Reykjavík University 12. Dec 2013

eefacta Mobile Timesheet
Windows Phone 8

Fall 2013

Einar Sigurður Sigurðsson
Guðjón Ingi Gunnlaugsson
Guðni Matthíasson

Instructor: Stefán Freyr Stefánsson
External Examiner: Hlynur Sigurþórsson

T-404-Loka
Department of Computer Science

Contents

1 Introduction **1**

1.1 About Unimaze . . . 1

1.2 The Project . . . 1

1.3 Software Used . . . 2

2 Product Design **2**

2.1 Code Architecture . . . 2

2.2 Data Storage . . . 2

2.3 Synchronization . . . 3

2.4 User Interface . . . 4

2.5 Testing . . . 4

3 Progress Overview **4**

3.1 Preparation . . . 4

3.2 Product History . . . 5

3.3 Product Plan . . . 7

3.3.1 Risk Assessment . . . 7

3.3.2 Methods . . . 8

3.3.3 Workplace . . . 8

3.3.4 Attendance . . . 8

3.3.5 Documentation . . . 9

3.3.6 Backlog . . . 10

4 Sprint Overview **12**

4.1 Sprint 1 - UI Design: Proof of Concept . . . 12

4.1.1 Burndown . . . 12

4.1.2 Retrospective . . . 12

4.1.3 Product Status . . . 13

4.2 Sprint 2 - Login and Databinding . . . 14

4.2.1 Burndown . . . 14

4.2.2 Retrospective 14
4.2.3 Product Status 15
4.3 Sprint 3 - Networking, Error Handling, and State Persistence 16
4.3.1 Burndown 16
4.3.2 Retrospective 17
4.3.3 Product Status 17
4.4 Sprint 4 - Reiteration 18
4.4.1 Burndown 18
4.4.2 Retrospective 19
4.4.3 Product Status 19
4.5 Sprint 5 - Finishing Features 20
4.5.1 Burndown 20
4.5.2 Retrospective 20
4.5.3 Product Status 21
4.6 Sprint 6 - Finalizing the Project 22
4.6.1 Burndown 22
4.6.2 Product Status 22

5 Conclusion **24**
5.1 Team Experience 24
5.2 Scrum 24
5.3 Programming For Windows Phone 24
5.4 The Future 24

1 Introduction

Eefacta Mobile Timesheet is a final project at Reykjavík University in collaboration with Unimaze. This report covers the progress, structure, and description of the project.
This project is the result of work in the autumn semester 2013 by Einar Sigurður Sigurðsson, Guðjón Ingi Gunnlaugsson, and Guðni Matthíasson. Our project instructor is Stefán Freyr Stefánsson, and the product owner and contact at Unimaze is Markús Guðmundsson. Work on the project began August 26th and ended December 12th

1.1 About Unimaze

Sendill is Unimaze ehf. is an Icelandic start-up company, founded in 2003 and began operating, specializing in electronic invoicing solutions, in 2006. The company is mostly owned by its founders and employees, but other shareholders include Íslandspóstur and DK hugbúnaður. Company logos can be seen on figure 1.

Figure 1: Company Logos

1.2 The Project

The goal was to implement and design a mobile timesheet application for Windows Phone 8. The app was to operate against an existing back-end system from Unimaze, eefacta. eefacta handles the creation and distribution of electronic invoices.

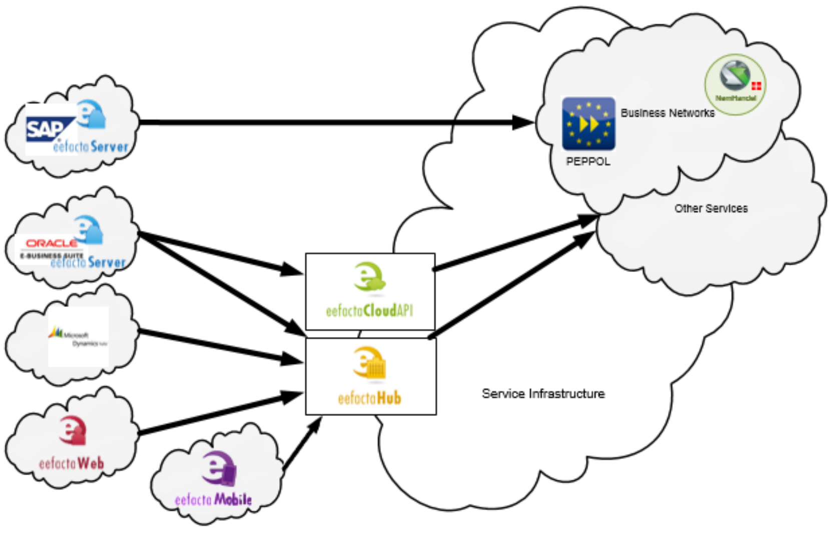


Figure 2: eefacta Layout

The timesheet application will become a part of the newest member of the eefacta family which can be seen on figure 2, eefacta Mobile, and is intended to provide another interface for employees of the companies using the service to increase efficiency. Companies using the service can define a list of customers, job categories and resources. An employee registered under that company can then register new customers, and insert entries indicating units of work completed based on the selected resource for a particular customer. These entries can then be billed electronically. this involves sending invoice requests to the Eefacta servers, where they are evaluated and sent.

1.3 Software Used

Developing for Windows phone requires Visual Studio. The school supplied the developers with Visual Studio 2012 Ultimate. The group was also supplied with extended trial keys for Telerik which is a UI framework for C#, providing easy data-bound input fields and forms, which Unimaze has been using for the eefacta Web interface. For version control Unimaze maintains a subverison server and the developers received VisualSVN trial keys for easy integration with Visual Studio. They also have a continuous integration server running on TeamCity, which automatically built new revisions as they were uploaded.

