Scrum

And

How to be Truthful About it

by pmbasics101.com

Scrum. Easy in theory, difficult to master. Will it work for your project? Does it work at all? How to introduce Scrum in an organisation. How to make customers follow the guidelines?

I can bet you have the same questions.

You need to understand how Scrum is intended to work. You must know the common problems that arise in Scrum teams. After that, you need to ensure you will be able to overcome fears and natural patterns of behaviour of the clients.

How will you do that?

I have a story for you. It will help you to get inside the mind of a client.

If you are not familiar with the basics of Scrum methodology, I suggest you read Scrum Guides first

It was a usual project with an unusual client.

John came to our company looking for a new vendor for a software project. He travelled with only intent to tour the office and meet people working there. He brought no project details yet.

I was assigned as a project manager for his potential account.

John is an open-minded and innovative person. He focuses on productivity and new technologies. He always on a look out for effective means to facilitate the work.

John had years of experience as a project manager. However, he never participated in software development. He was ready to try out new approaches and face the challenge.

All in all, I can surely say that he is a customer out of a dream.

But there is a catch.

He is a customer to full extent. In other words, he is only a product owner, not a sponsor. Therefore, John has a vision of a product. But he uses the money of other people.

That is important to keep in mind.

After the initial tour, we had a meeting where besides other topics we discussed Scrum.

John: When Does Scrum Work the Best?

To tell you the truth, it is always a sort of a problem to answer that question.

Why?

Marketing description of Scrum always comes first. You feel like you need to warn the customer about possible difficulties and pitfalls. Nevertheless, you don't want to scare him off with too many details.

Here is what came out.

Small Team is a Must

Scrum works best when the team is four to ten persons. When there is less than four, it will often be constrained. If it is larger than ten, it is harder to manage and coordinate it.

However, it is only the surface, a simplified explanation.

What's the real story?

I had to answer this question a bit later. Read on.

You Can't Win Without Engaged Product Owner

What's next? It is you, John. We need you to engage into the development process. You need to provide requirements and priorities.

And you must believe us.

You see, you pay for the team. The team delivers value. However, there is no way we can ensure that you have enough money to create a valuable product. And it is hard to predict if you will get a return on investment.

That is something Scrum omits. But no worries we will sort this out for you. Well, actually we will plan out the project at a high level the old way.

It is your responsibility to maximise the benefits for the team. Also, you must define what User Stories will create the most value for your product.

However, you can not influence decisions of the development team. You need to believe that they are doing their best.

Choose Vendor with Flat Organisation and Little Politics

You see John, there is a myth about the Scrum Team. It consists of you as a Product Owner. Me, as a Scrum Master. And our development team.

The problem is obvious. You have a boss and sponsors to report to. They have their own interest in your future product.

I also have a boss. In addition to that, we have several supporting departments. There are other projects around. Moreover, some of them share the same sponsors.

You and I must keep them all at bay.

Somehow…

You must ensure that sponsors and your clients do not conflict. So that we don't receive conflicting requirements from you. And they will push. They will want more value for each dollar spent.

From my side, I have to ensure that the whole world around our project understands agile values. I must shield the team from their influence and distractions.

However, at the same time, I must behave well. You see, our team doesn't work in a vacuum. We are fully dependent on their support.

Both Vendor and Customer Should be Agile

I can see from your face, John, that you already envision problems when two worlds collide.

In Scrum, there is only value. The team produces increments of the product. No tools to control the overall project progress. No metrics to report.

You only have the price of the team and product in its current state.

We will have to create a way to keep our reporting requirements in alignment. Because at some moment we will feel pressure from sponsors.

Constant Shipping to Market

Continuous delivery might be a solution. Releasing almost each increment to the users is the intended approach. It increases perceived value. You can get early feedback from users.

New metrics will appear. You can measure satisfaction, you can collect analytics.

But what's the catch?

It is hard. It is not really efficient by the standards of project management. You need to accept smaller increments.

It is a trade off.

