**Table of contents**

[1. Overview 2](#_Toc450308704)

[2. Situation : JESSE 2](#_Toc450308705)

[3. Task : JESSE 2](#_Toc450308706)

[4. Action : ARJEN 3](#_Toc450308707)

[5. Result : ARJEN 4](#_Toc450308708)

[6. Reflection : NIEK 4](#_Toc450308709)

[7. Transfer : DENNIE 6](#_Toc450308710)

# Overview

<https://meesterschap.wordpress.com/2013/06/09/starrt-methode/>

Good morning everybody, as students of the PXL we were asked to create a platform for the italent course. We will try to give you a brief insight in how we accomplished this task.

* I will first explain how we got this project
* Then, I will provide you with some details about the task we were facing
* Next, Arjen will explain some actions we took to resolve issues and results we’ve booked
* After that, Niek will reflect on how we did the job
* To finish up, Dennie will explain how to Transfer things we’ve learned to potential other projects

# Situation : JESSE

As IT students of PXL, we were asked to create a platform for students and teachers to launch ideas for the italent course. This platform will be used to provide students with ideas and stimulate collaboration between departments.

We accomplished most of the technical workload at home. Meetings with clients and coaches were organized in the PXL buildings and team meetings were held using Skype or face to face at OfficeCenter and PXL buildings.

Since this projects is a quite big one, we’ve decided to manage the workload using various tools like planning, automated building & deployment tools to speed up development work.

# Task : JESSE

So, about the task:

Students, teachers and potentially companies can post their ideas on this online platform but that doesn’t mean they have to get involved. The platform is purely an ideas pitcher. Something like kickstarter, but more transparent and not money-related.

Our task was to create this platform using the latest techniques. Because of this big assignment we had to split our team to get the most of everyone’s experience.

* Arjen did most of the frontend work
* Jesse was dedicated to Hibernate and Spring
* Dennie set up a basic Spring Boot backend and some security
* And Niek tried to resolve technical issues and related tasks.

Our goal was to create a web-based future-proof and secure Single Page Application with open source libraries and new techniques.

We started the creation of the platform by forming a plan. This plan was obviously intended to meet the requirements but we soon realized that the requirements were fuzzier than we first thought. This initial plan is currently still in development, so we decided to go Agile.

The task could get split into these sections:

* Planning : where we have choosen for a ‘SCRUM’ approach
* Analysis : requirement analysis was the most important one here
* Documentation : Not easy to do as a developer, but quite necessary
* Development : The actual creation of the platform

Of course we didn’t strictly follow these steps, one was often mixed with another

I will now give the floor to Arjen

# Action : ARJEN

So, what did we actually do?   
As a first step, we **brainstormed** a lot. We created an initial exploring analysis and realized that everyone had a different idea on how we should move on. So we decided to arrange some more meetings to discuss how we could meet the goals that were requested and at the same time combine our visions.

Next, we started the deeper **analysis** process while some of us were looking into the technical details of the application. We tried to get solutions for **technical issues** we were sure we had to overcome.  
For example: how do we upload videos or pictures? Will we link to them or host them on our website?   
The outcome of these questions were, of course, extremely important for further analysis so we arranged a ‘daily’ scrum meeting using Skype, every day at 8pm sharp.   
Since most of us have a full time job it’s not easy to attend all sessions, so we decided to only attend them if necessary.

When the biggest part of our analysis was done, we started the initial development. Using everyone’s own skills and expertise we managed to get a potentially shippable product in an extremely short matter of time.  
Of course, this product had only about 20pct of all requirements we had to meet.

Today, we are still iterating and resolving requirements and are excited to show you our final product during the final presentation.

# Result : ARJEN

The result is a **nice-looking responsive webapp** designed in AngularJS and Boostrap.  
The backend is created in Java using Hibernate and Spring.

The advantages of this approach is that it is possible to run the application on any device.  
Also, during coding, we could work the ‘model-first’ approach where we weren’t bored with database development. This approach saved us a lot of time since we didn’t have to code very much: the focus was coding the processes, not the technical aspects.

The maintainability of the product is very high. There are just a few references to specific classes and no repeated coding we tried to live up to the **High cohesion, loose coopling** paradigm to make sure the quality of our final product was good.

Another result of the implementation of this paradigm is that we have created lots of testcases. This also provides us with quality in coding and maintainability.

Niek will now reflect on all work we have done.

# Reflection : NIEK

Looking back at our goals, we should definitely do things slightly different in future projects.  
But we are proud of our final product and our team’s collaboration.

* Planning

Since we decided to plan everything up-front, we were able to delegate tasks between our team members. We were not very prepared to changing requirements and additional work, so we struggled along these obstacles by re-planning the entire release.

