Institutionen för datavetenskap
Department of Computer and Information Science

Final thesis
Replacing a physical console with an iPad
by
Björn Falke

ISRN LIU-IDA/LITH-EX-A--12/034--SE
2012-10-22
Replacing a physical console with an iPad

Master's thesis 30 HP, written by
Björn Falke
2012-10-22
1 Summary

This thesis report presents an iPad application built for replacing a physical display console on an indoor exercise-bike. The paper attempts to answer whether it's possible to do this sort of replacement and what possible consequences it could have.

The paper details the actual implementation and important design decisions such as Wi-Fi vs Bluetooth or iOS vs Android are discussed. The class structure and coding philosophy of the project are presented as well as choice of IDE and language.

Part of the purpose behind replacing the old console is to improve the user experience. This paper describes in detail the user interface of the developed application as well as guidelines for how such an application should be constructed.

The paper concludes that it is possible to replace the console on the EB200 exercise-bike with a tablet application and that it has the potential to be a superior solution.

2 Foreword

Wemotion AB have given me both the opportunity to do this thesis and the support to finish it. Thanks to everyone at Wemotion who have helped me in completing this thesis. A special thanks to Anders Östlund at Wemotion who have been my mentor for this work.

I would like to give thanks to my examiner Erik Berglund who have been very supportive, flexible and tolerant to my somewhat delayed conclusion of this paper. I also want to thank my opposition Joel Lennartsson.
# 3 Contents

## Table of Contents

1 Summary .................................................................................................................. 1  
2 Foreword .................................................................................................................. 1  
3 Contents .................................................................................................................. 2  
   3.1 Picture and table index ....................................................................................... 3  
4 Introduction .............................................................................................................. 4  
   4.1 Purpose and research question ......................................................................... 5  
      4.1.1 Framing of the question .............................................................................. 5  
      4.1.2 Purpose ...................................................................................................... 5  
   4.2 Terminology and clarifications ......................................................................... 6  
   4.3 Structure ........................................................................................................... 6  
   4.4 Method and sources ......................................................................................... 6  
5 Background ............................................................................................................. 7  
   5.1 Wemotion & Webike ......................................................................................... 7  
      5.1.1 Webike Home Edition ............................................................................... 7  
      5.1.2 Current console ......................................................................................... 8  
   5.2 Network protocol ............................................................................................... 8  
   5.3 Chain of communication .................................................................................. 8  
      5.3.1 Inside the bike .......................................................................................... 9  
      5.3.2 Springtail connection .............................................................................. 9  
6 Design decisions ...................................................................................................... 10  
   6.1 Why Gnat ......................................................................................................... 10  
   6.2 Platform choice ................................................................................................ 10  
      6.2.1 Development cost ..................................................................................... 10  
      6.2.2 Market segment ......................................................................................... 10  
      6.2.3 Prioritizing iPad ......................................................................................... 11  
   6.3 Bluetooth or Wi-Fi ............................................................................................ 12  
   6.4 Development environment .............................................................................. 14  
      6.4.1 Objective-C ............................................................................................... 14  
      6.4.2 C# (C-sharp) .............................................................................................. 14  
      6.4.3 Migrating to MonoDevelop ....................................................................... 14  
   6.5 Program specifications ..................................................................................... 16  
      6.5.1 Hard Specification ..................................................................................... 16  
      6.5.2 Soft Specifications ..................................................................................... 16  
7 Implementation process ........................................................................................... 17  
   7.1 Code structure .................................................................................................. 17  
   7.2 Class diagram .................................................................................................. 17  
   7.3 Network code ................................................................................................... 19  
   7.4 Security ............................................................................................................ 19  
   7.5 Graphical design process ................................................................................. 20  
      7.5.1 Usability .................................................................................................... 20  
      7.5.2 iOS design ................................................................................................. 20
3.1 Picture and table index

Illustration Index
Illustration 1: In-game footage from Webike home edition..............................................7
Illustration 2: Current console for the EB200 exercise bike...............................................8
Illustration 3: Picture illustrating the communication between devices..........................9
Illustration 4: Gnat class diagram.....................................................................................18
Illustration 5: Gnat welcome screen................................................................................22
Illustration 6: Gnat length chooser screen.......................................................................22
Illustration 7: Gnat program chooser screen.................................................................23
Illustration 8: Gnat biking screen....................................................................................23
Illustration 9: Gnat end screen........................................................................................24

Index of Tables
Table 1: Tablet market share by operating system fourth quarter of 2010 and 2011...........12
Table 2: Smartphone market share by operating system for third quarter of 2010 and 2011..12
Table 3: Differences between Wi-Fi and Bluetooth.........................................................13
4  Introduction

When apple launched the iPad 2010 a lot of people including myself were skeptical. Who could possibly have use of such a device? Many people deemed the device to large to put in your pocket and too weak to run any serious programs. Apple sold 14.8 million iPads worldwide, during 2010 and kicked the tablet marked into high gear. People everywhere including myself were starting to see the potential of the tablet device and started trying to find uses for it.

In most cases a tablet does what any other computer can do but it does it either better or worse. Reading digital documents for example, something you can do on your desktop or on your phone but its probably “nicer” to do it in a sofa with a tablet. Surfing the internet is another one of those. These are not very interesting uses since they are not new. But there are things you could use a tablet and only a tablet for.

The most classic example is physicians at hospitals. A study from early this year show that a quarter of all European medical doctors are using a iPad professionally. It has the mobility to be carried around, good enough usability and a big enough screen for the physician to be able to for example show a patient something.

Another example is board games. The screen on an iPad is big and good enough that you can play multiple people on the same device. And the multitouch interface gives the interactivity needed for social gaming.

The last unique usage of the tablet that intrigues me the most is the use of it as a controller. Especially musicians and DJs have started using iPads as a dedicated control surface for desktop music software. Other products are starting to appear, such as home entertainment systems that are using iPad/iPhone apps as a remote control. The industry is a bit slower but I imagine that robots in production facilities could be controlled via non integrated tablets in a near future.

This thesis is an experiment in using a tablet as a controller for an exercise device. The paper will describe the process of making such an Application for the iPad and present the resulting program. It will also explore some of the aspects that needs to be taken into account when using a tablet as a controller for another device in this manner.