2 Product Design

2.1 Code Architecture

When starting a new project using the Windows Phone framework there is a certain structural template which served as a starting point to be expanded with helper classes and essential libraries for the project. Given the original nature of the assignment to make the product available on both the Phone and RT tablet, the developers emphasized separation of concerns to be able to use the back end of the code for both versions. All database code is in one class and all code interfacing with the REST service in another. The project was split up into 3 parts, Shared library code (POCO objects, database structure, REST api, etc.), a Background Agent (Handles data synchronization and notifications), and the base project which contains the views and associated code. The project structure can be seen in figure 3.

2.2 Data Storage

For this application, the only logical choice for a local storage medium was Microsoft SQL Compact with LINQ to SQL. The data context for this is set up in such a way that the database structure is code first and each table definition can be used as a POCO object in the application. Model objects that could be created in the application, such as job entries, use database generated identifiers, with the actual ID field set to 0. On successful upload, the actual entry identity is returned and updated locally. Other model objects use the

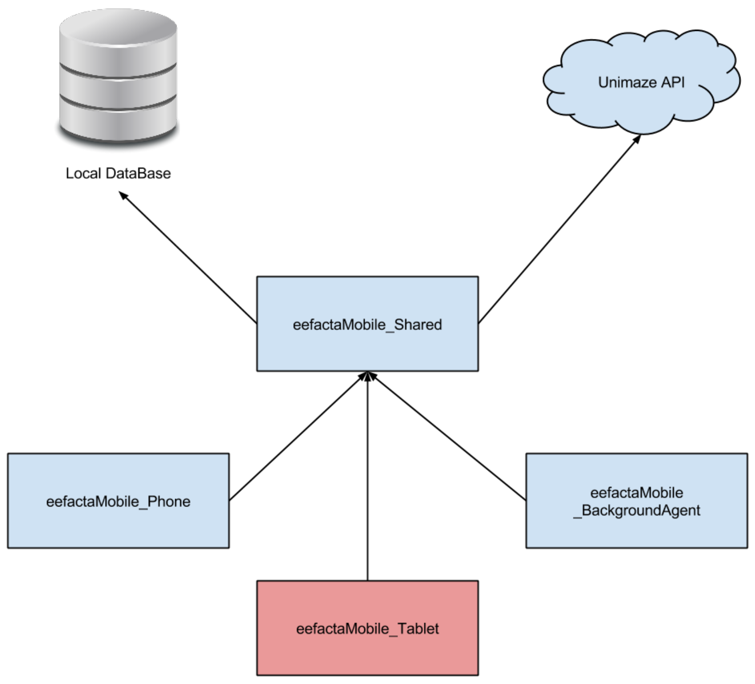


Figure 3: Current project setup (eefactaMobile_Tablet is red because it was cancelled)

identifiers provided by the REST service directly, as they can not be created locally. The database enforces no foreign key restraints because everything in storage is synchronized via the REST service, the back end database of which enforces these restraints.

2.3 Synchronization

Because the data can be altered on multiple devices (using different interfaces) simultaneously, the application needs to be able to stay up to date in order to give the user up-to-date data. This is done by the Windows background agent that automatically calls our sync functions every hour if the phone is online. If the application has not be used in 14 days it stops (An OS feature on Windows Phone 8). The phone does not require a network connection to use the application, so when no connection is available, new entries are stored locally with an “unsynced” flag until the phone connects to the internet. Then, the application will ensure that data on the phone that has been uploaded but is no longer available online gets deleted, while entries with the “unsynced” flag set are properly uploaded and updated. A conflict may arise when a job entry is edited on the phone and in the web interface without the application synhcronizing in between. When this happens, the device will alert the user that a conflict has occurred and replace the local version of the entry with the one on the server.

2.4 User Interface

The design of this application is based on guidelines by Microsoft to have a uniform experience with other windows phone applications. The goal was to allow the user to get as much information about his hours as possible over a variety of periods; the first screen showing all entries in chronological order, then the week and month overviews to be a natural extension of the initial page, accessible with swipes to either side. The week overview is a stacked bar chart showing hours registered per day in a given week; the bars link to a list view for the corresponding day, listing entries in more detail. The month overview is a calendar with numerical data on the number of registered hours per day. The calendar cells link to the day view just like bars in the week overview. By flicking up or down in the week and month overviews, the user can cycle through past and future weeks/months with natural looking animations. The application uses resource strings to accommodate multiple languages and currently supports Icelandic and English translations. Date and number formats follow the regional format of the phone. Between 5:00 and 6:00 PM each evening, the background agent displays a notification to the user, reminding him to log entries for the day. When data is changed in the application or the synchronization routine has finished while the application is open, the user is presented with a “toast”; a simple, elegant and unobtrusive notification that slides in on the top half of the screen, alerting the user of such events. Should the user become confused with the interface, every view has a “help” link in the application bar context menu. The link directs the user to a web page containing detailed information on available actions on the current page.

2.5 Testing

This project was not run with test driven development but the team kept a document to keep track of known bugs and in the later half of the project, when the team had a working prototype, the product owner and his associate performed user tests regularly and provided bug lists. During the final days of testing, the application appeared stable and bug-free.

