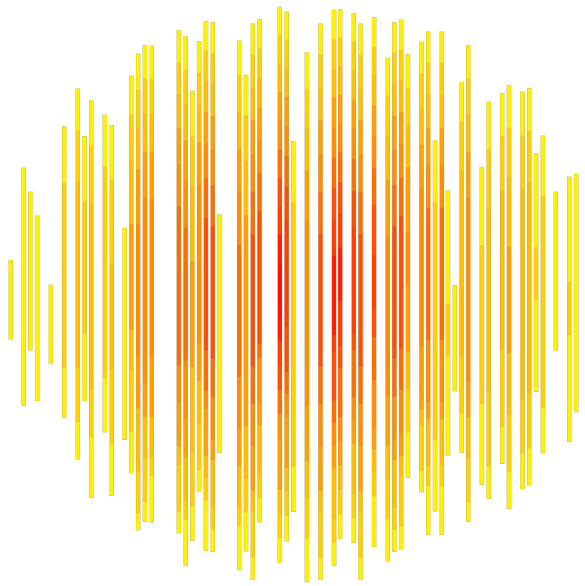
August 23, 1966

Jiangang Hao (jghao@umich.ed)
Brian Nord (bnord@umich.edu)
Christopher Peplin (peplin@umich.edu)
John Walters (jdwalter@umich.edu)

May 23, 2009



Abstract

August 23, 1966 is a project from the University of Michigan, supported by a grant from the GROCS (Grant Opportunities Collaborative Spaces) program in 2009. This report serves as documentation of the activities of the project over the course of the semester, as well as a guide for future groups to use or learn from the hardware and software created. This report is distributed with a compressed package that includes all of the source code, schematics, more detailed technical documentation and other media for the project.

Contents

Chapter 1

Proposal Summary

Chapter 2

Design Reviews & Meetings

Make sure to reference the videos.

2.1 Design Review 1

2.1.1 Questions & Sugguestions

- How to understand space and time and have a sense of space and time?
- Using some images of space and time can bring people closer to understand them.
- Math is the major barrier for most people to understand the complexity of the space and time.
- Astronomy in popular culture, like space travel
- How to convert the lensing to people?
- How to attract people to play with the stuff?
- Make something more tangible, such as a basketball.
- Constellations, in a game show.
- Google Earth
- Visualize things beyond our scale, like microscope.
- Exposite more about lensing, make it simpler to understand.
- Start from small scale to big scale.
- It is easy for people to forget these things. How to keep them remembering these?
- The speed of light, faraway object we see are actually many years ago. How to make people understand this better?
- The fate of life and our civilizations.

2.2 Design Review 2

2.2.1 Questions & Sugguestions

- Using voice to tune the star evolution
- Using heart beat to regulate star oscillation
- How to balance technology and principles? Too much Tech. stuff, are they better suited for presenting th central idea?
- Place the multi-touch table into a separate place so that to have a sense of reality.

2.3 Other Input

Chapter 3

Reflection

The members of the group individually reflected on some of the results of the project, technically, socially and otherwise. Rather than condense them into a single voice, we include them here separately to showcase the overarching individuality among the group.

Jiangang Hao The GROCS project we completed is a big achievement. With the goal of introducing astronomical events to average people, we built a multitouch table and a wiremap to help to reveal the complicated astronomical objects. Personally, I did not contribute as much as my collegues to the project, but I did not learn less than them from the project.

The major thing I learn is how to turn an idea into reality with iterative improvements. Our project started with the idea of visualizing the huge amount of data in simulation and real sky survey, but ended on displaying the evolution of a star on wiremap. Over the process, we keep shaping our methods to better reflect our goals. We achieved it eventually. Meanwhile, I learned the Processing language, an easy-to-use tool, to visualize and manipulate 3 D objects. It is really cool. Finally, the great ideas from our team members are really exciting. We turn out some really amazing things.

Brian Nord

Christopher Peplin The most important goal to me, and one we did accomplish, was to complete a vertical slice of an entire system. Our gallery installation and the hardware and software that support it touch an amazing number of topics, and I know everyone on the team has learned a great deal about their own and one another's fields in the process. I am extremely happy to have expanded my knowledge with hands on experience in:

- XML-RPC
- Java Servlets
- Multi-threading and databases in Java
- Sound in the Processing environment
- Library development for Processing
- Releasing a software project online with an open source license
- Multitouch input processing

- Graphics performance optimization
- Analog signal processing (filters, opamps)

I am coming away from this semester having created multiple tools, and I am eager to pass those on to anyone else with an interest in these topics.

