditional and responsive media we explore systems that enable enhanced shared experiences through modeling groups of users. A series of prototypes built with an experimental framework, MediaConnector, help us document observations and behaviors of participants.

MediaConnector is a peer-to-peer media-sharing framework that allows people to develop new peer-to-peer media sharing application. Through engendering each node with its own historical audit trail we can take a crawler approach and dynamically build group profiles and perform trend analysis.

Theoretical and practical work that leads to the final framework design is discussed. In particular experiments with GPS enabled cameras that explore metadata inter-relationships, networked tables to share photos and two construction tests of the MediaConnector framework in dynamic group level personalization of television and audio content.

It is intended that a "constructionist" approach together with new behavioral analysis will foster new and novel sharing applications to emerge. MediaConnector is evaluated by its ability to support the above approach in a community of users.

Thesis Supervisor: **V Michael Bove Jr.**
Title: Principal Research Scientist

This work was supported by the Digital Life Consortium, MIT Media Lab and by Kodak Research and Development

MediaConnector: A Gestalt media sharing system

By

Surjit Savji Patel

Thesis Committee

Thesis Advisor – V Michael Bove Jr.
Principal Research Scientist, Object Based Media Group
MIT Program in Media Arts and Sciences

Thesis Reader – Ted Selker
Associate Professor,
Context Aware Computing Group,
MIT Program in Media Arts and Sciences

Thesis Reader – Glorianna Davenport
Principal Research Associate,
Interactive Cinema Group,
MIT Program in Media Arts and Sciences

Acknowledgements

To my Mother and my sisters, Ela and Malti. Three incredibly strong women who kept me from going over the edge when I clearly didn't even know it was there. I dedicate this work to you.

To Suresh, Anil, Arvind and the rest of my immediate family for the support they have shown over the years in my over paced lifestyle and interests. I would not be where I am today without your support and encouragement. You are such powerful role models. And so talented.

I would like to thank my wonderful supervisor, friend and mentor Mike Bove Jr. On the shoulders of giants I would stand and many a times fall off. Your grounded guidance taught me a lot. I will always be grateful for my time at the lab.

To my readers Glorianna, Brian and Ted. Thank you for guiding me through this wonderful right of passage. Your wisdom and insight will always be with me.

Many thanks to David Allen and Tim Hartley. You two will never understand how much of an impact you had on my life when you invited me to join the BBC. You forever changed the direction of my journey and gave me a love for the craft of television and digital media. And yes, I still want to do that job very very badly ☺

Thanks to my friends at the lab Jim McBride and Edison Thomaz. We are legends in our own lunchtime. Thanks for all the advice and support when code and the "system" would invariably never work.

Thanks to Mike McGeachie, your dry wit and consumption of beer kept me sane. You are a good friend and an ally.

Thanks to Stuart Cannon, Chris McDonald and Phillipa Beswick and the rest of the ex-pat crew. It was great to share those moments with you as only ex-pats can.

Kent Whalberg, Chap Hanford and Matt Basham. How on earth could I ever forget you. One day I will earn enough money to be hypnotized. That's how. ☺ Friends for life and you know it.

To Gulliver Smithers, Keith Robinson, Paul Jackman and Huw Williams. Gin and tonics on the veranda then? Thanks for being proud of me and pushing me along. You have seen me at my best and worst.

Neal Cooper and Stephen Roberts. My oldest friends. Nuff said ☺ You saucy pachyderm hoover musicians you.

Karen Werner for all the support and trips to Bread and circus. Your are a pint sized blockbuster of a woman!!! I'd be an emaciated mess without you looking out for me.

Last but not least to Steven Davidson, David Mead, Gilian Cape, Sarah Crewdson, Mark Bell and all of my other "Madchester" friends. Please don't be upset that I couldn't mention you here. You know that I derive my energy and passion for life from you. Wherever you are I miss you. A lot.

Index

List of Figures

List of Tables

Chapter 1 – Introduction

"v. ***shared, shar·ing, shares***
v. tr.
-) To divide and parcel out in shares; apportion.
-) To participate in, use, enjoy, or experience jointly or in turns.
-) To relate (a secret or experience, for example) to another or others.
-) To accord a share in (something) to another or others: shared her chocolate bar with a friend"

Fictional scenario, a short hop into our very near future….
"John arrives home after work. The metro had been particularly groggy and heavy this evening with the heat of the summer. He switches on his television and picks up his media mail first before anything else. Ahh. Nothing new. It's the same old stuff re-sent to him from five different people. Today it could be a joke, or an image that one of his friends has doctored into a humorous image or a video that has been annotated by conspiracy theorists that show that J Edgar Hoover was collaborating with aliens from another planet. Hmmm. He wonders. Where do they get the time from….but wait. There is a media mail from his mother…..what could this be? Ah. They are photos of his new niece born only a few days. Mum being the kingpin of the family always making sure that things get sent out. He decides to annotate it with his words and send it on to his girlfriend. John sees that one of his friends is watching a music tv program. It's been while since they hung out at a gig together, so he messages him and says that he will join him. Johns TV set downloads the dynamic tv show and the show starts to re-render. It gets his listening profile and media profile from his MP3 player, his DVD, his PVR and the Internet browser. The show reports back to his friends instance and the two programs collaborate and intelligently re-cut a common show to make hours worth of viewing

that they both will enjoy. Minutes later some other old cohorts have joined in and their instances of the show conspire. And its almost like the old days when they were at college, flurrying messages crossing the networks, except they aren't staring at each other over pints of beer. They now live time zones apart around the world, but for one moment, everybody knew that everybody was there. And you know what? It meant something that the people he cared about were sharing this moment with him. He knew they were there."

This thesis documents the preliminary research and subsequent implementation of MediaConnector. MediaConnector is a media-sharing framework that proposes that a significant portion of the richness of media consumption comes from shared experience. It explores the notion that the ultimate value of the sharing lies not in any one facet of the process but that it is the whole (the media, the networks, the people and the services) that leads it to be greater than the sum of the individual parts. Also could a constructionist approach help create more engaging experiences for the community using such a framework to build custom applications?

Let's start this introduction by looking briefly at the basic materials.

1.1 Digital Media and Computer mediation

Traditional analogue media forms such as photographs, video and audio are steadily moving to digital forms for capture, storage and transmission. The digitization of media has some inherent advantages – intelligent mediation by computers and loss less replication. Computers lend themselves to the arduous task of searching methodically. Using applied algorithms this manifests itself as useful searching, collation and storage services for the user.

The user is then able to use these services to shape their consumption and experience of the media.

The majority of digital media involves the media being stored and catalogued on individual computers by conceptually simple database systems. These systems mirroring physical paper / celluloid systems provide two advantages. Their concepts are easily understood, because they mirrored, physical processes - store this here, get this for me from there - and because they were faster and convenient than their real world counterparts. For example it would take a human operative the best part of a month to go through an archive of photographs numbering in the hundreds of thousands and find the top 100 that best match a certain criteria whereas a network of modern computers could grind through that number in minutes. However the human still has one advantage. They can understand emotion and expression within an image or sequence of images

That leads on to the attributes that are used and required by the computer when searching – metadata. Metadata is, as the name implies, information about information. In the case of photographs it may be the time and place the photo is taken, by whom it is taken, subject matter and the contents of the photograph. Metadata is divided into three sources for the most.

1) Explicitly input – entered by human intervention
2) Secondary artifacts – Garnering contextual information at time of creation, consumption or from the analysis of the content itself (image / texture processing) e.g. photo taken at GPS co-ordinates 41 54 N, 12.27 E on date April 15th 2001, viewed at 42.216 N, 71.051W on April 16th 2001.

3) Chained / Inferred explicitly from other data / metadata sources. Taking the above example we relate the numbers to GPS co-ordinates and date information to diary events and geographical locations. We derive that the photo had been taken in Vatican Square on Easter Sunday 2001 and viewed in Media Lab on following Monday. Here we are effectively chaining metadata between various sources that would have a common reference to work from.

The next logical step in this is for the computer to use metadata to enhance the consumption of the media.

1.2 Networks and Communities

Participation in the Internet by people of all abilities has proven popular and successful. There is a huge body of research and writing about the subject and to even basically summarize it here would be out of the realms of this thesis. For now lets consider certain popular phenomena that are relevant to our discussion that are occurring on networks:

1) **Person-to-Person.** The popularity of messaging on the Internet – email is by far and away the most popular activity on the Internet. [37]. This implies that people want to talk to each other. With the addition of mime attachments email is increasingly used to transmit media and has led to certain peer-to-peer email chain phenomena. Richard Dawkins dubbed the idea being transmitted a "meme" and work by Rick Borovoy has looked towards mapping this and understanding the process behind the propagation [39]. It is rare to find individuals using email who have not encountered an image, MP3 or joke mailed on to them by a friend or work colleague.

The basic overview is this - An individual creates an email with a piece of media that they have authored (text, photo, animation or video) and passes it onto his community of friends. In turn these friends pass it onto other circles of friends who aren't members of the original mailing list. Supporting the "Six degrees of separation" theory, the email soon spreads, creating a shared experience People want to share ideas and experiences with other people. Recent examples such as "All your base are belong to us" phenomena and the "Nike Sweatshop" email media explosion [38][40][41]

Figure 1.1 The "All your base" phenomena and the focus of Jonah Peretti's "sweatshop meme".

2) **Micro Broadcasting** - The rise of homepages, user publishing, community sites, blogging, Shoutcasting, AboveStream.com. These are basic examples of people's desire to broadcast their material. They want to reach an audience, members of which are unlikely to know them personally. People have something to say and want to be listened to by more than one person. This "micro broadcasting" is an exchange that is characterized as one to many and unidirectional.

3) **Social Groups** - The rise and growth of communities – newsgroups, BBS's, online gaming – implies that people who have something to do or something to say would like to engage with people who share similar interests / experiences. These groups that are created for similarly minded people to engage in many to many relationships with the constituents themselves being the drivers of the relationships and identities in this sphere. Newsgroups are the most basic, oldest and most successful examples of these. Anyone can start one and anyone can contribute but only those interested would normally be attracted. The exchange here characterized by exchanges that are many to many and bi-directional.

4) **Social Groupware** - The rise of community applications – Instant Messaging, Napster, E-donkey. These applications are usually built around a single function, most popularly messaging and media sharing. The most famous of these would be Instant Messaging (IM) and Napster. IM allows real time chat rather like Internet IRC but in a more controllable manner and Napster will find and exchange media at your request. Empirical evidence suggests social traits can be witnessed here as in other web based social phenomena such as identity and grouping.

Answers as to why some of the above may have flourished (explored in Ch. 3), may be found in the idea that constructionist behavior is instrumental in community building. These communities have often created and extended themselves with user authored content and services, providing a center and sense of support to a community. To quote the author Howard Rheingold from a talk he gave at the British Broadcasting Corporation in June 1999 "The greatest value of virtual community remains in its self-organizational

aspects. Any group of Alzheimer's caregivers, breast cancer patients, parents of learning disabled children, scholars, horse breeders -- any affinity group that has a need or desire to communicate -- can start a Listserv, a chat room, a BBS forum. Using Internet social tools is a literacy, not a commodity."

In short, people need to communicate with other people and they can communicate more effectively when they are able to construct within an environment they control. [49]

1.3 MediaConnector – The Gestalt proposition for media sharing

The preceding two areas lead to a suggestion that there are certain attributes to successful network based computer mediated media sharing.

It is clear that there is no *one* necessary ingredient for network media sharing to be a total success. The mixture of the connectivity, the ability to author, the audience, motivations, the recommendations and other artifacts around the phenomena creates an environment that is conducive to sharing and community forming. The end result is that each of the components on their own would not be as useful as the whole created by the sum of all the parts. Borrowing from the Gestalt philosophy movement of the late 19th and early 20th century Germany [41] – we can say that the online phenomena outlined above are the result of a Gestalt entity, that is the people, concepts and software, where the whole is *more* than the sum of the individual parts.

I asked some questions. Would it be possible to create a set of building blocks that permitted users to create their own constructible and modifiable media sharing applications? Could this constructionist philosophy create better media sharing environments? Could it generate novel new applications and services? Fundamentally, is it possible to create a better, community directed way to share media and associated experiences? What is the true potential for people consuming media with others through networks?

This thesis describes the creation of MediaConnector, a small first step to explore these ideas, and its subsequent evaluation.

1.4 Reading this thesis

This document is made up of the following seven chapters and appendices

Chapter 1 – Introduction
This chapter outlines a rationale for the project

Chapter 2 – Theoretical Foundations
This chapter presents the precursors for my creating MediaConnector in a slightly more depth. It attempts to draw a thread of analysis from my observations

Chapter 3 – PhotoConnector and PhotoTable
This chapter outlines two important projects that contributed towards the project.

Chapter 4- Technology and Design
How, why and with what media connector is built. It presents design and technology choice rationales.

Chapter 5 – MBeat – MediaConnector in action
This chapter outlines the creation of one test application to illustrate MediaConnector in use.

Chapter 6 – Evaluation and results
This chapter presents the evaluation activities and their design.

Chapter 7 – Reflections and future directions

References

Appendices

This thesis formatted for readability and is designed to be read printed single sided and bound. Major illustrations and diagrams are inline.

Chapter 2 Foundations

My investigations of people sharing media has led me to examine literature and ideas in 5 major areas

1) Existing practice – Sharing applications are popular for this. What can we learn from their design?
2) Recommendation systems – When we have more choices than we can deal with how can we pick the best one?
3) Electronic communities and structural analysis – Can the field of social science and numerical analysis yield us insight and better network applications for electronic communities?
4) Constructionism – The ability for people to create and modify their environments has been shown to create engaging and special communities, is this a metaphor to pursue?
5) Online constructionism and communities – Looking at case examples of where this has worked.

Here follows an exploration of each major area.

2.1 Existing practice for Media Sharing applications

2.1.1 Survey

One of the first steps taken was to survey existing practice. I chose to survey existing practice [5] as a way of providing a broad view of common media sharing needs quickly. To do this we looked at 38 sharing applications in the public domain via one of the many central

indexes of file sharing applications available. (http://www.zeropaid.com/). As expected there is a common set of functionality. I will describe the most common features found:

Acquire file / Advertise File – as it implies the first thing that a media sharing application should be able to do is to download a file from a source. Symmetry implies that there has to be a mechanism for the file to be offered or advertised to facilitate this transaction.

Search – Once the number of files moderated reaches more than a few hundred then the system has to offer a computerized search mechanism to help locate the item that is desired otherwise the search task becomes tedious and not compelling. Media sharing environments feature people and resources that are continually appearing and disappearing. Search queries and their results tended to be relatively immediate with the option of having search requests persist if not immediately fulfilled. Also a computerized search is probably the only tractable way to quickly realize which opportunities for resource acquisition are available in such a changing environment.

User Identity selection – many media sharing systems are anonymous where the source of the resource or its consumer are not labeled. These may be both centralized client server systems or they may be peer to peer. At the very least the system requires some sort of network identity usually an IP address for both provider and consumer. However the other complement of media sharing systems allow the user to either identify themselves with a chosen moniker or the system issues one.

Group Formation – Many applications permit the creation of groups. Groups facilitate users with common mutual interest's or issues to

congregate and draw more people through its presence and identity. The action has two functions – practical and social. Socially you are more likely to find engaging people to interact with in a common interest group and typically they are more likely to own or have access to the content that you want. [29] A good example is a special interest newsgroup and a real world analog is that of a jazz bar, where members of the public who are jazz fans can congregate knowing that they will find other similar fans and jazz music.

Person-to-Person Messaging – To facilitate closer interaction between the members of a group there is messaging. This is usually short pieces of text.

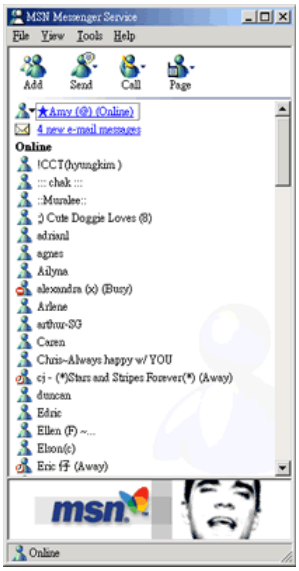


Figure 2.1 Microsoft Network's Instant Messaging client

Chat Rooms – real time text messaging with conversations that can be one to one and private, one to many and public and various permutations. The transactions are normally in real time.

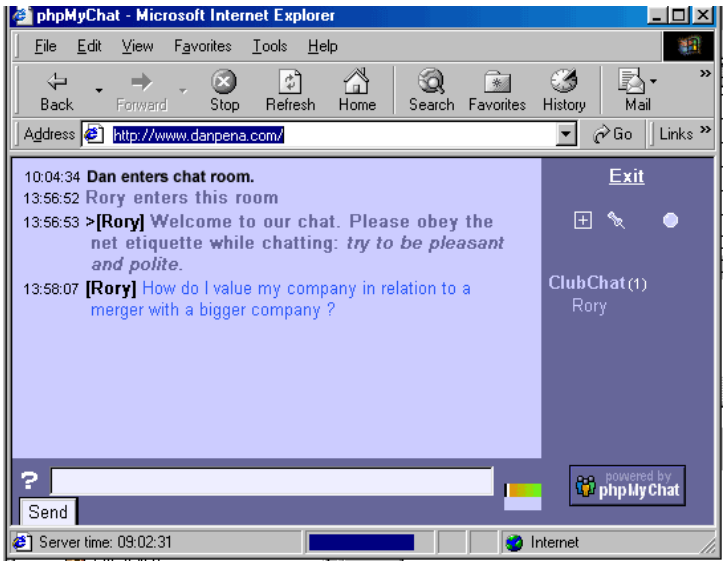


Figure 2.2 A typical community "Chat Room"

Publishing - This aspect allows users to create their own informational material and then to advertise and publish it. This allows people from that moment onwards to access it whenever they want. The audience may be restricted to a closed group. For example many "blogs" are a public way of broadcasting personal experiences in the hope that they relate or provide interest to someone else. [40] Making a posting on a bulletin board recommending something is a form of personal publishing as is publishing the reports from little league softball matches.

Recommendation system - These are analogous to the real world systems in which we find recommendations [29]. We often discover cues for recommendations when we ourselves have no idea of what it is we want. For example, we see a line of people queuing outside a gallery; obviously there is something that is interesting a large number of people and they cannot all be wrong, can they? Or we see a display copy of a magazine that is well thumbed, giving us a indication that it has been looked at and it may be worth us looking at. Similarly we are wary about restaurants or bars that have few people in them. We ask

what is wrong with them, why are people avoiding them? These social navigation cues, rather like a beaten path in a forest, indicate that there have been others before us who have trodden a similar path. [29] Vannevar Bush in his seminal 1939 article "As We May Think", proposes not only a system for Hypertext but that we may one day follow other people's paper trails through an information space.

We may get recommendations from newspapers, critics and friends. These sources usually make recommendations built upon their knowledge of the person or audience, their likely preferences and their likely behavior.
Recommendation systems are provided to help users make satisfying media choices. Actual systems and algorithms are explored later in this chapter. The survey found that recommendation systems were often provided in media sharing applications to offer new choices and to propagate recommendations that the user may be interested in. Typically they are used to provide alternatives that the user would not have normally discovered but would be likely to enjoy. These were offered in systems that attempted to make the file exchange process less goal driven and introduce an element of serendipity to the experience. People often find pleasure in making a new finding that is satisfying and unexpected.

Some of the media sharing systems analyzed were primarily based around recommendation. e.g. Audio Galaxy

2.1.2 Analysis of existing practice

Rounding up our survey of existing practice in media sharing (Napster like) applications there are some conclusions we can draw

1) Typically individual media items are shared but I found little evidence of collections being shared. E.g. I can share a picture but

not a photo album. I can share a video of a concert but not the allied press clippings and photographs I have collected. Technically there seems to be no reason preventing this or it may be that my sample did not show such features.

2) There is no monitoring or recording of transactions between people with people's consent.
3) There is no audit trail to media. We do not know where and when it entered the sharing pool.
4) No way of advertising ones own media preference to a given group except through the collection.
5) Metadata exists in unrelated clusters, for example we can relate a compact disc track name with some certainty to the databases of community maintained information at CDDB. This could further be related to the information at Amazon.com and the user reviews they contain.
6) No user modeling or behavior awareness is used. Related to (2).

Each of these represents an opportunity for further exploration and consideration.

2.2 Recommendation systems

2.2.1 Introduction

In media sharing systems there are typically two main modes of usage. One is goal driven. "I want to get this file". The other is more serendipitous in that sometimes the user will discover new media either through browsing or as the secondary artifact of a goal driven search (i.e. I didn't find what I was looking for but look at what I did find). Often we have to make choices without sufficient personal experience of the alternatives. A system can offer you some potential

choices by using an algorithm to recommend items. When a recommendation algorithm is used, it is usually based on data derived from a combination of the users behavior, other users behaviors, and stated preferences or from the content itself. These systems are specifically called 'Recommendation systems'.

Recommendation systems use metadata associated with the user and the media to make decisions. For example it will use the metadata that describes the genre and artist of a music track and it will use metadata about the preferences of the individual to see if that track would match their preferences (i.e John likes rock, Detroit Rock city by Kiss is a rock track). This implies that metadata is captured or inferred at some point for both the user and the media.
Recommendation systems are useful because they are able to deal with complex search spaces that a human recommendation would not be able to cover, especially for large sets of people and media. The volume of material is so large that any one person would be unable to filter through it in any reasonable amount of time to find suitable material.

In the following text I will only consider types of recommendation system that are particularly relevant to this thesis.