---

1 Apple Inc. (July 20th 2010). Apple Reports Third Quarter Results
2 Apple Inc. (October 18th 2010). Apple Reports Fourth Quarter Results
3 Apple Inc. (January 18th 2011). Apple Reports First Quarter Results 2011
4 Tyer, Dominic (Feb 15th 2012). Twenty six per cent of European doctors own an iPad [www]
5 Harvell, Ben (January 16th 2011). How musicians are using the iPad [www]
6 HomeTheaterReview.com (May 17th 2010). Xantech Using Apple iPad As Controller
4.1 Purpose and research question

Is it possible to replace the user interface console with a tablet application? What would the consequences be? Is the usability going to become better or worse? What other implications would this have?

These are some of the questions I asked myself at the start of this thesis work. These questions are however broad, general and out of scope for what can actually be answered within the time constraint.

4.1.1 Framing of the question

For the question to be clear we have to define it better. What is a console here? In this practical implementation we will only be looking at the specific console of the EB200 exercise bike. This bike and its console will be described in section 5 (Background). We will compare this design with my specific application developed for iPad. Of course both the console and the application can vary widely. Especially if we go outside of the realm of exercise bikes or even exercise equipment. If the equipment and software varies so would the answer to the initial vague questions vary. Through the very limited research that this implementation comprises we will thus not be able to answer the general questions.

The more specific question for the thesis is:

“Could an iPad application replace the console on the EB200 exercise bike?”

This is the question that this thesis work tries to answer with an implementation and research.

4.1.2 Purpose

The main reason for doing the experiment that Gnat (the implementation) involves is the interests of Wemotion. Section 5.1 (Wemotion & Webike) explains a bit more about the company and circumstances leading up to this thesis. In short, Wemotion hope that this work can be the start of a new product investment.

There is however another purpose in the form of research and personal interest. We live in a technical society that is changing fast. In this ever shifting environment it is inevitable to speculate in what the future solutions for today’s problems are. Mobile touchscreen technology might be part of some of these solutions. This is why I am interested in exploring the field further and try to find out what we can and can not do with the current and the future tablet technology.
4.2 Terminology and clarifications

Several names and terms that might not be common knowledge will be used in this paper. Here is a short list with explanations for some of the more obscure notions.

Wemotion – The company where this thesis work was made.
Gnat – Working title for my implemented iPad app.
Springtail – Wemotion's future exercise portal.
TED – Training Equipment Device. Most of the time, the exercise bike
Earthworm – Communication relay software for testing
Nuntius protocol – Webike Generic Communication protocol
IDE – Integrated Development Environment
Cadence – The number of revolutions of the crank per minute (rpm)
IB (interface builder) - UI creation helper for Xcode
Exertainment – Exercise and entertainment combined

4.3 Structure

The work of this thesis can be divided into three somewhat separate parts. Research, implementation and discussion. The first part is the background to the thesis. It will cover a wide scope of information that is relative to the implementation and also argue the design decisions made based on this information. The second part will account for the implementation itself. It will explain the application and what problems were encountered during implementation. During the discussion section the results will be presented and elaborated upon. Alternative solutions and hypotheses around the implications of the result will be presented.

4.4 Method and sources

The primary source of information for this thesis are experiences achieved during the development of Gnat. I have leaned two new programming languages in two new environments. I have also learned a lot about the more general question of making a software to replace existing hardware and about usability programming for a tablet.

Since this is a fairly new field there are not a lot of information available on replacing physical consoles with touchscreens. In printed sources there are very little information on the usability of touchscreens in general. Instead a lot of internet articles have been used and information have been acquired closely related fields.
5 Background

5.1 Wemotion & Webike

This thesis work have been done in cooperation with Wemotion AB. Wemotion is a small company located in Mjärdevi, Linköping. They develop exercise games and also have some consultant business. This thesis have been made in cooperation with the development team with the purpose of helping the company move its product into the mobile market.

5.1.1 Webike Home Edition

Webike HE is the main product that have been developed at Wemotion. It is a biking simulator that connects to an exercise bike. You plug in your exercise bike into a computer via a USB cable and can then play the game on your computer using the bike as a controller. The resistance in the bike is controlled by the game, influenced by hills, gear and difficulty. The speed of character in the game is in turn controlled by how fast the playing is biking on the exercise bike. The purpose of the game is to make the exercise more entertaining and thus fall into the category of Exertainment.

Illustration 1: In-game footage from Webike home edition
The game uses the same EB200 bike with the same console as I am using for my Gnat implementation. My app is however not supposed to be compatible with the Webike game. The purpose of Gnat is for when a user wish to exercise but don’t want to run the full Webike game.

5.1.2 Current bike and console

Webike is currently sold with the EB200 bike from Casall. The bike have two main components to it, the actual bike and the console. The whole bike including the console is built and developed by a Taiwan located company called Chang Yow. As a consequence the inner structures of the bike and console are unknown. That includes what circuit boards it has or what software it runs. The EB200 bike is a fairly standard mid-range priced exercise bike. The console have been modified with a USB port and can by request from Wemotion handle a fixed set of commands specified in the Nuntius protocol, described in the next topic. This console is the user-interface that I aim to replace with Gnat. However in this thesis work the console will never be removed since it is the only way to communicate with the bike.

5.2 Network protocol

Wemotion have made their own protocol for communication between a TED (training equipment device) and the computer. The protocol is designed for USB communication and is in use in Webike HE. To use it for Wifi communication Wemotion where required to change a few minor details. The protocol is called Nuntius protocol and contain a set number of commands that can be used to ask the bike for information or setting the resistance.

5.3 Chain of communication

Since a bike that might in the future use Gnat as standard is not yet developed a temporary way of testing the application had to be used. This was solved with a communication relay program running on a PC.

The TED (Training Equipment Device), in this case the EB200 bike, is connected to a PC via a USB cable. The PC is running the relay software Earthworm developed by Anders at Wemotion. This program have a Wifi connection with the iPad open. When information is sent from either the TED or the iPad, Earthworm sends it through to the other device. Since the wireless and wired communication uses slightly different protocols for the packages sent, Earthworm have to decode and then recode all the packages. In this way the EB200 bike can wirelessly communicate with Gnat on the iPad.
5.3.1 Inside the bike

There is of course a lower level communication going on within each of the three devices. The EB200 bike has a single sensor that fires each time the pedal passes by its position. From these ticks and a clock in the console the cadence is calculated. From the cadence the bike can then calculate other values like speed, effect and calories burned.