Our team used an online wiki (a TiddlyWiki) [5] to organize our thoughts and coordinate the project plan. I was very pleased with how this fit into the group's workflow. The wiki served as the default place to put any thoughts or meeting notes, which greatly simplified the process of sharing information.

Despite being well-organized, the project lost focus around the middle of the semester. I think the primary cause was the independent spirit of the group members. Each person's perception of the group direction was slightly different and ultimately led the team too far from any core mission. The end of the project was extremely rushed as a result. Our mission changed significantly (perhaps too much) after the first design review, and we lost some ground when some completed work was no longer as relevant. During this period, I continued work on the Twoverse backed, which ultimately played a strong role in the software system for the gallery. With better planning, we would have started work on the gallery display software sooner, giving us more time to test the entire system.

Overall, I enjoyed working with people outside of the computer science department. When I work with people with similar types of knowledge, I found that I began to make assumptions about ways of thinking and ways of looking at problems. This project made me realize how the "common sense" answer to me is often more software oriented not as obvious to a sculptor or physicist, who may have a more materials or theoretical solution. Accepting the existence of other solutions is sometimes a challenge, but ultimately a rewarding task.

John Walters My personal response to this project is one that highlights my great appreciation for the manner in which all members contributed to the final conception of the exhibition, even when it seemed that we may never see the project come to fruition. It was the delegation of tasks within a group dynamic that allowed each member to contribute significantly. By using each persons skills to their fullest potential, while at the same time being aware of the overall outcome, everyone was able to apply themselves as well as learn from this collaborative experience.

