Colour IT – The interview

Colour IT is a small company developing and implementing IT systems mostly for private customers. Colour IT currently has no project management system to handle tasks and time for their IT projects. The owner, Indigo Turquoise invited us for a meeting with him.

"Mr. Turquoise, could you tell us about the system you need?"

Mr. Turquoise: "Please call me 'Colour', everyone does", "Yes, I invited you because I want you to develop and implement an IT project management system, we can use at Colour IT."

"Sorry to ask, but why do you ask 1st semester students to make a system you could do better yourself?"

Mr. Colour: "We don´t have time, but the main reason is to provide a real life case for you to practice developing and implementing IT systems – and who knows, you could get a part-time job in our colourful company if you do well. And because you are working in separate groups, we have a chance to pick the best system to use in our company."

"Thank you, it sounds like a great opportunity for us. Do you have any requirements, use cases or maybe a class diagram so that we know what you want?"

Mr. Colour: "We will try to make it as close to any other IT project. Normally, our customers have no programming experience. Instead, a lot of knowledge about the system domain and sometimes but not always a clear idea of their future system. I will try to be that customer for you."

"OK."

Mr. Colour: "I can start talking about our current procedure. An IT pro…"

"Yes please."

Mr. Colour: "…ject in Colour IT… starts with an interview like this one. Here we try to understand the customer's world and get an idea of the system we should develop. After the interview, we formulate a list of requirements in form of user stories. Do you know about user stories?"

"Ehhm, is this requirements?"

Mr. Colour: "Yes, a functional requirement formulated from a user's point of view following a template 'As a […someone], I want to […something] such that […reason]', in other words, describing who want this feature, what is the feature and why is it relevant – who, what, why."

"Aha."

Mr. Colour: "We normally have an extra customer meeting or by mail get a confirmation that this is what the customer wants. In small IT systems, we make use cases modelling for the full system before the customer meeting."

"For a small IT system?"

Mr. Colour: "Yes in that case we almost follow a waterfall approach analysing the full system first. However, we normally use an iterative process and therefore do not make requirements and use cases for the full system in the first iteration. I know that this is on your 2nd semester syllabus."

"OK, use cases for the full system."

Mr. Colour: "Remember your focus, this may be the way you are working but the system you are developing is to be used in our company where we need to manage projects both using a waterfall approach and an iterative approach."

"Right."

Mr. Colour: "No matter in which order we do it, we end up having a document with the list of requirements. For each requirement we have 1) an id, 2) the user story text in the who-what-why template structure – remind me to get back to this, 3) an estimate for how long time we think we need to complete the requirement, 4) a deadline when it should be done, 5) a team member responsible for the requirement, 6) a status which is either Not started, Started, Ended, Approved or Rejected, and later 7) the total hours worked on this requirement and which team members worked on it."

"Ugh, that is a lot."

Mr. Colour: "Every requirement is split up into tasks. Every task has 1) the related requirement, e.g. its id, 2) a task id, 3) the task title or description, e.g. 'use case for requirement 1', 4) an estimate, 5) deadline, 6) responsible team member, 7) a status, and 8) the total hours worked on this task and which team members worked on the task."

"Do you know all the tasks in the beginning?"

Mr. Colour: "Only for small projects do we formulate all tasks in the beginning. If we do not use the waterfall approach, the tasks are formulated in different iterations. No matter what, when a project is done, we have a list of requirements and a long list of tasks – both including the precise time spent."

"What about the estimates?"

Mr. Colour: "We estimate how much time we need to develop, implement, test and document a requirement and we do it fairly good because of our experience. The estimates for all the tasks belonging to one requirement have to sum up to the estimate for the requirement, e.g. if one requirement is estimated to 20 hours, then the sum of the task estimates for this requirement is exactly 20."

"How do you know who is working on which task?"

Mr. Colour: "Every task has a responsible team member, which always is one of the members working on the task. Every day each team member reports to the Scrum master 1) on which task he is working, 2) if the task has ended or not. If the team member did work on multiple tasks, then the same information for each task. Besides that, the team member writes down the hours spent on each task that day."

"Scrum master?"

Mr. Colour: "A project team has one Scrum master, one Product owner and the rest as team members. Scrum master and Product owner are also team members but with special responsibilities. The Scrum master handles the process, documents who did what when, reports when a requirement is done, makes sure all are on track, and so on. The Product owner represents the customer and is the person approving a requirement when it has ended, prioritising requirements and for larger projects adding new requirements, removing requirements or reordering requirements between iterations."

"Should all the team members have the same access to the system?"

Mr. Colour: “We don’t want a login. In principle, they use the system in different ways, but only one at the time.

“There are 3 roles, a Team member, a Scrum master and a Product owner, right?”