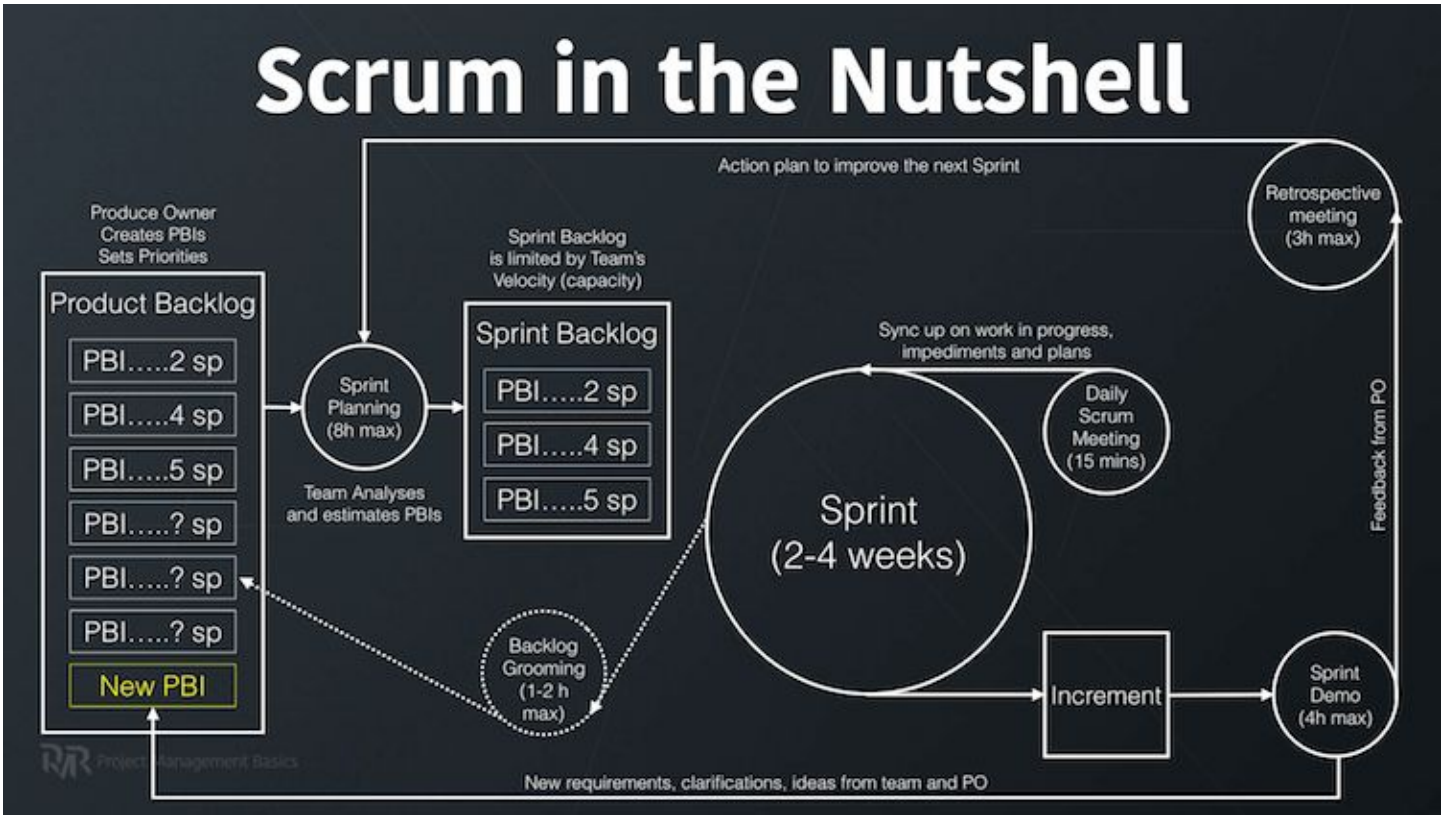
John: So, How Does it Work?

It is simple.

You are a Product Owner. Your primary tool is a Product Backlog. You fill it in with Product Backlog Items (PBI). In our case, it will be User Stories.