Since the bike needs a few revolutions before it can actually calculate the cadence accurately there is a delay from the start of your pedaling until any numbers are shown. Exactly what goes on inside the bike is unknown but there seems to be another delay too. The bike requires three sensory detections before it calculates a cadence value. After you have pedaled three revolutions it will take approximately another second or so before the console sends any data. There are no reasonable explanation for this delay but it limits the functionality of any device connected to the bike. Ideally the console would send the sensory data directly. This way we would get minimal delay and could calculate the Cadence and other values in the iPad. Instead we have to patiently wait for the console to calculate all values and then send them.

There will of course always be a slight delay since the data have to be send to another device, but this delay is extremely short. On a local Wi-Fi network it will be in the scale of milliseconds. Even with Earthworm relaying the information this delay is really small compared to the delay within the bike.

5.3.2 Springtail connection

Wemotion are currently working on the sequel to Webike. Part of the sequel is an online portal for interacting with other players and viewing statistics among other things. Connection to this portal have been built into Gnat. This is no essential feature of the app and does not help to answer my research question. The feature was implemented because Wemotion wanted to know how it would work and how hard it would be. Since it is of little importance to the thesis we will not get into more details regarding this feature.
6 Design decisions

Before starting working on the implementation of Gnat the where a lot of decisions to be made. What platforms should the app support? What standards will it use? What features will it have? Some of these decisions where later on in the implementation adjusted or changed. In this chapter these decisions will be discussed in detail.

6.1 Why Gnat

All projects at Wemotion have working titles from the insect world. The online portal for the sequel to Webike is called Springtail and the sequel itself goes under the name Mosquito. Since the application of this theses have some similarity to Webike and its sequel Mosquito in that it connects to the bike, but is a lot smaller, Gnat seamed to be an appropriate choice of name.

6.2 Platform choice

For the purpose of testing whether a tablet app can replace the physical console it doesn't really matter if the platform is iOS based or Android based. There are however a couple of aspects to look at when choosing platform.

6.2.1 Development cost

Android development is free. You don't have to pay to get access to the SDK or to deploy and launch the app. For iOS development you have to pay a yearly $99. This is an extremely low cost for any company so in the big scheme of things cost isn't really relevant.

6.2.2 Market segment

As shown in Table 1 and Table 2, iPad has a significant lead in the tablet market while Android is by far the largest operating system in smartphones. Market share is however not the only interesting aspect of the market. The average iPhone/iPad gamer is spending about five times more money on their mobile games than the average Android gamer. That is the result found from research firm Newzoo\(^1\), which looked at data from iOS and Android downloads as well as 17,000 survey respondents. Based on this knowledge it was decided together with Wemotion that iOS would be the primary target for Gnat.

---

\(^1\) Gaudiosi, John (May 5th 2012). New Research Shows Apple Still Winning the Video Game War Against Android
6.2.3 Prioritizing iPad

Initially the intention was to make an app that would work on both iPhone and iPad. Later on in the development I decided to prioritize the iPad solution. There are a couple of reasons for this decision. The first was time constraint. Making a GUI takes a long time and the GUI would have to be completely remade for the iPhone. The second reason was the difficulty to find a well working design for the iPhone. The old console have a surface area of 22.5×22 cm, with 8 buttons and a 6” display. To cram the same functionality into a 3,5” touch screen is a challenge. Especially since the user will be working out when using the device and might not have very good precision.

Even if the current version of Gnat does not run on an iPhone all non-GUI code are reusable for this device. All that would be required to get it working on an iPhone is to make the GUI and connect the GUI components with the right code.

<table>
<thead>
<tr>
<th>Global Tablet OS market share by %</th>
<th>Q4 ’10</th>
<th>Q4 ’11</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apple iOS</td>
<td>68.2</td>
<td>57.6</td>
</tr>
<tr>
<td>Android</td>
<td>29.0</td>
<td>39.1</td>
</tr>
<tr>
<td>Microsoft</td>
<td>0.0</td>
<td>1.5</td>
</tr>
<tr>
<td>Others</td>
<td>2.8</td>
<td>1.9</td>
</tr>
<tr>
<td>Total</td>
<td>100.0</td>
<td>100.0</td>
</tr>
</tbody>
</table>

*Table 1: Tablet market share by operating system fourth quarter of 2010 and 2011.*

<table>
<thead>
<tr>
<th>Global Smartphone OS market share by %</th>
<th>Q3 ’10</th>
<th>Q3 ’11</th>
</tr>
</thead>
<tbody>
<tr>
<td>Android</td>
<td>25.3</td>
<td>52.5</td>
</tr>
<tr>
<td>Symbian</td>
<td>36.3</td>
<td>16.9</td>
</tr>
<tr>
<td>iOS</td>
<td>16.6</td>
<td>15.0</td>
</tr>
<tr>
<td>Research In Motion (blackberry)</td>
<td>15.4</td>
<td>11.0</td>
</tr>
<tr>
<td>Microsoft</td>
<td>2.7</td>
<td>1.5</td>
</tr>
<tr>
<td>Others</td>
<td>3.6</td>
<td>0.9</td>
</tr>
<tr>
<td>Total</td>
<td>100.0</td>
<td>100.0</td>
</tr>
</tbody>
</table>

*Table 2: Smartphone market share by operating system for third quarter of 2010 and 2011.*


2 Gartner.com (November 15th 2011). Gartner Says Sales of Mobile Devices Grew 5.6 Percent in Third Quarter of 2011; Smartphone Sales Increased 42 Percent
One of the fundamental design decision for any wireless solution is what standard to use. The most commonly used ones are Wi-Fi and Bluetooth. There are several pros and cons to both techniques.