Mr. Colour: “Note that the Product owner and the Scrum master are also team members and function as developers like the other team members.”

Mr. Colour: “Actually, it could be nice to have a 4th role, let us call it a Project creator in lack of better words. He should be able to create a Project, assign team members, assign roles for team members and sometimes change their role and add new members to a project team.”

“How to present the data?”

Mr. Colour: “It is up to you to come up with a good design. The important thing is that it should be possible to search for a project to get the information in form of a list of requirements with status, estimated and used time, see or search for information of individual tasks, their status, time, etc.”

“Any other types of searches?”

Mr. Colour: “Yes if you have time, then a search by employee to find his related projects and maybe see his productivity, i.e. time spent related to the estimated time. And to see with whom he worked the most.”

“Should we use a database?”

Mr. Colour: “No, I have heard that you don’t learn about databases until 2nd semester. Therefore, I would like you to use files for persistence. Also, when an IT project is done it would be nice to store the entire project information in a single file which can be accessed when searching for historical data.”

“OK. Anything more?”

Mr. Colour: “Maybe it is time to tell you that some requirements do not follow the who-what-why template. The who-what-why requirements are called functional requirements or user stories. The other types are non-functional requirements and project related requirements. If for instance a customer want their users to change password every three months, then this would be a non-functional requirement. The project related requirements are the needed work to be done, but not like a feature to be implemented. In our case it is to produce a user guide, the updating of the documents, the system report and such. We need to spend time doing it and thus, have to include it as requirements and later as tasks.”

“And a graphical user interface?”

Mr. Colour: “Yes of course. Also, I want you to implement it in Java with the GUI in JavaFX in order for us to easily modify it.”

Mr. Colour: “One more thing: When all tasks for one requirement have ended, then the requirement should automatically be in ‘Ended’ state. The Product owner’s job is to test this requirement, normally a feature and if accepted, set its status to ‘Approved’, otherwise set it to ‘Rejected’. It could be nice not to keep all the tasks related to an Approved requirement in the same view as ongoing tasks.”

“Implemented in Java. Not a website?”

Mr. Colour: “We want a stand-alone application for internal use at Colour IT. In addition, a web access to our customers. On this website, our customers may find information about their projects like description and all the requirements with a status. No access to tasks, actual time spent and no information about team members”.

"OK, we need a little more information about this website, then."

Mr. Colour: "The website needs to look nice and professional - you know, nice colours, organised layout, and not too much text, but still enough so that they get an idea about Colour IT, the project team, and they should be able to see the status of their own project i.e. a short description of the project and then a table displaying all the requirements for the project, if the requirement is functional or non-functional, the deadline, and the status of the requirement. Oh, and use nice pictures and stuff to make it look good…maybe also music or something – nah, I'll leave that up to you."

"Right, so how should we show the status of multiple projects on the website?"

Mr. Colour: "That's up to you. We don't need the customers to log in or anything, the information that is displayed about the projects is not a secret. The important thing is that it is not all displayed in a big mess on one page on the website."

"OK, we'll figure it out. Could you sum up?"

Mr. Colour: "No, this is your job."

"Uuhm, let me try. A system to manage IT projects. A Project creator to create a project with team members, one being a Scrum master and one being a Product owner. A project has a list of requirements, some are functional requirements formulated as user stories with the 3 parts who, what and why, some are non-functional requirements, and some are project requirements. All of them in a list. I did not get if this list was formulated in the beginning or not."

Mr. Colour: "You are very good at summing up. About the list of requirements, for the system it is not important when they are formulated, just that we can add new ones at any time."

"And remove – and reorder, as far as I understood."

Mr. Colour: "Very correct. Just continue your summing up."

"Hmm ok. There is a list of tasks each with a link to a specific requirement, and a responsible person."

Mr. Colour: "A task has an id, a task text, a relation to a requirement, an estimate, a responsible person, a status. Team members are able to register their time spent on a task."

"And then different search options…"

Mr. Colour: "Exactly. Project, Team, Requirements, Tasks, Time registration and different search options in an application, and then a website for our customers. Simplicity is beautiful."

"Then there is the website that needs to have information about your company, team members, ongoing projects, and the status of the projects."

Mr. Colour: "That's right. Don't forget to make it look good."

"Hmm ok, I think we got it. We will start to formulate requirements and then get back to you."

Mr. Colour: "Could we say that you make the full analysis before we meet again? Hereafter I will come to your classroom to see if we are on the same page. If you have any questions, feel free to formulate these, collect them in your class, and send them. If urgent, I could stop by to clear up any misunderstandings."

"Perfectly ok, we'll do that. Thank you Mr. Turgo... Mr. Colour."