Chapter 4

Gallery Installation

4.1 Concept

4.2 Implementation

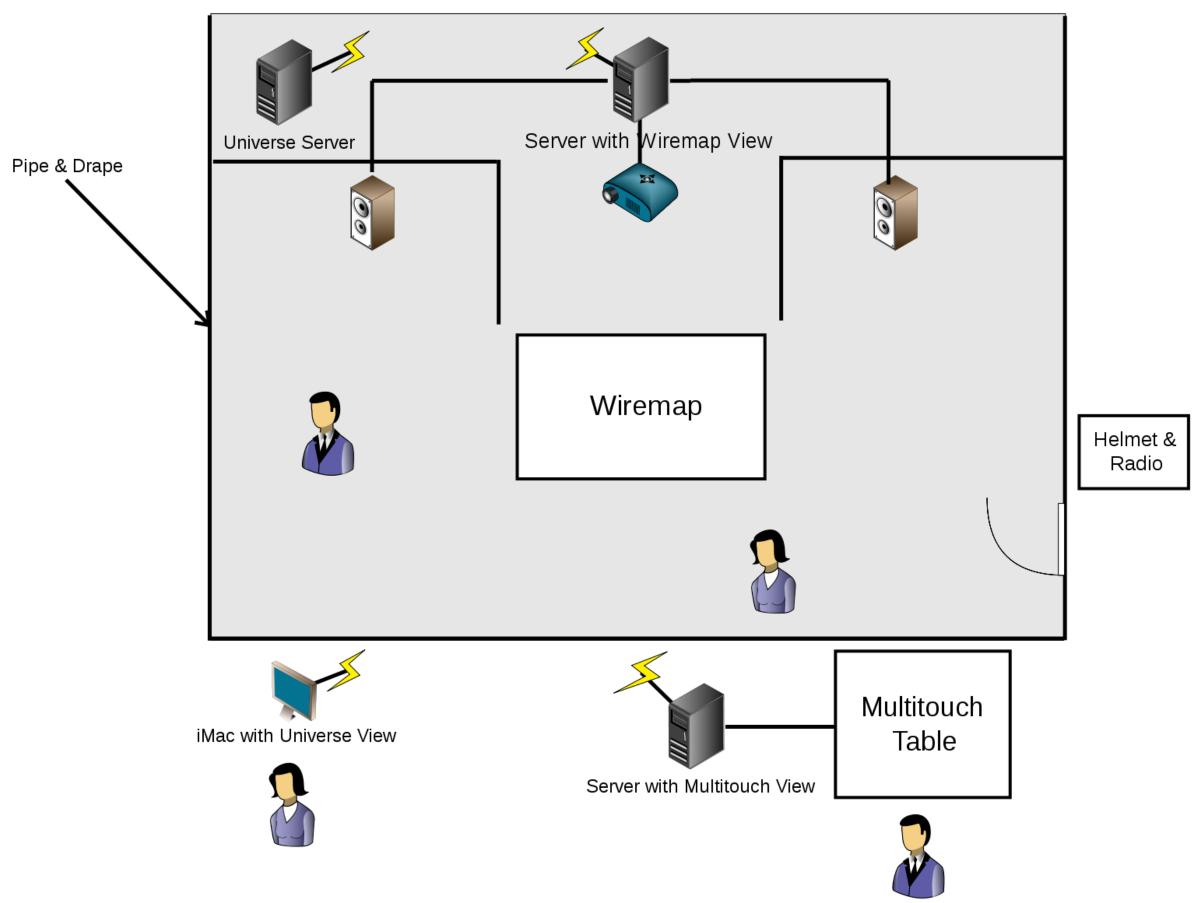


Figure 4.1: The layout of the gallery installation as implemented. All computers were networked with an Ethernet connection.

See the video included in the software package for a tour of the installation as it stood [4].

Chapter 5

Hardware/Software Documentation

This chapter describes the individual hardware and software systems built by the group. Some systems were directed at use in the gallery system, so the documentation is somewhat specific. In other cases, the system is easily generalized and the documentation reflects that. All of the source code and documentation for software usage can be found in the included software package [4].

5.1 Multitouch Table

The multitouch table created for the project is a rear diffused illumination design (see figure 5.2), and is housed in a coreten steel cabinet with recessed cooling fans and an access panel on the rear vertical wall. The touch surface is a 12” polycarbonate sheet with an adhesive projection film applied to the underside (acting as a diffuser for the projected image).

In addition, infrared light is also projected at the diffuser from below (inside the cabinet) the touch surface. The table currently uses an array of six multiple IR LED lamps. When an object touches the surface it reflects more light than the diffuser or objects far away from the surface. The change in light is detected by a web cam placed inside the cabinet, and the signal is fed to the computer for analysis.

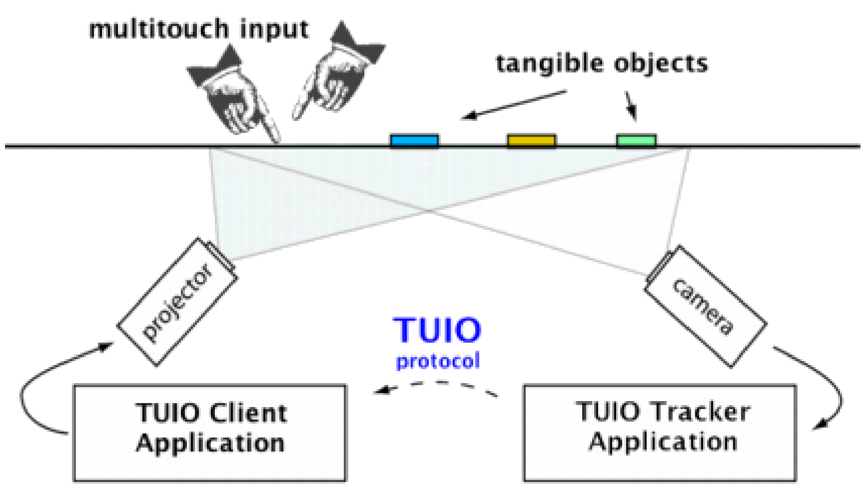


Figure 5.1: This diagram explains the flow of information from a finger press to the reaction on the screen. [17]

5.1.1 Software

The multitouch table uses The Beta, from the NUI Group [14], to process the video stream from the webcam. The Beta, tbeta for short, is an open source tool that analyzes a video to find tracking data for objects it recognizes as fingers or cursor devices. The software provides a great deal of control over the video parameters (high-pass filter, amplification, threshold, etc.) that adapts well to many types of multitouch displays.

Tbeta outputs the tracking data using the TUIO protocol [18], which is an open framework for receiving input events in various programming environments. For this project, the TUIO events sent by tbeta were received using the open source Java TUIO library in a Processing sketch.

