Solution

In general, a good user story should be written covering all the layers of the product. Should be an end to end user story. The solution for this problem is the coach the Product Owner and the team about this. For me works quite well when I explain to developers that do not matter if we deliver 200 back end user stories when in reality the customer cannot use them without the front end part. Below there is a picture from Ángel Medinilla that shows in a great way how stories should be built.

Solution

I believe the solution for this case is to find a strong and experienced Scrum Master. An experienced Scrum Master understands quite well both roles (Scrum Master and Product Owner) and can coach the Product Owner to allow the team to take responsibility for their tasks. One of the roles of Scrum Masters is to protect the team from intrusions, having a strong and experience strong master within the team will allow the team to remain isolated and protected from unwanted interferences, and at the same time will allow Product Owners to be coached on right behaviours.

Solution

Companies must understand that is not possible to have this scenario and they must provide an environment where each team has one single product owner.

Solution

My experience tells me this situation is quite difficult to resolve because most of the time this problem comes from managers above the Product Owner. In this case, the Scrum Master or the Agile Coach has a fundamental role to play. He is responsible for talking with the management (or with anyone else who is behaving in this way) and explaining how this behaviour has an impact on the company and the quality of the product. I have witnessed several situations where the Agile Coach was able to explain to Senior Management what was happening and was able to help management to change their behaviour.

Solution

To point out to the solution, I am using a blog post from Roman Pichler. "10 Tips to write good user stories".

Solution

The solution here is quite easy. The Product Owner should build a product burn down/burn up exactly as the Scrum Master does, but instead of being focused on the sprint, the Product Owner focuses on the release. A possible example can be observed below.

Solution

The job of a Product Owner is not simple; it´s a tremendous and complex job, sometimes the product is so complex that one single person might not be able to manage everything. I've seen some companies that brought business analysts to the team; these people took the responsibility of talking with clients to see what the market needs, leaving the responsibility of the technical product to the actual Product Owner. I saw this technique in several companies, and it worked quite nicely.

Solution

A Product Owner should not have more than two teams. Of course, this is a general rule, and it might be possible to have more, but if he/she has more than that, it can be difficult to give full support to the teams.

Solution

People approach me telling me that sometimes we do not choose the release dates and we have deadlines to be met. Therefore, there is no way to behave in a different manner in this kind of setup; the customers just want to have the product. I worked for a mobile manufacturing company, so I understand what dates set by the market means. Every year we had Christmas sales where the phones needed to be out; there was no other solution, so how could we manage this? The answer is simple: follow basic agile principles. The PO must educate the client that after every sprint the team has got something to show (with this the PO gets his demos). During the sprint demo/review, the client has the opportunity to provide feedback on what was delivered. And more importantly, has the opportunity to select the most important things that are on the backlog, enabling the team to always deliver the most value to the client. At the same time (and this is one of the most frequently forgotten practices), the PO checks the overall release status using Releases Burn Downs. This practice allows the PO to know exactly where the team will land, and discuss with the client about what can be dropped in the event the team is not able to deliver on time. If the client is not open to all these suggestions, the company should start to reconsider if the client is someone with whom the company would like to work in the future. The problems that will arise from future interactions will be so complex that the relationship will not benefit any of the parties.

Solution

This problem is quite serious and, in my opinion, is something that is part of the company´s culture (not caring about quality). The method I found most useful to reduce this problem is the presence of a Definition of Done Checklist. A Definition of Done Checklist is a simple list of tasks defined by the team that represents all the activities that must be done before a story goes into production. Without a Definition of Done Checklist, it´s easier for the Product Owner to push the team to cut corners; if this list is there, on the other hand, it´s possible for developers to show that a story is not done until all tasks that are part of the Definition Of Done are completed. My experience tells me this helps to reduce the problem. If you are interested to know how can you Mature Your Definition Of Done check this blog.

Solution

Each product should have only one Product Owner. My experience tells me if we have too many people responsible for a product we have none. Never assign more than one Product Owner to a product. If you have several products that are connected with each other, you might need to have a program. In that case, you should get a Program Manager or a Chief Product Owner as some companies call it, but never more than one person responsible for a product.

Solution

Each case is different, but as a general solution, I believe that we could try to use the same approach as in the first example. Having a business analyst within the team could help the Product Owner to get more time to help the team to deliver the right product.

Solution

In this case, I truly believe there is not much that I can suggest; management must understand that a Product Owner is extremely important for the company and only people who understand customers and the market can do a good job. Assigning people to these positions who do not understand the market or the customers is a recipe for disaster. I can only suggest that management assigns people based on quality and not based on political interest.

Solution

In both situations the Scrum Master's job is quite important; he must protect the team. He must explain to the Product Owner that the team owns their estimates. They are the ones deciding how much it takes to deliver that amount of work and there is not much he can do. The Scrum Master must explain to the Product Owner that the team's members are doing their best and he needs to trust them.

Solution

The solution here is quite simple, but not easy to implement at all. In this case, it is extremely important to get an experienced and strong Scrum Master. Great Scrum Masters can do their jobs in a well, but even more importantly, they can coach others in their jobs. Great Scrum Masters are great coaches as well, so having a strong and competent Scrum Master in this situation is fundamental to improving these issues.

Solution

Here it is quite important for the Product Owner to understand what the most important things are that must be delivered to the customers. Product Owners must understand the team cannot deliver everything at the same time, so they really must understand what is the top priority that should be achieved during the next sprint. When we applied this rule in some of my teams, teams got extremely focused, and they started to deliver much more stories per sprint. But more important than that, they started to deliver fully implemented features that could be shipped to the customer. Before, the teams were delivering a lot of things, but none of their features was ready, blocking the release to the customers.

Solution

The solution here is not easy to achieve at all. I truly believe this has more to do with mindset than with any agile process. I would suggest taking this Product Owner to a good agile training session to make sure that he understands what the roles are in Agile software development. To complement this, I would take him to a good training session about intrinsic motivators to help him understand what drives people to do a great job. If he would not like to attend any of these training sessions, maybe give him the book Drive by Daniel Pink. Hopefully, he will then understand that punishment or pressure will never motivate people. Hopefully, he will then understand that punishment or pressure will never motivate people.

Solution

I believe the Scrum Master and the team itself have an important role here. Together they must explain to the Product Owner that the aforementioned is his job. They must explain that without a proper backlog the team cannot do their jobs. The stories should be discussed between the Product Owner and the team. The Product Owner is the person ultimately responsible for keeping the product backlog in good shape. The Scrum Master and the team must show to the Product Owner that it is extremely difficult to do any work if the product backlog is not maintained. A good story must be clear, optimally written in the INVEST (Independent, Negotiable, Valuable, Estimable, Scalable and Testable) format with proper Acceptance Criteria.

Solution

The Scrum Guide says: "The Scrum Team consists of a Product Owner, the Development Team, and a Scrum Master." Just because the Scrum Guide says that, of course, there is no guarantee the Product Owner will consider himself part of the team—But, I think this could be a good start nevertheless. The Scrum Master or the team could explain to him that, according to the manual, the Product Owner is officially part of the team. On top of this argument, the team and Scrum Master could explain to the Product Owner they are not able to do a fantastic job unless the Product Owner starts to become more connected to them. This could be a nice topic for one of the Retrospectives.

Solution

The Scrum Master and the team must explain to the Product Owner that the design of architecture is not his responsibility. I always like to give this explanation: The Scrum Master and the team are responsible for "How": How the product is made. The Product Owner is responsible for the "What" part, meaning that the Product Owner should be the one saying what is supposed to be built.