3 Progress Overview

3.1 Preparation

Development began very late, because no product backlog was available until September 16th. In the meantime, the group’s assignment was to create a mock UI design and flow concept for the application. The user interface was then redesigned, taking into account increased knowledge of available user control modules, Windows Phone 8 design guidelines and new ideas from the product owner and development team.
Figure 4 shows a rough concept of the revised design:

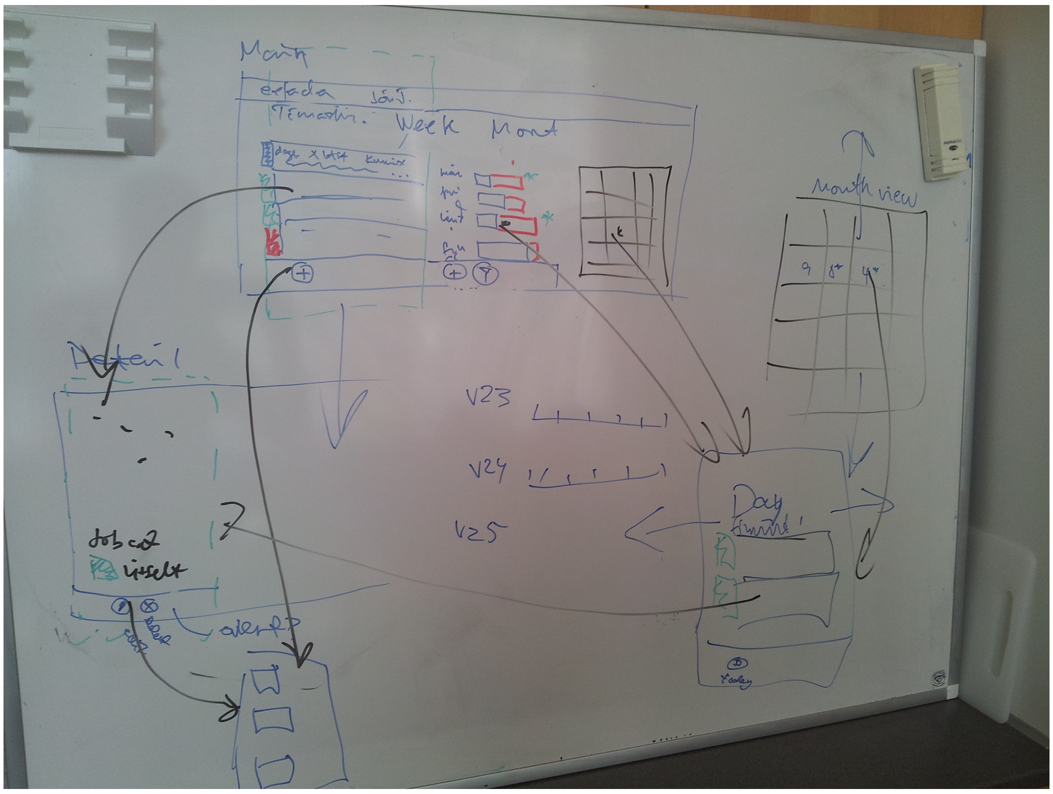


Figure 4: UI design for eefacta Mobile Timesheets

3.2 Product History

At first, the envisioned software suite that constituted the goal of the final project included, apart from the Windows Phone mobile timesheets app, a version of the app for Windows 8 tablet devices along with a sales application for the tablet. The developers soon realized that this could not be accomplished in the given timeframe, and consulted the product owner. The sales application was put on hold until further notice, but it soon became apparent that this reflected poorly on the overall plan, rendering it inconclusive by stating that the sales app might possibly be implemented if the team managed to finish the timesheets app in less than the due time. Therefore, in the early stages of development, the team called another meeting to insist that the sales application be completely removed from the project's backlog. This was granted, and as a direct result, the long term plan became easier to visualize.
Development began slowly. In the beginning, lack of resources and facilities slowed the process down, although once the development team had established a clear set of initial goals and implemented a navigable UI skeleton, progress was quickly made. With the office space provided in October, the rate of progress improved: The team no longer had trouble finding places to work and could meet with the product owner on a regular basis. The developers have since then improved upon the scrum workflow and overall product planning. Later on in the development process, the decision was made to reduce the scope

of the project further, restricting it to the phone application. This allowed the team to focus more thoroughly on lower-priority features that would improve the usability of the timesheets app.

3.3 Product Plan

3.3.1 Risk Assessment

Values are given on the interval 1-10. The risk factor (RF) is a product of each risk's probability (P) and severity(S). The risks are listed in order of their risk factors (RF).

ID	Description	P	S	RF ↓	Status
4	The applications expected by the product owner provide an unrealistic amount of work for one semester. **Mitigation Plan:** Start developing as soon as possible. **Damage Control Plan:** Negotiate reduction of the project's scope.	8	8	64	Resolved
2	The discrepancies in expectations towards this project between the product owner and the school persist. **Mitigation Plan:** Set up a meeting with the product owner explaining that the backlog and overall plan should reflect the priority of school grades over the end product(s). **Damage Control Plan:** Attempt to fulfill the school's specifications for the course while, in every way possible, still following the product plan.	4	9	36	Avoided
5	The learning curve of the Windows Application framework proves to be too steep. **Mitigation Plan:** Conduct thorough research before implementing anything new. **Damage Control Plan:** Negotiate further reduction of the project's scope.	4	7	28	Avoided
3	The virtual machines on which developers were expected to work on are not ready a month into the semester. **Mitigation Plan:** Acquire a list of software required to develop on the platform and request frequent updates on the status of the VMs. **Damage Control Plan:** Install the software and commence work on our own machines.	7	2	14	Resolved
1	The development team has no reliable working facility throughout the duration of the project. **Mitigation Plan:** Request a dedicated room in Reykjavík University. **Damage Control Plan:** Furnish Einar's garage.	3	4	12	Avoided

Risk 1 was out of the picture on October 1st, when Unimaze provided office space at Innovation House in Eiðistorg. The mitigation plan for risk 2 was eventually successful; after the second discussion about this problem with the product owner, the scope of the project was reduced. This also led to the avoidance of risk 4. The virtual machines addressed by risk 3 never became operational. One was deployed, but there were immediate problems regarding nested VMs (running the phone emulator on a virtual machine is apparently impossible), so the damage control plan was executed with great success. Risk 5 was quickly mitigated: If the developers had any doubts about how to implement a new feature, thorough research was conducted before any code was written.