2.2.2 Categorization of algorithms

Collaborative filtering

Collaborative filtering is a popular online and offline method to generate recommendations. The basic idea behind collaborative filtering is the social process of people recommending items to one another. [6] Collaborative Filtering makes recommendations based upon similarities between the interest profile of a user and those of other users. In effect it says, " You may like to do X based on the fact

that you have a profile similar to user B and they have done X and you haven't".

The main idea is to automate the process of "word-of-mouth" by which people recommend products or services to one another. If you need to choose between a variety of options with which you do not have any experience, you will often rely on the opinions of others who do have such experience. However, when there are thousands or millions of options, it becomes practically impossible for an individual to locate reliable experts that can give advice about each of the options. By shifting from an individual to a collective method of recommendation, the problem becomes more manageable.
Drawbacks to the algorithm include

- Needs large amounts of data to be reasonably on target.
- Most systems require explicit input of preferences. This is obstructive and a problem [6]
- The recommendations have no context. e.g. I buy six classical music CD's and one nursery rhyme CD. Another user who buys the same CD's will also get recommended the nursery rhymes CD as a serious recommendation because it doesn't know that I bought that CD as a joke for a friend.
- Profile model is statistically based and has no temporal weighting to indicate "trends and new interests, e.g. "I like ancient Greek tragedies but lately I have been reading a lot of comics."

Behavioral /Contextual Systems

Behavior has been used in Information Retrieval recommendation systems before, using pre-defined model behaviors to compare against and thus suggest solutions [10]. As described later the basis for another

system that uses a behavioral approach is Broadway [11]. This extends the behavioral approach using Case Based Reasoning (CBR) methods.[12] Case based reasoning methods try to identify the early stages of a search or exploration and compare it to resolved / completed cases to find a way to resolves the case being examined.

2.2.3 Survey of significant related projects

This section will briefly outline the salient points about a number of case studies in recommendation systems. I will, as in the preceding section, order them by the basis of the algorithm used. First I will consider Collaborative filtering and the behavior based systems.

Collaborative Filtering

There are numerous projects in this area in particular related to music and media recommendation.

Ringo / Firefly - Of particular interest has been the work done with Ringo [6,7]. Ringo is a music recommendation system that made personalized suggestions. People described their appreciation of a piece of music by rating it. These ratings added together over time to make up the person's profile. Ringo would compare user profiles to match likes and dislikes. Once similar profiles had been identified the system would predict how much the user would like or dislike the music by computing a weighted average of all the ratings given to that album by the other users that have similar taste.

The project launched on the web as an experiment with zero users and no data. Within seven weeks they had an audience of 2100 users and 500 messages a day (September 1994). People initially still wanted to share their opinions when there was no likely gain for their effort. Qualitative results from the test audience showed that Ringo

performed poorly in its early days when data set populations and user numbers were low and that as they both grew larger they became "unnervingly accurate".

Behavior based systems

This area of recommendation systems has its root in collaborative filtering and indeed may be considered a natural subset of it. The projects look at analyzing user behavior exclusively and try and extract useful recommendations based around behavior exclusively. It can be argued that they are essentially collaborative filtering but they differ enough to warrant examination separately.

An example will illustrate this better:

Broadway - This project proposed a new approach, called the Broadway approach, where the system recommends choices to a user that have satisfied other users that have *behaved* similarly. Previous approaches had modeled users using behavior and they had taken the approach of matching user actions to specific behavior models. The Broadway approach differs in that it models behavior using a set of defined observation variables. The system works by creating a time series log of a set of variables that describes the users behavior. The evolution of the variables over time describes the users behavior. The choice of variables is specific to the application field of the algorithm and the time series is clustered into "records" that describe a "session" of activity.

Case based reasoning (CBR) [19] is then used to find a solution to the current problem as it is posed. Broadway further enhances the CBR model by allowing cases with temporal indexes to be dealt with. Another interesting aspect is that query session "records" can be saved and then "completed" by other domain experts. This is an interesting

input that allows the system to provide a mentoring service by trying to spot behaviors that are driving towards one goal. e.g. From the pattern of activity it could be seen that a picture of Harrison Ford is wanted but not from the Star Wars films…a domain expert could "finish" a template that would the be available in such similar situations as a starting point to build a solution.

2.2.4 Analysis of recommendation systems

These systems have a user model that is incomplete or insufficient. More so the problem is that the models they do have do not take into account *changes* in the users behavior and *context*. They are static models. What is needed is a way to continually monitor the users (or groups) behavior for clues to changes in preferences and behavior to provide more satisfying choices. This should be a continual process to provide better context.[58]

Recommendation systems are useful in finding and deciding between media, people and affiliation groups.

2.3 Electronic communities

In this section I will consider the intersection of work between the sociological study of online communities and that of social network analysis. Each of the fields respectively has a large body of work, too large to cover, even in brief. However it is at their intersection and the powerful possibilities that Social Network Analysis (and allied to that graph theory) could bring to media sharing applications.

First we shall briefly look at the origins of social network analysis.

2.3.1 Social Network / Structural Analysis in the Social and Behavioral Sciences

Social Network analysis / Structural analysis had been developed by the social sciences to help form precise formal definitions of the relationships and interactions between people. They saw that this could help to provide an analytical basis to help answer questions about the political economic or social structures that people form.

In Social Network Analysis, relationships of people are represented as directed graphs. A social network is a set of actors (or points, or nodes, or agents) that may have relationships (or edges, or ties) with one another. Networks can have few or many actors, and one or more kinds of relations between pairs of actors. To build a useful understanding of a social network, a complete and rigorous description of a pattern of social relationships is a necessary starting point for analysis. Ideally all of the relationships between each pair of actors in the population is known. This is then represented as an array or a graph.

The actors are usually people and the links between them are relationship descriptors.

From the outset, the network approach to the study of behavior has two guides: (1) it is guided by formal theory organized in mathematical terms, and (2) it is grounded in the systematic analysis of data. First started in the late 1930's, it was not until the 1970s, when modern graph theory experienced progress and powerful computers became readily available, that the study of social networks really began to take off.

The usual representations for a social network are graphs, directed graphs and matrices.

This diagram shows an asymmetric relationship between two people.

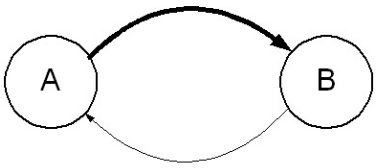


Figure 2.3 Social network of two people displaying an asymmetric relationship

The next figure shows the relationship between three people, A,B and C with AB having a symmetric relationship and AC having an asymmetric relationship. BC have a transitive relationship through A.

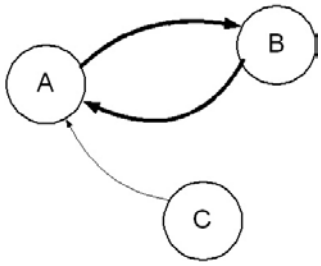


Figure 2.4 An asymmetric relationship between three people

Here are two graphs of commonly occurring social structures. The first diagram shows two social groups or cliques with one common member who is known by everybody in both groups, dubbed “a star”.

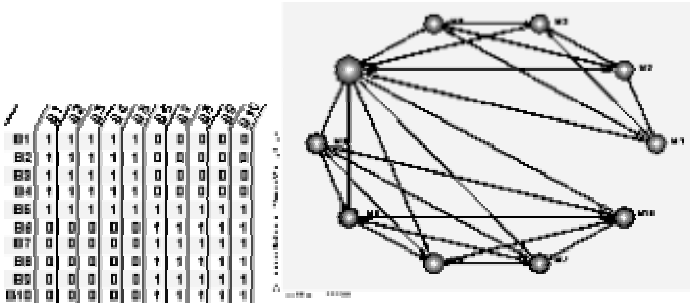


Figure 2.5 Network featuring two cliques and a star

The second one illustrates two cliques. However the two cliques are linked in that one person from one group knows somebody in the other group, thus forming a bridge.

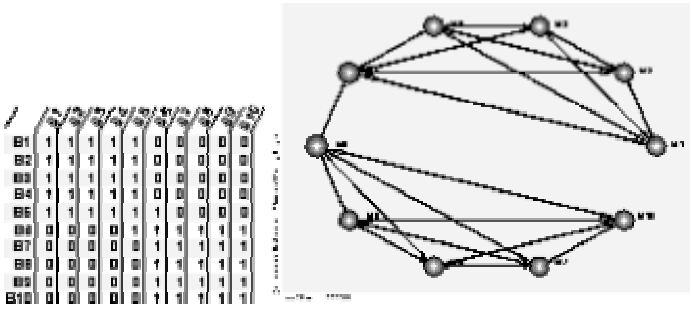


Figure 2.6 Network featuring two cliques and a bridge

[images and data courtesy of Lothar Krempel, Max Planck Institute]

The point is that structural analysis provides a very convenient and understandable representation of a social network that can be manipulated and analyzed by a computer. It would be possible to provide near real time analysis on a large network that it would humanly not possible to do.

2.3.2 Computer Supported Social Networks

Computer supported social networks (CSSN's) are social networks that are primarily mediated (communication) by computers. This is a good next step in the discussion of social networks.

Shared Interest and groups

Why do groups form? Shared experiences are important in social relationships. Lets consider the topic of conversation. If we were to meet a stranger we would often talk about the weather, world politics or some environmental feature that is present, because it is likely that we will have experience of these in common. When we encounter people who are likeminded (politics, interests, hobby, work) we are also likely to talk about the weather etc but also we are likely to talk

more about our mutual interest, as again we are more likely to have experience of those and be interested in talking about them. When talking to friends, we are interested in hearing about people we mutually know and when conversation turns to consider people that we do not know, our interest wanes.

Online groups such as Bulletin Boards / newsgroups, MUD's/MOO's [47], mailing lists and chat rooms provide ideal and easily accessible places for special interest groups to convene and find other members who have common interests. In fact it is possible that the only reason for coming to an electronic group is that it is possible for likeminded people to find each other and share experiences. [47]

Computer Supported Social Networks

When computer networks link people and other actors together, they are computer supported social networks. Computer supported social networks (CSSN's) [48] come in a variety of types such as electronic mail (email), bulletin board systems (BBS's) multi-user dungeons (MUD's), newsgroups and Internet relay chat (IRC). CSSN's tend to provide centrums, social support, information and a sense of belonging.

Opportunity

The opportunity lies in the fact that a methodology exists that allows a social network to be noted and analyzed by computer. [51]. In the case of a media-sharing network this could be by analysis from messaging transactions. Messaging is a communication from 1 to N members and hence defines a relationship.

Actor theory, which is a superset of social network analysis, allows us to further include media and other times like files and emails into the analysis. We can also see that other items in the media sharing food

chain such as the media itself and the relationships between Media and other media, media and people can be built. It is possible that we may find interesting results from this investigation into the "secret lives of media" and find patterns in media association. A visualization of this to the user community may yield self-reflection and present opportunities.

Also spotting trends may become easier and it may be easier to see who or which individuals are trendsetters and strong recommenders and weak recommenders. [52]

Because computer networks often mediate social networks, the social network approach gives important leverage for understanding what goes on in computer-mediated communication: how it affects the structure and functioning of social systems (be they organizations, workgroups or friendship circles) and how social structures affect the way computer-mediated communication is used. [48]

2.4 Constructionist behavior

2.4.1 Definition and Sociological overview

It would be best to start off with a review of the terms *constructionism* and *constructivism*. Recently they have caused me much confusion and may cause the reader some confusion. Both terms have their basis from the fields of epistemology. Constructivism, as defined by Piaget [20] and von Glaserfield [21], is the belief that the individual learns by constructing and reconstructing their model of the world given a social and cultural context. The learning occurs with the students awareness of their cognitive self-organization, i.e. the student must consciously think about trying to derive meaning of what is presented, and through that effort, meaning is constructed through the realization of their own

knowledge structures and relations. Piaget liked to emphasize learning through play as a way that this is done in nature. Constructionism is an educational method that is based on constructivism. It argues that people learn best when they are active participants in design activities [22] and that these activities give them a greater sense of control and personal involvement in the learning process [23]. Constructionism arises out of constructivism and is constructivism's physical manifestion.

Constructionism is most applicable to ideas relating to this thesis. It is the idea that an individual constructing something will achieve more understanding and meaning from it that is important for us. But what about a community setting? We see this in a quote from Shaw's theory of social constructionism. He analyses constructionism in a group setting – " Individual developmental cycles are enhanced by shared constructive activity in the social setting, and the social setting is also enhanced by the developmental activity of the individual". He further illustrates that Social constructionism is a useful tool for advancing the interests of a community [24]. Basically people helping themselves help themselves and creating new value for the community.

2.4.2 Constructionism in practice

Here we outline some projects that looked at constructionism in community and CSSN settings

Canard - Canard [25] had been a community-messaging project that had constructionism at the heart of the design. The idea being that communities with their varying and special messaging needs could author and create their own customized messaging systems. To quote the author "If as Dewey, Piaget and Papert and others have argued we learn through doing then it follows that to achieve more learning we

should facilitate more doing. If we want this learning to be in support of community, then we should design platforms for social construction – social settings that can be enhanced by the developmental activities of the individual. This tendency of construction leading to understanding typifies the role of hobbyists in other technological areas such as automotive, aeronautic and electronic. (e.g. early ham radio hobbyists who built or modified their equipment.)".

Central to the design of Canard is the hypothesis that open systems allow communities to adapt disparate telecommunications elements to their needs. Such systems allow community members to create applications that benefit the community as a whole. The system core consisted of a framework on which messaging systems could be built and used. It had an intermediate message model and this permitted many different messaging devices to be easily adapted to use the system. The system had been enabled with web interfaces, mobile handset's, pagers and other devices such as scrolling signs. To extend the framework the person would code write code in a simplified language.

The system was evaluated with deployment groups on campus over two years with technically literate members. The results can be further analyzed in the paper [26] but in summation that results were positive in that technically savvy power users adopted the system and some even created significant new services using the system. However it became clear that low volume or users with lesser technical ability were not able to contribute as significantly.

2.4.3 Analysis of Constructionism

The ability to create and modify ones own environment has been shown to be successful metaphor in many settings. Constructionism is

an idea that could potentially be applied to many more areas. It empowers the user and permits them to imagine and realize their own ideas.

2.5 Online Gaming, MUD's and MOO's

Of particular interest to our discussion is the rise of constructionism in online communities as a way for the community to express and fulfill their needs and create a more engaging environment.

2.5.1 MUD's and MOO's

An example of where constructionist ideals have been investigated and shown to be supportive of community building initiatives is in the project Moose Crossing.[49]. The research project directly investigated the possibility of using a MOO (MUD Object Oriented) [47] specifically as an environment for learning through community supported collaborative construction. The project had been aimed at children in the 8 – 12 year old range. Rather than just provide the tools necessary for construction the project wanted to create a self-supporting constructionist culture. Children could create creatures, objects and places and furthermore script them to exhibit behaviors. The project was very successful and the findings reflected the following

- Construction activities helped to create a particularly special, intellectually engaging community
- The design of all communities, not just those with an explicitly educational focus, can be enhanced by a constructionist approach
- Distributed models help scalability and social control of the environment

In addition, Bruckman suggests that there are guidelines for a constructionist approach to virtual community design, namely it should

1) Seek to maximize each individuals opportunities for creative expression and active participation
2) Start with the assumption that average people are smarter and more creative than is often assumed.
3) Provide software tools with low initial barrier to use and high a ceiling with what can be accomplished with them.
4) Encourage users to be creators of content, maintaining overall quality by enforcing a minimal set of community standards and establishing a distinction between private spaces and public spaces.
5) Provide infrastructure for community maintained support

In addition I also refer the reader to further literature research and empirical evidence in [24][23][27][48][49].

2.5.2 Analysis of literature from MUD's and MOO's.

From evidence in literature we can propose that there are three types of user

1. Those who are happy to participate casually in the system
2. Those who wish to extend its capabilities
3. Those who wish to create a new system for the benefit of others

Constructionist behavior works because it allows all three of the above to be satisfied. It permits the creation of functionality that people need to realize.

What we need is a media sharing framework that has a constructionist

approach but not with an epistemological goal in mind. The idea is that a constructionist approach will support the creation of "social constructions" thus enhancing the developmental cycle of the individual and the community [24]

2.6 Chapter Conclusion

In summary, creating interactive compelling experiences in which media content can be shared by people and systems is an endeavor that can only succeed if addressed from a multidisciplinary point of view. It requires elements from behavioral science, technology and engineering, and design [30]. The core thread in the topics covered above is people and their behavior.

Constructionism and behavioral analysis as core elements of an extensible system may provide suitable opportunities for observing novel new behavior.

Chapter 3 –PhotoConnector And PhotoTable

In this section I will describe two experiments that were very important as steps towards MediaConnector: PhotoConnector and PhotoTable.

PhotoConnector was a first stage implementation looking at how group media sharing and authoring could be affected by services provided to the users. It was implemented with tremendous support and assistance from Kodak Research and Development. They supplied the prototype digital cameras to and asked us what services they could offer and how novel new experiences could result. The work became very important in the evolution of MediaConnector and provided a test bed for ideas and sources of material to think with. It was a one off experiment and implementation that occurred at the same time as MediaConnector was being researched and provided important material to think with.

PhotoTable had been an experiment in networked media sharing and shared experience in physically separated social groups. The project being devised to provide answers at what could happen with a series of table surfaces that are interconnected. Primarily it was to see if communities could or would bond through the sharing of media rather than messages and what experiences the community would gain.

3.1 PhotoConnector

PhotoConnector is a project that looked at what potential benefits there could be for a consumer in a world of digital cameras equipped with

GPS devices. The main experimental question asked being “What happens when we have a uniform metadata relationship between differing content sources?”.

Project partner Kodak provided four prototype digital cameras for the purposes of the experiments. They consisted of Kodak DC 260 cameras with Garmin Wayfarer III GPS units. This GPS data would be written onto the JPEG header file and thus stored with the image.

Figure 3.1 The Kodak DC290 camera and GPS unit assembly.

The question that was asked is what could happen, when we have a world full of these cameras all producing images that have a common metadata structure?

Scenario

To provide a tool to think with we devised a potential scenario, “The Wedding”. It goes as follows:

“ Jane Doe had finally decided to give in and upgrade her camera to a new digital super GPS camera for her brother’s wedding. The day of the wedding arrives and she has a wonderful day out, taking lots of wedding pictures, mainly at the venue of the ceremony and the surrounding locations.

The next day after the wedding, she is at home and she decides to share her pictures with the world via her photo album agent that

resides on her PC. The agent takes the photos and shares them via a web mechanism. She browses through the photos and sees that they are wonderful. She wonders “hmmm. I wonder who else took photographs between an hour before my pictures and an hour after my pictures and within 3 miles of my pictures?” The agent goes and queries the system for all matching results.

The results return and she sees that she has all the other photos taken at the wedding with similar cameras…. she sees herself in some of the pictures taking photographs and she sees the ceremony from many different viewpoints. She also has the benefit of seeing photographs from other places as well, since there is no way she could be everywhere at the same time. She sees photographs of the Best man and groom having a drink before going to the church, she sees pictures of the bride leaving the hotel, and of people arranging the flowers just before the ceremony. Additionally, and rather magically, she sees pictures from a child’s birthday party, pictures from the hospital nearby where a child had been born during the ceremony, pictures from a village fair occurring nearby and photographs from somebody unknown’s garden barbeque.

For a moment she makes a connection and a bond with all the places and event in the photographs. She has shared an experience. The moments in time had been captured and each viewpoint is as special as the other.”

3.2 Technical Implementation of PhotoConnector

We decided to devise a system that would help us explore what could be done with media with a common metadata structure between the photographs to relate them.

A web based system was implemented that would permit us to easily modify and extend its functionality. The system would permit

- The upload of photographs
- The viewing and browsing of photos
- Ability to tag photographs with specific notes and text metadata
- Ability to add voice notes to photographs
- Ability to perform basic sort and selection queries on the photographs

To permit this we created a system that consisted of a MySQL database running under an Apache server with PHP CGI-Scripts. The images would be extracted of their metadata and they would then be stored within database records as BLOB fields for the images in the records. Other metadata would be stored as numeric or string values according to the type of data.

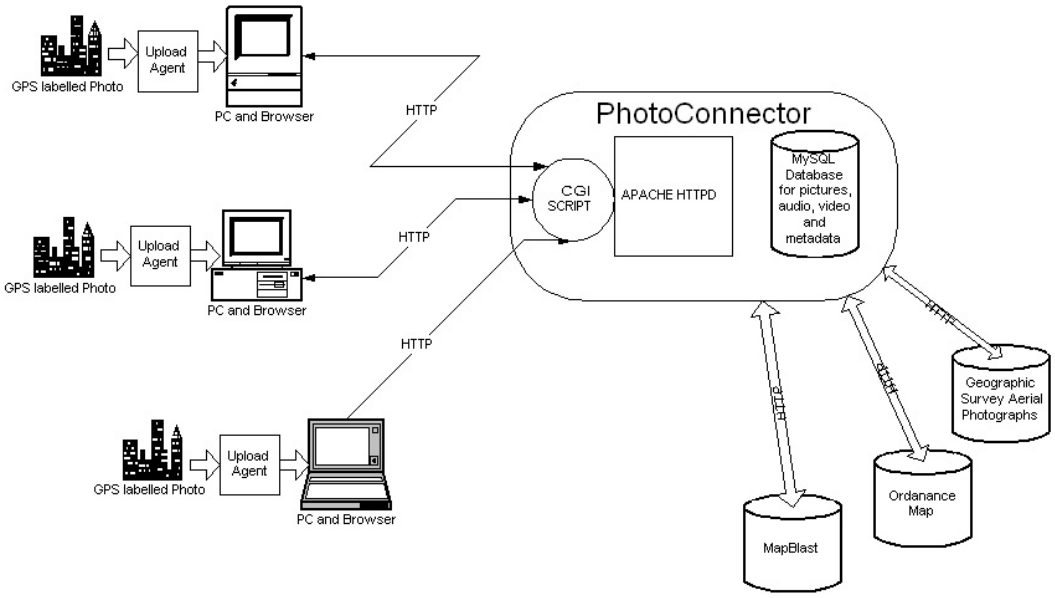


Figure 3.2 A Block System Overview of PhotoConnector

To make the experience as seamless as possible for the user we created a Java upload applet that would detect the images on a card reader, perform all necessary metadata extraction and then upload them to the server and the database using Multi-Part / Form encoding to upload both data and metadata to the PHP CGI-scripts.