<table>
<thead>
<tr>
<th>Specifications authority:</th>
<th>Bluetooth</th>
<th>Wi-Fi</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specifications authority:</td>
<td>Bluetooth SIG</td>
<td>IEEE, WECA</td>
</tr>
<tr>
<td>Year of development:</td>
<td>1994</td>
<td>1991</td>
</tr>
<tr>
<td>Bandwidth:</td>
<td>Low (800 Kbps)</td>
<td>High (11 Mbps)</td>
</tr>
<tr>
<td>Hardware requirement:</td>
<td>Bluetooth adaptor on all the devices connecting with each other</td>
<td>Wireless adaptors on all the devices of the network, a wireless router and/or wireless access points</td>
</tr>
<tr>
<td>Cost:</td>
<td>Low</td>
<td>High</td>
</tr>
<tr>
<td>Power Consumption:</td>
<td>Low</td>
<td>High</td>
</tr>
<tr>
<td>Frequency:</td>
<td>2.4 GHz</td>
<td>2.4 GHz</td>
</tr>
<tr>
<td>Security:</td>
<td>It is less secure</td>
<td>It is more secure</td>
</tr>
<tr>
<td>Range:</td>
<td>10 meters</td>
<td>100 meters</td>
</tr>
<tr>
<td>Primary Devices:</td>
<td>Mobile phones, mouse, keyboards, office and industrial automation devices</td>
<td>Notebook computers, desktop computers, servers</td>
</tr>
<tr>
<td>Ease of Use:</td>
<td>Fairly simple to use. Can be used to connect up to seven devices at a time. It is easy to switch between devices or find and connect to any device.</td>
<td>It is more complex and requires configuration of hardware and software.</td>
</tr>
</tbody>
</table>

Table 3: Differences between Wi-Fi and Bluetooth
For the purpose of the thesis either one would have been fine from a technical point of view. Bluetooth has a clear advantage when it comes to how easy it would be to connect your devices and set up communication. To connect a Bluetooth bike to your mobile device you would just need to pair the devices, a fairly straightforward configuration. To set up a Wi-Fi connection would be a bit more complicated.

There are two ways a Wi-Fi connection with the TED could work. Either both the bike and the mobile device are connected to the same Wi-Fi network via a router. The problem here is how you connect the bike to your Wi-Fi network. If you have an encrypted network, which most people have, you have to enter the password and settings somewhere and the TED will probably not have an interface capable of that.

The other way to set it up is to have the TED in ad-hoc mode and set up a peer-to-peer Wi-Fi connection. You can then connect to the bike's Wi-Fi network with your tablet. The only problem here is that your normal tablet can only be connected to one network at a time, so you would not be able to have an internet connection while connected to the bike. The solution would probably have to be a combination of these two configurations.

First the user starts the bike in ad-hoc mode so that a peer-to-peer connection can be established. When connected the user can configure the network settings of the TED. The user can find and set up other wireless networks. When your home network have been configured in the TED settings you reboot it in normal mode. The user then connects to the bike again but this time through the home Wi-Fi network. This may seem to be a very complicated solution, especially if the user have little or no computer experience. Luckily there are a couple of alternative solutions. If there is an option to connect to the TED via USB this can be used for the Wi-Fi configuration. The bike could also be made advanced enough that you can actually do this configuration on the bike directly. This would require a screen with GUI-software and a means of entering characters.

The decision was in the end was pretty simple. Apple requires that all Bluetooth devices that are connectable to a iOS device meet a set of requirements. For your device to be certified they have to verify it and you have to pay Apple for this verification. This is not only a costly process but also very time consuming. Since this thesis work is more of an experiment then a production pattern it would simply not be worth this effort and cost.

From Wemotion's point of view it is also a problem that UDK (unreal development kit) That the sequel to Webike is being developed in, does not support Bluetooth. They would not be able to reuse the techniques from Gnat if it was build on Bluetooth technology.

Wi-Fi could also be considered more flexible and foreseeing since most new devices like smart TVs and Media stations have support for it while not always supporting Bluetooth.

---

1 Apple Inc.(2012). MFi Program
2 Apple Inc (Oct 14th 2011). Bluetooth Accessory Design Guidelines for Apple Products
6.4 Development environment

When starting development for an iOS device you are not really presented with a choice of platform or language to program in. Everything is built up around the Xcode environment and Objective-C language you use in it. Apple have even implemented a few features making it impossible to completely skip using Xcode.

6.4.1 Objective-C

Objective-C is an object-oriented programming language that are primarily used in iOS development. The language is reflective, have borrowed traits from the smalltalk language and added it to a C foundation. This smalltalk way of handling object orientation is somewhat different from Simula-style languages like C++ or Java. In these languages the method name is bound to a section of code in the target class by the compiler. But with Objective-C the target of a message is resolved at runtime, with the receiving object itself interpreting the message.

Calling a method from an object pointer is in C++ done like this:
obj->method(argument);
In Objective-C, this is written as follows:
[obj method:argument];

6.4.2 C# (C-sharp)

C# (pronounced c-sharp) is a relatively new language. It is intended to be a simple, modern, general-purpose, object-oriented programming language. It was developed by Microsoft to have a language similar to Java that they owned and that was created for the Windows platform. C# didn't really blossom until 2007 when Version 3.0 was released together with the .NET framework 3.5. .NET (pronounced dot-net) is a huge reason for choosing C# as a developing language. Having access to all the pre made classes and functions that .NET includes makes development of larger projects a lot easier.

6.4.3 Migrating to MonoDevelop

MonoDevelop is an open source IDE for multi platform development. A few months into the thesis, development of Gnat was restart the with MonoDevelop. This meant that a lot of code previously written in Objective-C had to be rewritten in C# instead. There was a few reasons for this shift. One was that MonoDevelop allowed for multi-platform development. This meant that Gnat could be deployed with the same code to both iOS and Android. There are clear benefits to writing one set of code and then being able to deploy it for several different platforms. Development times can be cut and uniformity among the different versions of the program is easier to achieve. You are however still be required to make a GUI for both platforms separately.
which is why Gnat still only runs on the iPad. Since everything but the interface is reusable it would be relatively little work to get Gnat working on an Android based tablet.

