Documentation
Urban Instrumentation Project PIPE

Afroditi Manari
MediaArchitecture, M.Sc.
afroditi.manari@uni-weimar.de

Emir Genc
MediaArchitecture, M.Sc.
emir.genc@uni-weimar.de

Georg Erfurt
Medieninformatik, B.Sc.
georg.erfurt@uni-weimar.de

Kevin Schminnes
Medieninformatik, B.Sc.
kevin.schminnes@uni-weimar.de

Preetha Moorthy
Human-Computer Interaction, M.Sc.
preetha.moorthy@uni-weimar.de

René Levin
Computer Science and Media, M.Sc.
rene.levin@uni-weimar.de

Revathi Gurumoorthi
Human-Computer Interaction, M.Sc.
revathi.gurumoorthi@uni-weimar.de

Umar Akram
Human-Computer Interaction, M.Sc.
Umar.akram@uni-weimar.de

Vanessa von Jan
Medieninformatik, B.Sc.
vanessa.von.jan@uni-weimar.de

ABSTRACT

PIPE combines 3 individual placeable input modules and one attached output module to an interactive light-installation for walkway situations. Supported by the mobility of the input devices and the benefits of a central orchestration server, PIPE instruments public spaces and its main goal is to increase social interaction and the creation of shared encounters by providing a joyful and entertaining experience for one or more users.

This paper illustrates the way from analyzing existing projects, designing, building and testing the modules, to the final evaluation in a real-life situation.

Author Keywords

Social interaction; shared encounters; public interfaces; hardware; wireless communication; panStamp; orchestration software; product design; prototyping; human-computer interaction; in-the-wild study;

ACM Classification Keywords

D.1.5 Object-oriented Programming; D.2.3 Coding Tools and Techniques: Object-oriented Programming); H.5.2 User Interfaces: Input devices and strategies (e.g. mouse, touchscreen), Prototyping; J.4 Social and Behavioral Sciences: Sociology; J.5 Arts and Humanities: Architecture; J.6 Computer-aided Engineering: Computer-aided design (CAD), Computer-aided manufacturing (CAM);

This document was created as part of the projects "Instrumentation of Public Space for Social Interaction" and "Orchestrating Sensors and Actuators for the Urban Environment" of the Bauhaus University Weimar, Germany. The project was carried out by the authors mentioned above and supervised by Patrick Tobias Fischer, M.Sc, and Prof. Dr. Eva Hornecker during summer semester 2015.

INTRODUCTION

The project is split into four subgroups: output module, input module, orchestration software and evaluation. Each subgroup will describe it's local goals, leading towards the completion of the project as a whole.

The objective of this project is to generate social interaction in public space, triggered by digital design. Through a digital/physical installation we aim to attract multiple users and to create shared encounters.

The installation consists of two parts, the input module and the output module. For the input we use tubes with pressure sensors, that when stepped upon activate the output. For the output, on the other hand, we have chosen an LED chain that is attached vertically on a pipe.

A special software is necessary to deal with the different hardware modules. This tool exists in the form of RokkaSpot, which allows to communicate with the hardware, acquire data from input modules and send data to output modules.

The town of Weimar has an interactive fountain installed at the Herderplatz. An evaluation study was done on this interactive fountain, so that it can be compared with our own installation to test how much social interaction is actually produced.

RELATED WORK

The project is roughly based on two former projects of the Bauhaus University Weimar. The "Parasitic Interfaces for Public Environments" (2014) [1] aimed at creating an experience built around the idea of entertainment and design in a public space. During the project, a prototype was designed and manufactured. The prototype was used in a public test to analyze the general reaction of people when facing the interface. PIPE is based upon a similar idea, using the surroundings, like the architecture nearby, to create a public interface. The focus is different though. The parasitic interfaces were meant for public plazas, whereas PIPE was developed for walkway situations.

Another project of the BUW created the "Kick/Flickables" (2014) [2], a set of mobile devices communicating with each other and showing reactions to the user by emitting light in different colors and brightnesses. The systems aimed to create an interface in a different direction, opposed to the focus on screens. It was also meant to work as a catalyst for social interaction. The idea of creating social interaction, especially shared encounters, was a main focus of the project described in this document, too.

As part of project PIPE, a public interface in the local area was analyzed and evaluated. In the town of Weimar, at the Herderplatz, there exists an interactive fountain. It includes a simple non-digital multi-user interface. It attracts young people as well as old, serving as an inspiration for the project. Further evaluation expressed competition as well as cooperation when operating the fountain, which is further explained later in this document.

Looking for further examples, an interesting idea proved to be the piano staircase by “The Fun Theory” [3]. A video on the webpage visualizes a simple yet effective approach by putting piano-key-like modules on the stairs of a subway, producing sounds when stepping on them. It’s core idea is a different one, to make people use the stairs instead of the escalator. Still it fits PIPE’s idea of public interfaces in walkway situations quite well. The piano staircase successfully captures attention, an important property to create shared encounters, by using a combination of visual attracting design as well as acoustic feedback. PIPE uses a comparable input scheme, but works with visual output instead of sound. The output module is meant to catch attention from afar already, because the input modules do not cover the whole walkway and therefore may be “missed” by the potential user.

Finally, one main efforts of the project was to create “shared encounters”, a term well defined by the work of Katharine S. Willis et al. [4]. Generally, a shared encounter is an experience shared by multiple users, with the users acting in a specific “role”. It takes place in a shared space with high awareness between the different users. PIPE focues on the urban environment and passageway situations. The role of the users is to create different colors by stepping on the seperated input modules. The shared experience is the creation and output of a single color visualized at the output module.

OUTPUT MODULE

General Idea

The project ‘PIPE’ deals with social interaction within the urban setting, which requires people to involve actively in different scenarios. According to the aim of the project, those scenarios should be formed around architectural elements, which are crucial for the urban settings.

During the analyses of the urban settings, we were focusing on social interaction, therefore shared encounters, so that evaluating the social situations in an urban setting was the outset of the project. Initially those were indicated as two variations; walkway situations and plaza situations.

Plaza situations where considered as an extensive stage, where people can perform their everyday life activities and use the urban place vigorously. Hence it has been fixed upon taking those situations into account to analyze the encounters, especially the unintentional encounters. In order to analyze them we decided to go with the walkway situations instead of plaza situations. The difference of it is having more presumable contingences than the other one and it brings us closer to the architectural elements.

Design Process

Our initial concern was to find repeating structures on the urban, architectural elements, therefore to obtain the universal design approach of human-computer interaction methodology, and to start with analyzing the various attachment systems around us. Also through this repeating system, the form can be perceived as a whole in an urban setting, therefore it can enlarge the interaction space as far as possible.

Attachments, which generate the link between architecture and technology merging into architecture, must have been easy to apply on the surface and robust for the urban situations, considering the users behavior and the interaction rules. Once we got into the attachments first we were inspired from simple but often used methods, which we can come across in our surrounding environment.

The other concerns were to define the area of the project and create a stable display, which will be attached on the surface of an architectural element. Even though we decided to define the interaction zone in the beginning as 50x50 m^2, which is the maximum area to create a situated installation, after analyzing the walkway situations it would have been made more sense to narrow it down. The reason was that walkways are only considered as sidewalks; therefore it was not possible to create that big stable interaction zone. Another idea behind was, it should be easily discoverable, distinctive, for the possible users. Through all the concerns we came up first with an architectural element, which is repeating, easy to access, able to provide walkway situations, the downspouts on the facades; the pipes.

The pipes, downspouts, are standardly produced elements of urban architecture, which are often overlooked elements. We aimed to bring them in to the urban settings, assign them functionality and let them attract people rather then being invisible any longer. Besides, this installation aimed to be abstracted from place and actually it could stand on different walkway places.

By designing the interface and display related to pipes, we came up with different ideas. One of them was “the hammer game” to be realized on that vertical surface and to create a game, which participants could get into an interaction with different groups or individuals easily and find out the rules without any instruction or without guidance. The common rule of a hammer game is to apply some inputs and these inputs result to a vertically rising pattern, as if the participant leaves his/her own mark on the display, therefore individual footprints could be occurred on the architectural element.

Meanwhile another approach for the game idea, once we started thinking about vertical displays, was “the worm”. The worm was also considering vertical displays, the pipes, but both ways; up and down, as ascending and descending marks. The advantages of this idea were being able to develop it for the further interaction rules and to implement dual movement to an immobile display. For this movement we are determined to use a light pattern.

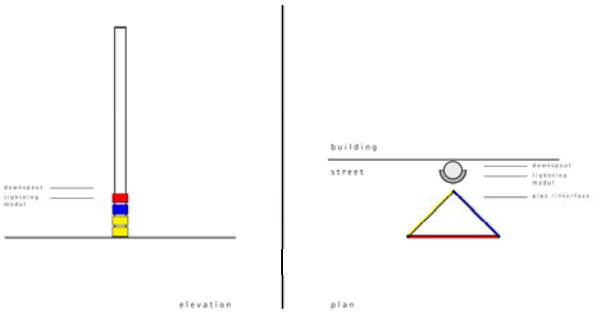

Figure 1.1: Design Idea.

The idea of worm was enabling to produce a segmented vertical display, which had made it possible to use each LED individually, in order to create pixelated light pattern. On the other hand to design a modular display, which has a repeating part to create a whole, allows us apart from having a dynamic display design, to produce from one sample and to assemble easily.

On the light display, according to color scale of LED chain, we supposed to see red, green and blue (RGB), but instead, we decided to have the pattern of the main three colors of the main scale; red, yellow and blue. The reason of using main colors is that we are used to see them and their intermediate colors in our everyday lives, therefore by using them in our design, the idea was expected to be more embedded in the urban setting.

As we were evaluating the input options for the pipe, we came up with many possibilities that we can acquire from the passers-by. As we thought the input methods, we suggested counting passers-by, sensing their motions or voice and even smell. Afterwards it has been agreed on using such an interface that participants or passers-by can step on, which consits of three pressure pads to represent the main colors (red, yellow and blue) and works by measuring the air pressure changes inside the pads.

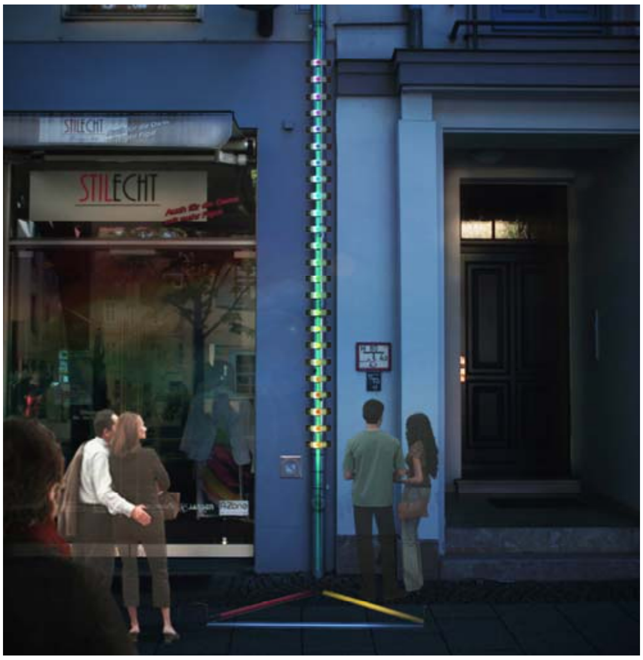

Figure 1.2: 3D Visualization.

Production Process

The production process of the output prototype can be divided in three phases: The first phase consists of the initial search of the form and at the same time testing various materials and attachment methods. The second phase refers to the prototype, which was presented at the summaery exhibition. At that point the form of the model had been finalized, while the materials used, were intended for indoor exposure and made of cardboard. The third and final stage refers to the final prototype, which was intended for real life situations. In this occasion, the materials that were used were finalized and displayed maximum robustness. While small alterations were made to the previous design due to different material attributes and fabrication method.

First Prototype(s)

The search for a form took place simultaneously with the testing of different materials and building methods.

Following the basic concept we focused on creating a structure that would encompass the LED chain and would be able to defuse its light. The parts that we focused on, were a diffusing case for the chain as well as a method to attach it on a pipe. From early on, however, we were inclined to create small separate cases for each LED light of the chain, instead of creating a holistic structure that would mix the different light colors.

The initial attempts included materials such as cardboard, nylon fabrique, metal wire, semi transparent foil, aluminum foil and spray paint. The nylon textile and the semi transparent foil were used for the diffusion, and in order to test their diffusion capacity we first conducted tests with an arduino board and a simple LED light, to check the illuminance as well as how we perceive the different colors through the different materials.