3.3.2 Methods

For this project the scrum methodology was chosen with Markús Guðmundsson as the product owner and Guðjón as scrum master, Programmers were all members of the team. The product owner decided to make the stories for the product backlog and the team received them on September 17th. A 2 week sprint length was chosen because of uncertainty in school schedules regarding return assignments for other classes allowing for more flexibility. To manage the process, Version One, an online scrum board, was used to track sprints. When the backlog was delivered the team came together and assigned story points to the stories. At the start of each working day the group was to sit down and talk about the previous work day and what the plan was for that day. After each sprint the team would meet with the product owner to showcase the work done in the sprint and plan what stories were to be in the next sprint , then the three members of the team would review the sprint and add tasks to upcoming stories.

3.3.3 Workplace

Unimaze did not have facilities available to the team when the project began so for the first 6 weeks they met at the university. On October 1st the team was provided with a brand new, spacious office at Innovation House, which enabled the team to focus more on what was important: The product.

3.3.4 Attendance

All three team members took 30 ECTS units this semester, the team planned what times they could work on the project together and set up a schedule.
This gave each team member 17 hours a week, on top of this we would also work Fridays when we had sprint reviews, this would result in 102 hours per sprint.

Table 1: Time plan

Day	From	To	Total
Tuesday	10	17	7
Wednesday	12	17	5
Thursday	12	17	5

3.3.5 Documentation

To help the team with managing the project a docs sheet was set up for the team to log their working hours and the accomplishments of each day. It was the responsibility of each team member to log their own work. During meetings with instructors and Unimaze employees notable items were written down for further discussion by the team.

3.3.6 Backlog

Status in story points:

TO DO:	0
IN PROGRESS:	0
DONE:	124
Canceled:	45

ID	Story	Points	status
s-001	**[DEV-ENV]** As a developer I want to set up my machine so it is connected to the Unimaze environment and is ready for development.	5	Accepted
s-002	**[UI SKELETON]** As a Developer I want I want to set up a UI skeleton with navigation beween forms so that navigation and conformance with UI guidelines can be demonstrated and reviewed.	8	Accepted
s-003	**[LOGON/REGISTER]** As a Developer I want to set up simple framework for login and registration so that the workflow can be verified, offline mode entered if applicable.	5	Accepted
s-004	**[SYNCHRONIZE]** As a Developer I want to set up syncronization for one type of data so that the synchronization framework can be verified and demonstrated.	13	Accepted
s-005	**[ONLINE/OFFLINE]** As a Developer I want to switch intelligently between offline and online mode so that the application gracefully syncronizes updates during connection changes.	13	Accepted
s-006	**[ERROR HANDLING]** As a Developer I want to gracefully handle and detect errors so that user experience is good and bugs are detected soon.	5	Accepted
s-007	**[STATE PERSIST]** As a Developer I want to gracefully resume from sleep state so that user experience is good and conforms with the platform UX guidelines.	3	Accepted
s-008	**[LOCALIZE]** As a Product owner I want the UI to be localizable and use the OS culture so that it can be used in different markets of the world.	5	Accepted

s-009	[**BASE DATA**] As a Developer I want to sync all base data and store on device so that it can be accessed quicly within the application.	8	Accepted
s-010	[**QUICK ADD/NOTIFY**] As a Mobile user I want to be able to quickly add time sheet entries so that entries are added quicly and easily.	5	Accepted
s-011	[**WEEK VIEW**] As a Mobile user I want to be able to see overview at a glance for a week entering so that wrong entries or missing entering entries can be prevented.	3	Accepted
s-012	[**MONTH VIEW**] As a Mobile user I want to be able to see overview at a glance for a month so that entering wrong entries or missing entering entries can be prevented.	5	Accepted
s-013	[**ACTION VIEW**] As a Mobile user I want to be able to instantly bill based on my entries so that I will receive payment promptly.	8	Accepted
s-014	[**STATISTICS**] As a Mobile user I want to see statistics of entered hours so that I can easily see how I am performing.	20	Future
s-015	[**ORIENTATION/FORM**] As a Product owner I want to be usable on multiple platforms and form factors so that it can be used by as many as possible.	25	Future
s-016	As a Developer I want to set up TeamCity so that I can have Continous Integration.	N/A	Accepted
s-017	[**Debug**] As a developer, I want to eliminate bugs specified by the bug report from the product owner so that user experience is better.	18	Accepted
s-018	[**Hand-In**] As a developer, I want to finish all documentation so that it can be handed in by Thursday 12.12.'13.	20	Accepted

4 Sprint Overview

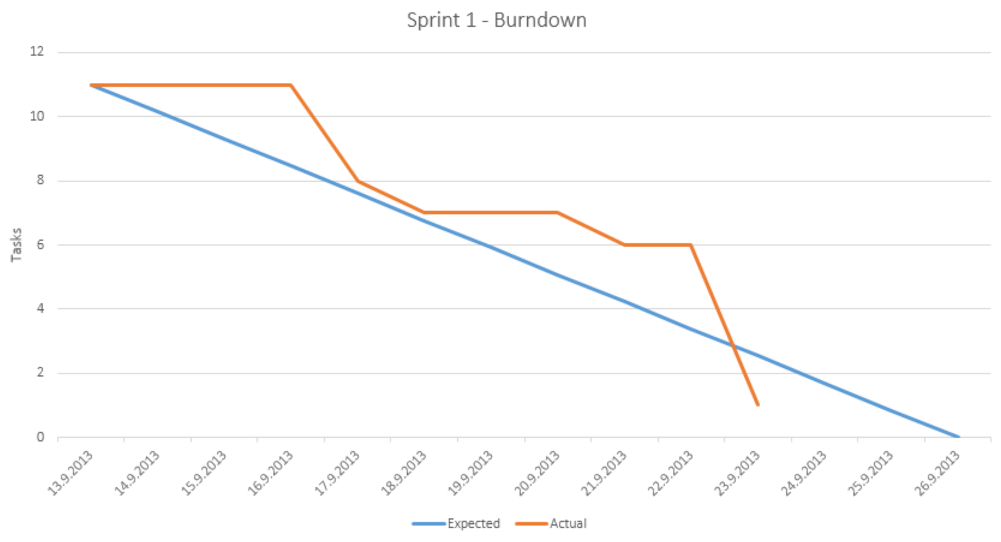
4.1 Sprint 1 - UI Design: Proof of Concept

The first sprint was intended for the development team to install the software and SDKs required to develop applications for Windows Phone and implement a navigable UI skeleton.