3.3 User interface and iteration of design

The first step had been to implement the image transporting Java application, which presented no real problems. With this in place we populated the database with test images. The Java transport program merely detected the presence of the camera flash memory card, parsed the images for their GPS info and then would pass them onto the PHP scripts using the Multipart / Form type of MIME encoding. This enabled a modular approach to design and testing of the system.

For the first iteration of the system I produced a set of pages to query the database. This yielded a system whose usability and intuition was low. The problem lay in that there were eleven variables in the search parameter to fill out and be aware of for most users. The search space as described by the parameters had been too complex to comprehend easily.

The design process then focused on the way that users would perceive services using basically geo-spatial metadata and time. One easier way to approach the search space problem is to use maps. It had been guessed that people would relate more intuitively with maps and timelines.

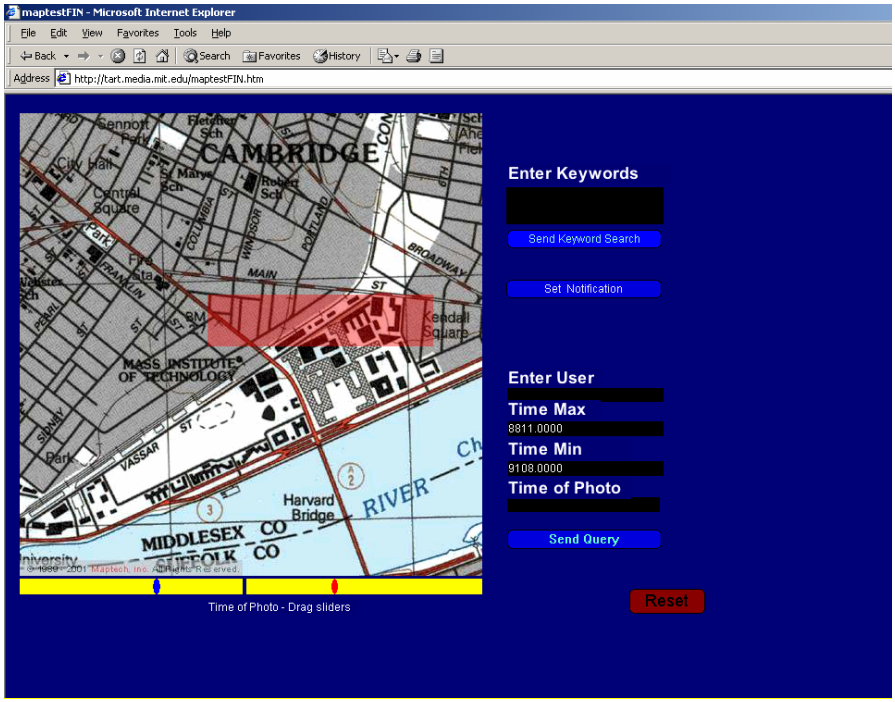


Figure 3.3 The map based GUI for searching in the database.

We can see that a map of an area can have a search space dragged out on it and in addition a time range can be set to look for items within a time period. Additional parameters include keyword searches. Also we can set a future monitor – if a picture gets taken within a certain set of parameters including future time then you will be notified.

Other aspects of the interface were simpler such as viewing results, modifying metadata pictures and annotating pictures with voice annotation and text / keywords.

To enable users to catalogue and index their own photographs and associated events we devised a prototype system based around the motif of a diary organizer. The idea is that we organize our memories and events temporally and that a timeline based metaphor such as a diary might be a good way to organize past "photo" events and future opportunities [54]. But a key feature that needed to be addressed was peoples desire to use photographs as instruments of social contact. It is

common behavior for people to show others their photographs to help relate and share experiences. [58] Photo albums of weddings and holidays are used to provide a way to keep in touch. Basically people use photographs as social instruments to communicate with each other. The physical diary provides a secondary function as well. It acts as a browser for ones own social network, by storing contact information and bringing together in one place a persons social contacts. This had to be something that we explored further. We also decided to make the diary shareable via the network so that people could keep abreast of new developments in the lives of people they know. If they had new photographs to share a flashing link would indicate this and they could click on it to allow them to view their photographs. The photographs communicate themselves but also people could the use the viewing activity to create a reason to call or email someone.. An address book would list friends and family entries and at the same time show links to their latest photographs. This provided a browser for their social networks and their networks news

Qualitative tests indicated that most people intuitively picked up the capabilities of the system and what could be done with it. The design was common sense approachable. People intuitively knew how it worked.

In effect we had created social groupware. More specifically set of tools to help one another relate to each other through the exchange of media. The diary provided a common metaphor to store and organize photographs and also to provide a browser on a person's social network.

GPS Co-Ordinates – A Common Metadata

In addition we discovered that a number of content systems on the web use GPS co-ordinates as a way to index and retrieve their content. As our system had been web based as well we were easily able to cross-refer our photographs to systems that could provide information to support our images as well. These included

a) Standard Street Maps
b) Geographical Survey Maps
c) Aerial Photograph collections
d) Weather services
e) Tourist information
f) Proximity and city guide services

Points (a), (b) and (c) were immensely useful and help set the context of a photograph, especially when the contents were ambiguous (e.g. two people you don't know, but where was it taken). It appears that latter three would be useful when used in the context of a tourist information system.

It also had been apparent that one could translate the GPS event into an event that could relate to a news archive for example, e.g In Boston on this day this happened. This had not been pursued at the time but an engine to transcode such events seemed perfectly feasible. Such an addition is likely providing a useful service addition to a sharing community.

3.4 User Experiences with PhotoConnector

People like to share experiences. The system was successfully tested with a small group of people proving that concept. Even with four cameras as source material generators we found interesting and novel

results. Most of the interaction design and iteration came from Kodak, and myself.

The qualitative results about the services that were provided (mapping, annotation, novel search etc) were very encouraging. It provides a strong case to creating a larger scale test with more cameras over a longer period of time.

What had been predicted, and seemed unlikely to occur in such a small test group, was co-incidence. In one event it appeared that a participant had been playing crazy golf with her children in Martha's Vineyard, whilst at exactly the same time (time difference accounted for) in London, England, I was wandering through the tropical greenhouse at the Royal Horticultural gardens in Kew Bridge. Actually relating the images to a map made the significance of the event much stronger. For both parties the revelation had been novel, of interest and a catalyst to further conversation.

Generally the experiement was succesful and novel new facilities that colloquially were described by participants as "cool" were produced.

For users making the connection between their own events and other real but hitherto unknown events had been one of the most compelling aspect of the system. It was the finding and sharing of an experience.

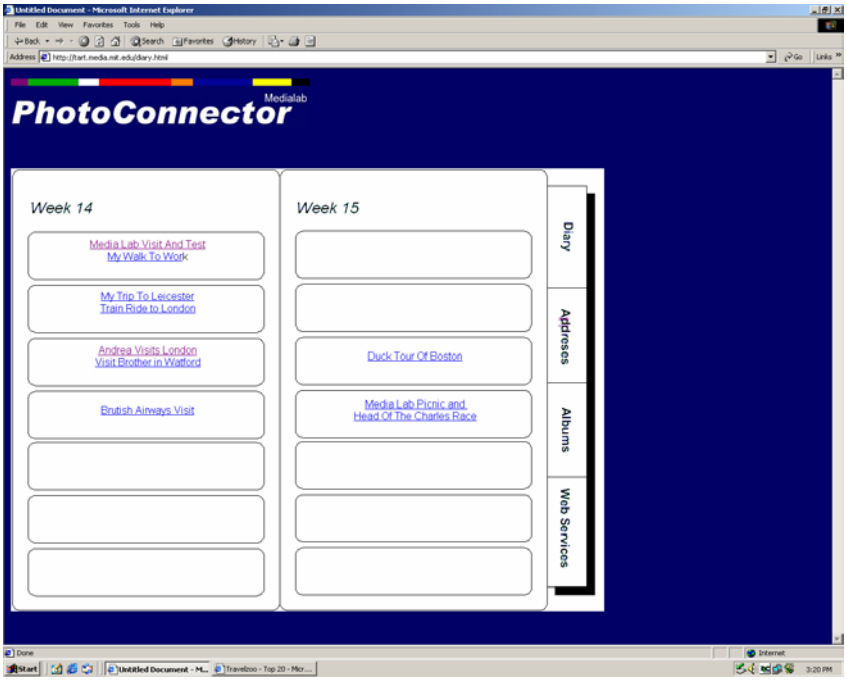


Figure 3.4 A figure showing the diary motif for organization. This permitted a way to view the activity of ones social network as well.

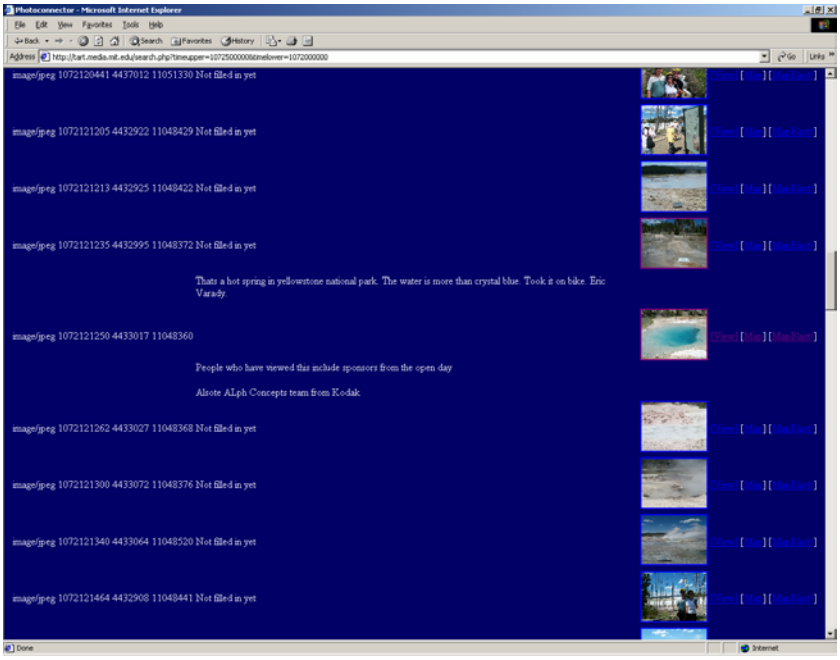


Figure 3.5 A basic return of results from a search

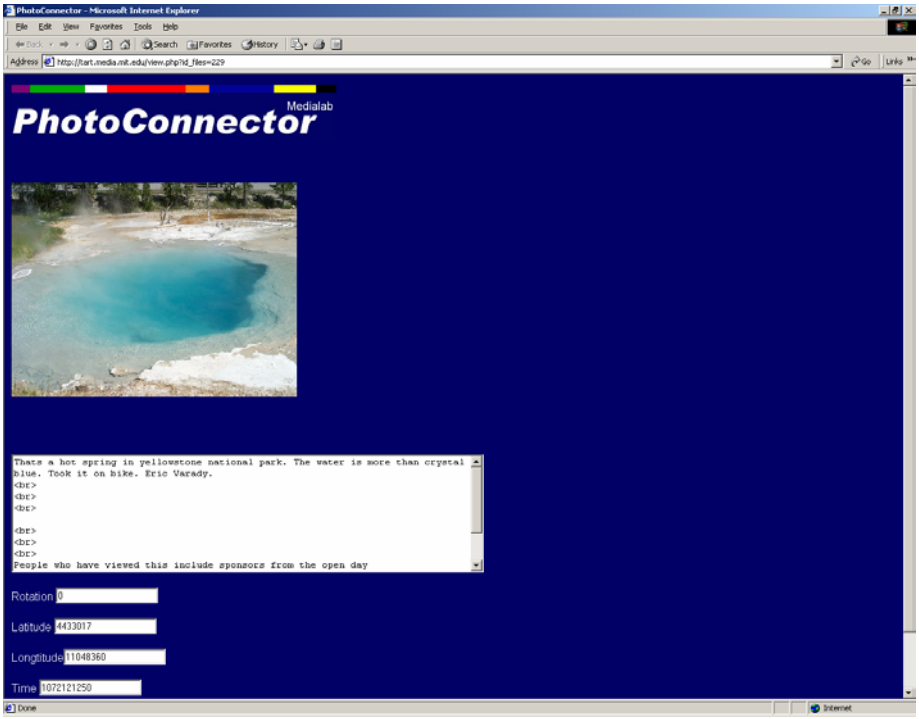


Figure 3.6 One of the photographs expanded upon for closer viewing

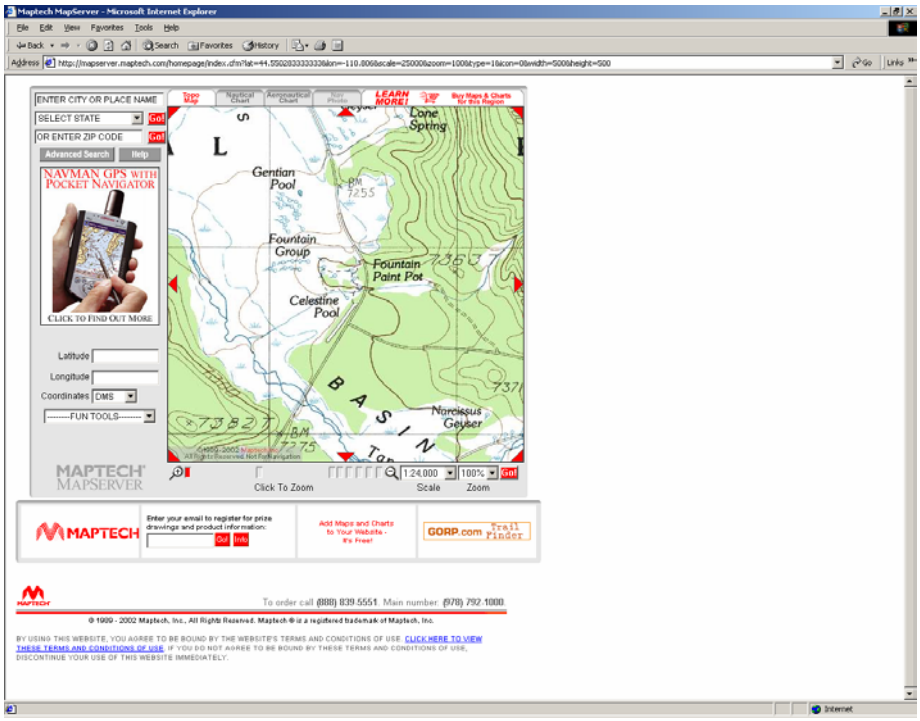


Figure 3.7 Geographic survey maps of the photograph's Location

Figure 3.8 A regular street map of the photograph location

3.5 PhotoConnector Conclusions

The system had been a one off project that was successful and provided a thinking tool as to what services are needed by a community of media creators and sharers. It provided material that lead to the ideas that autonomic frameworks that support the exchange and linking of metadata could help to augment media experiences.

With 4 cameras the system provided excellent results and insights. The results make a case for scaling up to have hundreds of users and then performing detailed studies on how the system evolves. Another important facet would be how the users behavior changes.

Key things that this project provided were

1) Evidence that a media sharing experience is enhanced by a collection of services. People enjoy the experience more and use the tools for that purpose.

2) A common Metadata structure provides multiple ways to access and view the same data. We can choose our own mappings and gain meaningful insight given appropriate tools and "windowing" tools.

3) The common metadata also provided a good way to relate various other 'bins' of content again adding meaning to the initial authored content. Again the correct tools must be provided.

4) The idea that user constructionism would provide better experiences for the individual and the community at large. Empower people to create their own tools.

5) An idea of the technical needs that a larger media sharing framework would require, such as massive scalability and ability to construct for it at varying levels of user ability.

3.6 PhotoTable

[A short paper that was an interim document is included in appendix E for further information]

Overview

Tables have a special meaning for us in our everyday life. They moderate the space between us when we meet. They hold our papers and documents as we share and discuss them. They prop up our feet when we watch television. They hold our plates when we eat and they enable families to congregate and eat. Socially and professionally they are important dividers of space.

In conjunction with my colleagues Edison Thomaz and Andrea Lockerd, we asked some questions. Could we extend the metaphor of the table across a distance? Could we eat meals together on different continents? Could we share documents synchronously and asynchronously? Could we share memories with photographs over networked coffee tables? We decided that the latter had been a good way to make a first exploration.

PhotoTable was a project that created a series of TCP/IP networked coffee table surfaces. People could upload photographs via a web-based form and then also be able to remove them using a web-based form. The idea is that what is seen on one table could be seen on another table that is a member of a defined group of tables. We hoped that community members would engage in providing new content and material.

3.7 PhotoTable Technical Implementation

Photo Table was implemented using the Isis programming language and ran in a Linux environment. The design of the program was trivial. Basically the system would play photographs out in sequence from a directory and loop.

Upload of photographs and PHP scripts running under CGI with an Apache web server had implemented the removal of photographs. People would use a web delivered form to upload and select media for removal.

Initially that was all that the table had as an interface. Upload and delete with a slide show that ran automatically. However people soon demanded that they be able to stop and view or skip ahead in the

collection. We implemented this as a haptic interface, using reed switches and magnets to produce the underlying interface mechanism.

The actual display consisted of a top projected image from a video projector projecting onto an opaque white surface. The haptic control could also manage the transform of the photo by 90 degree chunks so that people who were sat around the table could view the images correctly if so desired by merely passing the magnetic puck.

3.8 Setting Locations and description

Two tables were installed. One was placed in a social common area at the Media Laboratory in Cambridge MA and the other one in a similar area at the Media Laboratory Europe in Dublin, Ireland.

The idea was to build a better connection between the two locations and the people there. Since the system was completely open and 3anonymous, the community could manage any removal of offensive material if the case should arise.

3.9 Observations and Conclusions about PhotoTable

The system was installed in both locations and the transfer of photographs between the two locations was handled manually.

The Cambridge installation received a mixed reception as the bright projected image interfered with peoples television watching, another very important recreational activity in the same area. The system was treated with initial mixed reception with indifference setting in soon after that.

The Dublin system however received a warm welcome and was maintained by the community very well with new photographs

regularly uploaded and old ones deleted. Some of the images were of holidays or travels but the most in part were of doctored humorous images, cartoons and jokes, images from parties or of contentious messages. They also wanted to take the system and adapt it to their needs however in the end the few individuals who expressed an interest were unable to due to their barrier of entry being learning the Isis programming language.

Overall the experiment showed that people do want to communicate using images and that they are happy to use images in a public setting to relate themselves to a community. I also noted that the social activity of the two groups reflected itself in the social activity present in the two different locations mentioned, with the Dublin group being much more bonded, cohesive and about personal friendships and with the Cambridge, MA group much more transient in the space, reflecting the larger group present.

Secondly since we had no basis to know anybody there (except for myself, who knew a few people there) and vice versa, the images still didn't mean that much. The photo would essentially be of an anonymous person in some setting. This I think was important for the success in Dublin and the lack of success in Cambridge. The community knowing each other and being more socially engaged had more material in which they could relate the content with someone they knew or had some previous physical meeting with.

I concluded it was important to have some relationship basis for the shared experience of personal experiences.

3.10 Chapter Conclusions

PhotoConnector demonstrated that technology can be applied to create services that link together metadata to create compelling experiences. Metadata in effect creates multiple ways for the user to approach the same data using different conceptual mappings. The experiment suggested that the sum and interaction of multiple service tools lead to the whole media sharing experience.

PhotoTable demonstrated that people are happy to use images in public settings to help them share experiences and create bonding between others that they know. It also showed that some experiences are best shared between people that know each other otherwise the message is meaningless. It also showed that people want to create or change things to suit their community and context.

Chapter 4 – System Design and Implementation

4.1 Chapter Introduction

The previous chapters outlined theoretical and empirical precursors for some broader thinking. There is an opportunity in having a media sharing system that supports the following features

- Social group behavior – identity, gathering and advertising
- Media Sharing – file exchange
- Enhancing Shared Experience of media – behavioral cues for trend spotting
- Utilizing Social Network analysis – for understanding all the above
- Constructionist behavior – to allow the community to evolve by its own need fulfillment.

These are the key drivers for the design of the Media Connector framework. In this chapter I will outline the Design and the Technical considerations.