For further testing we started using the LED chain, as its light was brighter than the light of single LEDs. The chain we used was the WS2801, 12 mm Pixel Chain. For the wiring we followed a tutorial [5] while for the programming we used the Adafruit WS2801 Library for Arduino [6].

Our next attempt was to create a cubic case out of cardboard and semi transparent foil (Hobby Fun, Crea-pop folie, matt/gepr. 500) to test the diffusion in relation with the distance of the LED light from the outer surface. The closer the LED bulb was to the outer surface of the foil, the smaller its diffusion was, while when moved further away its diffusion grew. However we still perceived the light as a 'dot'.

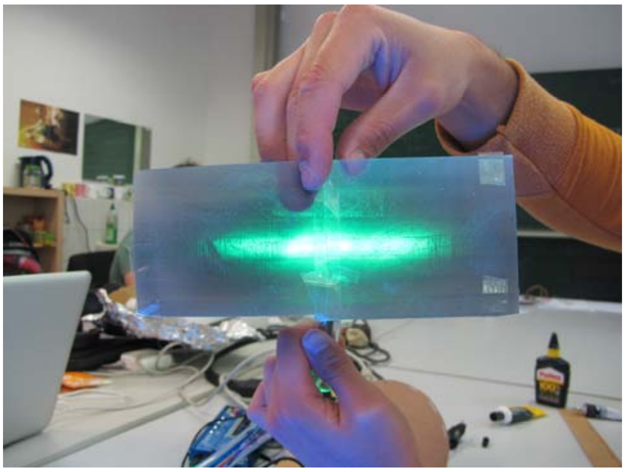

Figure 1.3: Horizontal Diffusion Effect.

The next attempts focused on different materials for the diffusion part, with the main focus on a custom made 'foil' created by polyurethane liquid that was cast on a 3d printed mold. Because of the linear print of the 3d printed surface, the foil appeared to have horizontal striping. This characteristic gave the diffusion a horizontal effect that we found interesting and adapted in our design. Other experimentations included adding color to the resin, as its original 'color' was transparent. We used blank ink to try to give the diffusion a darker shade that would make it fit in better in the urban environment. The result was a shade of grey that did not affect the diffusion result.

By that time the form of the case had acquired a cylindrical form that followed the form of the pipe. We considered various fabrication methods for the production such as molding, vacuum casting, vacuum forming, silicone brush on mold and 3d printing. Another idea included assembling piece by piece. However in the end we concluded in the idea of a folding case that will be combined with the diffusion material.

As far as the attachment methods are concerned, the research started in the early stages of the project. Clips, hooks, mounting tape, magnets, clamps, vacuum suction cups were some general ideas. For the attachment on pipes, one early idea was using a continuous metal wire that would serve as the skeleton of the structure while at the same time connect it with the pipe. The next idea was using a hose, which would incorporate the LED chain and it would be tied on the pipe.

Summaery Prototype

The prototype that was presented in the Summaery exhibition served as a grand rehearsal for the final model. By that time we had finalized the form we aimed for as well as the materials we wanted to use. However for the exhibition due to time restrictions we ended up using different materials, cardboard for the case and the same transparent foil we used in our previous experiments for the diffusion.

In order to complete the final design of the folding module, we used the help of 2D and 3D computer-aided design (CAD) software, Autocad and Rhinoceros 3d, and the parametric program, Grasshopper.

To finalize the folding case we went through a number of variations until we reached the optimal. The crucial points were the stability of the case when folded, the attachment with the LED chain, the attachment with the pipe and the connection with the diffusion material. The size of the case was decided based on the optimal distance of the LED bulb from the outer diffusion surface, in order to accomplish the best diffusion possible. The folding of the vertical and horizontal parts was done with the help of interlocking joints. While the connection with the diffusion material was done by using notches. To connect the case with LED chain we came up with the idea of a profile that would incorporate the chain and would be clipped to the cases. To attach the case to the profile, small parts from the case are bended to a vertical position so that they can be clipped to the profile with the help of metal rivets. Last but not least, the attachment on the pipe was achieved by using Velcro tape.

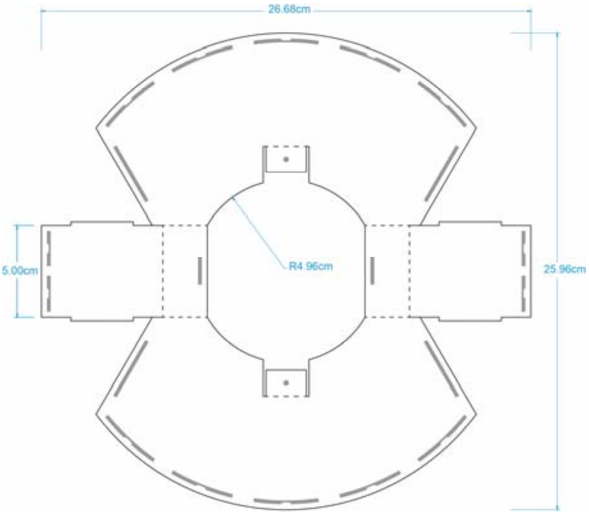

Figure 1.4: Module Layout.

To produce the cardboard prototype for the summary we used a Laser Cutting machine. The process included first designing the pieces in Autocad, then exporting them to Illustrator, with special care on the order they were transferred. The order is significant as it is the same order that the cutting machine will follow, and time is of essence, since the cost . The cutting includes three stages, the engraving, inner lines and outer lines. Therefore the design is organized in respective layers. To prepare for the cutting the cardboard needs to fit the Laser Cutting machine. When the material is inserted the cutting begins. First with the engraving, followed by the inner lines and finally the outer lines.The profile for the chain was also cut with laser Cutting. The semi transparent diffusion foil (Hobby Fun, Crea-pop folie, matt/gepr. 500), however, was cut by hand, as it was not suitable for the Laser Cutting machine.

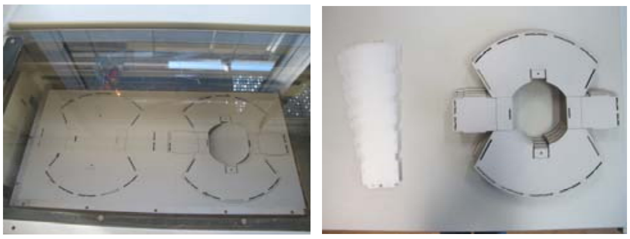

Figure 1.5 a, b: Laser Cutting Process.

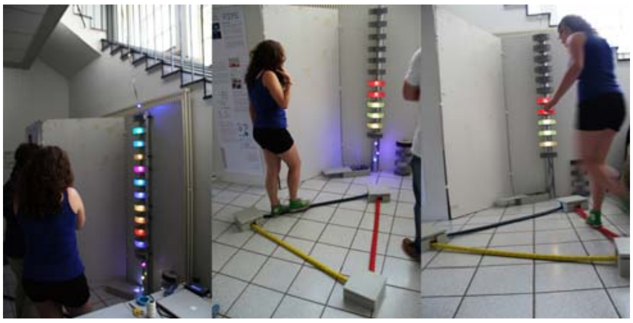

Figure 1.6: Summaery Exhibition.

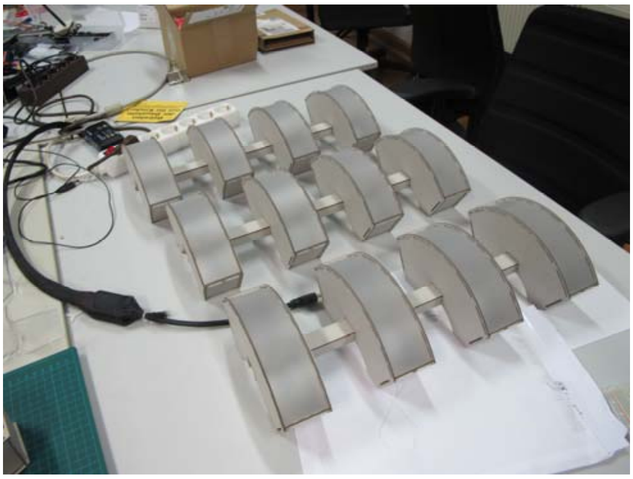

Figure 1.7: Summary Prototype.

Final Prototype

After the summary the final model of the installation followed. The final materials that we used were polycarbonate sheet of 2,00 mm for the case and a lens type light shaping film (Makrofol LM 297 H-M 040007) for the diffusion, which would give us the horizontal effect we were aiming for. Because of the use of different materials some alterations were made to the design, in order to fit the new material.

The fabrication methods we considered were Laser Cutting, Waterjet Cutting and CNC Cutting, with the latter being the one we chose. The process included again design in Autocad and processing in Rhinoceros 3d. In order to achieve the best result, we cut three prototypes, and every time improved various components such as the size of notches and holes and the engraving depths that were needed for bending/folding the model.

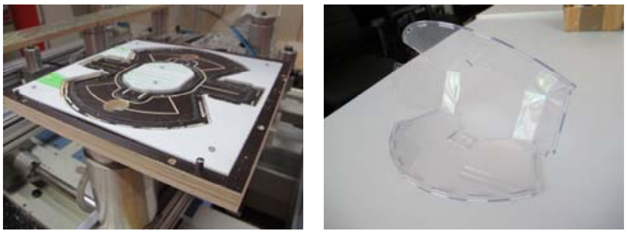

Figure 1.8 a, b (from left to right): CNC machine principle base, custom made to fit our prototype and used for all the modules and Final Product.

The challenge in this prototype was the bending of the polycarbonate sheet. To help the bending we engraved the edges that had to be folded. After many trials we managed the optimal ones. Engraving depths of 0,5 mm and 0,8 mm were not functioning, 1,0 mm and 1,2 mm were working. We ventured different methods for bending the cases, like hot line bending and cold bending with a break bench. However due to various functional restrictions we ended up bending the cases manually. For the hot line bending, due to the shape of the module, certain sides could not be heated without affecting the others. Also, polycarbonate sheets can be cold-bent up to 90-degree angles, depending their thickness [7]. However, as our sheet was 2 mm thick, it resulted in cracks and breakage at the time of bending.

Again the diffusion material was cut by hand. For the LED chain profile, we used a plastic cable channel (PVC Electrical Cable Channel, 3,00 cm und 1,2 cm 200 cm). Finally, as the polycarbonate sheet was transparent, paint was used to give it a more fitting color.

INPUT MODULE

Goals

Nowadays, the digital design of public space is still a difficult task. We intended to develop an interaction scenario where the technical part is integrated in the public environment. In this project we focused mainly on walkway-situations.

The system should consist of several wireless modules, containing sensors and actuators, which can be used in a large field of about 50*50 meters. We tried to create an intuitive design that is easily explorable. Our goal was to develop a multi-user setup that creates social interaction, especially shared encounters.

General Idea for Input Module

In order to create an installation that stands out, we wanted to avoid using conventional means for entering input. Buttons for example, only allow specific, predefined input values; therefore, they lead to a very limited interaction. In contrast, our module should enable different kinds of interaction, and also consider the amount of force applied by the participant. Furthermore, we wanted to create a visible interaction that one can see from afar in comparison to pushing a button, which requires only little movement.

Soon, these requirements prompted the idea of a tube filled with air lying on the ground. Pedestrians interact by stepping on this tube, thus changing the air pressure inside. Instead of just pushing a button, they can use one foot or both feet, jump or just gently touch the input module. These diverse methods should result in different outputs.

Before developing the exact interaction design, a first prototype was built to determine if these interaction methods even lead to distinctive output values and how reliable these measured values are.

Technical Realization

For realizing our general idea of a pressure-sensitive tube, several electronic parts are needed: a pressure sensor, motorized pump, panStamp, battery and battery board. *(see Figure 2.1)*

As mentioned before, the system should notice changes in the air pressure inside. Therefore, a sensor is required to measure the pressure. We used a piezoresistive transducer, the MPXV5050GP [8] which is handy because of its size and its optimization for microcontrolled applications.