4.1.1 Burndown

s-001 (5 points) **[DEV-ENV]** As a Developer I want to set up my machine so it is connected to the Unimaze environment and is ready for development.

s-002 (8 points) **[UI SKELETON]** As a Developer I want I want to set up a UI skeleton with navigation beween forms so that navigation and conformance with UI guidelines can be demonstrated and reviewed.



Time spent:	118 hours
Stories closed:	s-001, s-002
Storypoints finished:	13

4.1.2 Retrospective

The first sprint gave the developers a look at the work ahead by acquainting them with the programming environment and introducing Windows Phone data binding methods. However, as can be seen on the burndown chart, the team was lacking in knowledge of some scrum procedures. The y axis on the chart is measured in tasks, which may vary in scope, rather than an even unit of measurement such as estimated working hours.

The development team also became increasingly aware of the project's actual scale as they were better introduced into the development framework. The original idea was to make a Windows 8 RT tablet version of the same software and a sales application for the tablet; offering a "shopping basket" interface for travelling salesmen to register and bill sales. Team members agree that this would have taken much more time than approximated by the course specification.

What went well:

+ The work done in this sprint created a starting point for the development team.
+ Developers acquired a basic understanding of the software design guidelines for Windows Phone 8.
+ Visual design was mostly finished.

What should be fixed:

– Communication between developers and product owner needed to be more frequent.

– The product owner's expectations for this project to result in a market-ready set of applications were unrealistic given the timeframe.

– The developers needed to find working facilities as soon as possible.

4.1.3 Product Status

By the end of sprint 1, the product was nothing but a UI skeleton. The main page was a panorama page with three views: Recent Items, Week View and Month View. Apart from that, there was an About page, a Day View page, an Add/Edit View page and a Details View page. None of them contained any data but could be navigated to via temporary links in the main page action bar.

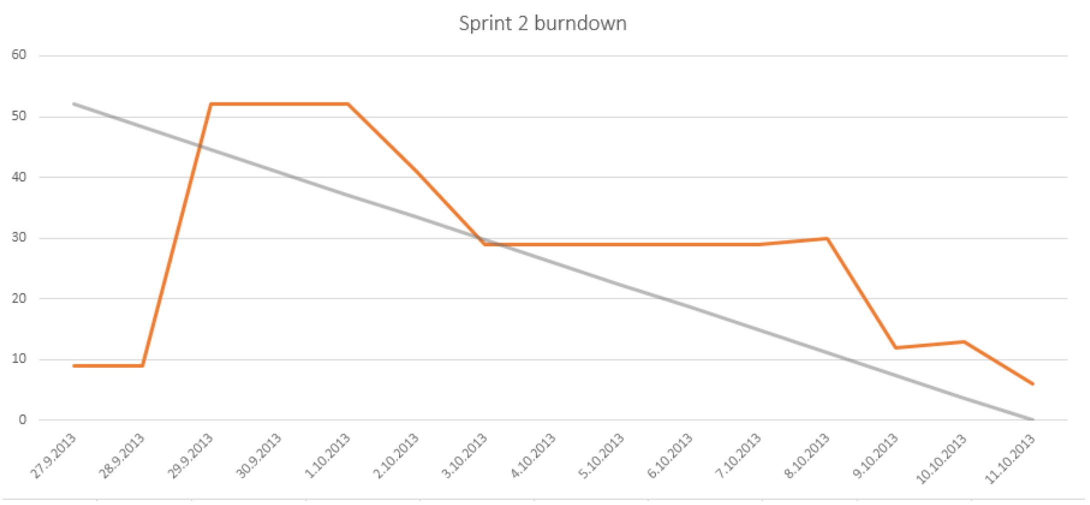
4.2 Sprint 2 - Login and Databinding

The second sprint was the first real programming sprint. The plan was to implement a simple login framework and handle basic representation of local data.

4.2.1 Burndown

s-003 (5 points) **[LOGON/REGISTER]** As a Developer I want to set up simple framework for login and registration so that the workflow can be verified, offline mode entered if applicable.

s-008 (5 points) **[LOCALIZE]** As a Product owner I want the UI to be localizable and use the OS culture so that it can be used in different markets of the world.



Time spent:	128 hours
Stories closed:	s-003, s-008
Storypoints finished:	10

4.2.2 Retrospective

Following a discussion about the shortcomings of the previous sprint regarding the scrum process, the developers started estimating working hours required to finish each task in order to make the burndown more accurate. For two days after the sprint began, the development team could not meet due to other school projects. Rather than assigning tasks to the stories as they were assigned to the sprint, most of them were added on the third day (as the burndown chart indicates). However, sprint 2 proved a great launching platform for actual product development, since most of the uncertainties relating to the product backlog had been clarified. Another contributing factor was the office space provided on October 1st, which greatly improved team morale and working conditions.

The second week of this sprint, 7.10. - 11.10. was the so-called "Disaster Week" at Reykjavík University. There were no classes or assignments, and as indicated by the burndown chart, the development team used the time to work on the final project.