4.2 Design of MediaConnector

The first stage was to create a list of specifications for functionality and features that Media Connector should encompass and also a profile of the intended audience:

Feature	**Why?**
API	To allow advanced users to access modular functionality from their programs
Scripting	To allow lower end users to access modular functionality to customize and create their own functionality
Monitor Social activity	Towards the goal of using social network analysis in the future
Monitor Media Activity	Towards the goal of understanding relationships between media items in an Actor network
Extensible – functionality	To allow new functionality to be added to Media Connector by advanced users
Multi Platform	To permit the broadest range of users to be accommodated
Monitor user behavior	Towards building a profile of the users consumption for others to access
Spot and monitor trends occurring and capitalize on them	To allow recommendation systems to gain advantage and provide early warning of "trends"
Functionality	
Create Unique User Identities	To enable users to have one identity
Create groups / Join Groups /Leave groups	To enable people to form special interest communities
Resource discovery (including peers)	To find a media item or service that is required
Advertise Resource / File	To aid in resource discovery
Download File	To aid in file acquisition
Unique file Identification	To allow files to be identified uniquely
Messaging to occur as XML structured documents	Allows other XML compliant programs to access messaging services within MediaConnector
Create an audit database	To help build behavior profiles

Table 4.1 Audit results of existing practice

4.2.1 System Assumptions

We started off by making some assumptions for the system and the audience (and to help constrain the project to reasonable resources for this prototype)

- One user profile per instance of MediaConnector
- That the platforms will be of similar capability
- Media items can be of any digital type
- Authentication would be minimal
- Security considerations are lax for the purposes of this project

4.3 MediaConnector Architecture

The high level architectural decisions are outlined one at a time

Scriptable application versus Component framework - I decided that a framework rather than a scriptable application would be appropriate for the aims of MediaConnector to be met. A single application would be too restrictive, however component architecture based upon a framework of unit functions would allow applications to be written by more capable users and also allow the same to extend its capabilities by adding new modules to the framework.

Centralized versus Distributed - The next decision had been that a distributed approach would be best in terms of system flexibility, persistence and performance. Popular media sharing applications based on similar architectures such as Napster, Kazaa and Gnutella have shown this to be a successful choice.

Group Ad-hoc profiling - The most flexible way to meet the requirements of the needs for behavior profiling, media profiling an

etc. is to have a database present that stores an audit trail for each node. If we were to collate data we could build group profiles on an ad-hoc basis based on the members present.

Scripting – I decided would allow for flexibility for lower ability or those with less motivated needs, to access the framework to either build applications or customize the functionality of their current applications
The design of scripting language has to accommodate the lower aspiration/ability user. In short the language has to compromise between ease of use or learning and flexibility to achieve desired results. This section is explored in depth during the implementation phase.

Service addition – each node should be able to offer, execute, advertise and download a new service. Protocols should be established well enough to allow the services to be usable by a wide variety of clients not just the applications in the node. One such service that is established and uses in the main TCP/IP protocols is that of HTTP based CGI-BIN programs. These can be downloaded and be executed locally or from other IP addressees.

XML services – to encode and decode XML documents XML parsers should be included.

API – to be exposed as a set of calls via native Java language implementation.

4.4 Implementation of MediaConnector

In this section I will describe the technical implementation of MediaConnector and describe the reasoning and basic functioning behind the various choices. The section on scripting will include a short debate about scripting language design and the reasons for the final choices.

We will initially start off by looking at the various components that make up MediaConnector.

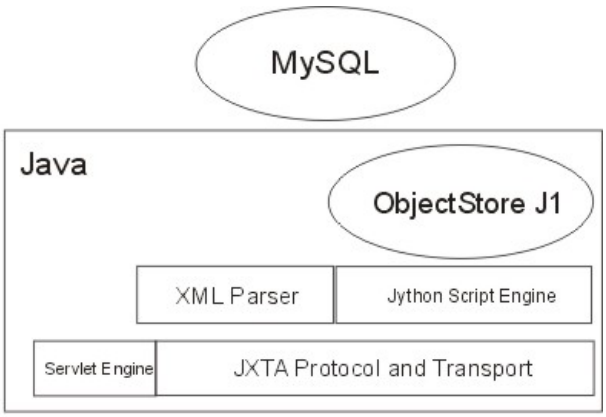


Figure 4.1 The MediaConnector Modules. As the diagram shows MediaConnector is essentially bringing together existing modules that together create a versatile functional base. The components are written in Java, which is also used as the glue between them.

Java – This language had been chosen to help reach our goal of platform independence. The language has a significant number of stable JVM releases and this boded will in its favor. Also its popularity since 1994 means that there is a large amount of material available to developers in design and debugging. Also it provided a greater choice of being able to integrate sub-systems that were written in Java. Some of those follow…

XML parser – Originally the popular SAX/JAX model [http://java.sun.com/] was evaluated and the conclusion had been that it embodied too many features and hence it also consumed a lot of memory and install space. Whilst working on another project with J2ME cell phones I discovered the NanoXML parser [http://nanoxml.sourceforge.net/]. The stripped down XML parser did exactly what we required. It returned a tree of nodes from an XML document and would take a node tree and write out an XML document. The functionality of this was made available through an easily instantiated object with exposed methods. The class library had been integrated easily with no modification.

Servlet server – To enable new services to be easily implemented and accessed I implemented a modified Java servlets server. Servlets technology provides a simple, consistent mechanism for extending the functionality of a web server without the performance limitations associated with CGI. Servlets have access to the entire family of Java APIs, including the JDBC API thus allowing easy integration with other database services via TCP/IP or connection to the core MediaConnector database. Servlets can also access a library of HTTP-specific calls. Reusability is of prime importance and it adds another element to the constructionist ideal for MediaConnector. This is in addition to the existing code base available. A full functional description can be found at the W3C website. [http://www.w3c.org/] .

The system had been proven during the development of a test prototype described in the next chapter. During the latter phases of development we wanted to implement peer – peer on screen messages and a “request” facility. These were implemented quite painlessly as servlets that

interfaced with the show-rendering engine via small text files. XML was not used, as there seemed to be no point as communication had been on the node itself. The services were published and participating viewers could access the respective web pages via their Internet browsers or name and pair value HTTP calls (e.g. another Java applet /servlets for example). In addition the binary executable could be downloaded from another server if needed.

JXTA – Originally it had been intended that I write my own Java based Peer-to-Peer protocol and actually I made inroads on this approach. By chance I met with some engineers from Sun Microsystems who were looking for developers to try out their new P2P protocol written in Java. After basic evaluation of its features I decided to make it the basis for the P2P protocol implemented in this project. The framework for the JXTA services I use in my project had been adapted from an example pattern by Sing Li of Sun Microsystems which he published in his book[56]. I extended upon his Content Management pattern with the required additional functionality that was needed by MediaConnector.

Overview – JXTA is an open source set of protocol definitions and Java calls that implement a true Peer- Peer system. The system had been designed to fulfill some of the potentials of developing applications in P2P, namely:

1) Enabling interaction between peers
2) Zero reliance on centralized resources and services for operation
3) Cope with extreme changes in network composition
4) Applications that thrive on Non Deterministic network topologies
5) Scalability to a massive user base

JXTA implements a pure P2P system that does not rely on any centralized resources to operate. Hence it cannot rely on a centralized naming system such as DNS (whilst tables are replicated the structure and allocation is administered by one body) and as such it implements a distributed addressing system. It operates conceptually like UNIX pipes and processes. When a peer (process) wants to talk to another peer a unidirectional pipe is set up and one process feeds data to another process. If a channel back to the source is desired then the process is repeated with another pipe in the other direction. Peers are found using a rendezvous service and the routing is completed by an address resolution and routing service.

I have intentionally kept this section brief. For further information the reader is referred to the JXTA platform documentation at http://www.jxta.org/ for detailed technical descriptions.

JXTA is an incredibly powerful yet complex protocol. In use the learning curve was steep and a lot of the features were not used due to complexity and performance issues. However once some work had been done in establishing a few of the basic protocol patterns the rest followed more easily.

Databases

Our specification called for the ability to infer and monitor social networks, build patterns of usage around media and build user profiles. As discussed earlier it had been decided to do this without direct user intervention by using a database populated by events. Although computationally expensive on a larger scale it affords the most flexibility for the various decided analyses and allows scope for evolution.

It is left up to the application designer to ensure that events are placed in the database duly when an event occurs. The database would be present at each node to permit true peer-to-peer decentralized architectures. The database would only contain information for that user and node.

Object Database vs. RDBMS

A decision had to be made as to which type of database to use. The most popular format for a database is the Relational Database Management System or RDBMS [33].

The main disadvantages of Relational Databases include their inability to handle specialist areas like spatial databases (e.g. CAD), applications involving images, special types databases (e.g. complex numbers, arrays) and other applications that involve complex and changing interrelationships of data. Otherwise their presence and use in commerce and general usage has led to their steady evolution and nowadays they operate very efficiently in their competency areas.

Object-oriented (OO) databases employ a data model that supports object-oriented data structures with all the benefits of OO design paradigms. OO databases provide unique object identifiers so that the objects can be easily located. This is similar to a primary (unique) key in the RDBMS model.. The data in object-oriented database management systems (OODBMS's) is managed through two sets of relations, one describing the interrelations of data items and another describing the abstract relationships (e.g. inheritance). These systems employ both of these relationship types to encapsulate data items with procedural methods. Thus, a direct relationship is established between the application and the database data model. The strong connection between application and database results in less code, more natural data

structures, and better maintainability. OO languages, such Java, are able to reduce code size by not having to translate code into a SQL or JDBC [34]

Implementation

It is the ability of the object database to handle complex data and relationship types that appealed the most, as it offered the most potential flexibility. I acquired and integrated the Fast objects J1 object database from Poet inc. [35]. Written in Java it integrated easily and the next step was the learning curve associated with using a new technology.

The database was surprisingly easy to use and had an incredibly small footprint. Performance was good and early tests indicated that it worked very well.

However during the course of MediaConnector's development the following problems occurred:
1) SQL queries, which are useful for reporting on data sets, would not work with the version we had. An expensive upgrade would resolve that but that was not justifiable for the investigation at hand.
2) It would not integrate properly with Java servlets and hence it would be hard for people to create new services that could use the database.
3) The paradigm shift had been too great to fully realize the potential of the database in the given time.

The solution to this had been to integrate the freeware MySQL RDBMS [http://www.mysql.org/], mainly to permit 1 and 2 above to be resolved quickly. It was decided to leave the OO database present so that its potential could be explored if so desired by future work. The databases are accessed via internal API calls and the MySQL RDBMS is also accessible via two Java servlets. The latter would permit remote

querying and update of a database from another peer.

4.5 API / Scripting language design

4.5.1 Scripting Introduction

Towards a primary goal of making MediaConnector extensible and reconfigurable by the user community I decided to permit people to facilitate this by exposing a Java API. The API is documented in Appendix A. This API permits a programmer to build a Java application from scratch that utilizes MediaConnector or to significantly modify an existing MediaConnector application.

Now this section will primarily focus on providing a similar facility for users who are not as technically literate and whose needs for flexibility are not as likely to be great. In essence these users will require modifying existing applications and creating new applications that are higher-level reconfigurations of modules in MediaConnector.

4.5.2 The case for scripting languages

System programming languages such as C and Java were designed for building data structures and algorithms from scratch using the most elemental components such as words of memory. Scripting languages, such as Python, REXX, Tcl/Tk , however assume the existence of a powerful set of components and are intended primarily for connecting components together. Scripting languages are often used to extend the features of components but they are rarely used for complex data structures and algorithms [50]. For example compare the code fragment sizes in Java and Tcl to perform the same task. (Place a ‘hello world’ button on the screen) [55]

Figure 4.2 Code fragments comparison from Java and from Tcl

Java

```
// Hello button using awt (jdk 1.0 compatible)

import java.awt.*;

public class hello extends java.applet.Applet {
+ public Button hello;
public Button bye;

public void init() {
+            hello = new Button();
+            hello.setFont(new Font("Times",
Font.PLAIN, 16));
+            hello.setLabel("hello");
+            this.add(hello);
             bye = new Button();
             bye.setFont(new Font("Times",
Font.PLAIN, 16));
             bye.setLabel("bye");
             this.add(bye);
}

public boolean handleEvent(Event event) {
             if (event.target == bye && event.id ==
event.ACTION_EVENT) {
                          System.exit(1);
+            else if (event.target == hello &&
event.id == event.ACTION_EVENT) {
+
             System.out.println("hello");
             } else {
                          return
super.handleEvent(event);
             }
             return true;
}

public static void main(String[] args) {
    Frame f = new Frame("hello Test");
    hello win = new hello();
    win.init();
    f.add("Center", win);
    f.pack();
    f.show();
}
}
```

Tcl

```
button .b -text Hello! -font {Times 16} -
command {puts hello}
```

The other major difference lies in the ‘typing’ of the language. (its ability to handle and enforce data types). Scripting languages are very weakly typed whereas system languages tend to be strongly typed. Scripting languages usually are less efficient (executing) than system programming languages because their components are chosen for ease of use and relative power rather than efficient assembly for the underlying hardware. In most cases however it is not the execution performance of scripting languages that’s an issue, it’s the construction time and ease of use that is its primary goal. They sacrifice execution speed to improve development speed. In practice ample evidence exists to support this proposition. The rise of the world-wide-web and the increase in the popularity of scripting languages is in part attributed to the need for rapid application development and the accessibility in terms of

knowledge and experience that scripting provides i.e. you need less than a systems programmer would.

4.5.3 Application Domain for the scripting language

For the scripting engine for MediaConnector we are trying to find the optimal balance between learning effort, Syntax complexity (readability and writ ability) and the amount of flexibility that can be exercised.

Our audience profile would be targeted to those who have little systems programming skills but are technically literate to be able to understand the basic principles of file sharing. Some of the more esoteric functionality within MediaConnector may still remain out of the grasp of these users however and this is where normally community support would step in to help the individual out. i.e. a more talented programmer in the community would step in to help write the functionality.

A diagram maps out the various languages with cost of learning against flexibility.

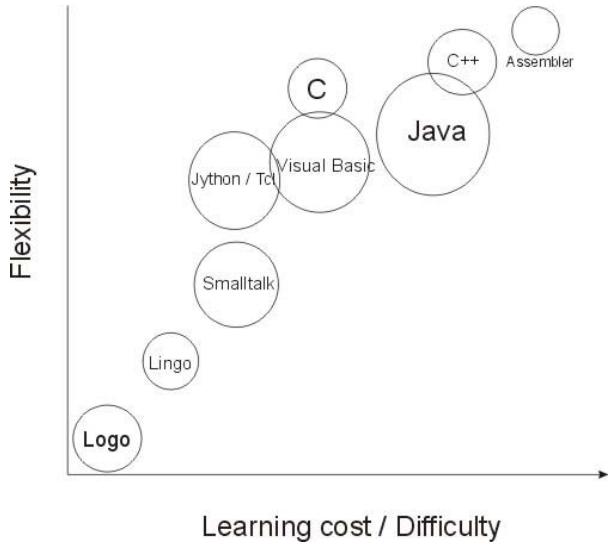


Figure 4.3 Diagram showing languages and their positioning in a flexibility / difficulty space.

In an ideal world we would like to be able to have non-technically literate people pick up the scripting language in no time and build customized and complex applications using the scripting language itself.

A quick, logical and cognitively straightforward design for a scripting language would only involve procedure calls mapped to keywords. There would need parameterization and basically this would lead to a sequence of steps to follow to initiate actions. However this would only allow extension of the application and not the creation of other functionality.

It had been intended that an original scripting language would be devised for MediaConnector. The early sketches however quickly revealed flaws in the language, specifically its lack of versatility. The language design project was dropped due to time constraints and a search was made for a language that could be implemented quickly and that found a median value for the three decision criteria of syntax, ease of learning and flexibility.

Jython, a Java version of Python [http://www.Jython.org/] was deemed to fit the bill. It was easy to integrate, the syntax is slightly terse but comprehensible, it had a high level of modularity as well and also servlets could be written in it to create public access services via the applet server. The following is an applet that displays hello world.

```
from java.applet import Applet

class HelloWorld(Applet):
    def paint(self, g):
        g.drawString("Hello from Jython!" 20, 30)
```

The language is short and concise and free of unnecessary and confusing syntax.

4.6 Chapter Conclusion

With the architecture and components chosen, we have created a flexible framework for the creation of media sharing applications and environments. The peer-to-peer facet of the architecture enables scalability. The use of open standard modules permits access to knowledge, code and supporting groups.

A significant amount of effort was expended on bringing the various modules together and making them work. Most of the extra work revolved around creating services and object classes that would provide the most flexibility in re-use, hence a generalized approach to functionality designs.

Some compromises were made due to time restrictions.
In particular much opportunity lies in experimenting with one aspect – the scripting language. Jython had been chosen due to its completeness, similarity to Python and the lack of time available due to effort expended on JXTA implementation complexity.

A successful basic framework, which meets all the primary design requirements of flexibility, scalability and constructionism, has been constructed. The next stages will explore the framework in use and the results of trying to create novel applications with relative ease.

Chapter 5 – MBeat – A design case in MediaConnector

This chapter will outline the development of a demonstration system that had been built on top of MediaConnector. The purpose of this exercise was to see if indeed MediaConnector is a good framework on which to build novel applications and to test its design parameters of ad-hoc membership and distributed peer-to-peer media applications. Could it deliver on its premise of construction, distribution and versatility?

5.1 Overview of MBeat

My background in television led me to devise a high level concept – a television show that would tailor aspects of its content to a given social group watching it. The idea is to extend upon the 'water cooler effect' [30] of shared experience amongst groups where the members of the audience know each other in some social or professional capacity. Secondly it is to challenge the common idea that a television show or a video is a fixed entity editorially [53][60][61][62]. I wanted to look at the ground between individual personalization that produces unique experiences [60][61] and mass media broadcasting. The idea of personalizing a media experience for a group size [62] where individual preference features are still discernable enough to be able to differentiate that group from other similarly sized groups had been interesting. Statistically you can show that as a group sizes increase, the group would start to resemble others as the features average out on larger sample sizes.

The format chosen was that of a television magazine style show with packaged content, i.e. a show that is made up of smaller edited

sequences of video content that are stories in themselves and are linked together to form a show by using a narrative linking mechanism such as a presenter. News shows, daytime magazine shows and music video shows are an example of these. Due to time, budget and asset availability it was decided that a music show format, MBeat, would be produced.

5.2 MBeat Show format

To achieve appropriate levels of personalization for specialist content, it was decided to use the archives of the viewers where possible mixed in with broadcaster owned material. Copyright issues aside, with technologies like Tivo, DVD, DivX and broadband this is feasible and not as far-fetched as once it would have seemed.

The MBeat model is novel. A broadcaster would use their editorial skills to create a semi-populated template for a show. Some content, like new releases and news items, are not going to be owned in the community yet. Neither would proprietary items like presenter links and titles and brandings. An analogy for the show would be like that of a customized car. The basic model is there but the customization of features is by the choice of the buyer.

The template is delivered, and then according to your consumption preferences it will adapt and populate the show using your content or broadcaster owned material. If during the viewing period members of a defined community, say your friends, were to tune in then the show would change the remaining play order (edit list) to reflect the change in audience preferences? For example imagine that person ‘A’ starts to watch the show whose interest mainly lies in chart pop music. The system will populate the show with pop videos of the preferred choice. Now imagine friend ‘B’ logs into the show and starts viewing person

'A's' show synchronously. The program accounts for B's preferences and adjusts the content to reflect items that they would both be interested in. This would extend linearly until such a point where we reach an average model of the group and hence an editorial suited for broadcast. We are interested in the smaller group level.

The show is constructed by sequencing video elements together namely videos, presenter links and graphics and interstitials.

The show had been designed to have sections that were editorially set and also to have sections that were changeable

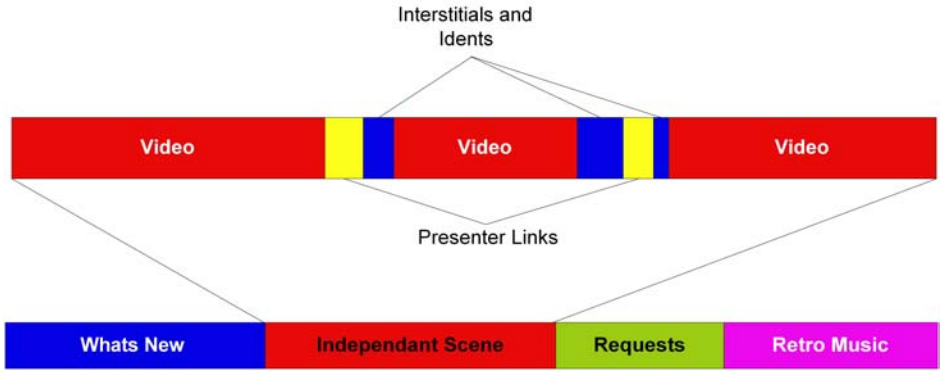


Figure 5.1 showing the organizational structure of a show composed of packages and an expansion of a package into the sequences that comprise it.

The final version did not have the latter whole show templated format due to time restraints but the concept was proven by the implementation.

5.3 Metadata Considerations

To implement the functionality required in MBeat, MediaConnector had to build a profile of a person's musical listening preferences and of the group. This I decided to do via a modified MP3 player. The MediaConnector system is extensible to accommodate any media (you

adapt the Media Watcher class) but for this test we focused on MP3 events.

Various types of algorithms were experimented with, from a basic algorithm that used simply statistical frequency to the final one, which tempered frequency with time.

The model is derived to account for human behavior. Purely statistical models such as Collaborative filtering are not able to account for context. For example lets take an average 30-year-old rock fan that has been listening to rock music for the past 15 years. His statistics would show that his major preference would be to listen to rock music which in the long term average would be true, but however it doesn't account for the fact that recently he has been listening to psychedelic flute music because he had been introduced to it by a friend and is keen on hearing more at the moment. To resolve this we implemented a simple algorithm in the playback engine thus where λ is the confidence score built from the product of two matrices (the collective preference and the clip metadata scalars) and K is a constant matrix value. As time t grows larger (i.e. its been a long time since we played that track) then the score matrix S evaluates to λ.

$$S = \lambda - K/t$$

This made sure that the same video wasn't played more than once in a close time span, to maintain editorial variety. The play list is edited with the highest scorer being placed in the list. The metadata recorded had been the following

Time : Track Name : Artist : Genre: Format

A modified MP3 player would transmit this to MediaConnector layer, which would then record it in a database table for media events.