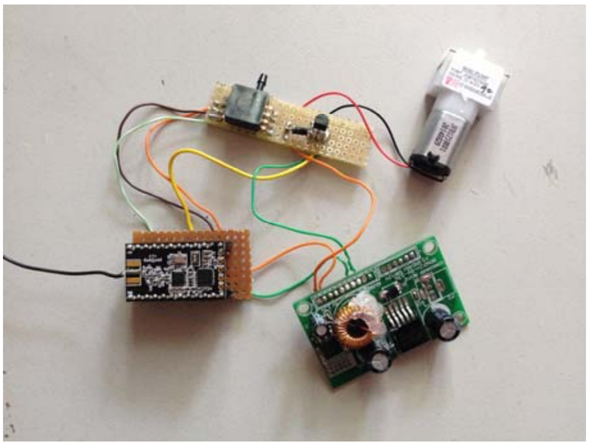

Figure 2.1: The wiring of the motor, battery board, pressure sensor and panStamp (from left to right).

The use of the pump is also straightforward. It fills the tube until a certain amount of air pressure is reached (as measured by the sensor) and then turns off. In case of decreasing air pressure (e.g. because of a leak or temperature changes), the pump turns on again and refills the tube.

To control the motor, a bipolar junction transistor (BJT), an electrically operated switch activated by the microcontroller, can be placed behind it. A diode should be placed across from the motor to prevent reverse currents. *(see Figure 2.2)*

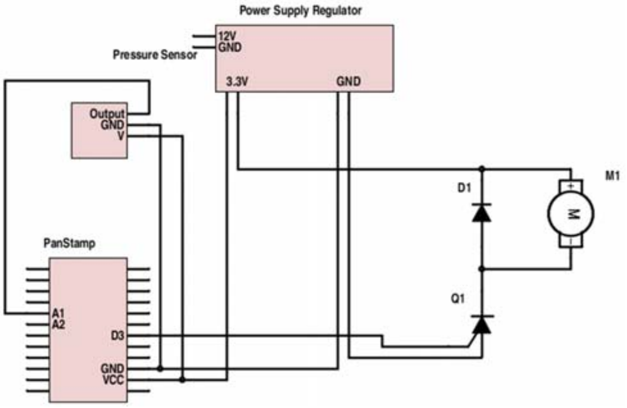

Figure 2.2: The schematic circuit of the wiring.

Initially, the motor and the pressure sensor were attached with tubes to a Y-shaped connector part; because they shared the tubing, the pumping generated 'fake tube steps'. This issue could be solved by routing two separate tubes into the input module.

The panStamp [9] is a low-power wireless multipoint control unit (MCU) with a transceiver. In our project, it sends the measured pressure values from the input module to the server, where they are interpreted. The use of the panStamp allows us to be flexible both during and after the design process.

In the beginning, the sensor and motor were attached to an Arduino. Because of the panStamp's compatibility to the aforementioned, the initial sketches could be transferred, only requiring small adjustments. Each of the tubes is attached to its own panStamp; therefore, the number and the arrangement of the tubes is not fixed and can be adjusted according to our findings in prototype testing.

After completion of the input modules, they are deployed to the street. When walking around the city, we noticed that the available space on the walkways differed greatly, ranging from 1,37 to 6,64 meters in width. On a broader walkway, the tubes can be placed farther apart, whereas on narrower walkways they are arranged more closely. Furthermore, the distance between the in- and output module can be adjusted as well according to the situation of the street - the input module can even be placed on the other side of the street, if needed. Moreover, the transport of the whole installation is also immensely facilitated, as each module can be carried on its own.

The battery also supports the flexibility provided by the panStamp. It must be connected to a battery board to convert the 12 V provided by the battery to 3.3 V needed by the panStamp, pressure sensor and motor.

First Prototype

On basis of the interaction design, we started to build the first working prototype. We did a material research to find a good tube material for our input modules. We compared different tubes by length, width, robustness and form.

We wanted the input modules to be big enough for two people to interact on one module at the same time. For appropriate measurements, it is necessary to follow the rules of proxemics [10] because they categorize the different types of personal space. The personal distance, in which people interact with good friends and family, is defined in the close phase from 46cm to 76cm and in the far phase from 76cm to 120cm. But as we don't want only friends to interact with each other, we also need to consider the social distance, that is defined 1.2m to 2.1m in the close phase and 2.1m to 3.7m in the far phase. Because of the fact that our interaction should be located on a walkway, we don't want our input modules to be too large. Otherwise they would block the whole walking space. Consequently, we decided for a length of 1.5m.

In terms of width, we wanted the tubes to have enough interaction space, but also keep them narrower than the length of a foot, because we wanted people to be able to fully step on the tube and therefore divide the air circulation inside of it.

Furthermore, we needed a robust tube material. The whole installation was going to be tested outdoor by day as well as night. So we planned on having people jumping on the modules, maybe even with sharp stones on the sole of their shoes. Additionally, there could be splinters of glass on the floor. Therefore, the tubes had to be relatively cut protected.

Another point to consider was the form of the tube. Usually people avoid to step on tubes, because when someone sees a tube on the floor he usually has the metaphor of a garden hose and does not want to interrupt the flow of water inside of it. But in our case, we want to animate people to step on it. One point to achieve that is to decide for a flat kind of tube to abstract the look away from the classic garden hose.

For our first prototype we chose the tube "Hi-flat hd" from HiFitt [11]. This is a PVC flat hose with textile reinforcement. The hose is usually used in the agriculture and according to it's fabricators information very flexible and robust.

In order to get the tubes filled with air and measure the air pressure inside of it, both ends have to be closed completely airtight. After testing a lot of different types of glues, clamps and even welding we decided for a combination between glueing and clamping.

Finally, all parts above-mentioned in *Technical Realization* are assembled and placed inside of a wooden box (30x30x12cm) at one end of the tube. On the other end, another pressure sensor was attached for the initial prototype.

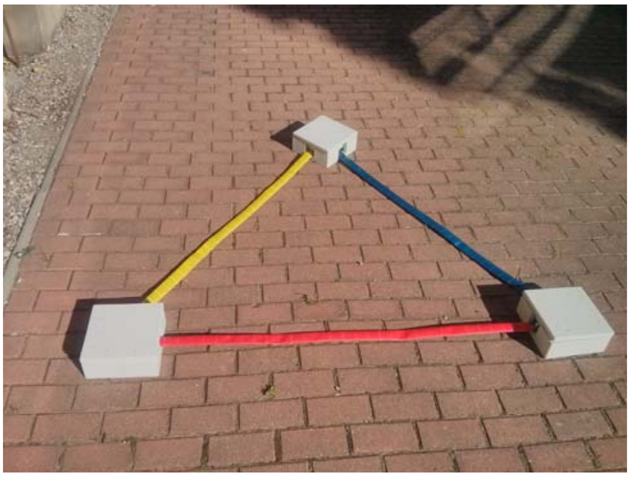

Figure 2.3: All three input modules are arranged in a triangle shape.

Interaction Tests

After completing one pressure-sensitive tube, it was time to assess the input module. The test should answer some of the remaining questions, such as: How sensitive are the sensors? Is there any scenario where the two sensors measure a different value? What interaction techniques are possible? Can one recognize if more than one person is using it at the same time or if a specific step technique is used?

The test setup was relatively simple; the test tube was lying on the ground and each pressure sensor was hooked to a computer where the measured data was first displayed as a numerical value on the screen. Later on, both sensors' outputs were combined in a graph using a Processing sketch

[12] *(see Figure 2.4)*. The changing values were observed real-time; therefore, we could directly see the impact our interaction methods had on the tube. We were two participants, so that single- as well as multi-user interaction could be experimented with.

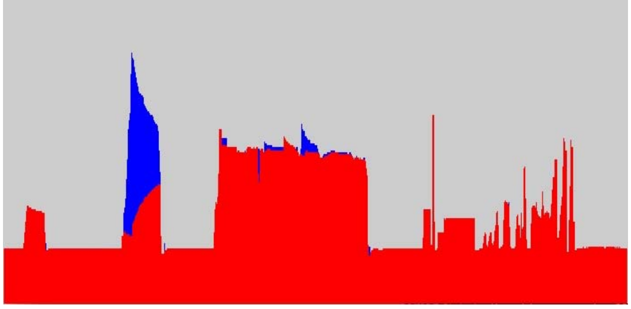

Figure 2.4: Processing output for vertical, horizontal step technique and using both feet.

Our procedure was to start slowly with one participant and simple interactions and gradually build the complexity by using another foot and/or bringing in the second participant.

In the beginning, the participant gently pressed the tube with one foot and increased pressure. He stepped on it vertically, rotated 90 degrees from the length of the tube *(see Figure 4a)* and horizontally, following the direction of the tube *(see Figure 4b)*. With the vertical step only parts of the foot touch the tube, whereas the horizontal step requires the rolling of the whole foot.

Afterwards, the participant began using both feet *(see Figure 4c)*, stepping on the tube one foot after the other. While standing on the tube, he applied more force by rising to his tiptoes. Then the participant jumped on tube with changing time intervals between the jumps.

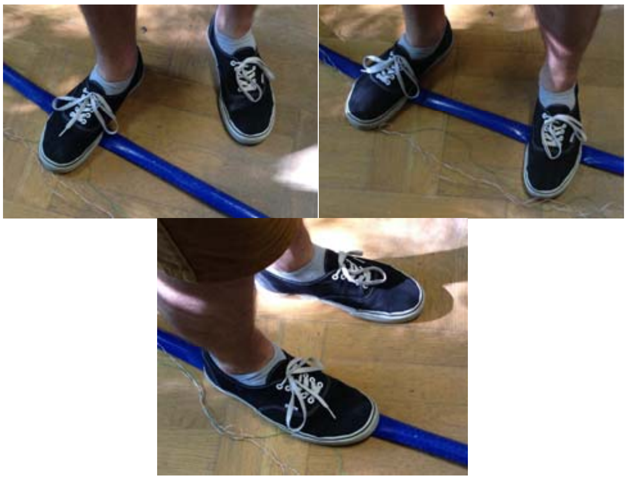

Figure 2.5 a, b, c (left to right): The vertical, horizontal step technique and using both feet.

Next, the other participant started interacting as well. The interactions were variations of the methods mentioned before; sometimes, both of the participants used the same techniques, while at other times, they used different ones.

With these changing interaction patterns, we tried to simulate the ‘random’ behaviour of a pedestrian walking past our installation.

The conduction of the experiment lead to many interesting findings, both convenient and inconvenient for the later interaction design.

First of all, the pressure sensor is working very accurately, as even slight differences in pressure are noticed. It also reacts very fast; consequently, even jumps could be part of the interaction pattern.

The highest pressure values can be achieved by closing a part of the tube completely and then adding force to the other, because the size of the area that pressure is applied to is much smaller. When using this technique, the pressure sensors also yield different pressure values. However, it is not always possible to completely divide the air circulation, depending on the weight of the participant and initial air pressure on the tube.

The other technique to obtain differing sensor values is to use the whole foot in a roll-movement, which is much more reliable.

Apart from the aforementioned methods, higher pressure values can be reached by adding more weight. Consequently, a heavier person or multiple interactors lead to bigger values.

Interestingly enough, the position on the tube where the interaction does not influence the value at all. The only exception to this rule is dividing the air circulation with one foot; the side on which the second foot steps results in a higher pressure value. Again, this method is rather unreliable.

In all the listed examples, there are two or more methods to achieve a specific pressure value. This ambiguity complicates the interpretation of data significantly. Basically, it is easy to predict the outcome of a step technique, but because of the many similarities it is very hard to conclude which method was used to achieve this particular result.

Interaction Design

Initially, each different interaction method should lead to a different outcome. But as the test results showed, the unique patterns (e.g. two different sensor values) do not always occur, making the discrimination very unreliable. If we decided to still include this information, sometimes the systems would recognize that more than one foot is used, other times it would not. In that case, the same interaction could prompt contrasting outputs. This seemingly random output could confuse the users and complicate the process of understanding the interaction rules. Ambiguity can be used to engage people more deeply with a system by encouraging them to interpret the situation for themselves [13]. However, in this case, we felt that the confusion would outweigh the advantages, thus making a simple interaction design more favorable.

As sensing small pressure changes turned out to be the most reliable aspect, it was decided to base the interaction design around it. When a person steps onto one input module, the pressure value will be transmitted to the server. There, the colours of all tubes will be mixed according to their relation to each other. For example, two people standing on yellow and one person standing on blue would result in a light green.

After deciding to only use the pressure values, it was unclear if it makes sense to keep both of the pressure sensors. Keeping in mind that our output should be as consistent as possible, we agreed to select the higher of both values. Thereby, the chance of having a completely closed tube that does not notice when there are more people on the other side is eliminated.