What went well:

+ Efficient communication amongst team members regarding uncertainties about product requirements and possible solutions thereof led to the creation of a more cohesive project plan.
+ The developers systematically wrote task definitions that could be used as acceptance tests and assigned them to stories.

What should be fixed:

– Communication between product owner and developers still needed improvement.

– The development team had to improve records-keeping and remember to log working hours and managing tasks every day.

– The team needed to write tasks for stories on the sprint backlog in the beginning of the sprint.

4.2.3 Product Status

By the end of sprint 2, the product had a rudimentary login routine, despite the fact that the back-end API had not yet been implemented by the programmers at Unimaze. Another new feature was the use of resource strings throughout the UI, allowing the user to view the application in both English and Icelandic.

4.3 Sprint 3 - Networking, Error Handling, and State Persistence

On the third sprint, the developers intended to implement a synchronization routine, updating the local storage with online data via RESTful API, an event handler to manage attempts at network access and make sure that the application maintains its state when switching between apps. The goal of the sprint was also to finish implementing core views, such as the main pivot and add view. One member of the team, Guðjón, was overseas during the weekend 18.10. - 21.10.

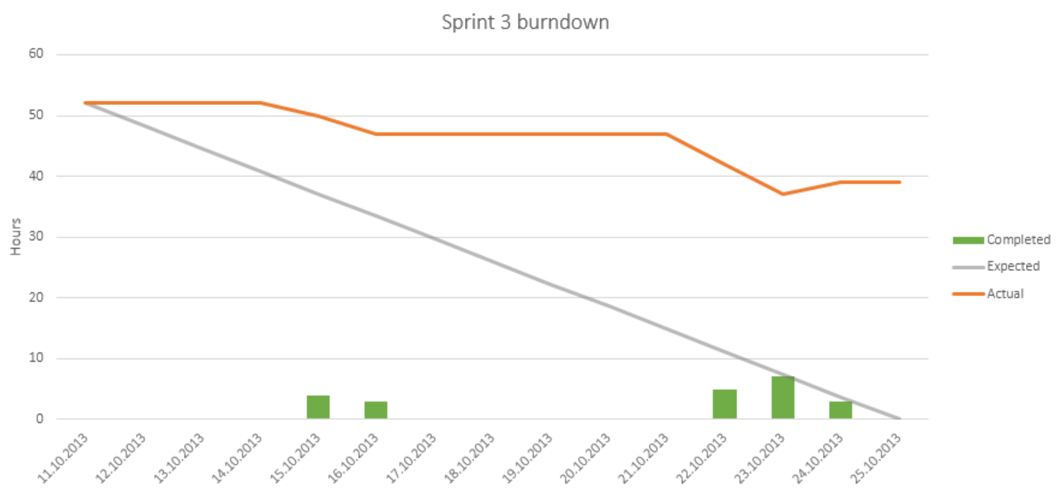
4.3.1 Burndown

s-004 (13 points) [**SYNCHRONIZE**] As a Developer I want to set up syncronization for one type of data so that the synchronization framework can be verified and demonstrated.

s-005 (13 points) [**ONLINE/OFFLINE**] As a Developer I want to switch intelligently between offline and online mode so that the application gracefully syncronizes updates during connection changes.

s-006 (5 points) [**ERROR HANDLING**] As a Developer I want to switch intelligently between offline and online mode so that the application gracefully syncronizes updates during connection changes.

s-007 (5 points) [**STATE PERSIST**] As a Developer I want to gracefully resume from sleep state so that user experience is good and conforms with the platform UX guidelines.



Time spent:	88 hours
Stories closed:	s-005, s-007
Storypoints finished:	18

4.3.2 Retrospective

Due to a heavy workload in other classes, and the absence of a team member over a weekend, the development team had few hours to spend on the project this sprint; so despite having implemented an event handler for network availability, a BugSense module to handle and log unexpected exceptions and a basic client to interact with the online API, the synchronization routine was still not working properly. Thus, the team did not achieve significant burndown. On the sprint review meeting, the decision was made to reevaluate some of the stories (s-004 and s-006) that had proven more difficult than originally estimated, and pass them on to the next sprint.

What went well:

+ The features implemented in the synchronization client gave the developers a good understanding of the RESTsharp library and asynchronous programming, which would prove useful in the sprints to come.
+ Online/Offline mode switching was completed.

What should be fixed:

- The team needed to find more time for the project during weekends.
- After meeting with the instructors, the need to start maintaining a view of total progress became apparent.

4.3.3 Product Status

By the end of sprint 3, the product used bugsense to log and report exceptions as they happened. It managed a compact local database for job entries and associated data. The RESTful API to access online data for synchronization and billing was still under construction by Unimaze, but some progress was made on the synchronization routines. Windows Phone 8 handles state persistence in such a way that the specifications were very easy to meet. If a new instance of the application was started, it replaced any previously running instance, and if the user switched between applications, the state was not affected.

4.4 Sprint 4 - Reiteration

Sprint 4 was used to refactor and improve on existing features, and finish unfinished tasks. One member of the team, Einar, was overseas 6.11. - 10.11.