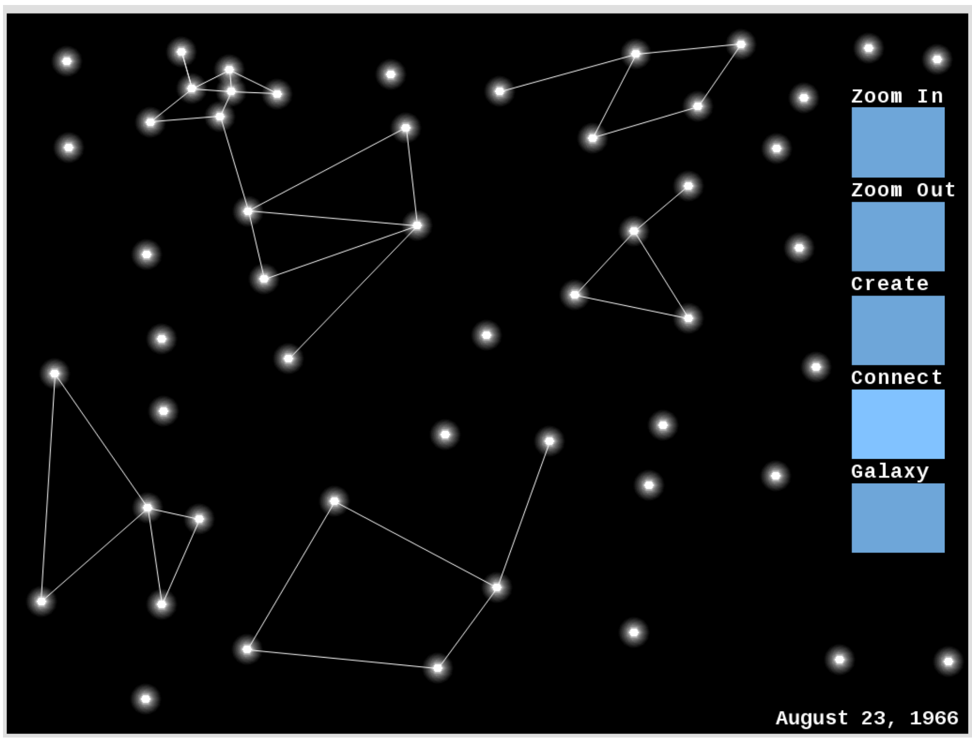


Figure 5.2: A screenshot of the Multitouch Client with a group of stars displayed at the default zoom level. The lines connecting the stars are constellations.

Figure 5.3: A screenshot of the Multitouch Client viewing the details of a single star.

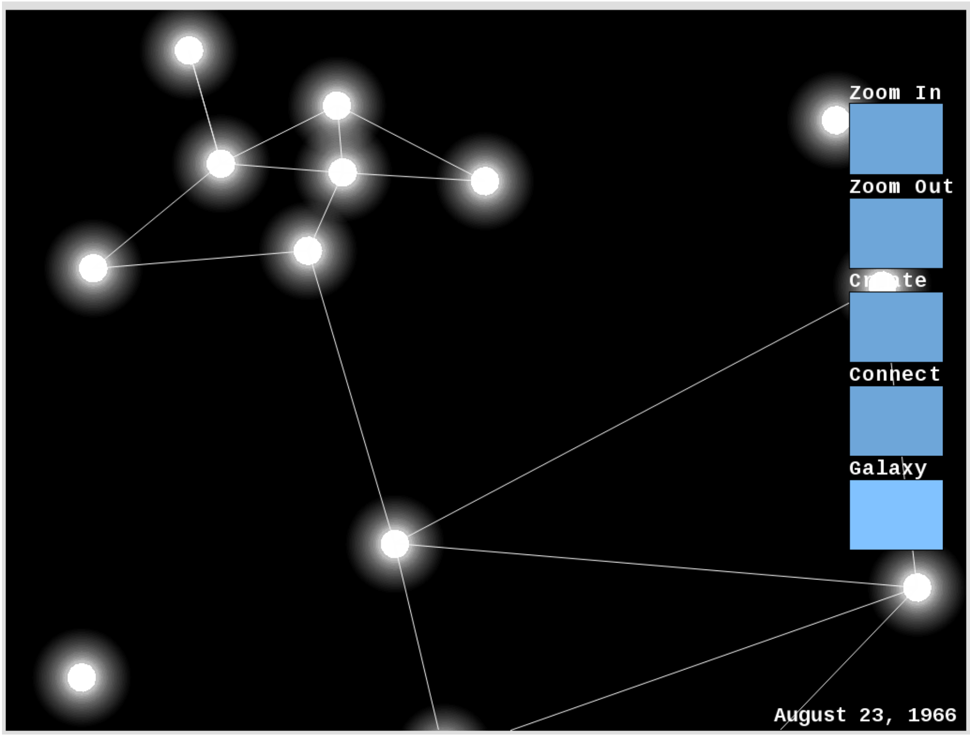


Figure 5.4: A screenshot of the Multitouch Client zoomed in for a closer look at a cluster of stars.