One of the main goals is to encourage multiple users to interact with the installation. For example, we could introduce a threshold for the pressure values; once it rises above the system knows two persons are using the tube. Unfortunately, two children playing with the tube possibly cannot amount to the needed values. As a result, the server should determine that multiple-users are interacting with the installation by not only looking at one input module but all three of them.

The tubes are arranged in the shape of a triangle, so in order to blend two or more colors users have to work together. The triangle was chosen because when several persons are using the modules at the same time, they are more or less forced to face each other. According to observations conducted by Kendon [14] , people engaged in a conversation naturally arrange themselves into a circular shape. Therefore, they are creating a shared inner space for their interaction, which is distinct from the surrounding space. By encouraging strangers to stand in this intimate arrangement, we hoped to trigger more shared encounters.

All in all, our interaction design is very simple. Users can discover the input modules on their own, trying stepping techniques and then seeing how they can have the largest effect on the mixture with their color. By just focusing on the pressure values, the design is intuitive, so even pedestrians with only little available time can grasp the general concept. Users that wish to interact longer can team up and try to mix a particular color together.

Findings on the Summaery

The Summaery was the first public test for our modules. A detailed evaluation of this event can be found in the chapter ‘Evaluation’.

The most interesting fact for our input module was, that some people did not understand where to interact. In consequence, we had to think about a new form of the tubes and the boxes to make it more clearly where to interact.

Besides that, we realized our hardware setup is working well. So our development of the final prototype could be mainly focused on the appearance of the input module.

Final Prototype

At the Summaery, the first public test for our modules, some people had difficulties understanding where to interact. They thought the interaction happens with the boxes and the tubes between them are just for connecting them. In consequence, we had to think about a new form of the tubes and the boxes to make it more clearly where to interact.

As the hardware worked well, for our final prototype we were only rethinking the design of the input modules. To make it more clear that the interaction happens with the tubes, we decided to increase the width of the tubes and to produce smaller boxes. Since there is nothing wider available in the section of garden hoses and similar tubes, we had to build them by ourselves. We used a sheet of rubber for fabricating new input modules. Our idea was to cut out two similar sheets (30x120cm) and glue them together. On one of the short sides we glued in a connector-piece, where the pressure sensor and the pump can be attached. We fabricated that connector from cast resin with the help of cardboard templates *(see Figure 2.6 a,b).*

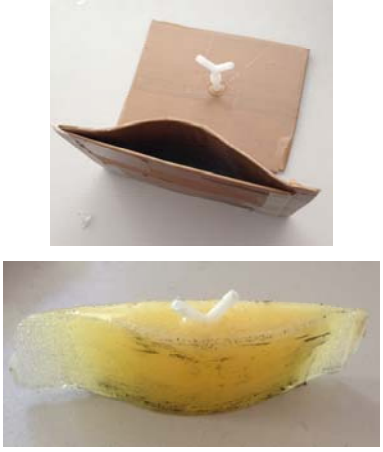

Figure 2.6 a, b: Cardboard template and finished cast.

For glueing, we chose REMA TIP TOP Cement SC4000 and E40 Hardener [15]. The problem with this procedure was, by inflating the tube, it evenly deformed to the bottom and the top. Therefore it was not laying solid on the floor and through this behavior it is very unpleasant to step on. Our solution was to make the bottom of the tube rigid. For that we chose a thin wooden board and glued a single rubber sheet on it. That made it much more stable and comfortable to step on the tube. Another change to the first prototype was, that we were now using only one pressure sensor. With the old tubes there were scenarios where it could have made sense to measure on both ends of it, but now the input area is so wide that it is very unlikely to create such a scenario. Furthermore, we built new smaller boxes. While the old ones were about 30x30x12cm, the new ones were only 30x12x6cm. By changing the scale between boxes and tubes, we made clear which is the part to interact with.

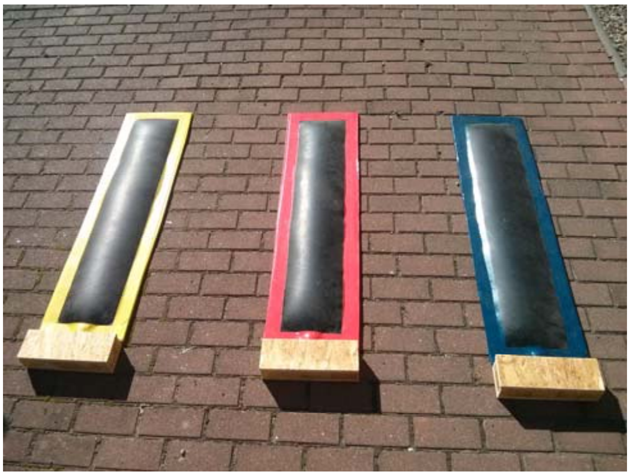

Figure 2.7: All three input modules can be arranged independently.

ORCHESTRATION SOFTWARE

Purpose of the Orchestration Server

The main purpose of the orchestration server software "RokkaSpot" is the management of one or more devices. A device is a collection of individual modules. The devices and modules exist in hardware as well as software. A module has a specific functionality, such as receiving input (e.g. using different sensors) or manage output (e.g. using actuators, LEDs, speakers). PIPE is a single device containing three pressure-measuring input modules and one LED-chain output module. The use of modules allows to easily replace parts of the device, resulting in faster iteration cycles in development and testing. Each module can be tested seperately and different combinations of modules can be tested easily, too.

The available communication protocols are based on the APIs currently integrated into the system. At the start of the project, it was limited to XBee wireless transmission [16]. As part of the project, panStamp technology is now also functional with RokkaSpot.

The software is built around module implementations, which store the data belonging to the hardware module and apply functionality, and UI views, which allow to manipulate the data by using a graphical user interface and trigger the execution of commands. These commands are implemented on the hardware side though. The orchestration server is not a tool to upload code or executables onto the hardware. It may only send messages to trigger given functionality of the hardware or receive messages from the hardware.

The server is meant to work as a connection between different modules, allowing them to interact in more complex ways with each other. The processing power of the hardware modules is usually limited. Using a PC or different high-end processing unit increases the possibilities of interaction between the hardware modules notably. This includes applying more complex algorithms like color conversion, audio analysis or pre-rendering of images.

Basic Flow of the Orchestration Server

This code flow description is based on the orchestration server version existing at the end of the project. The former version worked similar, yet with some small differences. These arose because of the introduction of interfaces and a mediator, as described in the following chapters.

The UI initialization and general application setup is done by an implementation of *IRokkaSpot*. The java main-call can be found separately in *PlazaPuckMain*. The current *IRokkaSpot* in use is *PlazaPuck2*. It initializes and spreads the mediator, builds the main UI using SWT and does the hardware code setup.

The first step of the user should be to start a device server. He may choose between an XBee-server (*XBeeServerV2*) and a PanStamp-server (*PanStampServerCC1101*). When doing so, the proper *IDeviceServer* is set in the mediator and started. Once the server is running, it may send and receive data. To do so, the address of the hardware devices must be known. These are stored within *IModuleInfo*-objects, which are created at runtime. There are different ways to gather the currently available modules in wireless reach. One is to send a message to every device in range and wait for an answer that includes the address of the device. This approach is used by the *XBeeServerV2*. Note that this method needs some code on the hardware modules that will answer the call of the server and send back the address.

The user can now create a device and add some modules using the UI. This will internally create SWT-composites out of the package "view". These composites visualize data received from the modules and/or provide buttons to send commands to the hardware. Note that an *IDevice* manages multiple *IModule*, with each module owning an *IModuleInfo*. These module info can be used to communicate with the hardware, while the module itself stores data and functionality. The device manages the modules and allows data transfer between them.

The hardware communication is done by the *IDeviceServer*, which is accessible to the views and modules per mediator. There is always only a single mediator instance, which holds the reference to the current device server. The device server sends all the messages and also receives all messages, yet modules may register a special listener to get notified on incoming data, too. This listener is specific to the used API though.

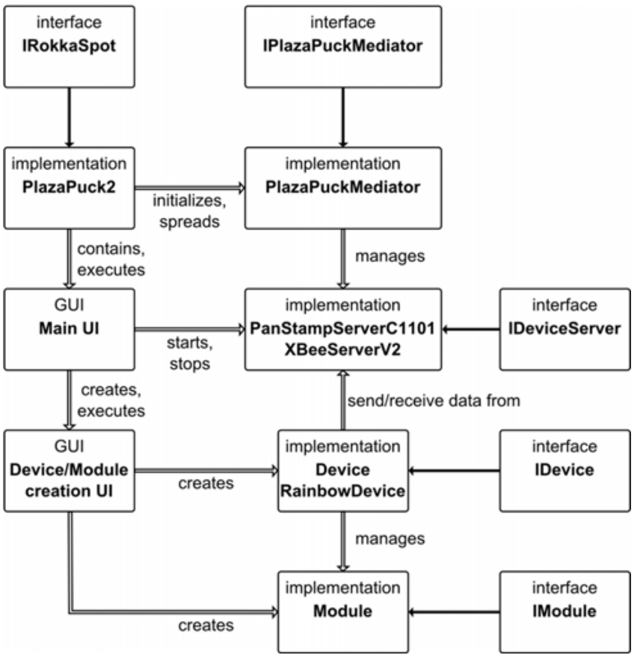

Figure 3.1: Simplified code flow in the orchestration server when setting up devices and modules.

Introduction of Interfaces

A class overview was created to analyze the existing code. This overview visualized an almost complete lack of abstraction using abstract base classes and interfaces. One of the main goals of the project was to allow the use of PanStamp along XBee technology, which use different drivers each. To achieve this, the code structure had to be changed to a higher degree of abstraction that would allow the replacement of the hardware drivers and corresponding APIs within the orchestration server with minimum effort.

The core of the hardware specific code was once the Device*Server* in the package “io”. It dealt with most of the XBee specific code, yet other classes used technology specific calls, too. To allow replacement of the Device*Server*, an interface *IDeviceServer* was created. The older implementations were renamed to *XBeeServer* and *XBeeServerV2*, while the new implementation would be the *PanStampServer*. Because of the use of multiple PanStamp APIs, this class would later be split into *PanStampServerLeGrange* and *PanStampServerCC1101*. The lack of any interface resulted in the other classes referencing the *DeviceServer* directly. A lot of classes had to be altered to use references to the proper interface instead.

More hardware specific code was found in the package “plazapuck.model”. Again, the missing abstraction made the introduction of interfaces a necessity. A core problem proofed to be the *ModuleInfo*. It stored API-specific data like device address and signal strength and used XBee-specific parameters. After creating *IModuleInfo* the code could be split into the hardware dependent *ModuleInfoXBee* and *ModuleInfoPanStamp*. The module info is part of the module, which describes a single part of independent hardware. It is realized via the class *Module*, which is now also abstracted by the top-level *IModule*, allowing diverse implementations. Multiple modules form a device, which is now divided into proper levels by *Device* and *IDevice*, too. There are also some container classes managing *IModuleInfo* and *IDevice*, which are now *IModuleInfoManager* and *IDeviceManager* as well as their current implementations *ModuleInfos* and *DeviceManager*. Some special implementations of modules to be used with specific APIs can be found in the packages “plazapuck.model.xbee” and “plazapuck.model.panstamp”. Note that these use inheritance to add the specific features of their APIs to the neutral module implementation found in “plazapuck.model”.

The code doing the program initialization and basics was loose. A new package “main” was created. This now holds the executables, config and logging code. A problem was the config-file system. There was some dead XML code present as well as a binary file implementation. Using a interfaces and cleaned up code, the new interface *IConfig* as well as the classes *XmlConfig* and *PropertyConfig* resulted. The log system was another small yet important part that was once hardcoded to use the log4j-library. It was then split into *IlogWriter* and *Log4JWriter*. Interestingly, the log system was not used a lot in the code anyway. Newly written code during the course of the project used the log system, but many old functions often simply used *System.out.Println*. A last change in the “main”-package included the abstraction of executables and extraction of the main-method. The main-method is now in a separate file, *PlazaPuckMain* (name of the old project), and uses an object implementing *IRokkaSpot* (name of the orchestration server software) to execute. This creates a standard among executables and forces some basic structure.

The changes to the core structure of the code were rather huge and there is a chance some references have not yet been updated properly to use the interfaces instead of the implementations. When continuing the project, interfaces and abstraction should be used continuously, since they allow easier code extension, replacement and repair. Also, abstract base classes may avoid redundant code and are a far cleaner solution opposed to copy and paste.