4.4.1 Burndown

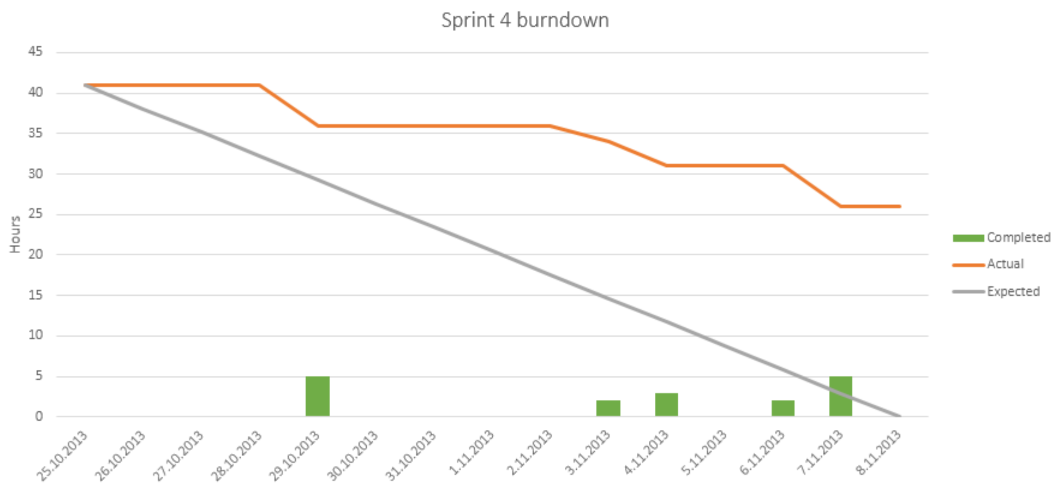
s-004 (13 points) [**SYNCHRONIZE**] As a Developer I want to set up syncronization for one type of data so that the synchronization framework can be verified and demonstrated.

s-006 (8 points) [**ERROR HANDLING**] As a Developer I want to switch intelligently between offline and online mode so that the application gracefully syncronizes updates during connection changes.

s-009 (15 points) [**BASE DATA**] As a Developer I want to sync all base data and store on device so that it can be accessed quicly within the application.

s-010 (5 points) [**QUICK ADD/NOTIFY**] As a Mobile user I want to be able to quickly add time sheet entries so that entries are added quicly and easily.

s-012 (5 points) [**MONTH VIEW**] As a Mobile user I want to be able to see overview at a glance for a month so that entering wrong entries or missing entering entries can be prevented.



Time spent:	88 hours
Stories closed:	s-004, s-006, s-009
Storypoints finished:	36

4.4.2 Retrospective

As the burndown chart clearly depicts, and as stated in the sprint plan, the team did not close a lot of tasks. This was due to poor sprint planning on the developers' part. Instead of creating a new story entailing the work ahead in collaboration with the product owner, and assigning relevant tasks immediately, the developers went on to work outside the scope of the sprint plan. This reflected well on the product, but poorly on the burndown and overall plan.

What went well:

+ Many bugs were eradicated.

What should be fixed:

– The team needed to make sure to plan each sprint in such a way that they could achieve their goals by following the plan exactly.

4.4.3 Product Status

Due to recent developments in the RESTful API and a better vision of the ideal project architecture, the login procedure was refactored to actually authenticate the user against an existing user base on the eefacta servers before any queries were made to the online database. The synchronization client was also refactored and could fetch all base data to populate the local data storage. Local entries were in turn displayed in their appropriate views.

4.5 Sprint 5 - Finishing Features

At this stage in the project the team needed to sit down with the product owner and look at the backlog to select and prioritize the remaining features that could be finished before the deadline.

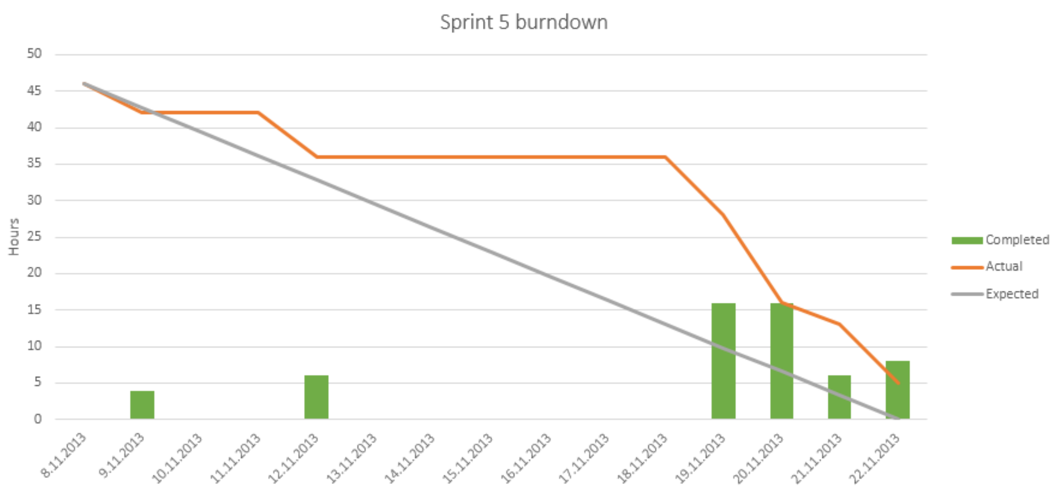
4.5.1 Burndown

s-010 (5 points) **[QUICK ADD/NOTIFY]** As a Mobile user I want to be able to quickly add time sheet entries so that entries are added quicly and easily.

s-011 (3 points) **[WEEK VIEW]** As a Mobile user I want to be able to see overview at a glance for a week entering so that wrong entries or missing entering entries can be prevented.

s-012 (5 points) **[MONTH VIEW]** As a Mobile user I want to be able to see overview at a glance for a month so that entering wrong entries or missing entering entries can be prevented.

s-013 (8 points) **[ACTION VIEW]** As a Mobile user I want to be able to instantly bill based on my entries so that I will receive payment promptly.



Time spent:	135 hours
Stories closed:	s-010, s-012, s-013
Storypoints finished:	18

4.5.2 Retrospective

Most of the features planned for this sprint were successfully finished, leaving only a single feature to be implemented before the feature-freeze of the next week: An asterisk to be displayed in the week view to indicate units other than hours on the corresponding day.