5.2 Pulse Oximeter (Hearbeat Monitor)

In researching ways to detect the human heartbeat, we found that photoplethysmography would be the simplest and least expensive way to bring an immediate and personal touch to the gallery installation. A photoplethysmograph is usually obtained with a pulse oximeter - this is the same device that grips a person's finger in hospitals to measure the heart & respiration rates.

Concepts A pulse oximeter simply illuminates the skin with light from an LED (usually infrared), and measures the luminance of the skin on the other side. Each cardiac cycle brings more blood to the extremities, and thus the finger is denser and less light passes through to the detector. When the blood flows away, more light is let through. This fluctuation can be recorded and the timing of the luminance peaks used for recording the heart rate.

This project required knowledge of photoplethysmography [15] [10], signal processing filters [11] and operational amplifiers [13] [8] [7].

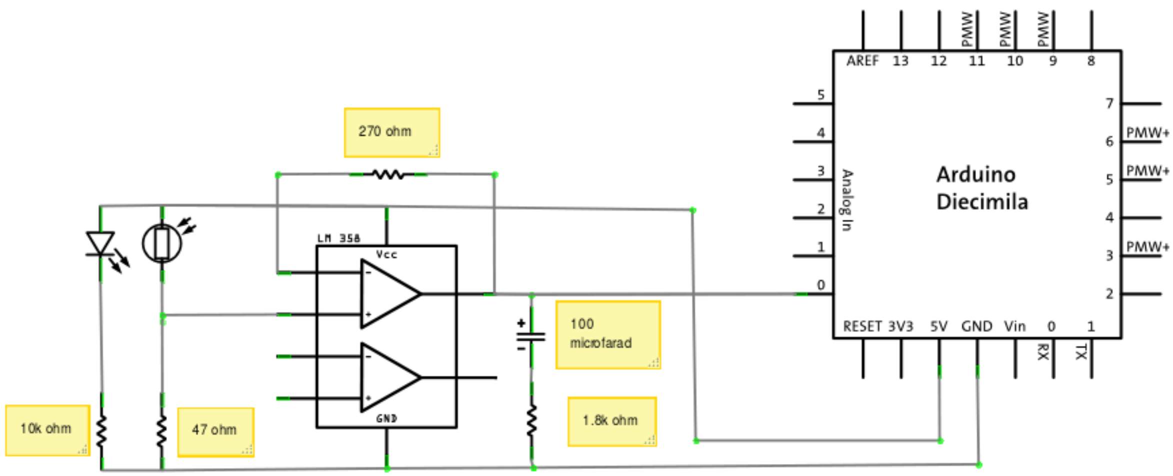


Figure 5.5: This schematic describes the circut used inside the pulse oximeter. It is provided as a Fritzing file in the software package.

Implementation This device is advantageous for its low intrusiveness. In our implementation, the visitor just needs to gently place their finger on top of the light sensor. Depending on the person, the shape, rate and range of the photoplethysmograph obtained can vary widely, but we found the results distinct enough to obtain a heart rate from almost every participant.

Our device used the amplified signal of an inexpensive photo-resistor passed through a high pass filter and captured by an Arduino microcontroller [3]. The microcontroller fed an averaged luminosity to a Processing [16] sketch on the host computer, which analyzed the signal for peaks. The peaks were then converted to a frequency, and passed along to the gallery software. The software is general to heart rate monitoring, and can be used for other applications that are interested in the data.

The electronics schematics are included as both a PDF diagram and as a file for Fritzing[12] and can be found in this package at `/august/doc/heartbeatmonitor`.

The source code for the microcontroller and for computer-side analysis can be found in this package at `/august/src/gallery/PulseOximeter` and `/august/src/gallery/WiremapClient` [4].

5.3 Twoverse System & Library

The software core of the August 23, 1966 project is a Java software system coined as "Twoverse," a parallel universe that exists only in the digital realm with some key points of connection to the real world. It can be considered an extension of the many-worlds interpretation - alongside our world, with its crumbling economies and warring nations, there exists a digital universe that is directed by you, the user. Just like here on Earth, many parameters are outside of your control. The interesting part is choosing what you can, timing as you may, and watching the results.