Introduction of a Mediator

A huge problem of the original code was the strict and rigid structure. Usually, in a project of this size, the general architecture should aim for a loose binding of the classes to allow adapting the code as well as possible. The rigid structure and cross-references decrease efficiency of multiple programmers working at the same time, because a change in one class usually affects multiple other classes. Increasing the level of abstraction is only a part of the solution, the cross-referencing requires a structural change in the general reference management.

This led to the introduction of the “mediator”-pattern into the existing structure. For consistency and maintenance issues, the mediator is implemented against an interface, *IPlazaPuckMediator* (naming convention is actually a little weird at this point, *IRokkaSpotMediator* would make more sense as well as *IPipeMediator*, because it was created to be used with PIPE). The implementation used within the project became the according class *PlazaPuckMediator*. The mediator allows to access the most important components of the software system, creating a single global way of referencing critical data and functionality. Instead of cross-referencing objects, each object in need of critical data has access to the mediator. This builds up a simpler software structure which is build around a single managing object in it's center.

The mediator needs a way to be accessed. Often, mediators use the “singleton”-pattern to allow global access. Yet this would not allow a simple replacement and neglect the idea of abstraction. This is why a single mediator object is created and passed along to the objects using it. To allow simple spread and use while keeping the reference to the interface instead of a class, an interface was introduced which is implemented by any class using the mediator. It is *IPlazaPuckMediatorAccess*. Classes including this interface have access to a getter and setter method, allowing other classes to set the mediator (if they own a valid reference) or get the mediator (to use it or receive a valid reference). The abstract class *BasePlazaPuckAccess* exists to avoid code redundancy where possible. Because Java does not allow inheritance from multiple classes, the use of *IPlazaPuckMediatorAccess* is more common though.

Note that the mediator does not manage every single object in the whole application, but only a few objects that are used frequently. These include the current *IDeviceServer*, the *ILogWriter*, the *IConfig* and the *IDeviceManager*. The mediator uses the hardware abstraction introduced in the previous chapter to allow a simple replacement of the used API. For example, if currently a *XbeeServerV2* is in use and the software needs to switch to a PanStamp version, only a single object has to be replaced. Because the current *IDeviceServer* is only referenced through the mediator, every subsystem in the application will automatically use the replaced PanStamp-server. This central and abstract structure core allows to maintain, replace and add features more easily and also enables multiple programmers to work at the code at the same time without interfering each other.

The mediator is not the optimum solution, though. A better approach would be to use a communication framework based on events and event listeners, including a central event manager transmitting the events to their destinations. These systems are common in frequently changing structures with varying size, like modern game engines and servers. The advantage is that any system can be added or removed at any time because of an almost optimized loose binding between the system components, managed around the event manager, which remotely replaces the mediator. Yet implementing an event based communication structure between the objects is not a simple task, especially if the system components already exist and do not follow any given standard. The mediator was the easier solution and, given the circumstances of time, working power and existing documentation, the only feasible option.

The mediator tends to continue growing over time and creates a single large class while the application adds up features. The event management does not need to be altered as the project advances, which is one of it's main advantages. Also, the use of the mediator creates a strong connection between the objects managed by the mediator and the objects accessing the mediator. Event based communication built around callback methods is far superior in terms of loose binding, because if a system is removed or changed, no other system is directly influenced. A switch to event based communication instead of direct referencing might be a good idea if the orchestration server software is about to implement even more features.

Use of Observer Pattern to replace SWT-Bindings

SWT contains a feature to bind the UI to specific data. Once these bindings are set up, changing a value in the bounded object leads to an automatic update of the UI. These bindings were used in some of the views of the orchestration software, yet problems arose when interfaces and the mediator were built into the core structure. Unfortunately, the bindings seemed to be incompatible with inheritance (which includes the implementation against an interface). The bindings had to be replaced in some cases.

As easiest and most versatile solution the design pattern “observer” was chosen. The data became *Observable*. Once the data changes, it sends an update request to it's observers. The *Observer* is the UI, which repaints and/or updates specific UI elements depending on the data changed. This simple approach works rather well and is easy to maintain. Since the basic pattern is already given in Java, it is also very simple to apply.

Still some problems exist. As a known issue, performance slows drastically if the UI-update is called too often or in too close intervals. The update-calls should be optimized as much as possible to avoid lag in the UI. Whether this is a general problem or special to SWT is currently unknown. Another problem is the update of the relationships. Some objects that are dynamically created need to be added as observers and removed later on. Forgetting to do so may cause malfunction or exceptions. One can easily lose overview of the current state of observables and observers. Introducing a global management could be a possible solution. A custom and improved observer implementation might be another option. The current system works, but it is only used in a few places. A general replacement to the SWT-bindings needs to be developed, yet the observer-approach might be a good base. The event based communication system already discussed would solve the problem without having to introduce any additional data-binding feature. The UI could simply listen to value-update-events and react accordingly.

PanStamps

Beside the implementation of an orchestration server, the second part of the project was the development of a public installation out of different modules based on a radio infrastructure managed by the mentioned orchestration server. To reach this goal the choice fell on panStamp microcontrollers in addition to the already supported XBee modules.

PanStamps are microcontrollers based on the arduino platform produced by panStamp S.L.U. [17]. They support general arduino features like digital and analog read/write to different pins and onboard computations and are designed to be small and energy efficient. Their main difference to arduinos is the integrated radio chip, which provides the possibility of wireless communication between different modules. More precisely the used micro controllers are panStamp AVR 2 and they contain an Atmega328p MCU and for radio transmission a Texas Instruments CC1101 RF interface [18]. The panStamp itself has 10 digital an 8 analogue pins that can be used for programming in the Arduino IDE with a slightly different naming [19].

With dimensions of 17.7 x 30.5 mm it is very small and suitable for embedded and mobile systems with needs for radio-based communication and lightweight design. Depending on the particular use case, panStamps can be very energy efficient. They can be set to a “sleep-mode” and just wake up for sending sensor-values, which leads to battery life times of more than one month. With a higher frequency of transmitting and receiving data the duration gets shorter but is even suitable for mobile devices.

Radio Communication

Supported by the CC1101 chip, panStamps are able to communicate wireless via radio transmission. For the use in panStamps CC1101 chips can handle two frequencies: 868 MHz and 915 MHz for sending and receiving data, assumed the microcontrollers are equipped with an antenna in the correct length of 164.3 mm or 155.9 mm.

With a range up to 70m panStamps are appropriated for the use as controlling units in mobile or spatially separated modules.

PanStamps can be programmed with the common Arduino IDE. Due to better handling of hardware based code for the CC1101 chip, there exists the “LowLevelLibrary” within the panStamp-Packages for the Arduino IDE.

This library contains all necessary commands for sending and receiving data with the C1101 chip. Therefor it provides an own package format (CCPACKET) and different solutions for easy settings of frequencies, transceive and receive rates and addresses.

As extension to this low level API just with the most important functions for radio based communication, there exists the so called SWAP API. This one is based on the SWAP Protocol.

SWAP stands for Simple Wireless Abstract Protocol and this protocol is mostly common for its use in home automation and provides a more complex infrastructure.

In brief the base for communicating with the help of the SWAP Protocol are global registers. These are network-wide uniquely identified units of information, which can be set with different data values, like measured sensor values, I/Os or configuration parameters. Historically this register approach was developed and inspired from Mobus, an abstract protocol from the 1970s with proven flexibility and adaptability for almost any type of application. The idea of abstract registers provides a way to transmit data independently from its type and origin. Without concerns about functional profiles or restrictions in data types, the SWAP Protocol has a very small firmware footprint and enables easy implementations [20].

SWAP describes 3 different message types: Status Packet, Query Packet and Command Packet.

Status Packets are always sent if a register value changes, to inform the network about the actual value. Also they are the response for a Query Packet containing the queried register value and after the receive of a Command Packet a Status Packet is sent to inform about the changed register. Query Packets are used to query values from specific registers on a remote device and Command Packets are used to control values of a specific register.

This provides easy to use functionality to check sensors from remote devices. For example if a register is updated with new measured sensor values, automatically a Status Packet is send. In combination with the well working sleep function of the SWAP-library this offers options for really low energy applications, such that the device just wakes up to update the register and automatically sends them via a Status Packet to a modem and just returns to the sleep state with a very low power consumption.

Due to its common use in the sending and collecting of sensor values from different modules to one modem and its possibilities to control and set registers over the air, the SWAP – Protocol was used in the first prototype implementations.

Modem

One project-requirement was the implementation of an orchestration server, which does the computation and forms the "heart" of the communication between the different modules. To reach this goal the server needs a hardware interface to enable communication between the orchestration software and the modules: the modem.

On hardware-side the modem is built out of two components: one panStamp and one panStick, which can physically hold the microcontroller and supports serial communication between PC and controller via USB. This is done by a FTDI FT231XS chip, which serves as a bridge between USB and serial. The modem runs an arduino sketch (*modem.h*) for translating the incoming packages in a computer readable format and vice versa. [21]

Java APIs

Conditioned by the already in outline existing orchestration server, it was necessary to use an API which enables the Java-application to communicate via serial with the panStamp network. For this purpose two APIs were qualified for the use in the project: “panstamp-java” by Gideon LeGrange [22] and a bundle of APIs for different programming languages by Norbert Truchsess and Daniel Berenguer, in particular the “java-swap-library”. [23]

LeGrange’s API is a relatively young project started in May 2014 with some small issues and not much documentation in code and a small how-to and wiki. Truchsess’ and Berenguer’s implementation is an older one with a bit more documentation in code, but non-existing how-tos or wikis and it had no updates since April 2011.

With LeGrange’s library it was possible to communicate with the first prototype modules and to read out and set registers remotely, without drastic changes in its implementation.

Module Software Delevopment

As already mentioned, the aim of PIPE was the wireless orchestration of different wireless-modules. The used hardware modules, more precisely described in previous chapters, were three input and one out module.

In the early phase of the project, the software of these modules were based on the SWAP structure and the measured sensor values were written into the registers and sent automatically via a Status Packet to the modem. Also it was possible to set the registers on the output device to display the colors. This worked well for the use with one prototype. But when testing it with two or more active modules a really extreme loss of packages occurred and a fluent operation of the system was not feasible. The reason for this problem was the high amount of data, which was sent by the network. Due the fast update rates of the input modules of 10 per second and the sending of 10 Packages per second to the output module there were 40 sends and receives every second. SWAP is only implemented in a way such that only one package can be send at one time, otherwise there will be collisions.

After specifying the problems with the SWAP – Protocol some fundamental changes had to be done in the orchestration-server and the modules.

LeGrange’s library provides no obvious solution to directly send CC1101 packages in the correct format to the panStamps, but Truchsess’ and Berenguer’s does this in theory. The infrastructure of transceiving packages was changed to this library. The implementation of the CC1101 code in this API was intended to use just as an interface for the actual SWAP code. For this reason the whole modem (*CcModem.java*) and protocol (*CcPacket.java*) code had to be changed and customized in a way to produce valid CC1101 packages that can be interpreted by the output module and incoming packages can be evaluated by the orchestration-server.

Also the module-sketches had to be changed to be able to communicate via CC1101 Packets. For this the whole sensor-reading process was modified and the structure for sending packets was completely rebuild, too.

PIPE

The corresponding classes in the orchestration server for the developed modules are named *TubePressureModule* and *PipeLEDOutputModule* and they are extending the Module class. The first one consists of an ID to identify the exact input-module out of the three given and one pressure value (to use the first (summaery-) prototype there is a second pressure value). Also it provides functions to get and set the ID and the values. The output-module-class has variables for the used LED-address, a code to run different actions on the output-module and for the color-values. There are options to set the colors and to adjust the minimum and maximum address of LEDs. Also functions to set “timer-blinking”, “wipe-out” if the module is “full” and a theoretical screensaver on the module are implemented in this class. Providing two operating-modes the *PipeLEDOutputModule* can handle colors in RGB as well as in RYB and has the possibility to convert the colors in the needed one. For the use with the actual output-module there is a filter-function to get better colors on the LEDs.

Both classes are extend by *TubePressureModulePanStamp* and *PipeLEDOutputModulePanStamp* to support the PanStamp specific functions for sending and receiving of CcPackets. The input-module implements a *CcPacketHandler* which is called when a CcPacket is received and writes the data to the associated pressure value and notificates the observer about the update. Also it implements a *MessageListener* for the first used SWAP-Protocol. The PanStamp-class of the output-module is able to create a valid CcPacket with all information the output needs: destination- and source-address, the LED-address and function-code and also the color-values. The sending itself is done by the *DeviceServer*.