5.4 MBeat Architecture

The design had to fulfill a few design requirements

- Record metadata about a persons media consumption habits
- Permit ad-hoc groups to form of variable size
- Permit the player app to get the profile information from individual nodes / profiles
- Record metadata without user intervention
- Provide messaging and request services for the show
- Register viewer presence / non presence
- Each node should be capable of rendering the show
- Provide a way for the viewer to express dissatisfaction with the algorithms choice

The sequence of operations would be best described as follows:

- A node initiates the playback of MBeat and advertises this
- Another node seeing this advertisement would choose to join the group that the rendering is occurring on
- It downloads the MBeat program and timing information
- The primary node re-calculates the play list. It does this by first polling to see which clients are present in this group. For those that are there it downloads their statistical data and forms a composite model. The play list is then re-built.
- Sibling nodes acquire play list from primary rendering node
- Sibling nodes acquire any deficit media via p-2-p resource discovery. The media in effect ‘ripples’ through the group

- Primary node publishes a list of services such as messaging and requests
- Primary node maintains current membership information

This lead to the architecture formulation as described here for the demonstration.

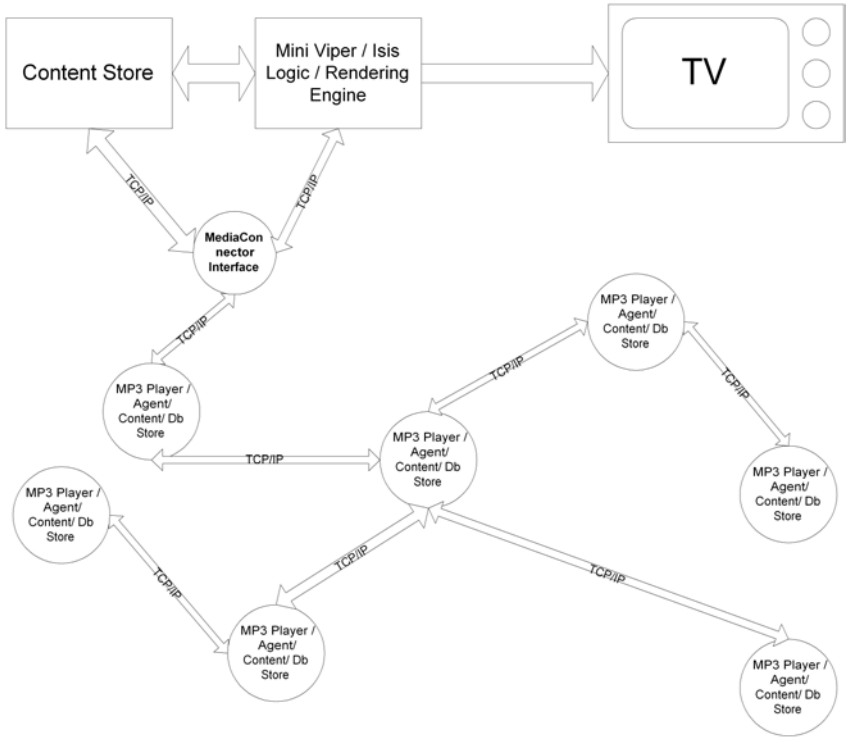


Fig 5.2 Showing the structure of the MBeat television application

5.5 MBeat Technical Build

The project had three major parts:

1. A player / playback engine that would render the television program. This was not written in MediaConnector but had been written in Isis [53]. It interfaced to MediaConnector via text message passing via HTTP calls to servlets.

2. An MP3 player that would record information about track, artist, time of event and similar

3. Underlying network protocols to allow groups to form, message and query each other

The first part was probably the most straightforward to implement and had been implemented in a manner as to keep its operation as simple as possible. The design worked in conjunction with the Viper [53] annotation system and a modified Viper playback engine. This allowed the video clips to be annotated with metadata thus

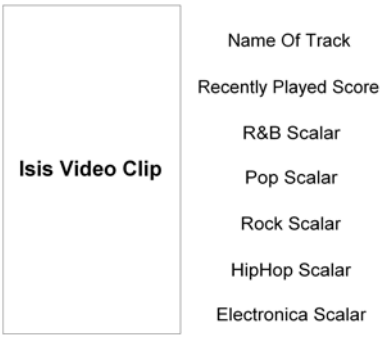


Figure 5.3 Diagram showing the metadata attributes assigned to a music video clip in MBeat

This is necessary to allow the system to identify and relate between the various pieces of video.

The original idea with the MP3 player had been to look at the last accessed information for various media file types by an agent that would collate the information. However it was discovered that most of the intended platforms don't publish that information (UNIX does and windows has 1 day resolution but none of them do under Java). In this case it was decided to modify an open source MP3 player to change the last modified date of a file and then to have the Media Watcher

class monitor for new reads (changes in modified date). This created a sequence of media events (When and what things are played, watched, read etc). The modifications were made in Java to the JLGui Java player.

The networking and protocol layer had been relatively simple to create with the building blocks of MediaConnector. (It was difficult to get there but after that it had been relatively simple). First of all a node is configured to act as a rendezvous for the audience. A group was published and a couple of servlets were created to provide the messaging and request services. Further detail about the protocol operation is given in chapter 4. But basically once a node joins a group then there are certain group services provided such as a global resource discovery and acquisition system (Search and acquire media). Additional services such as messaging are created and published.

The actual coding had been done by calling a sequence of Java routines from the MediaConnector main thread class (ShareFrame) and building the service from there. The application could also have been built by using the API in a separate class or the Jython engine. As this was the first test it was decided that it was prudent to minimize the risks initially especially as there were likely to be flaws present.

5.6 MBeat User Interface / User Experience

The user interface had been designed to be as seamless as possible. The idea being that by taking contextual usage information from a persons MP3 player or viewing habits the system would then aggregate this with others information providing a basis from which to form an edit list. Also a metadata table had been populated on the database that could be correlated against the event info so that the

genre of a track could be identified and tagged to an event.

One consideration came clear early on during development. What if the system chose wrong or there was an exception in the choice it made? We decided to add a like and dislike button which is accessed via a remote control mechanism. The controls appeared on the screen. The action would fire an event into the database so that this could be registered for future notation. (See Figure 5.10).

The next stage had been to ensure a good supply of content to the system. In the end the system contained over 35 digitized videos from various popular musical genres. In addition a set of interstitials and idents were filmed with an actress and later music and Foley added to make them appear professional. This would make the viewing experience a little less jarring for people who are normally used to high production values.

The final consideration was that if this is indeed a social system, as the implication is there by the fact that it is designed as a social (though not necessarily co-located) viewing experience, then people will want to do synchronous social things such as messaging. We added on screen messaging and also we added the ability to request a particular song, which would basically override the system and play that song if it is available. The messaging and request system were enabled via Java servlets on the program rendering MediaConnector node.

Other wise user interface cluttering and complexity had been kept intentionally low.

5.7 MBeat Testing and Observations

A group of six nodes were set up with one of the nodes acting as the rendering node, the node that would start of creating the show. During the show the number of the people and their combinations were changed and the resulting change in the editorial policy of the show monitored via a visualization system to show editorial policies.

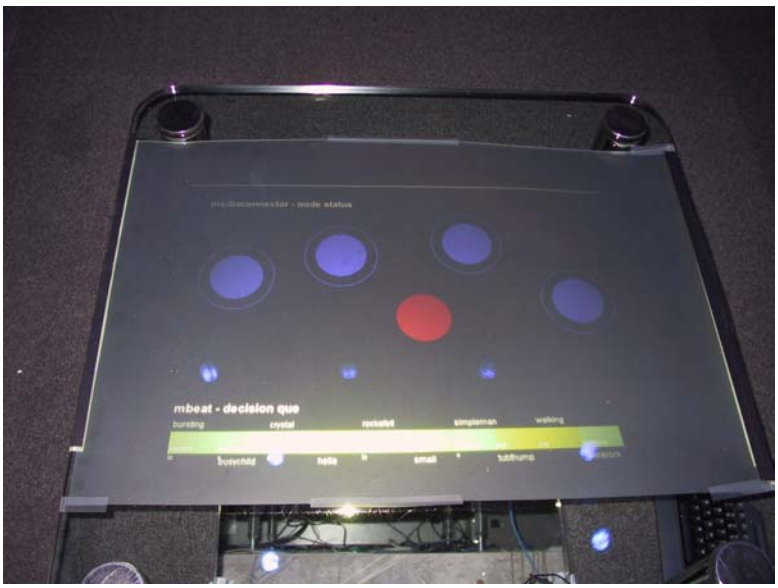


Figure 5.4 The MBeat visualization table, implemented to gain perspective into what decisions it is making. It aided debugging and also provided useful feedback to viewers.

The show performed as intended. Empirical results showed that the show did indeed cut the show according to group level profile and bias.

It had been decided to have only one playback engine as we only had regularly accessible Isis machine, but running simultaneously another MBeat beat client on another machine proved scalability and media transfer. In effect we had a playback engine with a six-strong data source for the audience, with each data source running on separate machines.

Even with 60 videos there is not enough content to prevent near term repetition of a song and that since the profiles were created and did not reflect real people, it is hard to say whether the show would have been enjoyable by such profiles.

However the concept and feasibility of a group level personalization of content had been deemed reasonable and the idea seemed attractive to potential participants.

Also the ad-hoc scalability of the system had been shown and this is mostly due to the JXTA architecture and the minimal message passing and distributed architecture for a media application.

Figure 5.5 The MBeat setting. Screen shows the presenter.

Figure 5.6 A closeup of the MBeat visualisation table

Figure 5.7 A music video playing

Figure 5.8 showing the MBeat logo from the ident.

Figure 5.9 A music video playing

Figure 5.10 A graphic ident. Note voting controls in lower left.

Figure 5.11 The presenter (Alicia Peyrano) making a link.

Figure 5.12 A music video playing – note voting controls in lower left.

Chapter 6 – Evaluation Design And Results

6.1 Chapter Introduction

To test the flexibility of MediaConnector and its ability to iterate quickly it was necessary to create another application. This would not only test if the framework performs but also whether the applications are engaging and can help extend shared experience somehow.

I will describe the creation of GardenConnector, which extends upon the group level dynamic personalization present in MBeat and applies it to another shared social situation. The idea is to create an ambient "Jukebox" that reflects the current tastes of the members of the social group currently present in the area. The evaluation has two components. They are 1) The MediaConnector framework and 2) The GardenConnector test application.

To better understand MediaConnector's effectiveness we have to evaluate it against its goals of constructionism, ad-hoc scalability and event auditing.

The GardenConnector had been evaluated on its design aims of being compelling for the audience at hand and acting as a social catalyst in the test area through shared experience.

The evaluation is occurring in a short period due to time limitations. Ideally the evaluation should be with a community of members using the framework over a long term (3-4 months), by constructing and extending with MediaConnector. The results presented are an indicative test of how in the long term MediaConnector would evolve an perform.

6.2 Experimental setting and idea

6.2.1 Section Introduction

The experiment target had been a social grouping by environment in the MIT Media Laboratory on the third floor. Known by its residents as "The Garden" it is a large open area enclosed by two person offices. At the time of writing the majority of garden residents are graduate students.

The daily activity is characterized by people often drift out of their offices to mingle and socialize. Often there are large groups of people moving through with demonstrations taking place. The environment has often been characterized as being noisy and so residents are not so prone to disruption by noise.

The relationships are mostly typified by being described as worker-colleague work relationship drive with a few good social bonds between some persons. As with most social settings, people are curious to know more about each other.

There had been a jukebox in the garden area before that had been basically a PC that played back digitized music. However due to past inequities in usage the system was take away and not used. People did comment that something like the jukebox would be a great idea if there were a mechanism to make access fairer and reflective of all participants' tastes.

The idea was to create a central jukebox client that would schedule and play peoples music. Individuals would have an MP3 player that they would use in normal usage. The MP3 player would set the last modified file attribute when it reads an MP3 file from a defined folder. An agent would watch when the file had been read and then would

transmit this info to the local database for storage. The jukebox then can poll any MP3 players (and hence users) present for information on recent choices played. The jukebox can then ask the player to send over a chosen track and the jukebox will then proceed to play the music. Hence it is a jukebox that will play music by participation. Community members have to engage to have their 'say'. Also the choice of music may help to reveal the tastes and preferences of the community of garden members to others.

The valuation period occurred over a period of four days, which maximized the time available.

6.2.2 Design of Framework Evaluation

The framework had been evaluated by building, deploying and iterating GardenConnector. I myself undertook the construction within the MediaConnector framework, as the likelihood of people having the drive and desire to engage the learning curve, however minimal, is limited if not unlikely (4 days). Instead the evaluation proposes that over a longer given period the likelihood of people taking the initiative will be great enough for them to overcome the barriers to entry, namely learning the API, for them to use the system. What we have to look at in the context of this experiment and its timeframe is whether the application generates new community needs and whether those needs can be fulfilled with reasonable cost. This will be gauged in the context of having to write a program from scratch in Java compared to scripting in Jython.

The proposed application and its design pattern allowed us to look at whether it scales and for us to postulate scalability on a massive application scale.

Looking at the audit databases and mining them for data provided us trend and behavior analysis.

6.2.3 Design of GardenConnector Evaluation

This is done qualitatively with a questionnaire at the beginning and some questions at the end of the study period. The combinations aims were to try and ascertain the following

- Did they enjoy the system and its iterations?
- What did they think of the system?
- Did they have initially any aspirations for the system?
- How well did they know people before and after the experiment?
- Did they think it has helped the community of the garden?

The questionnaire tried to quantify people's familiarity and knowledge of others in the garden area through a scale measurement system.

6.3 Design of the GardenConnector Application

The system consisted of the jukebox (of which there is one instance) and many instances of the client.

6.3.1 Jukebox

The jukebox was essentially a client running a MediaConnector JXTA instance. In the instance JXTA had been initialized to provide the file sharing layer and the CMS content manager, which would handle all file transfers. In addition this instance of a JXTA node would have its transports configured to act as a Rendezvous service. So that covered the ability to find other JXTA nodes (hence users) and to be able to handle the resolution of node addresses and content transfer from other nodes. Internal API calls made the extraction of nodes, their addresses,

and names much easier. This is essentially a pretty basic and standard instantiation in most circumstances with little or no modification.

The next stage had been to create the functionality that would implement the 'jukebox' facility for playback. First the following algorithm was designed

Repeat forever

Get list of peers out there

Pick one from random

Contact it for its most recently played track service

Download this title

Play it

The resultant code in Jython looked like this

```
import java
from com.wrox.ea.jxtashare import jythonintface

h = jythonintface() # instance our API into MediaConnector

while 1:
       peerlist = h.getpeerlist()  # get peers out there
       numberofpeers = len(peerlist)
       #normalise it to produce right range for array index
       numberofpeers = numberofpeers - 1
       index = -1
       index = h.random(numberofpeers) # Pick one at random
       peerchoice = peerlist[index]
       track = h.getonetrack(peerchoice) # get track
       h.downloadfile(track)
       h.delaything(10)
       trackplay = "c:/jukebox/" + track
       h.playMP3(trackplay)               #play it
       h.dellasttrack()                 #delete it
       #iterate again
```

It had been decided to add one more method to the interface in Java to implement the MP3 player, which was the following code in Java.

```
public void playMP3(String mp3name){

System.out.println("MP3 PLAYER CODE HAS BEEN CALLED");
try
        {
         // Open player
        String paramline = "java -classpath
c:/mediaconnector/javalayer02/classesjavazoom.jl.player.jlp \""+ mp3name +"\"";
//+ mp3name;
        try{
                // open up Javalayer and run it
                Process mp3player = Runtime.getRuntime().exec(paramline);
                                Thread.sleep(240*1000);
                mp3player.destroy();
            }catch(IOException e){    System.out.println(e); }


         }catch(InterruptedException e){System.out.println(e);}
}
```

It is done in Java so that it could be used again easily in the future. Also more importantly we had the benefits of thread control over the command line Java MP3 player. This is useful when a corrupt track or similar caused the thread to hang. And as we shall see later it became useful to quickly implement another feature.

Also each node needed a servlet to answer the query of which track had been played most recently.

```
public class onetrack extends javax.servlet.GenericServlet{
          public void service(ServletRequest req, ServletResponse res) throws ServletException,
IOException {

                        res.setContentType("text/plain");
                        ServletOutputStream out = res.getOutputStream();

                        // who is this data from? read name file and display
                        FileReader fr     = new
                        FileReader("C:/MediaConnector/servers/jo1_0b7/webapp/host/root/nodename/name.tx
                        t");
                        BufferedReader br = new BufferedReader(fr);
                        String thisnodename = new String();
                         thisnodename = br.readLine();

                        Connection con = null;
                        ResultSet rs;
                        Statement stmt;

                        try {
                              // find and instance the MM JDBC driver
                              Class.forName("org.gjt.mm.mysql.Driver");

                        }catch(ClassNotFoundException e)
                                    {System.out.println(e);
                        };

                        try {
```

```
                                con = DriverManager.getConnection("jdbc:mysql://18.85.45.22/mcdb");
                                stmt =con.createStatement();
                                 rs = stmt.executeQuery("select * from mc_transaction where
uid=\""+thisnodename+"\" order by lastplayed desc");
                                String str = rs.getString("name");
                                out.print(str);
                                stmt.close();
                        }catch (SQLException e){
                                System.out.println(e);
                        }

                }

}
```

6.3.2 The GardenConnector Client

The garden connector client consists primarily of an agent thread that watches a directory for new reads. The reads signify that the item has been accessed. Currently I have restricted its monitoring to one directory only but there is no reason why this could not be extended. The idea is to open the monitoring of it to all document and media types. The MediaWatcher class thread implements this. However in implementation it was discovered that Java does not access that information and in the end we modified a player to change a files last modified tag and adjusted our code to monitor that field. The method involving last read could be done via JNI implementations on native platforms or actual applications modified to pass that information to the MediaWatcher thread for processing.

6.3.3 Overall Structure

The structure of Garden Connector can best be described as follows by a diagram.

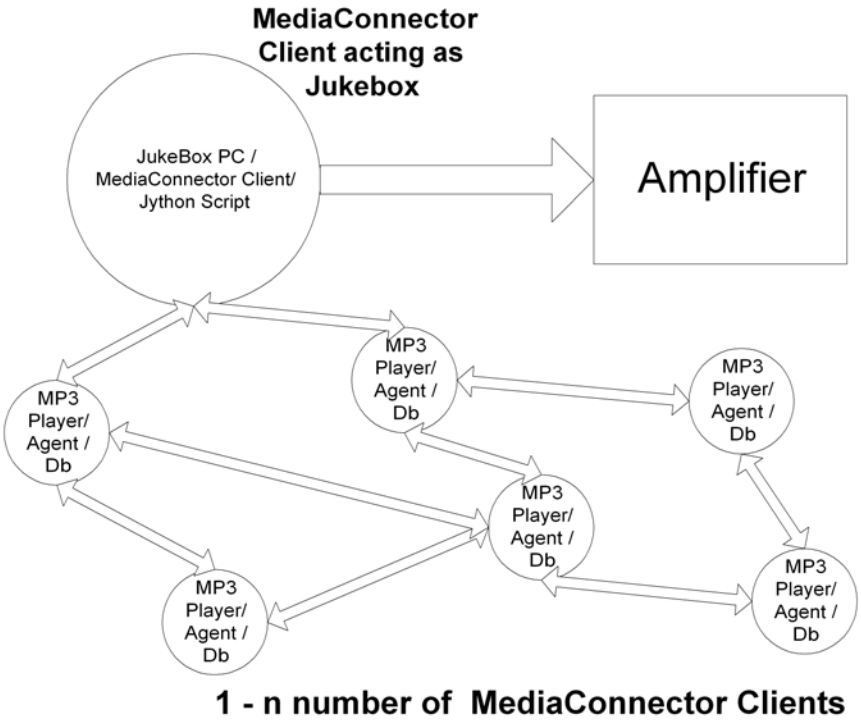


Figure 6.1 A diagram of the GardenConnector architecture.

Each circular node, regardless of size, represents one machine on a tcp/ip network. The architecture also provides a distributed content management system that minimizes data replication between the nodes and works out the most efficient way to transport data between any two points. That functionality is provided by the JXTA system.

6.3.4 User Interface and User Experience

The design also had to take into account what features the system should offer as user interface to its functionality.

Initially the design had been about minimalism. The system would collect contextual information and use this to help set the music play

list. People ideally would have just been using their favorite players and the agent would recognize and note this. However the limitations of Java forced me to modify a Java MP3 player to amend the last modified attribute, an attribute that is available. Hence initially the system had only one interface a Java MP3 player.

As my later [6.4.1] notes show the system did build some new elements that were requested and implemented through community need rather than design.

6.4 Framework Evaluation Results

The MP3 player application took only four hours to build and debug including hunting down a command line player. For a media sharing application the right modules were available to script. Essentially we needed modules for resource discovery and file transfer and those were available through scripting Java. The additional modules for the MP3 Player were not available immediately and those had to be written.

The second phase had been to debug the system. There were three parts to this, one was the actual Jukebox algorithm, the second was the file transport system and third had been the MediaConnector system itself internally.