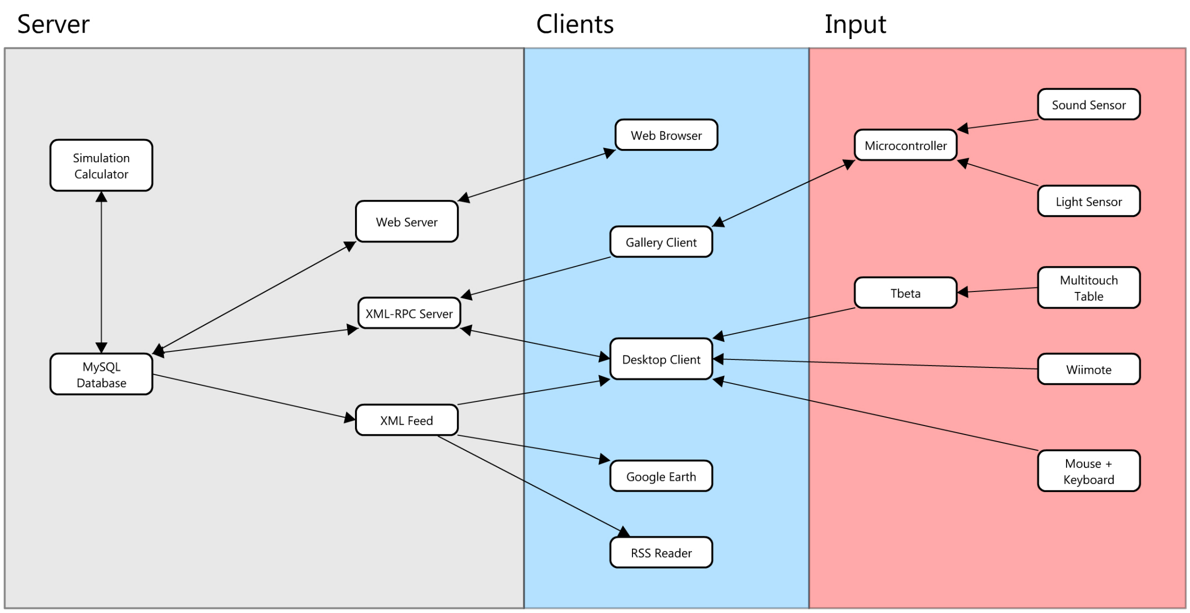


Figure 5.6: The Twoverse architecture can be split into three levels - server, client and input. This diagram includes some unimplemented elements, such as input from a sound sensor.

System Design The broad concept of Twoverse includes mechanics and partial system specifications for a massively multiplayer online game. The scope was minimized due to time constraints and to better fit the gallery installation. However, even with a smaller scope, a complete vertical slice of the entire system was implemented and used. Downsized from a universe of many types of objects, the system currently supports a universe made of stars with a few properties, and constellations that connect them with meta-objects known as links. Scalability was stressed from the beginning of development, so extending the universe with new objects will require a minimal amount of work.

Server The system relies on a central server to provide the following:

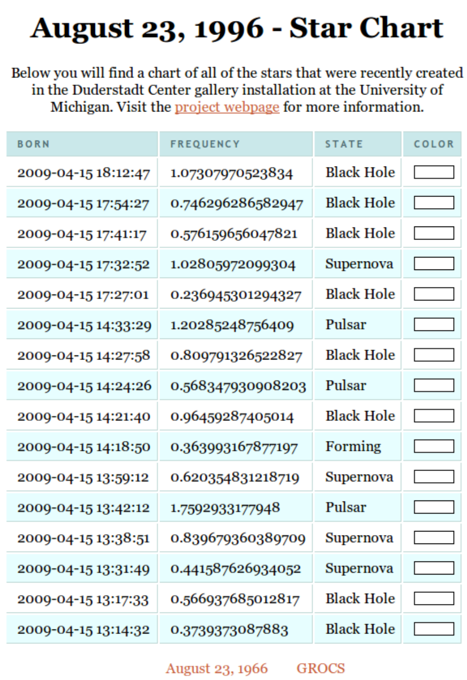
- A persistant database of all objects in the universe and their current state, using MySQL
- User account management, as well as authenticated session negotiation

- A public API for interacting with the universe, via Apache XML-RPC [2]
- Client pull style updates for minimizing bandwidth requirements, via an XML feed
- A web-browser based frontend to view the status of objects in the universe, via PHP

Client Using the public API, many types of clients are possible. This includes graphical, text-based, mobile, e-mail, etc. The clients implemented for the gallery installation are graphical clients written using the Processing development environment and include the following features:

- Users can scroll and zoom around a graphical universe of glowing stars
- Users can click on an individual star to view a close-up view and additional details about its creation and properties
- Users can create a new star in the universe, and watch a 3D visualization of their star's formation
- Users can draw constellations that connect the stars in the universe, and leave them for other users to see
- Users can visit the gallery website to view a table of all of the stars in the universe, their properties and current status (see figure 5.7)

The graphical client uses the XML-RPC API as defined by the server, and updates its local cache of the universe via the server's XML feed.