When developing for iOS through MonoDevelop you still have to use Xcode to some extent. The UI is designed in Interface Builder where you can place and adjust graphical components as well as giving them properties such as “clickable”. The design is saved in a .xib file and you usually have one of these files for each screen in the application. Each component and its properties is then connected to an object in a corresponding Objective-C class. This is how it works both when working in Xcode and MonoDevelop. To be able to use these objects in Mono a translation file have to be created. This is done automatically by MonoDevelop and will give you access to these UI objects and their properties in C# code.

A big argument for migrating to C# is my own and Wemotion's knowledge base. Neither I nor anyone else at the company had any previous experience developing in Objective-C. From the company's standpoint it is also beneficial to use the same language in all projects.

Using C# and MonoDevelop grants access to a subset of the .Net framework. This is a rich source of wrappers and classes that can be used instead of coding everything from scratch. In the Gnat project it was mostly notable when writing the network code which was a bit tricky in Objective-C but very easy in C#.

A drawback to this work flow is that not all functionality will work. Several properties and features that are possible to use in Interface Builder will not get properly linked to MonoDevelop. You can for example not use the UI component “barButton” since the “touchUpInside” functionality of this component has not been implemented in Mono. There are however ways to solve these issues. If nothing else works you can code the behavior of the component in an Objective-C class that can then be consumed and used in C#. But this is of course not very time efficient.
6.5 Program specifications

Together with Wemotion we agreed on a specification list for the implementation. This list was composed in the very early stages of the thesis work but most of the fundamental features are still up to date. In the result sections these specifications will me discussed in more detail, as well as explanations why not all of them where implemented.

6.5.1 Hard Specification

The specifications is divided into three categories, Shall Should and Want.

Shall
be able to change the resistance of the cycle
be able to show current speed
be able to show the distance cycled
be able to show time cycled

Should
be able to show the wattage generated
be able to show calories burned
be able to show the pulse
be able to show the average speed
include alternatives for pre-programmed workouts
include alternatives watt-based training

Want
that there are real-time graphs that show trends in speed / pulse / watts etc.
that there are alternatives for pulse based training
that there is a ghost mode (match your own best time / minimum heart rate / maximum watts on a preprogrammed path)
that there is a connection to Springtail
that there is a connection to Facebook / twitter / blogs
that there is a pinch to zoom in the graphs
a long term exercise planner

6.5.2 Soft Specifications

• Good usability
• Fast reaction times
• Good looking
• Simple and clean
7 Implementation process

7.1 Code structure

Initially the plan was to create an app that would run on both iPhone and iPad. Later on in the development when migrating to MonoDevelop the app was supposed to work on android-phones and -tablets as well. The current version of Gnat only works on the iPad but this strive has still affected some of the design decisions. When programming for multiple devices it is important that you keep your code modular. Each functionality should be separated so that you can easily exchange it if it for some reason doesn't work on another device.

The modular thinking has been in consideration for most of the development of Gnat but there are a few places that can be improved drastically. The GameScreen class contain most of the logic controlling the training session. This is of course not good modular programming since the UI parts is different for different platforms. One of the most needed improvements of Gnat is probably to break out the logic in the GameScreen class into a GameLogic class that handles all times and distance counting.

7.2 Class diagram

Here an overview of Gnat will be described with a UML diagram. The Diagram contain almost all classes of the program but not all the functions and attributes of the classes. There are a lot of default functions, such as `viewDidUnload()`, `ShouldAutorotateToInterfaceOrientation()` and many more that are not included. The diagrams are also quite simplified. A button is actually both an attribute that is connected to the xcode interface builder file and a function that handles the action when the button is pressed. In the UML diagrams UI components are just shown as attributes that are assumed to contain the event and action of that component. The navigation bar element are compressed into one attribute and all labels and other UI components are not included, to give a few examples. Most of this is done to save space so that the whole diagram can actually be viewed in this document.

For each Screen class there are actually four files. Two .xib file that describe the UI of the screen. One for iPhone and one for iPad. One .designer.cs file that handles the translation of IBActions and IBOутlets from xcode to monodevelop. The fourth file is the .cs file that contain the C# code that describe the behavior of the screen.
Illustration 4: Gnat class diagram
7.3 **Network code**

When Gnat is started a new thread is created for the purpose of handling network communication. This is necessary since network communication involves a lot of waiting. While waiting for a network reply the app can not simply be idle. That would mean that the GUI would freeze and user experience would be horrible. Instead a separate thread is handling the network communication, including waiting for packages, parsing and sending.

This line of code creates a new thread that calls the Connect function.

```csharp
Task.Factory.StartNew(() => this.Connect(hostname));
```

The first thing done in the connect function is to create a new TcpClient that handles the network communication. Being able to use this .NET class that wraps socket handling, timeout events and buffers makes network coding a lot easier and faster to program.

This line of code creates a new TcpClient that will handle the network communication.

```csharp
TcpClient client = new TcpClient(server, DefaultPort);
```

After a whole package is received it has to be parsed according to the protocol specifications. This is done in a separate class that also handles creating packages that will be sent.

7.4 **Security**

During the implementation relatively little effort and thought have been put into security. It is however a matter that needs to be looked into carefully if Gnat is developed further. A flexible general platform like a tablet give more possibilities for security leaks than the custom hardware and software in the standard console. Using wireless communication also poses threats since the information is theoretically accessible for anyone in range.

But as Brian Shea writes in his IT security book\(^1\) “Is it worth protecting?”. We should start by consider what the actual consequences of unauthorized access would have. The data sent in between Gnat and the bike is fairly uninteresting. If you were to listen to the communication the only information you would get is training related. Cadence, Pulse, effort generated and resistance in the bike. This information is probably not worth putting effort into protecting.

Alternatively someone could intervene in the conversation between Gnat and the bike, pretending to be either party and send sneaky messages. This way a scoundrel could manipulate either device and get access to futher information. In the EB200 console there are however no interesting data to access. As far as manipulating, the worst you could do is to change the resistance or maybe make the device freeze.

---

\(^1\) *Shea, Brian (Apr 1st 2002). Have You Locked the Castle Gate? Home and Small Business Computer Security*
All apps on the iPad are sandboxed. This means that my Gnat application could never access data in for example the Photo Album. So if someone were to highjack the communication between the iPad and the bike, pretending to be the bike. The worst they could possible do it to cheat with the training results.