The jukebox algorithm needed modifying. It became clear that the service had been repetitively playing the same song within a reasonably short time. This came to light especially if there were only a few clients active where it is likely that person would be selected again and also they may have not listened to anything new. The solution to this had been to implement another function that randomly picked a track from all of the currently available tracks from all the currently available clients and to modify the Jython jukebox code to

watch out for repetition. The final code is available in appendix [D]. Most of this modification had been done within half an hour.

The file transport system managed to prove itself to be unreliable when crossing different subnets. MP3 files that crossed over different subnets were prone to corruption making the tracks skip. Initially it had been thought to be a software playback problem but when the files were tested with other more robust commercial players they were found to skip similarly. This problem consumed a huge tract of time with no resolution as the problem lies within JXTA. However at the time of writing the main lead to resolving this is to move away from just TCP and use HTTP as a more reliable transport stream. This can be achieved two ways, either in JXTA or by extending the web servers to provide the file transport with two servlets to read out and read in file.

The third debugging task had been within the MediaConnector framework itself. Some errant coding was responsible for thread collapse when peer membership changed. This had been resolved easily by ensuring array size when passing between the Java and Jython namespace. There is still a small processor load spike in one of the threads. This is due to the way that JXTA builds unique hash keys of the files it is sharing. A way to minimize this is to reduce the frequency of its key building and potentially for a lot of applications it could be done only once.

For the period of the trial the system has worked well and most users are currently pleased with the system. Some of its functioning has been irksome with users most notably the corrupt transport causing MP3 files to skip. However this will be resolved.

6.4.1 Iteration

As expected the community engaged itself with the new system in its physical space. The space and its dual function as a recreational and as a working space caused some tensions and differences in needs to arise. Iteration and the desire to construct or extend upon the system was most definitely noted [appendix B]. There were suggestions made as to what functionality that they would like to see added to the system. Most notably some were the same requests from different people.

1) An 'Eject' button accessible by the web to allow a song that is not playing correctly or a song that is irritating to be dislodged. A 'Blacklist' to be set up of songs that receive persistent ejection.
2) The MP3's to play properly *all* of the time. At the moment about 10 percent skip regularly.
3) The system to play a song to its end and then immediately play the next one. At the moment a song is played for four minutes and then killed and a minute gap left in between. This is a coding choice made by myself to make thread control easier
4) Download and swap files
5) Ability to find out whose song is playing currently
6) Ability to tag songs with notes / recommendations
7) An ability to see who is logged in and tune into what other people are listening into

All of these requests could implemented in the following way.

1) An eject button: - kill the MP3 player thread
2) Improve skipping of audio – improve transport reliability, improve player reliability. Use a native code player for Jukebox machine.

3) Play end-to-end - Related to (1). Instead of timing a thread, wait till thread dies at end of track.
4) Download and swap files – this is trivial to implement as that is fundamentally what MediaConnector is designed to do and has a GUI that is currently disabled to allow this.
5) Whose song is playing – again trivial as the data exists for the decision made. Decision has to be made as to how to get that information to the user.
6) Ability to tag songs – Not so simple as there are many approaches to this. Best idea is a centralized repository of information that relates a song title to an information card.
7) Tune into others – Very possible by using download. I decided to implement a basic version that instead of playing the item would just return a list of the 5 most recent tracks played. If the person wants to download them they can via the GUI. Also a servlet has to be added to the servlets server to provide this information.

In the time available to implement and report these iterations I decided to prioritize the implementation of 1,2,3,4,5 and 7. I will implement 6 when more time is available or better still let a member of the community implement it as a future project.

(1) and (3) were solved by using a command line DOS mp3 player as the jukebox is an old Windows2000 machine with a Pentium II processor. (2) had been improved but not solved by switching transports within JXTA to HTTP. (4) Is enabled by switching on the GUI interface with a flag setting called from Java via Jython. (5) had been enabled by writing out a short HTML page to the web server and having it auto refresh every 10 seconds. (7) Is implemented by a similar method to (5). A servlet had been written that returned the last five tracks played, in fact a simple adjustment to an existing servlet

yielded the required results again showing the rapidity that re-suing code can provide.

6.4.2 Data analysis

One of the important elements of MediaConnector that had been designed to be in it is its ability to help infer behavior patterns and social networks via its database at each node.

To infer social networks we really should have had messaging provided as a service.

In the GardenConnector experiment we had the audit database enable to record the following items for a play of a track or a download.

- User id
- Track Name
- Timestamp
- Play or Download

In addition we can also have the information about the persons collection. This information structure can allow us to determine the following data analysis

- Person profile via collection
- Tracks most often listened to
- Tracks listened to by time
- Collection changes over time
- Collection elements common between persons
- Presence of people by time of day
- Most active listener
- Most active down loader
- Matching people through collection and behavior patterns

From the limited time of the experiment, we have limited data and as such the results are not likely to prove much in this area. To have better results we require larger data sets so that we can filter results for the noise and gain long term perspectives that would only appear in large sets.

However the analysis did prove that certain of the above can be defined always, such as most active listener and downloaded. The others can be analyzed as well but it is unlikely that we will see any significant patterns over the short term, which will yield any insight into the behavior. Results and analysis are included in appendix [C].

6.5 Application evaluation results

Even before the project was deployed people were excited by the proposition as laid out initially. People were keen to engage with the system and the concept laid out.

Generally the feedback on the system and its functioning was very positive and well received. Even those members of the community who were not actively taking part have found it interesting and pleasing. An effect of the short trial has been to attract those who were initially skeptical into using the system and people are being added as of writing.

The ambient reflection of one's taste in music as a side artifact of their normal behavior is compelling as is the way that the system can keep changing as they do. They do not have to maintain a central directory of MP3's and maintain them as the underlying system takes care of it.

Users generally described the system as “cool” and “really neat”. [see Appendix B for real user feedback] One of the most common wishes was that it worked more reliably. When it wasn’t working properly it would be jarringly obvious to those in the area. It is interesting that these “problems” instigated the most response and desire for iteration. The reader is referred to appendix B for further detail and actual feedback.

People managed to connect more through the discussion surrounding the new stimulus, whether it had been to disagree about the music being played, or finding they had previously unknown similarities in taste or wanting to own the music being played. Discussion would soon turn to “who’s that music from” which eventually led to the need to develop some of the additions described above.

People were keen to extend the system by suggesting additions and this is good. Attempt to try and ascertain whether people would like to build with it produced varying results with some liking the idea a lot and some not really being concerned with that aspect at all.

People learned more about other people and their tastes and will continue to do so as the system runs on and evolves through additions.

6.6 Conclusion

With MediaConnector we created an application that was engaging to the community and were able to do it in a few hours. The framework was flexible enough to show iteration and implementation of ideas within a fairly short time. Both the application and the framework show future promise. There are issues to solve with the framework’s technical performance but these can be expected of any new system in the time given for its development and evaluation. It has proven itself

to be malleable enough to create new applications very rapidly and flexible enough to be extended as need has required. Whilst there is a core library of functionality available it is clear that the vocabulary needs to be extended and it looks likely that this will indeed happen through community support.

It is also likely that MediaConnector is only suitable for an application class or design pattern centered around sharing media either centrally or peer to peer, but that should be clear to most people who view it for what it is. If the domain is likely to stay within this area then two things will happen. 1) The generation of a vocabulary of modules that will quickly yield the majority of functionality that is required and 2) further advancements in scripting via Jython and the appearance of script libraries that can be used as design patterns for re-use.

It is hard to say whether the evaluation by myself of its constructional ability is accurate enough as there are many factors clouding the clear resolution of an unambiguous result. A better evaluation would occur over a period of months with a community supporting itself. However viewing the experiment with myself playing a role that could easily be played by another, then we see that the proposition of a constructionist framework for sharing has possibilities for succeeding. Opening up access to varying levels of technical proficiency would have greater effect on its evolution and spread.

My own personal reflections are positive and I offer them here with the caveat that they are my own experiences. Having music in the garden improved the ambience and atmosphere considerably. People were motivated to talk to one another and also any bad music acted as a catalyst between people to talk. Being the owner of the project I was the attention of some well-meant and jovial abuse, and this was good

as I turned it around into interactions between other people. Rather by accident than design I finally got to know the people in the garden much better and even found points of shared interest with people. I was amazed by how much people wanted to engage with something that offered social bonding. We had a common cause and no one was too upset at the flaws. Rather they wanted to resolve them so that the system would function as they had intended and then add more features.

Abstracting slightly, we see that the system is a set of web servers that can be scripted and have peer / resource discovery. It is the latter that is the critical part of this framework. JXTA has shown itself to be both a boon in its functionality and a hindrance in its complexity and performance. It is likely that JXTA will remain for resource discovery and that other methods be employed to take some of its unreliable features offline. The system by being written in Java can be made multi platform with some minor effort. With Jython we have a powerful weakly typed language that makes it easier for people to learn and experiment with the system. Their only constraint is that brought about by reassembling modules of higher-level functionality rather than coding from the base up. However the rapidity factor balances up the proposition.

In short the MediaConnector framework has shown good potential as a constructionist framework for the creation of media sharing applications. The next stage is to see how it evolves in the future with communities that embrace it.

Chapter 7 – reflections and future directions

With MediaConnector we were able to create a media sharing application in a matter of a few hours and then easily iterate to the demands of the community using it. The contribution of MediaConnector has been to show that a constructionist environment for developing media sharing applications is a sustainable idea. The peer-to-peer architecture scales well and is intuitive in most peoples minds as a network structure. Communities should be empowered to create and modify their own applications without an education in computer programming. Even if MediaConnector is not the ultimate system for such efforts then something that one-day fulfills that need is required. Allowing people of varying levels to contribute and alter their clients to suit their needs will allow a greater range of use cases to be satisfied. Secondly the thesis shows the potential for integrating further the fields of social network analysis and behavioral monitoring as ways to provide new metadata for new and novel applications to emerge. It is the whole, not the sum of the individual parts that is important.

Suggestions for the future direction include

1) A formal evaluation. Ethnographic studies and formal evaluation of the framework should take place. The community should be committed and support provided to help maintain momentum.
2) Creating a central forum. Open source projects from Linux to ping – clients have commonly created forums to allow the community to support itself and resolve issues.

3) Allow viral distribution of the framework by having an open source license policy and distribution.
4) Creation of applications where social networks can be resolved. Essentially they will need messaging to allow networks to be identified and gauged. Again events should be added to the database.
5) Optimization of the code underlying MediaConnector, currently occasional bursts in computation can cause 100 percent CPU load. Reliability is good but open for improvement.
6) Jython as the scripting engine had been a choice made given time constraints. It would be good to explore once again the creation of a custom language for MediaConnector to use. The language should be as far as possible user extensible as well.
7) Explore further the use of XML to create multi media documents that are scriptable using Jython or similar. One potential is to look at integrating the Water programming language [57] into the system.
8) Take this one further and create a visual programming language that can be parsed into XML and hence into MediaConnector. This is a broad and very speculative route of inquiry.
9) Exploring the use of Jython with servlet engine, something which is possible, but not yet fully explored.

I also propose that we merge elements of CSCW and Computer Mediated Communication research for a specific new area – 'Computer Supported Social Entertainment Applications' to create a very multidisciplinary approach to the architecting and design of social applications. From online gaming, to media sharing applications to SMS messaging, these applications are primarily designed to connect *people* together, *not computers*.

References

[1] Orwant, Jon, Doppelganger Goes To School: Machine Learning for User Modelling, 1993, September, Cambridge, MA, Massachusetts Institute of Technology

[2] Orwant, Jon, For Want of a bit the user was lost: Cheap user modeling, IBM Systems Journal, 35, 3&4, 398 – 416, 1996

[3] Bollen, Johan, Evaluation of Digital Library Impact and User Communities by Analysis of Usage Patterns**,** *D-Lib Magazine*June 2002, Volume 8 Number 6, ISSN 1082-9873 http://www.dlib.org/dlib/june02/bollen/06bollen.html

[4] Bollen., Johan, Group User Models for Personalized Hyperlink Recommendations. *In LNCS 1892 - International Conference on Adaptive Hypermedia and Adaptive Web-based Systems (AH2000),* pages 39-50, Trento, August 2000.

[5] Messerschmitt, David G. *Understanding Networked Applications: A First Course* ,Morgan Kaufmann Publishers, Sept 1999

[6] Maes, Pattie and Shardanand, Upendra, Social Inofrmation Filtering: Algorithms for Automating "Word of Mouth", *Proceedings CHI 1995*

[7] Upendra Shardanand, "Social Information Filtering for Music Recommendation", *MIT EECS M.Eng Thesis*, Learning and Common sense group, MIT Media Laboratory, 1994.

[8] Special Issue on Information Filtering, *Communications of the ACM*, Vol. 35, No. 12, December 1992.

[9] Scott Deerweester, Susan Dumais, George Furnas, Thomas Landauer, Richard Harshman, ``Indexing by Latent Semantic Analysis", *Journal of the American Society for Information Science*, Vol. 41, No. 1, 1990, pp. 391--407.

[10] Yan T.W, Jacobsen, M., Garcia-Molina, H. & Dayal, U (1996) From user access patterns to dynamic hypertext linking. *Computer Network and ISDN systems*, vol 28, 1007 –1014.

[11] B. Trousse, M. Jaczynski, R. Kanawati, Using User Behavior Similarity for Recommandation Computation: The Broadway Approach*, 8th international conference on human computer interactions (HCI'99)*, Munich, August, 1999.

[12] Kolodner, Janet. *Case-Based Reasoning*. Publisher San Mateo: Morgan Kaufmann, 1993

[13] Kim, Youngmoo and Brian Whitman (2002). "Singer Identification in Popular Music Recordings Using Voice Coding Features." (ISMIR2002)

[14] Whitman, Brian and Paris Smaragdis (2002). "Combining Musical and Cultural Features for Intelligent Style Detection." (ISMIR2002)

[15] Logan, Beth and Salomon, Ariel. A Music Similarity Function Based on Signal Analysis *2001 IEEE International Conference on Multimedia and Expo (ICME2001),* August 2001

[16] Lieberman, Henry Rosenzweig, Elizabeth And Singh, Push , Aria: An Agent For Annotating And Retrieving Images, *IEEE Computer*, July 2001, pp. 57-61

[17] Webseek, visual search engine tool http://disney.ctr.columbia.edu/webseek/

[18] J. R. Smith and S.-F. Chang. Multi-stage Classification of Images from Features and Related Text, Proc. Fourth DELOS workshop, Pisa, Italy, August, 1997

[19] Kolodner, J. Case Based Reasoning. Morgan Kaufmann, 1993

[20] Piaget, Jean (1954). The Construction Of Reality In The Child. Ballantine Books, New York, 1954

[21] Von Glaserfield, Ernst, Sensory Experience, Abstraction and Teaching. In Constructivism in Education, Lawrence Erlbaum Asociates Hillsdale, NJ 1994

[22] Papert, Seymour, A., Instructionism vs. Constructionsim. In The Childresn Machine, Basic Books, New York 1993

[23] Resnick, Mitchel, Bruckman, Amy, & Martin, Fred, Pianos Not Stereos: Creating Computational Construction Kits. *Interactions* , Vol 3, No. 6, September / October.

[24] Pinkett, Randal D. (2000). *Bridging the Digital Divide: Sociocultural Constructionism and an Asset-Based Approach to Community Technology and Community Building.* 81st Annual Meeting of the American Educational Research Association (AERA), New Orleans, LA, April 24-28 2000

[25] Chesnais, Pascal, Canard, 1st International Symposium on Wearable Computers (ISWC '97)

[26] Haase, Kenneth, FramerD:Representing knowledge in th large, IBM Sytems Journal, 35, 3&4, 381-397, 1996

[27] Ultima Online gaming environment http://www.uo.com/, 2002

[28] Feller, Joseph, and Fitzgerald, Brian. A framework Analysis of the Open Source Software Development Paradigm, Proceedings of the twenty first international conference on Information systems , 2000 , Brisbane, Queensland, Australia

[29] Dieberger, A., Dourish, P., Hook, K., Resnick. P, and Wexelblat A. Social Navigation. Techniques for Building More Usable Systems. ACM Interactions, November/December 2000

[30] Janse, Maddy D, Stienstra, Marcelle and De Ruyter,Boris. Increasing the Television Experience with Shared Media. AEI Conference on Media Futures, Florence 8-9 May 2001

[31] Jain, R, "Digital Experience", Communications of the ACM. Vol.44,pp.38-40, March 2001.

[32] Curtis Eaton, B, Pendakur, Krishna, Reed, Clyde G. Socialising, Shared Experience and Popular Culture. Simon Fraser University, Dept Of Economics, Discussion Paper May 2000.

[33] Codd, E.F. *A Relational Model of Data for Large Shared Data Banks.* Communications of the ACM, 13(6): 377-387. 1970.

[34] Rao, B.R. *Object-Oriented Databases: Technology, Applications, and Products* McGraw-Hill. New York, 1994.

[35] FastObjects by Poet http://www.fastobjects.com/

[36] Smith, B.K., Blankinship, E., Ashford III, A., Baker, M., & Hirzel, T. (2000). Image Maps: Exploring urban history through digital photographs. In T. Ishida & K. Isbister (eds.), Digital Cities: Technologies, Experiences, and Future Perspectives (pp. 326-337). Berlin: Springer-Verlag.

[37] Nielsen Net Ratings , Report - Global Internet Trends Quarter 1 2002 , Stamford CT May 9 2002

[38] Peretti, Jonah – My Nike Media Adventure – The Nation, April 9th 2001

[39] Borovoy, R., Martin, F., Vemuri, S., Resnick, M., Silverman, B., and Hancock, C. "Meme Tags and Community Mirrors: moving from conferences to collaboration", in Proceedings of the ACM 1998 conference on Computer supported cooperative work. pp 159-168

[40] Peretti, Jonah - Culture Jamming, Memes, Social Networks, and the Emerging Media Ecology - The "Nike Sweatshop Email" as Object-To-Think-With – Online Essay, http://web.media.mit.edu/~peretti/nike/index.html

[41] in R. Fabian, ed., Revised English version of "Mach und Ehrenfels: Über Gestaltqualitäten und das Problem der Abhängigkeit", *Christian von Ehrenfels. Life and Work,* Amsterdam: Rodopi, 1985, 85-111

[42] Henry Lieberman, Elizabeth Rosenzweig And Push Singh Aria: An Agent For Annotating And Retrieving Images, IEEE Computer, July 2001, pp. 57-61

[43] Out of Many, One: Reliable Results from Unreliable Recognition, in ACM Conference on Computer-Human Interface (CHI-2002), Minneapolis, April 2002.

[44] Terveen, L., Hill, W., Amento, B., McDonald, D., Creter, J., PHOAKS: A System for Sharing Recommendations, Communications of the ACM, pp. 59-62, Vol. 40, no. 3, March 1997

[45] Paul Resnick et al. "GroupLens: An Open Architecture for Collaborative Filtering of Netnews", Internal Research Report, MIT Center for Coordination Science, March 1994

[46] Brian Whitman, Gary Flake, and Steve Lawrence. Artist detection in music with minnowmatch. Proceedings of the 2001 IEEE Workshop on Neural Networks for Signal Processing, September 2001

[47] Bruckman, A and Resnick, M. "Virtual Professional Community: Results from the MediaMOO project", Proceedings of 3CyberConf, The Third International Conference on Cyberspace, May 1993

[48] Wellman, B. and Gulia, M. Net Surfers Don't Ride Alone: Virtual Communities as Communities. In Kollock, P. and Smith, M. (Eds.) Communities in Cyberspace: Perspectives on New Forms of Social Organization. Berkeley: University of California Press, 1997

[49] Bruckman, Amy (1997). "MOOSE Crossing: Construction, Community, and Learning in a Networked Virtual World for Kids." PhD Thesis, MIT Media Laboratory 1997

[50] Ousterhout, J. K. *Scripting: Higher level programming for the 21st century*. IEEE Computer (Mar. 1998).

[51] J. Kleinberg. "*The small world phenomenon: An algorithmic perspective*". Proceedings of the 32nd ACM Symposium on Theory of Computing, May 2000

[52] Granovetter, M. The Strength Of Weak Ties: A Network Theory Revisited, Social Structure and Network Analysis, edited by Marsden, P. and Lin, N. Beverly Hills: Sage, 1982, (105-130).

[53] Agamanolis, Stefan, Isis, Cabbage, and Viper: New tools and strategies for designing responsive media, PhD dissertation, Massachusetts Institute of Technology, 2001

[54] Gelertner, David, The Aesthetics Of Computing, Phoenix Publishers,1998

[55] Ousterhout, J. *Additional Information for Scripting White Paper*, http://home.pacbell.net/ouster/scriptextra.html

[56] Li, Sing, Early Adopter JXTA, Peer To Peer Computing with Java, Wrox Press, 2001 ISBN 1-861006-35-7

[57] Plush, Mike, Fry, Christopher, Water Programming, Published by Clear Methods ISBN: 0972006702

[58] Finke, Jon - Rensselaer Polytechnic Institute Monitoring Usage of Workstations with a Relational Database, Proceedings of the Eighth Systems Administration Conference (LISA VIII) (USENIX Association: Berkeley, CA), 1994

[59] Rodden, Kerry, University of Cambridge Computer Laboratory, How Do People Organise Their Photographs?, Proceedings of the British Computer Society IRSG 21st Annual Colloquium on Information Retrieval Research, April1999

[60] Evans, Ryan George, LogBoy Meets FilterGirl: A Toolkit for Multivariant Movies. Master's thesis, MIT Media Laboratory, February 1994

[61] Fitch, Stephan, Cinema Server = s/t (story over time), An Interface for Motion Picture Design. MIT Media Laboratory MS Thesis 1992

[62] Pan, Pengkai, I-Views: a Storymaking Community of, by and for the Audience. MIT Media Laboratory MS Thesis, 1999

Appendix A: MediaConnector API

A.1.1 Java API

The core of the application was built using a template created by Sing Li of Sun Microsystems for his book Early Adopter JXTA[56]. This was a JXTA template that uses the CMS subsystem of JXTA to handle file discovery and transfer. If you need to get to the level of actually handling advertisement publishing and advert expiry control then you are better off writing from scratch. If you want to extend an existing java application with a P2P facility then this is an option. This api basically modularizes and makes some of the JXTA stuff easier.