August 23, 1996 - Star Chart

Below you will find a chart of all of the stars that were recently created in the Duderstadt Center gallery installation at the University of Michigan. Visit the project webpage for more information.

BORN	FREQUENCY	STATE	COLOR
2009-04-15 18:12:47	1.07307970523834	Black Hole	
2009-04-15 17:54:27	0.746296286582947	Black Hole	
2009-04-15 17:41:17	0.576159656047821	Black Hole	
2009-04-15 17:32:52	1.02805972099304	Supernova	
2009-04-15 17:27:01	0.236945301294327	Black Hole	
2009-04-15 14:33:29	1.20285248756409	Pulsar	
2009-04-15 14:27:58	0.809791326522827	Black Hole	
2009-04-15 14:24:26	0.568347930908203	Pulsar	
2009-04-15 14:21:40	0.96459287405014	Black Hole	
2009-04-15 14:18:50	0.363993167877197	Forming	
2009-04-15 13:59:12	0.620354831218719	Supernova	
2009-04-15 13:42:12	1.7592933177948	Pulsar	
2009-04-15 13:38:51	0.839679360389709	Supernova	
2009-04-15 13:31:49	0.441587626934052	Supernova	
2009-04-15 13:17:33	0.566937685012817	Black Hole	
2009-04-15 13:14:32	0.3739373087883	Black Hole	

August 23, 1966 GROCS

Figure 5.7: A screenshot of the star chart in a web browser during the time the gallery was open.

Library The client and server for Twoverse share many common functions, and were designed to inherit their source from the same tree of code. They both stem from a Twoverse Java library, which includes many utility classes, shared functionality and the server executable. The Twoverse library provides these features and many others:

- Database wrapper class for a persistent universe - could be used to run SQL database client-side
- XML-RPC servlet for serving XML-RPC requests
- Thread-safe universal object manager for maintaining the state of the universe
- User session manager
- Small unit test suite for core classes
- Processing camera wrapper class for simplifying the elusive camera() function
- Flexible 2D/3D point coordinate class

The source code as well as an extensive HTML reference is included in the software package at `/august/src/Twoverse/doc` [4].

See the project wiki [5] for more information on the motive for Twoverse and possible future plans.

5.4 Gallery Software Setup

To replicate the gallery installation, the following systems & equipment are required:

Server

- MySQL server
- Web server, root pointing to <location of august package>/www
- Ports 80 & 8080 open
- Java JRE 1.5 or greater

Multitouch or Desktop Client

- Multitouch table
- Hardware accelerated graphics
- Network connection to server (can be over Internet)
- Sound output (ideally to a radio headset)

Wiremap Client

- Wiremap
- Projector
- Sound output (ideally to a sound dome)

5.4.1 Server Setup

1. Create an "august" user for the MySQL database
2. Initialize the august database using the SQL script found at `/august/database/create.sql`
3. Run the compiled Twoverse server to check the database connection and create the XML feed file: `java -jar twoverse_server.jar`
4. Create a symbolic link to the feed.xml file that was just created in the web server's root directory (this should be /august/www/)

5.4.2 Multitouch or Desktop Client Setup

1. Plug the radio headset into the audio output of the computer (if you are using a radio)
2. Open the client sketch (either Multitouch or Desktop version) and modify the configuration parameter for the server IP to point to the correct address.
3. Configure the Wiremap server IP configuration parameter in the CreateMode.pde file to point to the address of the Wiremap client (if you are using one)
4. Run the sketch in Processing, or export a compiled application to simplify restarting the application

5.4.3 Wiremap Client Setup

1. Plug the sound dome into the audio output of the computer (if you are using a sound dome)
2. Open the client sketch and modify the configuration parameter for the Multitouch Client IP to point to the correct address of the Multitouch Client (not the Twoverse server)
3. Run the sketch in Processing, or export a compiled application to simplify restarting the application

The Multitouch or Desktop client connects to the Twoverse server and manages the creation and storing of new stars. The Wiremap client receives a message from the Multitouch Client when a new star is being created, which initializes the sound and visuals for the star formation experience. See the Twoverse library documentation for more detailed information on this relationship.