The only reasons to strengthen the security would be to avoid cheating and pranks. Since none of these aspects are critical, and could probably be achieved by other means even with tighter security, the conclusion is that Gnat do not need to be made more secure.

7.5 **Graphical design process**

User friendly design becomes essential when working on an application like this but design is also the overall appeal of the program. Good design is also something that allows the application to grow and change when the user gets more experience. Preferably without changing the interface to the user too much.

7.5.1 **Usability**

Usability is one of most important aspects of an app like Gnat. If the app is not easy to use, intuitive and can get you started quickly there is no point for the user to want it over the old console. A lot of the arguments for having a tablet app instead of a console with physical buttons are from the developer or production company's point of view. Usability is one of the few things that the end-user see and care about.

The development of Gnat have been guided by Jakob Nielsen's "Ten Usability Heuristics"\(^1\) as well as a strive for user-friendliness of the interface. Jakob Nielsen have also done a couple of iPad usability evaluations that have inspired the design.\(^2\) Gnat is a very minimalistic app which in itself highly benefit the usability. However no tests or evaluations of the usability have been done. This is something that is highly recommend to be done if Gnat is to be developed further.

7.5.2 **iOS design**

Apple are relatively strict when it comes to the design of apps for iOS devices. They have a rigorous design guideline documentation and will not approve apps that stray too far from it.\(^4\) This principle is understandable from a usability point of view. Having all apps follow the same basic design means that the user will always recognize fundamental functionality.

\(^1\) Nielsen, Jakob (2005). *Ten Usability Heuristics*

\(^2\) Nielsen, Jakob (May 10, 2010). *iPad Usability: First Findings From User Testing*

\(^3\) Nielsen, Jakob (May 23, 2011). *iPad Usability: Year One*

\(^4\) *Apple Inc. (Last updated: 2012-03-07).* *App Design Strategies*
There can also be drawbacks to having uniformed design. Here are two examples in the case of iOS development:

The navigation bar is always on top in a iOS device, but this is also where connectivity information is shown as an overlay. Thus it can be difficult to navigate when you for example are broadcasting your 3g internet.

A iOS app can not have a close function. Apple's argument is that "that is what the home button is for". The problem is that the home button doesn't actually close the app it just minimize it. So if you want to actually terminate an app you have hold in your home button and terminate it manually. This is not a problem for most applications since there are no reason to actually terminate them. But if you have an application that can run in the background but you don't always want it to be running in the background, you have a problem.
8 The Gnat application

8.1 The design of Gnat

The application consists of five main screens. At the start you are greeted with a Welcome screen.

Clicking the start button takes you to the Length chooser screen. Here the user will choose a length for the training. It can either be a distance or a time.

Illustration 5: Gnat welcome screen

Illustration 6: Gnat length chooser screen
The next views the *Program Chooser Screen*. There are currently four different programs to chose from, Interval, Ladder Mixed and Manual.

Clicking start will then combine the length and program chosen into a training session. A countdown will activate and give you 5 seconds before the clock starts.

In this exercise screen you can see all the relevant information such as speed, distance, pulse, cadence, time to go and even more. You can also change the resistance via a resistance slider at the bottom. Either when the goal distance / time is met or if the user clicks STOP the app will go to the *End Screen*.
The app also contains two more screens, The Bike Screen and The Springtail Screen. In the Bike Screen you will be able to connect to a bike. This is however not implemented. At the moment you enter the IP-address of Earthworm client in the code. The Springtail Screen is there for logging in and out of the Springtail service.

8.2 Specification results

Gnat in its current form is not a polished product and would need to be developed more before sending it to the app store. In terms of product specifications the app does meet most of the requirements set up in the initial stages of this thesis work. All the Shall requirements have been met. All but one of the Should requirements have been met and one of the Want requirements. The shall requirement that was not completed is the watt based training.

8.2.1 Shall requirement

Even though the design of Gnat has changed a lot from the initial drafts when we agreed on the specifications the most basic requirements are still the same. The Shall requirements cover what the app needs to be able to do for it to have any possibility of replacing the old console. Completing these requirements shows that the iPad can communicate with the bike and that the user can interact with the iPad.
### 8.2.2 Should requirements

These requirements are kinda necessary for the application to be useful but they are also a bit design specific. The design of Gnat is also the main reason why wattage-based training was not implemented. Four programs with four time settings and four distance settings gives 32 program combinations. This would seem to be enough variation in training programs. A wattage based training program could be added to the training program page without changing the design of the app. However the biking page would have to be different with some element that helped the player stay on the designated wattage. There would also be need for some mechanics in the back end that currently does not exist. There was just not enough time to complete this requirement.

### 8.2.3 Want requirements

<table>
<thead>
<tr>
<th>Want</th>
</tr>
</thead>
<tbody>
<tr>
<td>that there are real-time graphs that show trends in speed / pulse / watts etc.</td>
</tr>
<tr>
<td>that there are alternatives for pulse based training</td>
</tr>
<tr>
<td>that there is a ghost mode (match your own best time / minimum heart rate / maximum watts on a preprogrammed path)</td>
</tr>
<tr>
<td>that there is a connection to Springtail</td>
</tr>
<tr>
<td>that there is a connection to Facebook / twitter / blogs</td>
</tr>
<tr>
<td>that there is a pinch to zoom in the graphs</td>
</tr>
<tr>
<td>a long term exercise planner</td>
</tr>
</tbody>
</table>

These leisure requirements where to be implemented if time allowed. Time have however not been abundant and therefore few of them where completed. They are also very design bound and with the current design of Gnat other want requirements that was not even though of in the beginning of the project would be better suited.
8.2.4 Feature list

With Gnat you can

- change the resistance of the cycle
- see your current speed
- see the distance you cycled
- see the time you cycled
- see the wattage you generated
- see the calories burned
- see your pulse
- see your average speed
- choose from 32 preprogrammed workouts
- upload training results to Springtail

8.2.5 Soft specifications

It’s harder to determine whether or not the application meets the soft requirements. Since no testing have been done I can only give my personal opinion in this matter.

Gnat has good usability, the chosen design really streamline the users actions and make the application intuitive even for first time users. There are of course always improvements that can be made and I have a list of things that I would like to change and make better.