All modules are combined in a class called *RainbowDevice*, which contains three *TubePressureModules* as Input and one *PipeLEDOutputModule*. This class is responsible for the whole interaction process by doing the computation of the output-values out of the input-data. After it is running, it starts blinking the first LED and monitors if someone is using the input-module. If the default-pressure-value is exceeded it starts the color computation by using the pressure-values and the relation between them, this calculated color is send to the output to get immediately feedback of the actual user-interaction. To reduce the amount of data, packages are only send if the values changed. After the chosen interaction time the last value is locked and will be the color that is shown on the LED-output-module. To inform the user about the next LED it blinks again 3 times and depending on the user's action he can set the color for the next one or steps down from the input module and the LED continues blinking to tell the following operator the current position.

If the highest LED is reached, the *RainbowDevice* sends the command to "wipe-out" the output-module via the *PipeLEDOutputDevice* and starts again.

The whole communication via the CC1101 Protocol, used by the modules, is mainly organized by the PanStampServerCC1101, an implementation of the IDeviceServer. It provides the functions for registration of the *CcPacketHandlers* and sending data.

EVALUATION

In the Wild Evaluation

To evaluate instrumentation in the urban space and how social interaction occurs, we incorporated some of the evaluation techniques to help gather data for our analysis. We used techniques that was suitable in our context to collect data. We categorized the participants with respect to age and as a group or a single individual for our observation and evaluation. Having to evaluate the social interaction in the urban space, what is social interaction [24], Social interaction is the way people talk and act with each other and various structures in society. It may include the interaction a family has together (eating, sleeping, living together) or bureaucracies that are formed out of the need to create order within the interaction itself. Social interaction in psychology [25], defines it as 'Social interaction is a process of reciprocating stimulation or responses between two people. It develops competition, interaction, influences social roles and status and people for social relationships'.

From our discussion as a project team, we derived our perspective of what is social interaction. We find that it is the way people talk with a mutual understanding between each other, a conversation taking place between two or more people, social encounters: request and response, mutual orientation, intentional, verbal or nonverbal actions between two people, and to avoid another individual is also a form of social interaction. Finally, one main efforts of the project was to create "shared encounters", a term well defined by the work of Katharine S. Willis et al. [26]. Shared encounters can also be expressed in different ways in respect to its context, with regard to our instrumentation PIPE, we focus on the urban environment and passageway situations. Considering, that space also changes the interaction amidst individual personage, we had to give in aforethought when placing the modules. For our evaluation we took into consideration of the different factors that we carried out when we observed the participants while interacting with the modules. And also the shared encounters between two different individuals or between two different groups. We also looked into the factor of multiple users interacting on the input module. We setup our modules i.e. 'PIPE' in the night, on the pathway in Amalienstraße, Weimar. We executed the project from 20:00 to 22:00 and we set them up on Tuesday, Wednesday and Thursday to see how social interaction takes places between people on the streets during these days.

Methodology

The data gathering technique that we used was the observation method, where close attention was paid to the users, in order to gather the required data of people interacting with the instrumentation setup. From the different variations in observation [27], we used in the field observation at which point we get a full and true story through observing what the users are doing or how to achieve a task in their natural environment. The other one being, direct observation triggering instantly and observing the situations and taking notes of an interaction that was instantaneous and unique. In addition we also applied the indirect observation where video recordings were collected to analyze later. It is also useful to track users' activities that cannot be present over the duration of the study. Adopting and using notes, interaction logs, photos and videos are several main techniques. Furthermore, we recorded and photographed few instances, where the behavior of the participants were different from other people.

Pilot Study

In order to have a better understanding of social interaction among people, we conducted a pilot study to understand better, how people interacted with each other in urban spaces. As part of the project PIPE, a local public interface was analyzed and evaluated. We made the pilot study at Herderplatz, Weimar where there exists a special interactive water fountain installed by the city. The interactive fountain is a small setting where it has to be activated by putting pressure on a specific stone plate, which acts like a huge button i.e. a user has to jump on a small stone in order for the water to splash out from the ground. An evaluation study was done on this Interactive fountain, so that it can be compared with our own installation of an interactive digital display, "PIPE", to test how much social interaction it produced.

These are the following spaces that were identified for our study for the interaction at the fountain. The Interactive

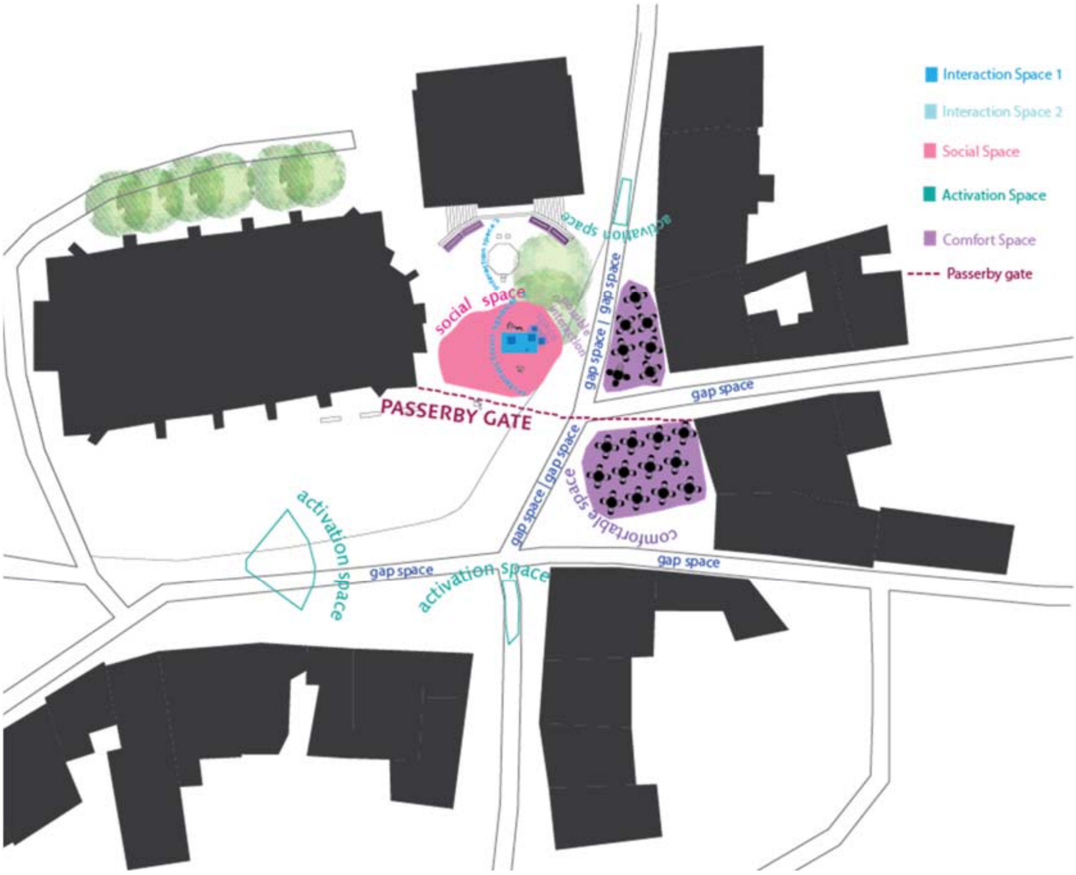

Figure 4.1: Overview of Herderplatz.

Space - where the people interacted with the fountain, Social Interaction Space - where people interacted with each other socially and also observed the interactive fountain, Comfort Space - where people usually were sitting and relaxing irrespective of the fountain, Activation Space - is the space from which the fountain is visible and Display Space is the whole region around the fountain. From the model proposed by Fischer et al. [4.6], we were also able to identify the above different interactive spaces around the fountain, which enabled us to do the observation of the fountain seamlessly. For the pilot study we categorized our users as children, teenagers, adults and elderly .One of the members from our testing team had to note down the number of people directly interacting with the fountain, the number of people who observed how to use the fountain before interacting with it, the number of people who passed by the fountain and had a look at it but did not interact, people who just passed by ignoring that the fountain existed. Hence, depending upon the people's behavior towards the fountain we also proposed four different categories of people as passers-by, observers, participants and performers. Here are a few definition to explain the different interaction zones and modes of interactions.

Performer - The person, who jumps on the brick to interact or play with the fountain water outlet.

Participant- A person who is indirectly interacting with the fountain by just standing in the interaction space.

Observer- The people who are watching closely and observing the activity going on in the social interaction space.

Hidden Queue- People who are waiting for their turn to interact with the fountain.

Courtesy Leave- People who move from the fountain so that the next person can interact with the fountain.

Passers- By- People who are just passing by the fountain and not paying much attention to what is happening.

Passers- By (with a Glance)- People who notice(glance) at the interface while passing by the interface.

Shared Encounter- When two distinctive individuals or members from other groups interact with each other. The two possible encounters were:

Talk- People having only conversations across the groups/individuals usually in Social Interaction Space.

Action- People coming together to play with water.

The members of the evaluation team investigated throughout the day in Herderplatz where the fountain is located and found and came to a conclusion that five time slots fit well for our observation. The final observation was done in late spring, the weather was cloudy/ rainy on Tuesday and Friday in the month of June, with five different time slots of one hour duration. We started our observation at 06:40 to 07:40 in the morning, the time before school begins for children. The midday slot from 10:30 to 11:30, we were able to observe interaction when people were going to work, having their coffee break and when shops were beginning to open. The evening slot from 15:30 to 16:30, the weather was sunny, it was the time that children finished their school, this helped us get good data over how many children and teenagers interacted with the fountain. The next slot was from 17:00 to 18:00, as this slot enabled us to gather data when people finished work and were heading home. The final slot was from 18:00 to 19:00, the closing time of shops and bakeries around Herderplatz. We used this time to note and observe if people interacted with the fountain before they went home.

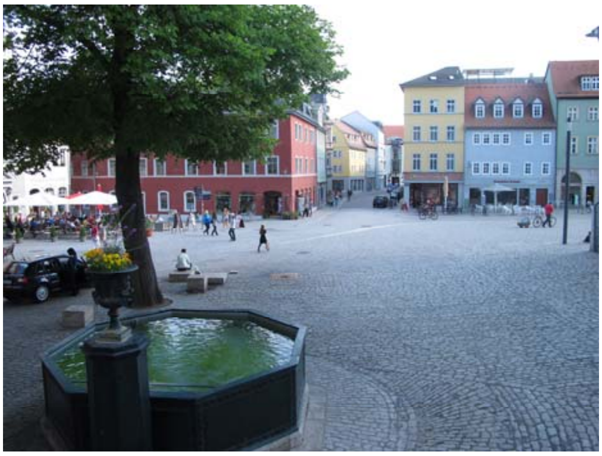

Figure 4.2: Fountain at Herderplatz.

The notion style that we used for our study of the interactive fountain are shown in table 1 and 2. There were two researchers who were involved in the observation study,

where one of the researcher documented the interaction of the fountain using the notations from table 1 and the other researcher documented the observers, hidden queue patterns and shared encounter with the help of the notations used in table 2.

From the observation and by analyzing the data, we were able to find which age groups of people interacted more with the fountain and also whether it attracted more groups or individuals. We also came across many shared encounters where people just gestured or talked with each other. The data we collected gave us a rough overview of the different factors that we considered for social interaction in urban space. The overall result of the observation in Herderplatz in as follows. We were able to find from the observation of the data analysis, that about 12.5% of people interacted with the fountain and generated about 4.8% shared encounters among the interacted people. However, the fountain generated 0.9% shared encounters of all the people in the display area. 2.5% of the people were observers, who inspected and watched how the previous person interacted or played with fountain before trying it themselves. From the overall data that we perceived 85% of them were passers-by i.e. who passed by the fountain but did not interact with the fountain.

NOTATIONS	REFERENCE
E	Elderly person
A	Adult
T	Teenager
C	Child
⬭	Represents a group
AC	Group of Adult and Child
ACA	Group of 2 adults and a child but child alone interacts.