5.5 Wiremap

The Wiremap is an innovative projection technique that displays a 3D image in space using a standard computer projector. The projector throws a beam of light on an array of vertical wires. From the focal point of the projectors lens, all the wires are evenly spaced from one another and have a corresponding distance from the projector. With that information (both a horizontal and depth coordinate), and using some careful calculations on the computer, we can project simple images at various depths in the field. From any perspective besides the projector position, the wires appear randomly placed and the image becomes visible.

Our implementation uses mason's string for the field, 3/4" plywood for the top and bottom alignment/hanging boards, and standard nuts and washers as anchors for each string. Our map has 256 strings, placed in a randomized dimension of depth through an equal number of holes in both the top and bottom alignment boards. The strings are secured with bolts on the top board and weighted down with a washer below the bottom board. When the top board is raised to 8ft, the wires become taught and can be aligned to a 90 degree angle with the floor. The Wiremap must be calibrated each time the projector is positioned - this includes making sure the wires are parallel, the projector sees the wires evenly spaced, and there is no unnecessary tilt or keystone in the projector's image.

Figure 5.8: The 256-string Wiremap installed in the gallery.

5.5.1 Wiremap Software Library

In order to facilitate quicker prototyping and make the Wiremap software more accessible to the team, we wrote a simple Processing library for rendering certain shapes in the Wiremap field. The library replaces the source code provided by the creator of the Wiremap [1] by reducing code duplication and abstracting most of the implementation details away from a user who wishes to simply draw a sphere, rectangle or sliver in the field. The library also includes a novel calibration method developed in response to inaccuracies in our Wiremap.

Abstraction The Wiremap library gathers the coordinate conversion and wire selection math into a single class. The previous method required duplicating a set of functions in every Processing sketch that output to the Wiremap. Now, the user creates an instance of the Wiremap class and provides a few key measurements of the physical interface as well as a text file listing the wire depths. The calculation is done as necessary, and not exposed to the user.

Coordinate Systems One key difference between the original source and the Wiremap library is the coordinate system used for each plane. Previously, the coordinates of X, Y and Z were all physical inches and matched the actual dimensions of the Wiremap. To facilitate quicker transitioning from a regular Processing sketch (using the standard 2D renderer) to one for the Wiremap, the X and Y were changed to be in the standard, Processing-style pixel coordinate system.

The Z plane remains in inches, as there is no obvious relationship between Z space on the screen (which is infinite in both directions) and Z space in the Wiremap field (limited by the physical dimensions). Thus, Z coordinates in the field range from 0 to the field depth.

The library has been released under the Apache open source license, and will continue to evolve after this project's completion. See the library's documentation for details on installation and usage [9].

Bibliography

[1] Albert Hwang, *Wiremap*,
http://wiremap.phedhex.com

[2] *Apache XML-RPC, Open Source RPC Framework*,
http://ws.apache.org/xmlrpc/

[3] *Arduino*,
http://arduino.cc

[4] *August 23, 1966 Wrap Package*,
http://august231966.com

[5] *August 23, 1966 Wiki*, included in [4] and online,
http://august231966.com

[6] *Bazaar Distributed Version Control System*,
http://bazaar-vcs.org/

[7] Bill Bowden, *Operation Amplifier (Op-Amp) Basics*,
http://ourworld.compuserve.com/homepages/Bill_Bowden/opamp.htm

[8] Bill Klein, *Signal Chain Basics: Operational Amplifier - The Basic Building Block*,
http://www.planetanalog.com/features/showArticle.jhtml?articleID=202801320

[9] Christopher Peplin, *Open Source Wiremap Library*,
http://code.google.com/p/wiremaplibrary

[10] Erik Cheever, *E72(a) Project - The Photoplethsymograph*,
http://www.swarthmore.edu/NatSci/echeeve1/Ref/E72Projects/E72Cardio/E72Cardio.html

[11] Erik Cheever, *Frequency Response and Active Filters*,
http://www.swarthmore.edu/NatSci/echeeve1/Ref/FilterBkgrnd/Filters.html

[12] *Fritzing, Electronics Prototyping Tool*,
http://fritzing.org

[13] Harry Lythall, *Operational Amplifier Basics*,
http://web.telia.com/~u85920178/begin/opamp00.htm

[14] NUI Group, *The Beta*,
http://teta.nuigroup.com

[15] "Photoplethysmograph." Wikipedia, The Free Encyclopedia.
`http://en.wikipedia.org/wiki/Photoplethysmograph`

[16] *Processing*,
`http://processing.org`

[17] reacTIVision,
`http://reactivision.sourceforge.net`

[18] *TUIO Protocol*,
`http://tuio.org`