The app itself have fast response times. When you click a button it almost immediately reacts. This have never been an issue since the complexity of the app is fairly low. No special optimization have been needed to meet this requirement. There is another side of it though. The reaction times of the bike is very slow. This will give slow reaction times when the user for example changes the resistance of the bike.

Gnat is at the moment not very Good looking. It has a clean and app-typical design but there are no art to improve the user experience. This aria could defiantly be improved. Even though the app is not super good looking its very simple and clean. The fairly low complexity once again helps to achieve this goal.
9 Analysis of results

The implementation of gnat shows that it is possible to replace the traditional console on EB200 with a tablet one. From this implementation alone we can however say very little about how well Gnat replaces the old console. There will, as touched on before, always be several factors influencing how “good” a solution is. The only thing shown is that there in this very specific circumstance, a solution that involves a tablet with Wi-Fi communication is possible.

9.1 Practical use of result

For Wemotion Gnat and the knowledge gained during development is very valuable. There is a high chance that Wemotion will be part of the development of a Wi-Fi compatible exercise bike. If that is the case Gnat will most likely be used and developed further. So not only is the implementation valuable but also this report which details what needs to be done next to improve Gnat.

9.2 Validity of Gnat

The implementation only proves that it is possible to use a iPad as controller/console for the EB200 bike not if it is advantageous to do so. To answer this question with certainty would require more research but based on the knowledge gained during the thesis I can ponder over the pros and cons.

Touchscreens are generally harder to use than an interface with physical buttons. Especially when the user is distracted doing something else. For example, research presented at the Driver Distraction and Inattention Conference at Chalmers 2011 show that drivers easily get distracted by touchscreen interfaces in the car.¹ Technical innovation may however fix some of the issues of touchscreen applications before they even fully get acknowledged.

9.2.1 Tactile sensations & Haptic technology

Touchscreens may not lack physical feedback for much longer. Both Disney Research², Senseg³, Immersion⁴ and others are developing technologies that could allow touchscreens to give physical feedback. Some of these technologies might be out in the market as soon as within a year.

¹ Norberg, Stefan; Rahe, Ulrike (December 15th 2009). Touchpad as interaction input control for use of in-vehicle infotainment systems
² Bau, Olivie; Poupyrev, Ivan; Israr, Ali; Harrison, Chris (2010). TeslaTouch
³ Senseg.com (2012). Senseg Is Haptics Re-Imagined and Realized
⁴ Immersion Corporation (2012). TouchSense® Haptic (Tactile) Feedback Technology
9.3 Abstraction of results

With more time it would have been interesting to take the research to a more general level. The fascinating more abstract question here is: “To what degree can we change consoles and physical interfaces to mobile touchscreens?”. As mentioned on in the ingress of this thesis we can already see more and more of these usages everywhere. Your phone can be used to control your car, stereo and TV. Are these just fun ways to use your phone and brag to your friends or is it a beginning of a trend?

Where would it be possible to use your phone or tablet as user interaction tool? Truth is probably more or less anywhere where you can use a touch screen. For example, today some modern stoves have touchscreens instead of the traditional knobs. Would it be feasible to use your phone or tablet to control your stove? Would this have any advantages or disadvantages?

9.4 Alternative solutions

The way that Gnat solves the problem of replacing the old console with a touchscreen is of course not the only way it could be done. There are numerous alternative approaches. Here we will touch upon some of them.

9.4.1 Alternative hardware

There a huge difference between an integrated touchscreen solution and an external tablet solution. The interface for the user could look the same in both cases but implementation and production would be completely different. Very early in this thesis focus was on a wireless tablet solution. It would however be entirely possible to instead have a built in touchscreen and computer on the bike. Some of the points brought up in this thesis, such as touchscreen versus real buttons would still be valid. Other factors would be completely different. A company that are looking to realize this product should look into the matter carefully and make a cost/benefit analysis of whether a built in computer with touchscreen is an alternative.

9.4.2 Alternative software

A lot of thought have been put behind the general design of the Gnat application. Still, there are a lot of completely different approaches and Gnat is probably far from the best way to design this kind of application.

An alternative could be a single screen approach, where something similar to my bikeScreen is always shown. From there you can simply change different parameters and settings. Another alternative would be to simply copy the old console in touchscreen form. That is to have the same seven buttons in the bottom and a display presenting the same information above that. This solution would of course not use any of the potential presented by a tablet.
If the specifications in this thesis are not set in stone you could even have a very abstract and artsy approach. Maybe not show any numbers at all but instead demonstrate speed and resistance by blobs floating around with shifting speeds and colors. The imagination is pretty much the limit of how to design an app and its impossible for me to say whether my solution is better or worse than any others.

9.5 Future work

Gnat has successfully completed most of the requirements that were set up in the beginning of the development. The application has also managed to answer the fundamental question asked in this thesis. But it is far from a completed product and if development were to continue there are several aspects to work on.

9.5.1 User tests

User tests is probably the single most important next step. If the old console were to be swapped for Gnat or a similar touch application in the future it is necessary that users find the new solution superior to the old. Otherwise it is not worth the extra cost. If the tests show that a touch application is the way to go there are also numerous decisions inside the application that end-users should help determine. Most of the end-customers will probably not have a lot of IT-experience.

9.5.2 Features to implement

There is a point in keeping the application simple. More features are not always better. With this in consideration there are still several things that would improve the application. To bring up a few examples, wattage and pulse training that where mentioned in section 8.2 would probably be valuable. As well as long term training programs that can also be found in the early Want requirements.

With the current design real time graphs would probably not be very valuable while cycling, but graphs in the end-screen showing the result of the training session in more detail could be very interesting. In Webike there is a test called VO2Max that gives an approximation of the users oxygen intake capacity. This could be implemented in Gnat as well.

9.5.3 Graphical improvement

The graphics of Gnat can be improved a lot. This includes both making the current content better looking with background pictures, custom buttons and fonts, but also includes creating new graphical content. Something that I envisioned at the start of development was a small animated biker that would bike along the height-map of the exercise-session. Thus showing the progress of the exercise and give something to look at while biking.
9.6 Conclusion

After completing this thesis work I have proven that it is possible and my view is that it is also beneficial to use a tablet as a portable controller or console for an exercise-bike. Swapping the static physical interface to a flexible tablet application gives so many possibilities. Mini-games, statistics upload, training program downloads, better usability and easy to update software to mention a few.

