Cool Boyz 2001
Adrian Lindberg, Axel Ljungdahl, Erik Jansson , Sebastian Nilsson, Jonatan Nylund, Jonathan Sundkvist,, Matthias Andersson,

Three things we learned from the Lego exercise

The Lego Scrum exercise was a great experience regarding how it is to work in an agile way. Building a city in Lego within several teams consisting of six to seven people each in three fifteen-minute *Sprints* efficiently highlighted the strengths and weaknesses of the Scrum development process. There were lessons to be taught in every stage of the work process, from learning to set a reasonable working velocity for the group to making sure not to overdeliver on the requested modules and features. Three specific insights we learned from the exercise are described in detail below.

Overdelivering features

One of the striking realizations from the exercise was the innate feeling of wanting to create something that you, as the model builder, thought would be a nice feature. Even though there was no specification in the contract of each model, somewhere along the way new features snuck their way in. This feels relevant to software developers as well. Most of us getting into development do it because we like to create things that we're proud of, and this exercise showed us that while a new feature might make the product better, it's not what the client ordered, and if they end up not wanting it, it's a waste of time and resources for the company.

Communication

The importance of communication became abundantly clear during the exercise. In the first sprint, there was a lack of communication with the client, between different groups and even within the group itself. This resulted in poor work efficiency and a substandard product that didn't live up to the client's expectations.

During the course of the exercise, we learned the importance of having both a detailed, exhaustive initial interview and a continuous stream of communication with the customer. When ordering a piece of software, while a customer will likely be able to describe the required major features, the smaller details may have to be brought up by the developer, both due to simple absence of details as well as lack of experience in areas such as user interface design.

Clear communication within the group itself was also found to be essential to a functioning group dynamic and a smooth workflow. It became clear that without communication and clear division of tasks, the other group members were no longer seen as an asset by the individual group member. In a group project, particularly one in which the group strives to build modular pieces that need to fit together (like in software engineering), it becomes essential to talk to each other. In our project we will strive to have meetings and code reviews so the structure and goings-on of the project will become clear to all members.

Time estimation

A challenge that arose quite early in the process was the difficulty of time valuation. In the initial phase, when one was setting the team's velocity as well as each contract's function points, our group had a tendency of overestimating our ability as well as underestimating the contracts in general. This resulted in a lack of quality as well as hindering the communication within the group, due to time restraints.

A conclusion that can be drawn is that it could be helpful to do a few contracts/features first and draw from that experience, in order to better value the functions points of the other contracts. In the case of the lego exercise, this manifested itself in the shape of us downgrading our velocity in order to better meet the expectations.