We learned that it’s not a good idea to plan up-front too much because of the changing requirements and unplanned obstacles we had to overcome before other tasks could get completed.  
After some weeks of planning and analysis we decided to go ‘Agile’. We dropped the entire planning and just planned up to the next client meeting. This approach was way better since our requirements to this short amount of time were very clear to everyone & all team members knew they had to resolve their tasks before the next sprint could get started.

* Meetings

We held lots of online-meetings. We tend to discuss lots of ‘irrelevant’ information during these meetings since we all wanted to go into detail. These should have been more to the point since development-time gets lost during long meetings where too little is decided.

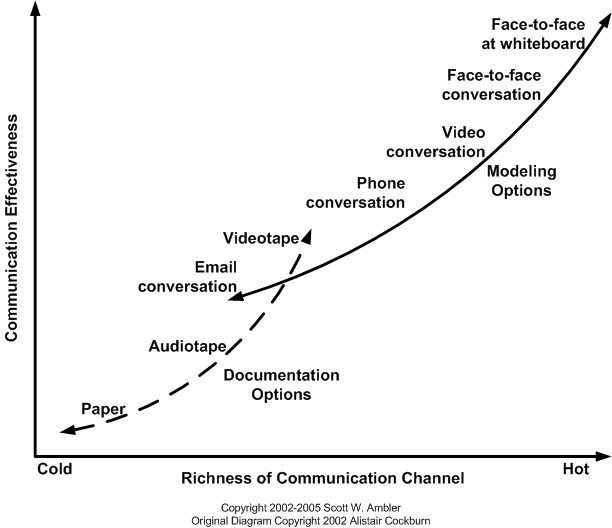
* Deciding the requirements

Requirements are the first step on creating product. It’s impossible to create a product right from the first time. We thought we could, so we created a detailed analysis as documentation of the final product.

Since these requirements changed over time, the ‘final analysis’ got deprecated.   
In future projects, we would only create detailed analysis for requirements that are 90pct sure to implement. Reducing loss of time should be a priority that all team members should take into account.

* Communication

Daily Skype meetings, weekly face-to-face meetings & multiple client meetings were more than enough to get the product we wanted to achieve. Since communication is a key to success for all projects we took this seriously. We not only discussed ongoing problems but also kept a list of features we could implement and improvements on already-developed requirements.



Since face-to-face is the most effective communication, we decided to plan some meetings at OffiCenter. During these meetings, most of the requirements were discussed since they are the foundations of our software. During phone conversations we held every day, we discussed the progress & current issues.

* Tooling

Every developer has his favorite toolset. We wanted to make sure we could use the individual chosen tools. Because, of course, development could go much faster when using tools we know.   
For example, Arjen likes to work with IntelliJ/Webstorm, while Jesse prefers Eclipse

# Transfer : DENNIE

**So, for future projects we would most certainly keep:**

* Daily communication using Skype & weekly face-to-face communication

Communication is a key for collaboration, so it’s necessary that all team members are dedicated to communicate and provide comments on work so far. That’s the only way to accomplish the team’s goals.

* Known & supported open source technology with a broad community

Choosing the ‘right’ technology is never easy. We have learned that the opensource community provided splendid documentation and questions you may have are answered within hours or days.  
This is a big plus in limited-time development.

* Stable build- and deploy processes

The continuous integration software build and deployment process is easy to implement and testing is much easier since everyone is testing on the same environment. We’d like to take this approach to other projects as well.

* Planning tools

Last but not least, a project alwys starts with planning ahead. You can use just an excel sheet or plain document to provide to your team. We have choosen for a complete solution from Rational.  
This tool provided us with a burndown chart, defect tracking, sprint & release planning and so much more. It’s easy to install an a great tool for versioning as well.   
The tool helped up get on track when planning tasks/defects and team members efforts.

We are convinced that a good planning tool can make or break a project’s deadline.

**We would most certainly change:**

* Getting the requirements clear before further actions

New IT projects should go the ‘Agile’ way. But we have learned that it all starts from requirements.  
When the requirements are not clear, there is no point in creating an analysis.

We will definitely take this into consideration since we have lost lots of time in analyzing requirements that became deprecated or had to change for one reason or another

* Don’t plan ahead too much: create time for a retrospective

Planning and agile go hand in hand, but only if you do not plan too much in advance. The planning should be flexible enough to re-locate work but tight enough to keep the team going.

Our planning was way to tight. We didn’t have lots of time to reflect after some time and as a result we had to change some requirements in a later stage. It’s important to reflect after each sprint so changes in requirements are quickly determined.

**Things we could do different, but were good enough:**

* More face-to-face meetings

Face to face communication between team members gets the most results. Just one a week or one each two weeks is the bare minimum to get the project done in time.

Phone calls, if course, also work but are not that effective since you miss the body language of your team members.

* Create less analysis, only when needed

To conclude, analysis is necessy, but no too much. It’s time comsuming and most of the time ends up in a box or on a drive. We’ve learned that it’s better to create analysis when needed for development work. Not the other way around.

I would like to thank you all for your attention