There are however technical and design difficulties that has to be solved in order to get a streamlined product. In this thesis work I have discovered and presented some of these difficulties as well presented solutions for some. I hope that this paper will help Wemotion or anyone else that wants to develop this idea further.
10 Sources

10.1 Web sources

Tyer, Dominic (Feb 15th 2012). Twenty six per cent of European doctors own an iPad [www]
<http://www.pmlive.com/digital_intelligence_blog/archive/2012/feb_2013/european_doctors_ipad_use_ownership>
Retrieved: 15/4 2012

Harvell, Ben (January 16th 2011). How musicians are using the iPad [www]
Retrieved: 15/4 2012

HomeTheaterReview.com (May 17th 2010). Xantech Using Apple iPad As Controller
<http://hometheaterreview.com/xantech-using-apple-ipad-as-controller/>
Retrieved: 15/4 2012

Apple Inc. (2012). MFi Program
Retrieved: 4/5 2012

Gaudiosi, John (May 5th 2012). New Research Shows Apple Still Winning the Video Game War Against Android
<http://www.forbes.com/sites/johngaudiosi/2012/05/05/new-research-shows-apple-still-winning-the-video-game-war-against-android/>
Retrieved: 10/5 2012

Retrieved: 8/5 2012

Gartner.com (November 15th 2011). Gartner Says Sales of Mobile Devices Grew 5.6 Percent in Third Quarter of 2011; Smartphone Sales Increased 42 Percent
<http://www.gartner.com/it/page.jsp?id=1848514>
Retrieved: 9/12 2012

Nielsen, Jakob (2005). Ten Usability Heuristics
<http://www.useit.com/papers/heuristic/heuristic_list.html>
Retrieved: 5/9 2011

Nielsen, Jakob (May 10, 2010). iPad Usability: First Findings From User Testing
Retrieved: 9/12 2011
Nielsen, Jakob (May 23, 2011). *iPad Usability: Year One*  
<http://www.useit.com/alertbox/ipad.html>  
Retrieved: 5/9 2011

Bau, Olivie; Poupyrev, Ivan; Israr, Ali; Harrison, Chris (2010). *TeslaTouch*  
<http://www.disneyresearch.com/research/projects/hci_teslatouch_drp.htm>  
Retrieved: 5/5 2012

Senseg.com (2012). *Senseg Is Haptics Re-Imagined and Realized*  
<http://senseg.com/technology/senseg-technology>  
Retrieved: 5/5 2012

Immersion Corporation (2012). *TouchSense® Haptic (Tactile) Feedback Technology*  
Retrieved: 5/5 2012

Apple Inc (Oct 14th 2011). *Bluetooth Accessory Design Guidelines for Apple Products*  
Retrieved: 2/2 2012

Apple Inc. (Last updated: 2012-03-07). *App Design Strategies*  
Retrieved: 10/3 2012

Apple Inc. (July 20th 2010). *Apple Reports Third Quarter Results*  
Retrieved: 12/4 2012

Apple Inc. (October 18th 2010). *Apple Reports Fourth Quarter Results*  
Retrieved: 12/4 2012

Apple Inc. (January 18th 2011). *Apple Reports First Quarter Results 2011*  
Retrieved: 12/4 2012

Norberg, Stefan; Rahe, Ulrike (December 15th 2009). *Touchpad as interaction input control for use of in-vehicle infotainment systems*  
<http://www.chalmers.se/safer/ddi2011-en/program/papers-presentations>  
Retrieved: 5/5 2012

Fidgeon, Tim (September 05th 2011). *Tablet PC and iPad - usability guidelines*  
Retrieved: 10/12 2011
10.2 Printed Sources

Shea, Brian (Apr 1st 2002). Have You Locked the Castle Gate? Home and Small Business Computer Security
Addison Wesley, Boston

Hillegass, Aaron (November 22nd 2011). Objective-C Programming: The Big Nerd Ranch Guide
ADDISON-WESLEY, Indianapolis

Michaelis, Mark (January 2011). Essential C# 4.0 3rd Edition
ADDISON-WESLEY, Boston

10.3 Personal communication

Östlund, Anders (2011-2012). Senior Developer at Wemotion AB
Upphovsrätt
Detta dokument hålls tillgängligt på Internet – eller dess framtida ersättare –från publiceringsdatum under förutsättning att inga extraordinära omständigheter uppstår.
Tillgång till dokumentet innebär tillstånd för var och en att läsa, ladda ner, skriva ut enstaka kopior för enskilt bruk och att använda det oförändrat för ickekommersiell forskning och för undervisning. Överföring av upphovsrätten vid en senare tidpunkt kan inte upphäva detta tillstånd. All annan användning av dokumentet kräver upphovsmannens medgivande. För att garantera äktheten, säkerheten och tillgängligheten finns lösningar av teknisk och administrativ art.
Upphovsmannens ideella rätt innefattar rätt att bli nämnd som upphovsman i den omfattning som god sed kräver vid användning av dokumentet på ovan beskrivna sätt samt skydd mot att dokumentet ändras eller presenteras i sådan form eller i sådant sammanhang som är kränkande för upphovsmannens litterära eller konstnärliga anseende eller egenart.
För ytterligare information om Linköping University Electronic Press se fölagets hemsida http://www.ep.liu.se/

Copyright
The publishers will keep this document online on the Internet – or its possible replacement – from the date of publication barring exceptional circumstances.
The online availability of the document implies permanent permission for anyone to read, to download, or to print out single copies for his/hers own use and to use it unchanged for non-commercial research and educational purpose. Subsequent transfers of copyright cannot revoke this permission. All other uses of the document are conditional upon the consent of the copyright owner. The publisher has taken technical and administrative measures to assure authenticity, security and accessibility.
According to intellectual property law the author has the right to be mentioned when his/her work is accessed as described above and to be protected against infringement.
For additional information about the Linköping University Electronic Press and its procedures for publication and for assurance of document integrity, please refer to its www home page: http://www.ep.liu.se/.

© Björn Falke