A.1.2 Performance Notes

The content sharing thread currently has a processor load spike everytime it runs. This is due to the thread building unique file object keys by opening them and using the content. This is used to verify what is and what is not on the list. The code can be improved by using a better algorithm for storing and adding changes. The code is in the FinderThread class . Basically, the bigger your files and the more files you have then the less often you should run FinderThread. With most applications its fine at the current setting.

On large file transfers there is data corruption. This manifested itself in our trial as skipping MP3's. The problem lies in JXTA and is needs attention.

A.1.3 API Listing

```
class ShareFrame extends JFrame
```

```
public void RunJythonScriptingEngine(String
filename_and_path) throws PyException
```

This calls the Jythons scripting engine. Pass it the script to process and complete path. Default is ./ .

```
public void startMMmonit(String folderpath)
```

This is an agent that monitors a given folder for media usage activity. This call starts the MediaMonitor thread. This thread is designed to look for last accesses of files based on the last modified attribute of the file. It is up to your programs that work with the files to modify this. Future improvement could use JNI to call and get the last accessed attribute, which opens up functionality to other common programs.

```
public void startJxta()
```

Yes, this starts JXTA subsystems and gets the whole thing started. It creates a default group Medialab and joins its. To create and change groups use the following (see groupmanager as well). There is no authentication presently though that facility is available as well through JXTA.

```
GroupManager manager = new GroupManager(group,group);

PeerGroup p = manager.addGroup("ANotherGroupName","net.jxta.impl.membership.NullMembershipService"," Null Membership Service",null,false);
```

```
public void startWebserver()
```

Starts the Jo! Webserver.

```
public void startMonitoringPeers()
```

Starts the peerMonitor thread that starts to listen out for and regularly monitor which peer nodes are still functioning and which are not. I.e who is there and who is not.

```
public void EasyStart(boolean windowflag, String sharefolder)
```

Yes the easiest way to start JXTA services. One call and that's it. Specify the sharing folder and whether you want the filetransfer drag and drop window to appear. E.g

```
Shareframe.EasyStart(true, "c:/mc_media/");
```

public class **CallingCardServo** extends Object

public boolean **writeCard**(String filepath)

This makes the node publish its IP address as an XML file that the JXTA CMS will share with the others. Basically think of it as a presence flag.

public String **StripIPaddressCard**(String cardfile)

public String **StripIPaddressCard**(String cardfile)

This will read in an XML file or string and return the IP address as a string

public class **DbServicesMYSQL** extends Object

public boolean **excSQLquery**(String sqlquery)

This will execute a one shot SQL query with the MCDB database in MYSQL. Returns true for success and false for failure

public ResultSet **excSQLquery2**(String sqlquery)

This will execute a one shot SQL query with the MCDB database in MYSQL. Returns a resultset object contains all the output from your query

public class **GroupManager**

public void **leaveGroup**()

Causes node to exit current group.

public void **purgeGroups**()

Causes a node to purge all group advertisements it has stored.

```
public Credential joinGroup(PeerGroup
   newGroup,StructuredDocument credentials, String
   authenticationMethod) throws
   GroupManagerAuthenticationException,
   ProtocolNotSupportedException
```

Causes the node to join the given group. See example in shareframe class.

```
public void getRemAds()
```

Causes a message to be sent to peers to return GroupAdvertisements from the DisocveryService.

```
public void JoinGroupByGroupID(PeerGroup
parentgroup,PeerGroupID grpid)
```

```
public PeerGroup addGroup(String groupName,String
   groupMembershipClassName, String groupDescription,
   ModuleImplAdvertisement advertisement, boolean
   conditional)
```

public class jythonintface

This class is the main way that the system is exposed in Jython. However you can still use it in Java as well.

```
public Vector getpeerlist()
```

This method returns a Vector (a list in Jython) that represents all the nodes currently still responding.

```
public void delaything(int secs) public static CMS
```

This is a delay. Waits for n seconds.

```
public int random(int sizeseed)
```

returns random number between 0 and sizeseed.

```
public void downloadfile(String filetodownload)
```

Given a string of the file it instructs the CMS to find it and download it into the share directory.

```
public String getrandomtrack()
```

Selects a random MP3 file from the global CMS list.

```
public void playMP3(String mp3name)
```

Plays the given MP3 file in a javalayer 020 player.

`public Vector` **`files_on_offer`**`()`

Gets the complete CMS files global list as a list.

`public void` **`displayURL`**`(String url)`

Opens a browser with the URL specified.

`public class` **`MediaWatcher`** `extends Object`

`public Vector` **`ScanNewReads`**`(String directory)`

Returns a vector of files in the directory and their last modified dates.

`public Vector` **`ScanNewReads3`**`(String directory, Vector mainvec)`

Returns a vector of files that have changed given a comparison list. Use the ScanNewReads method to create a base list then feed it to this function.

`public class` **`NameGetObj`** `extends Object`

`public String` **`getname`**`()`
Returns the name of this node. Set in a txt file in the nodename folder of the web server directory.(name.txt). A servlet also can set this file or it can be set manually.

`public String` **`getnumber`**`()`

Returns the number of this node. File located with name.txt in same folder (see above).

`public class` **`NetUtilz`** `extends Object`

`public String` **`thisNodeAddress`**`()`

Will return this nodes address.

`public boolean` **`SendURL`**`(String purl)`

Will send a URL. Useful for name value pair transmission to CGI scripts

```
public boolean NodeThere(String passIP)
```

It will ping an IP address to see if that JXTA node is still functioning.

```
public class SharedListModel extends
AbstractListModel implements SearchListener
```

```
public SharedListModel(PeerGroup pg)
```

This will create the base level list for the CMS instance that is attached to the given group.

```
public Vector listOfPeerCards()
```

This will return a list of peer-cards that it can see via the CMS.

```
public Vector listOfContent()
```

This will return a list of files that the CMS can see for the group.

```
public class XmlServo extends Object
```

```
see also NanoXML documentation.
```

```
public XMLElement read_and_parse(String
xml_file_and_path) throws Exception
```

Returns an XMLElement tree given a path to a file.

```
public boolean writexmlfile(XMLElement intree, String
filename) throws Exception
```

Will write an XML file to the given location given a tree of nodes of type XMLElement.

```
public XMLElement add_a_node(XMLElement intree, String
name, String key, String value, String datavals)
throws Exception
```

```
public XMLElement modify_a_node(XMLElement intree,
String name, String key, String value, String
datavals) throws Exception
```

Appendix B: Observations From the GardenConnector application

B.1.1 Overview

This section presents some of the material and user feedback from the experiment. An example initial questionnaire is included as is some of the responses from it.

The questionnaire was designed to try and provide a quantitative reasoning on how well people knew each other before and after the trails period , how often they listened to music and got recommendations, and also qualitatively on what people expected from the system. With the quantitative section the intention was to gauge whether people playing their taste in music would lead to others having better understanding of their personalities. The quantitative question is still included here even though a follow up had been cancelled due to probable inaccuracies in participants being able to quantify their relationships with people. A colleague advised me that this is likely to yield varying results according to peoples mood and for my given sample set of 10 people and the period of time it was not likely to be indicative and accurate.

B1.2 Sample Questionnaire

Questionnaire 1

This questionnaire is completely voluntary. Please answer questions that you wish to and leave those that you do not. Your name is not related to the answers below.

Q1. On a scale of 1 to 10 rate your level of awareness and frequency of contact with the following people. Make your best judgment using the following guidelines.

For *awareness*

The scale is approximately as follows

1- I don't know them at all
2- I know of their name
3- I can relate the name to the face
4- The above and I can acknowledge them with a nod or smile
5- I know them well enough to say hi
6- I know them and converse
7- I know them well enough to make conversation
8- I know them well enough to have a good conversation
9- We are friends and know a fair amount about each other

10-We are very good friends and know each other well

For *frequency of contact* (email or face to face)

1- I do not make communications with them
5- I make communication with them occasionally or in passing (once or twice a week)
10 – I make frequent intentional communication with them (more than once a day)

Name	Familiarity	Frequency of Communication
Darren Butler		
Tom Gardos		
James Seo		
Parul Vora		
Jim McBride		
Wilfredo Sierra		
Ivan Chardin		
Gerardo Vajello		
Sean Wheeler		
Vidya Lakshmipathy		
Stefan Marti		
Yi Li		
Sunil Vemuri		
Aisling Kelliher		
Chris Lyon		
Tim Gorton		

Q2 .How often do you listen to music on your workstation?

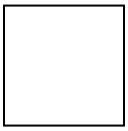
1- Never
5- fairly regularly (for a period every few days)
10-Almost constantly whilst I am using the workstation

Q3 How much would you like to have music playing in the general garden area?

1 – Not at all, it's a bad idea
5 – Neutral
10 – I would like that very much

Q4 How often do you pick up ideas for new listening or media in general (films, radio, events etc) from the people in the above list?

1- Never
5- Ocassionally and infrequently
10- Very frequently, almost always



Q5 Describe how you feel about the idea of a Garden Jukebox…

Q6 Can you think of features that you might like implemented with it? Broad level descriptions are fine.

Q7 Is there any other type of media sharing or messaging application that could be useful for the community in the garden?

B1.3 Results and Analysis

B1.3.1 Answers

Q1- The initial survey returned some interesting results. As expected the quantitative analysis showed that people knew a subset of the whole community well and the others whilst familiar were not known well.

Q2 – The average score was 7.86 . Most of the people who volunteered for this experiment are actually large consumers of music (scoring10) and a few were average or median consumers(5or 6). Basically most people listened to music often and had a large collection of MP3's.

Q3 – The average score was 7.58. Most people were very enthusiastic and a fewe were neutral about having music in the garden area.

Q4. – The average score was 6.0. There was a wide range of feedback here. Some individuals had listed 10 and the lowest had been 2. Most people middle around the 5.0 mark, occasionally picking up recommendations for new items in the garden area from other people.

Q5. – Typical responses were enthusiastic. Correlated against the score in Q3, people also now expressed caution at the system being dominated and with it interfering with those people who did not want it. Responses include

"It's great as long as it maintains a balance between users and doesn't get dominated by 1 or 2 individuals (especially if they love techno)."

"Lets try It"

"Good but possibility of being distracting / annoying (I work in the public garden space all the time)"

"I think it's a good idea. We should go for it"

"I would appreciate it more when people are congregating in the garden as opposed to working. Although I love to listen to music while working, others don't and I wouldn't want them to shut themselves in their office because of it."

"Despite some record of misuse*, a DJ can frequently be entertaining and sets the mood for social use of the space"

* = this refers to the old jukebox system in the garden and it misuse.

Q6. This section got a few responses or was left blank. Responses included. The section indicates that people have desire for the system and for it to evolve beyond the stated functionality.

"User stop lists, Play list construction, recommendations, "Mood matching music" play list, voting mechanism"

"Volume control. Somewhere where you can control it."

"Seeing who is on and what they are listening to"

"auto play play mode where I can listen to a carbon copy of another's play list, etc in real time or asynchronously. Access to other peoples libraries."

"Privacy control, ability to see and download current song, ability to kill / current song"

" A web interface to view available tracks and upcoming tracks. Using online music databases such as CDDB would allow tracks to be categorized, allowing the style of music to be modified for the time of the day (who wants heavy metal first thing in the morning after a late night out). CDDB would also allow info to be retrieved about the current track."

Q7. This section tried to establish whether they could think of any applications that they would want to see built that were not related to the JukeBox idea but still had a social theme. Responses were

"How about a Boss tracker /early warning system so we know when to stop messing around. It would also be useful to know when free food was available"

"Information of presence…. Where you are, if you're out of town, how long you've been gone for, when you're coming back, etc"

"would be nice to know who is into what music and easily find who has been listening to what"

" A system allowing messaging to the garden at specific time, with no caching or message history maintained"

"The TV should show whose track is playing. We should be able to vote music off. We should have some sort of messaging as well with this"

B1.3.2 Observations

I think the observations are

- People want the jukebox
- People have well defined small social groups with strong ties and extended social group with weaker ties
- They don't want the jukebox to infringe upon with people. Democracy is a must.
- They want to know more about other community members
- They want to see it evolve / action change to it. This indicates some precursors for constructionist behavior.
- They want applications to "connect" more and message more with their social community members.

B1.4 Iteration requests

B1.4.1 Brief

Participants were informed as they joined that they could email me at any point with requests for what they would like to see implemented. Some were also communicated to

me verbally. Ideally it would have been great to see them implement their own desires but in the given time frame and with incomplete documentation I decided to act as the implementation agent.

B1.4.2 Emails and Requests

Some examples

```
Date: Wed, 14 Aug 2002 14:52:05 -0400
From: xxxxxxxxxxx
To: surj@media.mit.edu
Subject: Ideas for the Juke Box

    [ The following text is in the "iso-8859-1" character set. ]
    [ Your display is set for the "US-ASCII" character set.  ]
    [ Some characters may be displayed incorrectly. ]


Hi Surj. My idea is that what if (as a member of Media Lab) upload all my
mp3 files to a common server, and anyone else with the same privileges do
the same thing. If say, xxx has music that I already have, then it would not
play it again (avoid repetition!) and... anyone who is a member of this
service could be selected randomly by the juke box.

Ok... here is the whole idea: I upload my music once. xxx, xxxxx and anyone
upload their files at the same directory, and whenever I play my music...
the server will identify the song that has been palying all day long, and
may be are similar to xxx or xxxxx. So... instad of uploading every
time, you do so only once. The system must compare and then play the most
popular song.

I know this sounds quite complicated... If you have questions about my idea
please I'll be deligthed to make it clear.
Hope this can contribute to the ML Juke Box Community
_____________________
xxxxxxx xxxxxxx
MIT Media Laboratory
(617) 253-xxxx



Date: Fri, 16 Aug 2002 11:35:42 -0400 (EDT)
From: xxxxxxxxxxxx
To: surj@media.mit.edu
Subject: jukebox suggestion

we need to be able to "veto" really awful songs anonymously, so that truly
awful music doesn't continue to play. perhaps once a couple of people veto
the song it stops playing, and if it gets stopped X times it gets
blacklisted?

also the songs need to stop skipping.

xx


Date: Wed, 14 Aug 2002 15:38:07 -0400 (EDT)
From: xxxxxxxxxxxxxxxxxx
To: Surj Patel <surj@media.mit.edu>
Subject: mediaConnector request.


dear mediaConnector technician.

i have started using your new tech-savvy media player.  i was wondering
```

whether i could see what other users were logged on and what they were presently listening to. also, is there a way i can set my player to just listen to whatever else someone else chooses? psuedo deejay/radio stlye? or at least get at songs that they have in their collection so i can be a copy cat?

i think these would all be valuable additions to your already rocking software.

xxxxxxxxxx

p.s. my mp3s always skip when i play them. why?

Date: Fri, 16 Aug 2002 13:50:53 -0400 (EDT)
From: xxxxxxxxxxxxxxx
To: surj@media.mit.edu
Subject: while I'm requesting features

how about a link on that web page to download the currently-playing song (and maybe the last one that played so we don't have to click so fast)

xxxx

Date: Fri, 16 Aug 2002 14:22:34 -0400 (EDT)
From: xxxxxxxxxxxxxxxxxx
To: Surj Patel <surj@media.mit.edu>
Subject: media connector suggestions.

dear media connector tyrant.

when i worked at the radio station, each ceedee had a sticker that unintentionally provided deejays with room to leave comments on certain songs or the album in general. half the fun of playing an album you hadn't heard was using the comments to help you decide what to listen to......it would be nice if people could leave comments on songs in your system...suggestions, recommendations for people that like their other stuff...but to prevent it from becoming a generic bulletin board, perhaps commenting should be limited to the owner, or some other such guideline/limitation.

an avid mediaconnector us

er.

After an addition had been implemented

Date: Mon, 19 Aug 2002 18:31:20 -0400 (EDT)
From: XXXXXXXXX
To: surj@media.mit.edu
Subject: "kill" function

surj,

the ability to kill a currently-playing song is a great addition. just a few minutes ago I was talking to several people in the garden and a song came on which we all reacted negatively to. so I pulled up the bookmark and it stopped. it's a very good way to keep the system unobtrusive in the public space.

although I had suggested some voting mechanism for killing songs, the single objection fits well with what you had said at the beginning about anyone being able to turn down the volume, etc. perhaps after several kills it could take a song out of the set being played....
XX

B1.4.3 Analysis

The emails show string desire and aspiration from the community to tailor their experiences and media consumption. This is a good indicator that uptake of tools may be likely in such an environment.

More importantly it illustrates that the basic precursors to a constructionist desire are present in media sharing communities and that the ideas developed are implementable and realistic.

B1.5 Post Experiment Interviews

B1.5.1 Brief

Post experiment, I decided to do qualitative interviews again. This time a questionnaire was not used but generally I wanted to gauge people's reaction to the jukebox and its evolution in the space over the last few days.

Only three questions were asked. One was what they liked about the jukebox and during the five days and the second was what they disliked about the jukebox and the five days of the experiment. The final question asked what else would they do differently Responses are quoted below.

B1.5.2 Responses

Date: Thu, 22 Aug 2002 13:39:19 -0400 (EDT)
From: XXXXXXXXXXXXXX
To: Surj Patel surj@media.mit.edu

Subject: Re: Garden Jukebox
> Queston 1: What did you enjoy or like about the last 5 days in the garden
> when the jukebox was running?

ambient music in garden, ability to hear new music from what others nearby
listen to (instead of the random songs on the net or the "popular" songs
that radio stations limit themselves to)

> Question 2: What did you dislike or find annoying in the last 5 days with
> the jukebox running?

bad music would come on occasionally and we'd have no recourse but to turn
down the speakers until the "kill" functionality was implemented. Also
there are times when it's nice to have a quiet workspace instead of one
with music going all the time.

> Question 3: Suggestions to improve it?
less random algorithm to find songs of interest, ways to rank songs so
they are played more frequently or banned. way to download
currently-playing song. (I haven't tried the new one yet.)

stop it from halting my machine every 30 sec or so--this really makes it unusable.

XX

Date: Thu, 22 Aug 2002 15:09:27 -0400 (EDT)
From: XXXXXXXXXXXXXXXXXXXXXXXXXXX
To: Surj Patel surj@media.mit.edu
Subject: Re: Garden Jukebox

> Here we go....
>> Queston 1: What did you enjoy or like about the last 5 days in the garden
> when the jukebox was running?
>
There was a sense of company even at odd hours. Being alone in the garden, hearing music that I came to associate with other people, made me feel less alone at times.
>
>> Question 2: What did you dislike or find annoying in the last 5 days with
> the jukebox running?
>
I found it a bit annoying when I logged on to the system, and there seemed to be no feedback in terms of music played by the system. It seemed the system 'favored' heavy users, and as an occasional user of the system, I felt a bit shut out.
>
> Question 3: Suggestions to improve it?
>
Stop lists. Logging on prevents certain songs from being played.

XXXX

Date: Thu, 22 Aug 2002 16:09:37 –0400
From: xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
To: Surj Patel surj@ml.media.mit.edu
Subject: Re: Garden Jukebox

i hope this isn't too late.

> Queston 1: What did you enjoy or like about the last 5 days in the garden
> when the jukebox was running?

i liked the initial surprise at music playing in a usually quiet space. although i work in an office where the music isn't audible all the time with the door closed, it was nice to have good ambient audio when i did stroll out. muzak with good music.

> Question 2: What did you dislike or find annoying in the last 5 days with
> the jukebox running?

it was too short a period of time for me to notice any annoyances.

> Question 3: Suggestions to improve it?

would be nice to have visual info - on the tv, or projected on the wall - on the song. i could glance at it if i liked the song. more visual info on any queues of songs about to played would be good too.

- There is a new version available that also has download and ability to see
  > what other people are playing. Email me if you are interested.|>
-

Date: Fri, 23 Aug 2002 11:33:27 -0400 (EDT)

From: xxxxxxxxxxxxxxxx
To: Surj Patel surj@media.mit.edu
Subject: Re: Garden Jukebox

> Queston 1: What did you enjoy or like about the last 5 days in the garden
> when the jukebox was running?

coming into the garden and hearing music.

trying to figure out how the program was choosing the songs it was playing.

seeing surj suffer.

> Question 2: What did you dislike or find annoying in the last 5 days with
> the jukebox running?

talking off my headphones and then hearing all of the songs i had just listened to.

having to add all of my songs to the player one by one.

only hearing my songs. it'd be nicer to hear songs that i hadn't heard and be exposed to other people music.

> Question 3: Suggestions to improve it?

have it take quarters and have bright flashing lights on it.

make it double as a cotton candy machine and a photo booth.

Some responses collected by interview

" I'm used to playing music in my office and sometimes the ambient music from the garden clashes. Its frustrating as I then want to close the door. It's like having two separate audio spaces"

" The MP3 player and agent caused a slow down of my machine and this was not good because I have an older machine. I didn't want to use after a few times. Is there any other way I can join in and contribute"

"The main thing is that the garden is sometimes a work area and sometimes a real social hangout area, what we need is something that can recognize that and be quiet when I want to work and play cool music when were all hanging out. That'd be cool. Maybe that or just based on time. It'll be quite during the day and then in the late afternoon start playing"

"I like the idea of having shared ambiance enhanced by music, but am a little weary of having a strong desire to turn it off at some points. I think the method of choosing the songs (who is there, what they've been listening to) sparks fun conversations and preference sharing though."