What went well:

+ The product was practically completed in terms of features and usability.
+ The developers worked closely with their Android counterparts and the product owner. This allowed them to quickly eliminate uncertainties as they arose and focus entirely on implementing the remaining features.

What should be fixed:

− The team had difficulty adjusting the application to an ever-changing RESTful API.

4.5.3 Product Status

The application was up and running. It showed a list of all entries in time-order on the first pivot panel of the main page, stacked bar-chart of registered working hours for each day of the week on the second pivot panel and a calendar displaying working hours for each day on the third. The Week View bars and Month View calendar cells brought the user to the appropriate Day View when clicked. In the Recent View and Day View, the user could view details on each entry and edit them or add new ones. The user could also add new customers in the Add/Edit View. The Action View was introduced, showing a list of all job entries for a specified customer, allowing each to be selected and billed. The user could filter entries in the action view by customer, date (to and from) and billing status.

4.6 Sprint 6 - Finalizing the Project

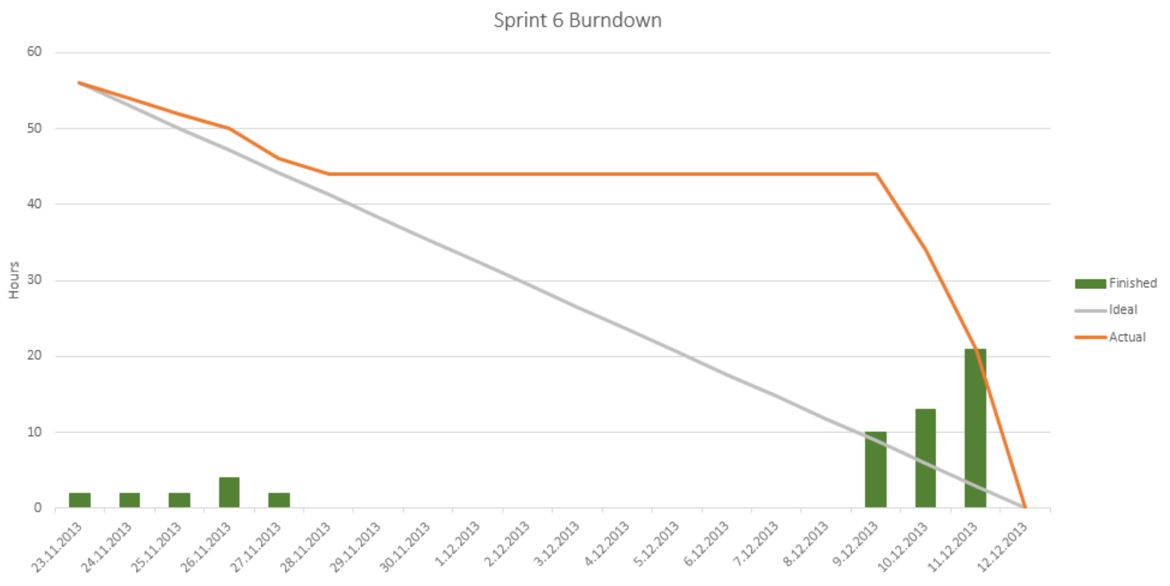
The development team quickly got to work finishing the week view before the feature freeze 27.11. The exam period can clearly be seen on the burndown chart, ranging from 30.11. to 9.12. As soon as the exams were over, the team began preparing for the final handin and fixing minor bugs and usability issues.

4.6.1 Burndown

s-011 (3 points) **[WEEK VIEW]** As a Mobile user I want to be able to see overview at a glance for a week entering so that wrong entries or missing entering entries can be prevented.

s-017 (18 points) **[DEBUG]** As a developer, I want to eliminate bugs specified by the bug report from the product owner so that user experience is better.

s-018 (20 points) **[Hand-In]** As a developer, I want to finish all documentation so that it can be handed in by Thursday 12.12.'13.



Time spent:	212 hours
Stories closed:	s-011, s-017, s-018
Storypoints finished:	41

4.6.2 Product Status

The application, in addition to the features mentioned above, now showed an asterisk by the names of days that had entries not measured in hours in the week view. Some minor issues with the action view and month view had been fixed, and animations when flipping between days and weeks were made to look smooth and natural.

5 Conclusion

5.1 Team Experience

Work on the project went well and the team spirit was great. From early on everyone had a similar vision of the end product, and yet, each had their own unique effect on the outcome. Communication and interaction within the team extended well beyond the scope of the project and resulted in some good male bonding over beer and music.

Given that Unimaze is a very small company, and the developers had limited access to their programmers for most of the project, they did not get the industry connections a lot of other groups get when they are introduced into the workflow of big companies so the methodology was refined mostly by their experience with it over the semester.

5.2 Scrum

When the project started, all team members thought they had a good notion of how the scrum workflow works. This turned out to be wrong. One thing that went crucially wrong with the application of scrum was that deadlines were never set in stone. During sprint planning meetings a lot of story points were taken on without any intention of finishing them all, so the team's story point capacity was never properly taken into account. This resulted in stories being moved between sprints a lot. When the team realized that this was not good for progress, the workflow was reevaluated and adjusted to fit their needs. However, in the 15 weeks, the process was refined based on feedback from the project instructor and the team's experiences using the scrum methodology. A lot was learned about Scrum and the improvements in its use over time were extensive.

5.3 Programming For Windows Phone

Going into this project no team member had experience programming for Windows phone but all had previously coded in C#. Nobody had heard of Telerik, but with the quality of their online support the team knew that this would not pose a problem. Additionally, because of windows phone's low market share there is a definitive lack of information on solving basic problems and getting started with development.

5.4 The Future

Future work on this project involves implement a view to resolve sync conflicts as they happen and adapt the code to work on Windows Tablet 8. For this to happen, the views and view model classes have to be re written and the user interface redesigned to utilize the larger screen space of the tablet.