Table 1: Notations used for Data Entry – Interaction

NOTATIONS	REFERENCE
AC	Group of Adult and Child are **observers**.
ACA	Group of 2 adults and a child are in **a hidden queue.**
ACA ⬭ ➡	Shared Encounter

Table 2: Notations used for Data Entry - Observation

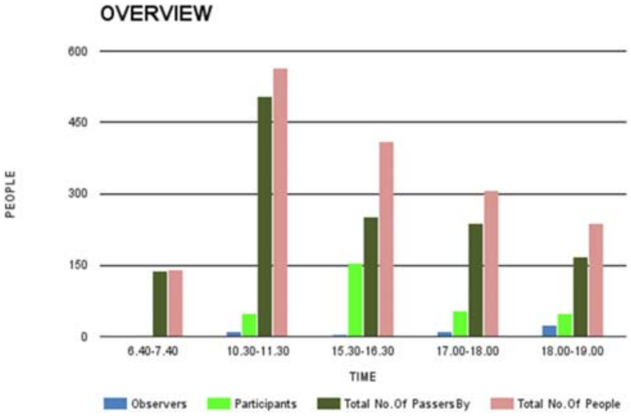

Figure 4.3: Overview of the interaction at Herderplatz.

PIPE Field Study

The goal of the study was to gain insight into the following questions:

(i)What is the impact of digital artefacts on people behavior and participation? (ii) How much effective it is to generate Shared Encounters? (iii) What are the common interactions of people to use the P.I.P.E? (iv)What are the differences interactions encountered when using a walkway and a plaza?

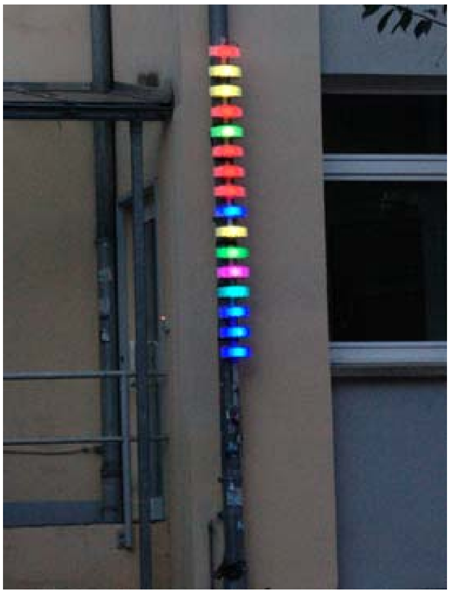

Figure 4.4: A fully lit output module (LEDs).

Location Description

The contextual setting of the space was characterized by several distinct zones. Based on our experience from the pilot studies, we situated the system on walkway area to maximize potential overlapping situations for both intentional interactions with the installation as well as unintentional. *Observation* We observed that PIPE created different zones of engagement around its surrounding area. Outside of the interaction space, passers-by engaged with PIPE in a number of other zones, such as social interaction space and comfort space. Different behavior is noticed for the people passing by and the people observing. Nearly half of the passer-by have noticed the system and reflected various gestures even took pictures of it. However, this active interactions are more common in groups than individual (passive), such as people in group discussing from the distance about what's going on.

The curiosity of knowing led the people to have different sort of interactions. Some people just stand on one input module to see how it reacts, and then they step further on other input modules. Some people jumped on different module and try to play with it like a game and they also expected some game kind of results while jumping. We noticed that participants are more likely to approach the display in groups. The following sequence repeated multiple times with different groups that a group passes nearby and one person notices the LEDs and starts interacting. The others then take part to fill all the LEDs. This is similar to the honeypot effect in-the-wild field studies [28], where a one's interaction encourages others to interact. One person who was already interacting was also trying to explain another group that how it works and then they join him. In such way it generated both kind of shared encounter i.e. talk/verbal and action, the activity based and the communicative one.

Most of the people could not understand clearly the purpose of the PIPE in one look, they had to interact with it to grasp it better. Out of their curiosity most people want to explore the things happening around their environment and surrounding. As the PIPE is visible from a certain distance (display space), most of the people when reach in display space have noticed the change in the surrounding and then acted accordingly to their choice.

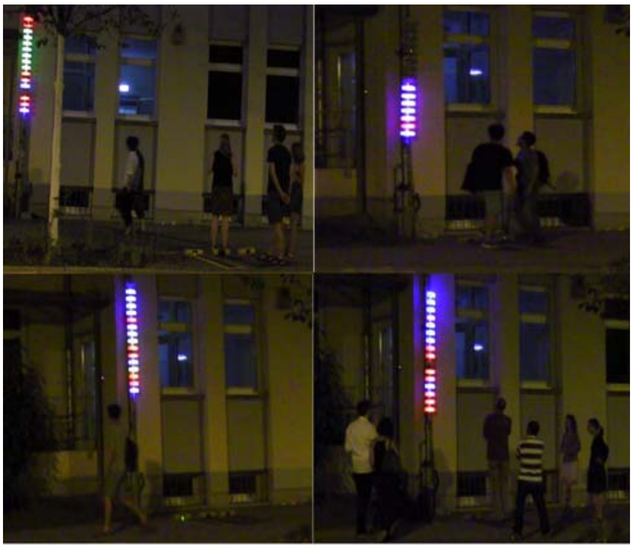

Figure 4.5: People interacting with PIPE.

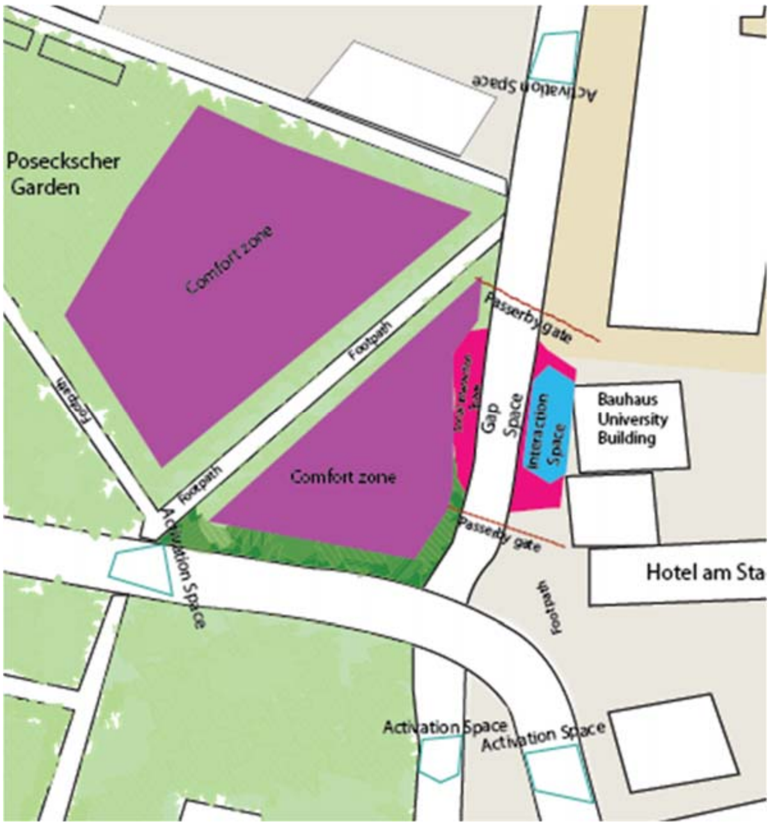

Figure 4.6: Overview of the different spaces.

As PIPE is installed on a walkway along the road, so people going on their vehicles also take a glance on it, especially the people in the bus. The interface even attracted the people who were driving by, for example a person driving by in the car stopped by and interact with it, which also implies that they crossed threshold from peripheral to focal awareness techniques like interacting with interfaces. Some users were so determined to light up the whole output module and when the output interfaces were cleared they gave a feedback by shouting out in joy, which helped us to understand that they were so interested in interacting with it and how well the metaphors encouraged the users to interact with interface for a longer time.During the four hours of the testing period in-the-wild; 450 people were in and around the display area, in which 162 people glanced at the PIPE and the same amount of people (162) did not notice the PIPE at all, almost 89 people performed the interaction while 37 people observed the interaction.

Results & Discussions

The PIPE was deployed in-the-wild in the month of August during night time on Tuesday from 21:00 until 22:00 and Wednesday from 21:00 to 22:00, the weather was warm (12-13°C). And it was also deployed on Thursday from 20:00 until 22:00 in the month of September, when the weather was cold (6 – 8 °C). On the whole, it was evaluated for three days. With the help of direct observation and video recording we were able to propose four different kind of persons: passers-by (without a glance), eye-beamers (passers-by with a glance), performers and observers.

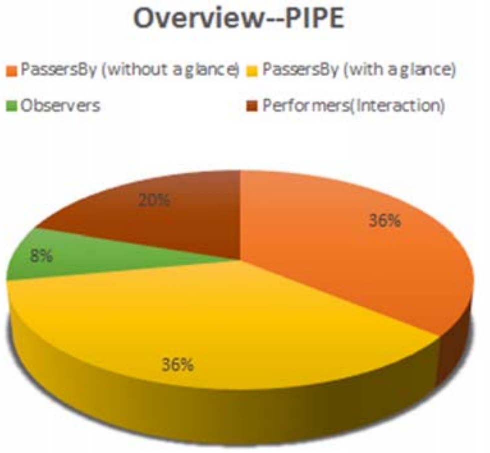

Figure 4.7: Overview of PIPE in-the-wild.

Overall interaction was 20% of the total people in the display area, which we thought was a great success as the project was tested during night. When we observed all the interactions we were able to find out that 80% of interactions were done by adults. As we expected there was not much interactions by children and teenagers as it was night time. Around 16% of elderly persons interacted with the PIPE.

Even though as expected most of the interactions were done by adults and only 15% by elderly people and 4% by teenagers. The flow of adults was only higher than any other age group of people in the display area during the evaluation. However, the behavior of the elderly people towards the PIPE was quite surprising, as almost 50% of them in the display area either observed or interacted with the interface (PIPE) and only one fourth of them did not notice the interface at all when they crossed the display area.

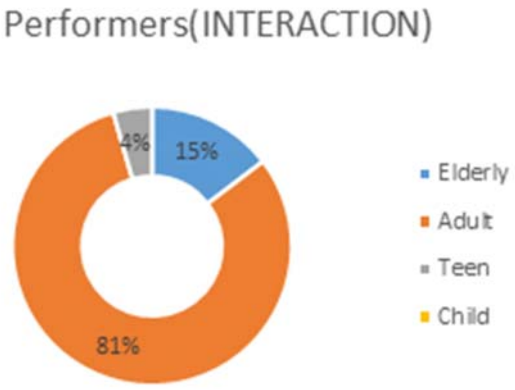

Figure 4.8: Interaction by performers of different age groups.

From the evaluation in-the-wild, we were able to find out that most people were in a group of two or three who involved more with the PIPE, rather than the people who were not in groups (Individual person).Our Interactive device had three separate input modules for multi users to interact with it simultaneously and that was a great success in-the-wild as it attracted more groups than Individual person. During evaluation in total 40 interactions were made in which 33 was made by group of people.

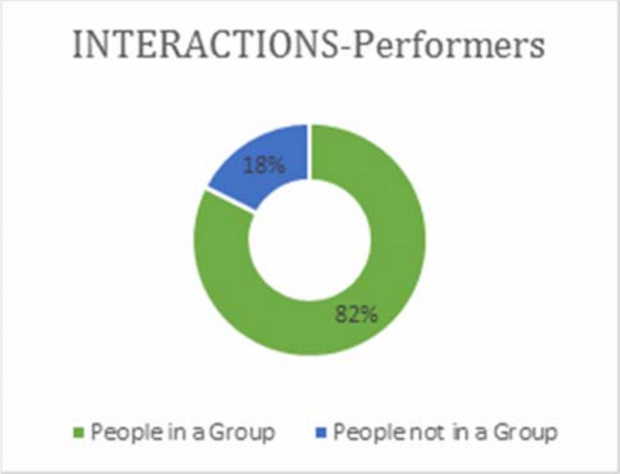

Figure 4.9: Performers of the interactive PIPE.

The individual persons did not interact much with all the three input modules laid out on the walkway and almost half of them (50%) did not notice the modules at all on the area. Even though the flow of people not in a group was higher than the groups of two or more people. We think they did not show much interest in interacting with the modules as they were in the pathway and they were going from one source to another destination. Although, we were able to lay out the input differently, where the input modules were placed further away from the output module almost closer to the road. We obtained some viable interactions when placed farther away from the output module, however the size of the input module obstructed the pathway for people riding their cycles. We then had to change the position of the input modules closer to the output module in such a way, that it was easily visible to the person interacting with the modules. Moving it closer to the PIPE, helped us gain better data of interactions, shared encounters, observation and hidden queue and also not blocking the walkway for bike riders.

As the modules were very much friendly to multi-users there occurred shared encounters during evaluation; even though it was not many shared encounters but still we had four shared encounters. The shared Encounter was 0.8% of the total people in the display area which we think is a good rate of social interaction as the testing was done during the night time as all age group of people won’t be present there and the weather was around (15-20 °C).