B1.5.3 Conclusions

The experiment did show that some members were more willing and proactive as predicted in my theoretical research

Because there was something that they engaged with and felt ownership of they were more proactive towards its evolution and design iteration. Most changes were implemented fairly quickly and easily thus proving one of the design parameters of the

system to some degree. This is good and shows a good precursor to constructionist frameworks being successful

Most people enjoyed the system and the experiment had been a success. The infrequent data corruption problem caused the systems enjoyment to be marred by the fact that MP3's skipped. However community support is high and the GardenConnector continues to evolve

Appendix C: Audit Data Analysis From GardenConnector

C1.1 Overview

In this section the database from the experiment is visualized in three ways to see what data we can glean about patterns of behavior and usage. Each visualization diagram is commented upon.

C1.2 Visualizations

The first plot is of activity, by person per day over the 8 days of the experiment. Each line represents one person and their timeline of events. See figure C1.

This second chart represents activity share by person, with most active user having the largest segment of the pie chart. Basically who is listening to more music and engaging with the system than anybody else.

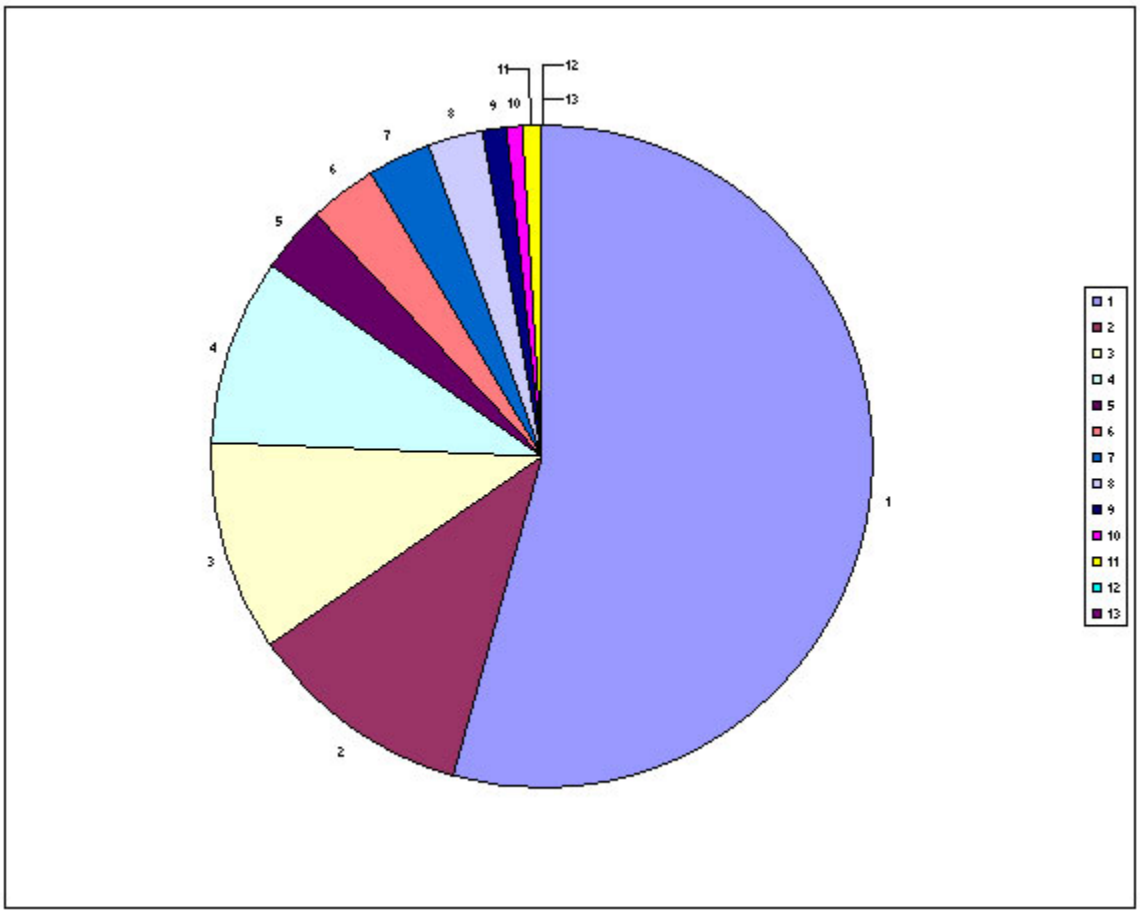


Figure C2. A pie chart showing most frequent listeners by magnitude

This was basically a count of events fired by a person into the database. As downloading had been only introduced towards the very end, I have filtered the data for download events.

Lastly I performed a simple statistical count on the filename occurrences in the database and matched them and counted them. This basically would provide a list of songs that were more popular than others, basically which songs have been played the most and how many times.

Amazed.mp3	9
Always & Forever.mp3	8
Blaze Of Glory.mp3	7
A Stroke Of Luck.mp3	7
I was only 19.mp3	7
Gone Away (Single Version).mp3	6
China Girl.mp3	6
Arms Wide Open.mp3	6
Lady in Red.mp3	6
All I want is you.mp3	6
Bach - Double Violin Concerto Vivace.mp3	6
Desert Rose.mp3	5
Crush.mp3	5
Amazing.mp3	5
Everybody Hurts.mp3	4
Endless Love.mp3	4
Hotel California(Live-The Millenium Concert).mp3	4
Black The Sun.mp3	4
The Unforgiven.mp3	4
Thank You.mp3	4
Nothing Else Matters.mp3	4
Hero (English Version).mp3	4
Here And Now.mp3	4
Everything I Do I Do It For You.mp3	3
phdblues-ea.mp3	3
For You I Will.mp3	3
The Freshman.mp3	3
The trick is to keep breathing.mp3	3
My Way Home.mp3	3
Falling.mp3	3
IGGYPO_1.MP3	3
ralph stanley - Little Birdie.mp3	3
03_-_Portishead_-_Roseland_NYC_live_-_All_mine.mp3	3
08_-_Portishead_-_Roseland_NYC_live_-_Glory_Box.mp3	3

Figure C3 A table showing track popularity by listening count.

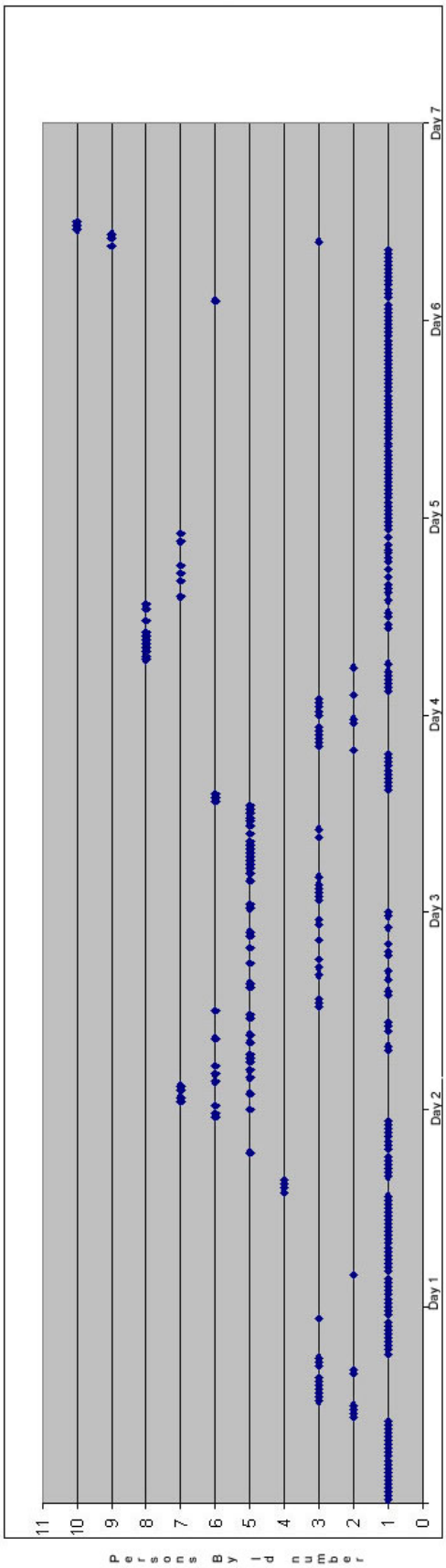


Figure C1 A plot of all events during the experiment over 7 days. Each horizontal ruling represents on person. Each plot represents one event.

C1.3 Conclusions on data analysis

Even working within the bounds of the small data set we had we can get some basic and obvious results like those seen in Figure C2 and C3. It is very easy to quantify how much and when with a system like this.

But the key findings can be seen in figure C1. This is a simple plot against time of people's events. Each horizontal ruling represents one person's timeline. The key thing that this does allow us to see is emergent patterns of behavior and to spot correlations of behavior from one individual against that of another. For example we can see that during day 1 user 1 is the dominant user with users two and three experimenting with the system as early adopters. Days two, three and four become much more interesting though. On day 2 we have another set of users join the fold and they start using the system early on in the day, as the day progresses some of the previous users are starting to log on to make their music heard. Another interesting pattern to note is that there appears to be turn taking, taking place. If you look at the pattern of interactions between users 1, 2 and 3 around the day 4 mark you will see that either by co-incidence or by design they are individually listening to music when the others are not. This basically could be that they are only playing music when they hear no other music being played.

Other signs are like those seen in the line of user 8. This user had been enthusiastic and then never used the system again. Basically this pattern shows interest and then waning interest. When questioned later his system had been not responding properly and files were transferring to the main system corrupted, hence his motivation was lost and he did not feel engaged. The signature of user 7 is also very similar. User 5 shows a signature that indicates growing enthusiasm and the suddenly stopping.

User 1 was very prolific and more often than not his music was heard above others, as statistically he was online more often and hence "present" to the system.

Other analysis could be centered around the music, its genre and time. Was there a certain favorite genre that people liked to listen to at certain times of the day? This when correlated to the feedback we received in B1 could help supply the context aware services people have described as wanting from the system. Again with time series data over a long time we could also build behavioral and preference profiles and match them to times of the day.

Matching this to other metadata and recommendation we could end up with some compelling experiences.

Appendix D: Example code from GardenConnector

D1.1 Java

Excerpted from Jythonintface Class

```
 public void playMP3(String mp3name){

        System.out.println("MP3 PLAYER CODE HAS BEEN CALLED");
        try
               {
                  // Open player
                  String paramline = "javaw -classpath
/mediaconnector/javalayer02/classes javazoom.jl.player.jlp \""+ mp3name +"\""; //+
mp3name;
                  try{
                        Process mp3player = Runtime.getRuntime().exec(paramline);


                       File fileob = new File("C:/jukebox/kill.txt");
                       int j = 0;
                       int k = 888;

                       while((j<150) && (!fileob.exists()) && (k == 888) ) {

                              fileob = new File("C:/jukebox/kill.txt");
                              try{
                                     k = mp3player.exitValue();
                                     System.out.println("thread still there " + k);

                              }catch(Exception e){System.out.println("thread still there
" + k);}
                              Thread.sleep(2000);
                              j++;
                       }

                       // mp3player.waitFor();
                       mp3player.destroy();
                       // originally mp3player.waitFor();
                       }catch(IOException e){
                              System.out.println(e);
               }

               }catch(InterruptedException e){ System.out.println(e);}

       }
```

D1.2 Java Servlet

Kill code service for the jukebox

```
import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;
import java.util.*;
import java.net.*;

public class kill extends javax.servlet.GenericServlet{

  public void service(ServletRequest req, ServletResponse res) throws
ServletException, IOException {
```

```
                    FileWriter filehandle = new
            FileWriter("c:/jukebox/kill.txt");
            filehandle.write("kill");
            filehandle.close();
            res.setContentType("text/plain");
            ServletOutputStream out = res.getOutputStream();
            out.println("THE SONG IS DEAD");

      }
}
```

D1.3 Jython

Final Iteration Jukebox

```
import htmllib
import formatter
import urllib
import javax.swing as swing
import java
from com.wrox.ea.jxtashare import jythonintface
h = jythonintface()

track="whatever"
oldtrack="juml"

while 1:
      peerlist = h.getpeerlist()
      numberofpeers = len(peerlist)
      #normalise it to produce right range for array index
      numberofpeers = numberofpeers - 1
      index = -1
      if numberofpeers > 0:
            index = h.random(numberofpeers)
      if numberofpeers == 0:
            index = 0
      if index > -1:
            peerchoice = peerlist[index]
      #get a track name from them
      track = h.getonetrack(peerchoice)
      #download it
      if track != oldtrack:
            oldtrack = track
      else:
            track = h.getrandomtrack();
            oldtrack = track

      h.downloadfile(track)
      #wait 60 seconds or so
      h.delaything(60)
      #play it
      trackplay = h.getnexttrack()
      trackplay = "c:/jukebox/" + trackplay
      h.playMP3(trackplay)
      #now delete the file
```

```
        print "DELETING LAST TRACK NOW"
        h.dellasttrack()
        h.delaything(5)
        #iterate again
```

Final iteration user client with other peer monitoring via a small HTML browser.

```
import htmllib
import formatter
import urllib
import javax.swing as swing
import java

ACTIVATED=swing.event.HyperlinkEvent.EventType.ACTIVATED
ENTERED=swing.event.HyperlinkEvent.EventType.ENTERED
EXITED=swing.event.HyperlinkEvent.EventType.EXITED

# we can bring in the Jeditor pane class here which can do basic HTML stuff

# we could actually make it part of a standard library.

class HtmlBrowserWindow(swing.JFrame):

    def __init__(self, urlString="http://www.media.mit.edu/"):
        swing.JFrame.__init__(self, title="HTML Browser", size=(500, 200))
        self.contentPane.layout = java.awt.BorderLayout()
        self.htmlPane = swing.JEditorPane(urlString, editable=0,
                hyperlinkUpdate=self.fHyperlink, size=(470,170))
        self.contentPane.add(swing.JScrollPane(self.htmlPane),
                java.awt.BorderLayout.CENTER)
        self.status = swing.JLabel(" ", preferredSize=(300,20))
        self.contentPane.add(self.status, java.awt.BorderLayout.SOUTH)

    def goToUrl2(self, stringy):
                self.htmlPane.setPage(stringy)
                self.goToUrl2("http://www.luc.edu/libraries/science/jesuits/1900.html")

    def fHyperlink(self, hlEvent):
        if hlEvent.eventType == ACTIVATED:
            self.htmlPane.setPage(hlEvent.URL)
        elif hlEvent.eventType == ENTERED:
            self.status.text = hlEvent.URL.toString()
        elif hlEvent.eventType == EXITED:
            self.status.text = " "

ozzy = HtmlBrowserWindow()
ozzy.show()
ozzy.goToUrl2("http://www.w3.org/MarkUp/Wilbur/")

#instance the jythonintface here as a  object
from com.wrox.ea.jxtashare import jythonintface

h = jythonintface()
b = h.getpeerlist()

#constantly cycle through a list of peers and review their listening habits

while 1:
        b = h.getpeerlist()
        for x in range(len(b)):
                URLstring = 'http://' + b[x] + '/servlet/NodeStats3/'
                ozzy.goToUrl2(URLstring)
                h.delaything(5)
```

Appendix E: PhotoTable– Fall 2001 – Project Summary

PhotoTable:
Designing Furniture for Sociability

Surj Patel and Edison Thomaz
MIT Media Laboratory
Shareable Media
Fall 2001

Abstract

Social relationships represent an intrinsic part of our human nature. It is a definite part of who we are and how we define ourselves to others. However, the sustainability of social relationships and physical communities have been gradually undermined and weakened by globalization and how nomadic we have become. We believe that as technologists, we have the responsibility to design new reconciliation mechanisms that bridge the gap between physically separated groups of people. In order to address this important issue, we built a piece of furniture, a coffee table, which will serve both as the conduit and as a natural interface used by people in different locations to share media with others. Our hypothesis is that sharing media with others is a powerful and implicit way to convey awareness and presence among members of a social networks separated by distance.

"Come, come. Sit by the table.

Here let me show you some photos of my son.

He's studying abroad now. Some university in America, MIT or something.

I hope he's well. I hope he's happy. He cant be eating properly. I know him too well. He was the most grueling nine months I ever had :-)

We use to often sit around this table with cups of tea and look at the photo albums. It's strange. If you think about it he was not even born when some of these photos were taken. But still you could not tear him away. His brothers and sisters liked the photos as well. But they are gone as well now. All over the place and all very successful.

I hope they remember their mother.

I wish he was here now. Right now. Looking at the new photos of his sisters wedding.

Oh look. Here are some photos of him when he was younger. It's funny how this table reminds me of the family passing the photos around. Tea, biscuits and photos. All of us together.

I wish he could share this moment. Maybe I should get a copy of these photos sent to him.

I bet he has an album on his coffee table. I know he's looking at them right now."

-- *Scene from a fictional family.*

Introduction

As the world becomes more fast-paced and globalized, our friends, our family and ourselves scatter across cities and foreign nations. Increasingly in pursuit of new challenges and experiences. One of the consequence of this geo-social phenomenon is the weakening of the physical communities and the social ties that they support.

Social relationships represent an intrinsic part of our human nature. Its a definite part of who we are and how we define ourselves to others. We believe that as technologists, we now have a responsibility to design new reconciliation mechanisms that address this socio-physical dichotomy and bridge the gaps within groups of people in this era of mobility. In other words, we strongly believe that we need better tools and devices that keep us connected and maintain affinity with other people. Tools that make us more aware of the people that we care about, regardless of their geographical location. Tools that promote discourse through the exchange of thoughts, ideas and experiences among groups of people. Tools that give us the means to reflect and perceive our world and the values of our social networks from a different angle. Tools that make us comfortable enough to knock on a neighbor’s door when we run out of sugar.

It is our belief that one of the many elements that serve as the foundation for a social relationship is the notion of shared experiences and, consequently, the exchange of representations of these experiences. An example of such a representation might be a

video or a set of pictures which depict an event or a trip, such as a wedding or a friend's trip to Europe. Although today we have the tools to record and create these experience representations fairly easily and inexpensively, the process of sharing this content is not nearly as straightforward as it could be. Or let me re-phrase that slightly. Not as familiar and comfortable as it should be.

Therefore, the research question that we would like to address is how to make media sharing more natural. The ultimate goal is of sharing experiences among physically separated groups of people. We would like to explore the theoretical notion of 'social objects' to extend the media sharing experience to the physical space.

The Coffee Tables

In order to explore these ideas, we built a piece of furniture, a coffee table, which will serve both as the conduit but also as the interface used by people to share media with others. These tables will also be networked together, linking the social space in the MIT Media Laboratory in Cambridge Massachusetts to a social area in Media Lab Europe, Dublin. One shared space between two physically disparate locations.

The tables will display photographs submitted by the members of the greater International Media Lab community. It is intended that these photos will be of social events, but also we hope that students will take the initiative to use the tables for other creative outcomes. The tables will have one common collection to start with this will increase to accommodate localized collections. All collections will be accessible by all tables. It is one shared space. Community members can upload and delete via web based interfaces. They will all be connected via TCP/IP initially and the MediaConnector project in later stages. The aim is to provide an non invasive, serendipitous and socially situated means of building connectivity.

We believe that the resonance between the tables, their locations in social situations and the fact that we are all Media Lab Community will help the two communities feel more affiliated, more linked and more empathic to each other. We want to know what our brothers and sisters are up to in Dublin and at the same time share our moments with

them. We are the same people but an ocean separates us. Using digital technology we want to give the sense that we are always in the next room. We are all in the same boat. We are the same corpus.

The location of the coffee table was inspired by the ML Garden area coffee table and its social significance to the people who frequent that area. But also important is the general social significance of any table. We eat at tables with our families and friends. We meet with people across tables. The space defines our social boundaries and relationships. We sit down and work at tables. Tables are everywhere and orthogonal to our vertical plane of common experience. But the common room coffee table more so. We leave magazines on there, photos for the group to view, books for people to pore over and albums that share captured moments with friends and family. We feel comfortable and safe leaving things there, feeling that the community will respect them. And these "serendipitous flashes" provide us with a sense of connectivity and affinity.

Networking Physical Spaces

Much work has been done in the area of teleconferencing and collaborative shared workspaces. But why don't we extend these ideas and work areas to what we normally consider "less serious' activities. Can we network physical environments together whose primary purpose is sociability rather than work? How would we treat these? More importantly, how would we perceive these?

Our interest in exploring objects and furniture in social spaces stems from several observations made by the psychology and sociology academic communities:

- People use objects to communicate
- People use objects to establish meanings about themselves
- Objects allow or facilitate certain kinds of social behavior
- Objects require certain kinds of social behavior

These tables will network the social spaces within the various Media lab outposts and bring them together on the level of the student. Not sponsors. Not faculty. Not politics. Not geographic boundary. The students feeling like one global corpus through a universally accessible and maintainable system of networked tables. Non-invasive and completely serendipitous. Like the glimpse of a colleague in the corner of your eye as they walk past, the table will flash colleagues and their moments past when your going about your normal business. But should you take more interest the table will permit you to pause and admire or scan through and find what you want to find.

State of the Project

At the moment, one of the tables is already at Media Lab Europe. The installation of the electronic components and sensors will take place in Dublin. During the month of January 2002, a second table will be built for the Media Lab US and connected to the first one, finally closing the network and social link between the media coffee tables. In the near future, we plan to build a third table in order to make the media furniture network accessible to the extended Media Lab community located at 1 Cambridge Center.