Interaction Patterns

During our evaluation in-the-wild, we recorded all the reactions of the people around our public interface using video recording to analyze the behavior of the people. At the beginning most of the groups, except the people who did not experience honeypot effect, took a few seconds to realize how the input modules were synchronized to the output module. Mostly everyone wanted to jump on the input modules rather than stepping on it for the adverse reaction on the output module as there were lots of leg space on the input module. We also came across two hidden queues, where people waited for their turn to interact with the fountain.

There is another interaction pattern which was repeated several time is that some people perceived it as game task and wanted to fill all the LEDs which makes them to interact longer. As they spend more time they get better understanding of the relationship of input and output modalities. And when the whole chain of LEDs are filled they made different joyful gestures by speaking loudly or making winning gestures by doing high five and clapping. While other people were interacting, some people going on the bikes also stopped to observe closely and one women also interacted. One of the unique interaction patterns were, that a group of four users interacted with it simultaneously and at one point they even tried to jump on the module at the same time (by counting 1,2,3,jump). Few users jumped on the input module for so long in excitement that the pressure pump tube got pulled off which we had to correct later; this situation occurred thrice during our evaluation. The input modules were not that fragile but the force the users gave on the jump made the pump tube to be released. In another interaction, there were two adults who interacted with the pipe until all the LEDs were lit and refreshed. Furthermore, an elderly couple made a small dance performance on the pipes to get different colors on the output module, which made it really interesting to know how people think in different ways to interact with the pipe. Some users even tried to walk over the three input modules as they walk in the pathway to see the behavior of the output module. Some users even tried to ride their cycle over the input modules to see the behavior of the output module. One of the passers-by rolled over his suitcase, over the input modules yet he did not notice that it was an interface to be interacted with, as observers we assumed that he thought that they were just planks of woods lying there as he did not notice the output module.

Two of the interactions lasted for a duration of approximately 3-4 minutes, which was also the longest duration of the interaction during our testing and most of the interactions were for an average of one minute per interaction. The PIPE Interface has even made the elderly people to interact with the digital media without much hesitation as the design was so user friendly for all the age group of people.

One of the most common interaction pattern is that jump on the single input module randomly to see how It would play. Few people find out about how to use multiple input module to mix the colors on LEDs. The usage of metaphors were very helpful for the passers-by to notice the interface in the area and as the input modules were so intrusive that they had to step on it which also increased the threshold of passers-by to interact with it.

	Plaza (Fountain)		Pathways (PIPE)	
	N	%	N	%
Passer-By	1303	85	324	72
Observer	50	2.5	37	8
Performer	307	12.5	89	20

Table 3: Summary of Plaza and Pathway Observation

One of the major difference between the walkway and plaza places that there is not enough observational and comfort space around the path ways. These difference leads to different observational and interaction behavior. In our observation the number of observing people and the length of interaction time is higher in plaza places artifacts than the pathways. The other factor is element of surprise, which is more in pathways as the display space is limited in certain directions. So some people interacted unintentionally first with the PIPE and then they realized.

Issues and Limitation

Input modules were not clearly visible from the distance, so to make more sense about how and what is happening, one has to come closer to the PIPE. That is why for most of the people from a distance, were attracted by the color and blinking LEDs, which enabled them to approach closer, so that they could discover more the relationship of the input modalities with the LEDs.

Due to the certain brightness of LEDs, we did the evaluation study only in the night time, so there were not many children on the street at that time. So we could not observe the interaction behavior of children. However, the users had a slight difficulty initially to understand that they had to step on the input module for three seconds for the color to be set on the output module. Most of the participants tried interacting or stepping on just one pipe at a time, not realizing that stepping on two pipes gave a different color in the output module. We also had problems in having to reset the panstamp often to let the modules start from the beginning as, it did not refresh for the next user to use the module.

CONCLUSION AND FUTURE WORK

The output part of the project was most of the time dealing with features of the material of polycarbonate and the forming methods. The problems, eventually we were confronted with, were choosing the bending methods for polycarbonate (heat bending, cold bending, manual bending). Without using machines for applying those methods, the process of forming the material did not go correctly. Therefore the joints did not align each other and the measurements couldn’t be accurate. We used folding with the method of laser cutting from cardboards for the prototyping, and it was working perfectly. Forming method could be improved.

Another part from the output module, which can be improved in the future production, is the attachment method of the modules to the pipe. Even though using Velcro did provide a good grip for that purpose and it was not visible on the installation. Also because of its flexible feature it was easy to apply. Besides, this solution can be replaced with the other methods, which can be more pertaining to the design language.

Since all of the input modules were not perfectly airtight, the next step could be to fix that. An approach could be to glue two rubber sheet together and then glue it on the wooden board, because rubber-rubber seems to bond better than rubber-wood. Another interesting point could be to change the design of the input modules from the longish form to another form like a square or a circle. That would have the advantage that it would remind nobody of a tube

anymore. But there is also an disadvantage - to still fit the proxemics, the interaction area would need to be even bigger and in that case not only the little pump but also the available space on a walkway could reach their limit. Nevertheless it could be interesting to observe how the interaction would change.
Another thing to try is to merge the box and the tube to one part. For example, the wooden board could be a bit longer, so that the box can be built directly on it. In this way the connection point between box and tube would be more stable and could also be hidden. If the whole setup should be more compact, it would also be possible to merge all three modules to one with three air chambers.
A way to make the input modules more eye catching could be either to take coloured rubber sheets or to arrange LEDs in the corresponding colour around the modules. There are many different possibilities how to change the appearance of it, but in all cases the hardware would essentially be the same. Therefore, prototyping with different forms of the input modules would be easy.

The orchestration software saw some major cleanup. Adding abstraction to the structure allowed the exchange of hardware dependent code components. The new structure therefore allowed to add the PanStamp implementation working along the XBee implementation. In theory, adding any kind of technology for hardware communication is now possible. This is a huge success, but there are still problems left. The hardware abstraction is not yet completely finished, especially in the user interface. The current solution is built around technology dependent views, using inheritance to avoid redundancy.

There is also no good solution to the communication between modules within the orchestration server by now. In PIPE, a specific device class was written to manage the modules and exchange data. A general approach does not exist. Using an event-based system could grant software modules to communicate without the necessity of knowing about each other. Properly creating such a system requires time and it is usually easier to built the software around the event-management instead of an insertion into existing software.

The integration of panStamp based modules for now is done without following a common standard. For future projects with panStamp-support it would be recommended to introduce a standard format or protocol for them, such that it is possible to provide easy communication between the modules in always the same way.

Another solution would be to construct a communication-protocol for all modules that is automatically converted in the needed one for panStamp or XBee.

In this study, we have observed various kind of system interactions and social interactions with respect to different spatial areas around the system. Our findings suggests that digital interactive artifacts like PIPE have positive impact in the urban placement. It also helped the people come together in certain regions and talk to each other or play with the system. It did not achieve very high rate of generating shared encounters among strangers; however, the social interaction among the members of the group who interacted with the PIPE was high, almost all the group interactions had shared encounters among themselves. Although, under some of the system limitation it fulfilled our expected results. In a nutshell, we can say that people interacted comfortably with our prototype.

REFERENCES

1 Parasitic Interfaces for Public Environments, Bauhaus-Universität Weimar (2014).

2 Movable, Kick-/Flickable Light Fragments Eliciting Ad-hoc Interaction in Public Space, Bauhaus-Universität Weimar (2014).

3 The Fun Theory: Piano Staircase[online].
URL: http://www.thefuntheory.com/piano-staircase
[09.10.2015].

4 Katharine S. Willis, George Roussos, Konstantinos Chorianopoulos, and Mirjam Struppek: Shared Encounters, Springer-Verlag London (2010).

5 Burgess, Phillip (2012): 12mm led pixels – wiring [online]. Homepage: adafruit
URL: https://learn.adafruit.com/12mm-led-pixels/wiring
[30.09.2015].

6 DiCola, Tony (2012): Adafruit-WS2801-Library [online]. Homepage: GitHub
URL: https://github.com/adafruit/Adafruit-WS2801-Library
[30.09.2015].

7 Makrolon Polycarbonate Sheet [pdf, online]. URL http://www.curbellplastics.com/technical-resources/pdf/polycarbonate-fab-guide-makrolon.pdf

8 Freescale Semiconductor: MPX5050-Technical Data - Integrated Silicon Pressure Sensor On-Chip Signal Conditioned, Temperature Compensated and Calibrated [pdf, online]. URL:
http://datasheet.octopart.com/MPX5050DP-Freescale-Semiconductor-datasheet-10000675.pdf
[30.09.2015].

9 panStamp S.L.U.: Homepage-panStamp [online]. Homepage: panstamp.com
URL: http://www.panstamp.com/
[30.09.2015].

10 Boundless (2015): Proxemics - Social distance between people is reliably correlated with physical distance: intimate, personal, social and public [online]. Homepage: boundless.com.

URL: https://www.boundless.com/communications/textbooks/boundless-communications-textbook/delivering-the-speech-12/effective-visual-delivery-65/proxemics-263-7998/https://www.boundless.com/communications/textbooks/boundless-communications-textbook/delivering-the-speech-12/effective-visual-delivery-65/proxemics-263-7998/
[30.09.2015].

11 Hifitt: HI-FLAT [online]. Homepage: hifitt.it
URL: http://www.hifitt.it/de/catalog/passaggio-aria-agricoltura/products/manichetta-hi-flat-hd-passaggio-acqua
[30.09.2015].

12 Processing: Processing Homepage [online]
URL: https://processing.org/
[13.10.2015].

13 William W. Gaver, Jacob Beaver, Steve Benford: Ambiguity as a Resource for Design [pdf, online]
URL: https://www.blasttheory.co.uk/wp-content/uploads/2013/02/research_ambiguity_as_a_resource_for_design.pdf
[13.10.2015].

14 Adam Kendon: Spacing and Orientation in Co-present Interaction, Pennsylvania (1988).

15 Rema Tip Top (2009): SC4000 CEMENT BONDING PROCEDURES [pdf, online]. Homepage: rematiptop.com
URL: http://www.rematiptop.com/technical/ind/REMA-TIP-TOP-SC4000-Cement-Bonding-Procedures-Rev4.pdf
[30.09.2015].

16 Xbee: Official Xbee Homepage by Digi
URL: http://www.digi.com/lp/xbee
[10.10.2015].

17 panStamp S.L.U.: Homepage-panStamp [online]. Homepage: panstamp.com
URL: http://www.panstamp.com/
[30.09.2015].

18 panStamp S.L.U.: Homepage-panStamp Product Info [online]. Homepage: panstamp.com
URL: http://www.panstamp.com/product/panstamp-avr/
[30.09.2015].

19 Berenguer, Daniel (2014): panstamp [online]. Homepage: GitHub
URL: https://github.com/panStamp/panstamp/wiki/panStamp%20NRG%202.-Technical%20details
[30.09.2015].

20 Berenguer, Daniel (2014): SWAP [online]. Homepage: code.google.com
URL: https://code.google.com/p/panstamp/wiki/SWAP
[30.09.2015].

21 panStamp S.L.U.: Homepage-panStamp – Product Info [online]. Homepage: panstamp.com
URL: http://www.panstamp.com/product/panstick/
[30.09.2015].

22 le Grange, Gideon: Github- panstamp-java [online]. Homepage: github.com
URL: https://github.com/GideonLeGrange/panstamp-java
[30.09.2015].

23 Truchsess, Norbert: Github- panstamp-java-SWAP [online]. Homepage: github.com
URL: https://github.com/ntruchsess/panstamp/tree/device_panstamp/java
[30.09.2015].

24 Wikipedia: Social interaction [online]. Homepage: Wikipedia
URL: https://simple.wikipedia.org/wiki/Social_interaction
[30.09.2015].

25 Psychology Dictionary: What is SOCIAL INTERACTION? [online] . Homepage: Psychology Dictionary
URL: http://psychologydictionary.org/social-interaction/
[30.09.2015].

26 Katharine S. Willis, George Roussos, Konstantinos Chorianopoulos, and Mirjam Struppek, Shared Encounters, Springer-Verlag London (2010).

27 Rogers, Y., Sharp, H., Preece, J. *Interaction design: beyond human-computer interaction*. 3rd ed., Wiley, 2011, pp. 456 ff.

28 Brignull, H. and Rogers, Y. Enticing People to Interact with Large Public Displays in Public Spaces. In Proc. of Interact '03, IOS Press (2003), 17-24.