You are responsible for keeping it the backlog full with clearly defined PBIs. Also, they should always be prioritised. You remember you need to ensure that the team is working on most valuable items, don't you?

We will work in cycles called Sprints. Usually, they are two to three weeks.



Scrum Framework.

Here is a crucial deal:

The development team has a velocity. It is a number of Story Points that shows the amount of value they can generate during one Sprint. You can not influence or negotiate on the velocity.

What happens next?

We will have a Sprint Planning meeting. The development team will take the first User Story from the Product backlog item. We will analyse and discuss it. The team members will ask questions to clarify what needs to be done.

Right on the meeting, we will make an estimate. We will not go into full details. We will just evaluate how difficult it is in relation to other User Stories. Team will state it in Story points.

Once everything is clear, we will move the User story into Sprint Backlog.

After that, the team will take the next User Story in the Product Backlog.

We will repeat estimation process until we reach our Velocity.

After that, we will continue planning. We will decompose User Stories into the task. Clean up the Sprint Backlog and prepare the whiteboard. As well as other small arrangements.

Then, the Sprint will start. The team will be working on the User Stories in the Sprint Backlog. There will be Daily Scrum meetings where we will sync up and clarify all the questions.

At the end of the Sprint, there will be a Sprint Review. We will demonstrate you the increment we created. You will have an opportunity to give feedback and update future backlog items.

After that, we will have a meeting to discuss how we can improve our work and generate more value in the next Sprint.

And the cycle repeats.

It is important to note that all the events in Scrum are time-boxed. It ensures that the team uses an appropriate amount of time for project management actually.

Dmitriy: So, What do You Think about Scrum, John?

John was delighted. It seemed like it charmed him.

Lightweight Project Management Approach

So, there is only one document I need to maintain and one meeting a day I have to attend to. It looks really simple. Though, I understand that I'll have to add extra efforts behind the scene.

Scrum is Transparent For a Client

Yeah, with such approach I should clearly see what I pay for. In just three weeks, you say? Planning together is great. Demo against the plan in several weeks looks fantastic.

Moreover, it less risky approach. I can terminate the contract at the end of a Sprint in case things get ugly. Don't you afraid of that?

Sure as hell we do. But I just smiled.

John was jubilant. As I found out later, he was preparing for another everlasting project with an unclear outcome. Well, he is a pro in managing that kind of work.

But what's the real story?

What did Customer misunderstand?

- There is a reason to call an iteration "Sprint". There is no time to pause. You can not wait for approval. There is no way to "think it over". Moreover, it is Sprint after Sprint, then Sprint again.
- Is it difficult to make the right decisions every day during 15-minutes long meetings? Once you start changing your mind too often, the team gets demotivated.
- How hard is it to live and work in the rhythm of Scrum cycles? You have to be ready to participate in Demo and Planning meetings. You need to be accessible to the team. It is easy. However, only when you don't have to do anything else.
- What? We can not just add this super important task to the sprint. Why?
- Scrum never performs as advertised just out of the box. Performance metrics are also "agile".

That is something John had yet to understand.

Later we went to lunch. In the end, we were sitting and drinking coffee. It seems that John tried to catch me off guard. Tested my certainty in the approach.

The series of the "Why" questions that followed scratched quite a bit of problems.

John: Why do Scrum Development Teams Work?

Can we have a large team? Is it possible to manage communications overhead?

What's the catch?

Why does Scrum Team have to be that small?

These are the questions that worried John before we could start the project. Definitely, he was planning to ramp up the team to speed up development.

What is the usual story?

When there are more than ten persons in it, the communications and self-organisation suffer.

Really?

You are a Scrum Master. Your primary responsibility is to facilitate the work of the team. Can't you manage it?

It is a trap. Business doesn't understand that.

Therefore, my answer is different. It was well prepared for a person just like John.

Resources Levelling Disaster

Scrum Team is a whole, John. You can not treat team members separately. Otherwise, the whole system breaks apart.

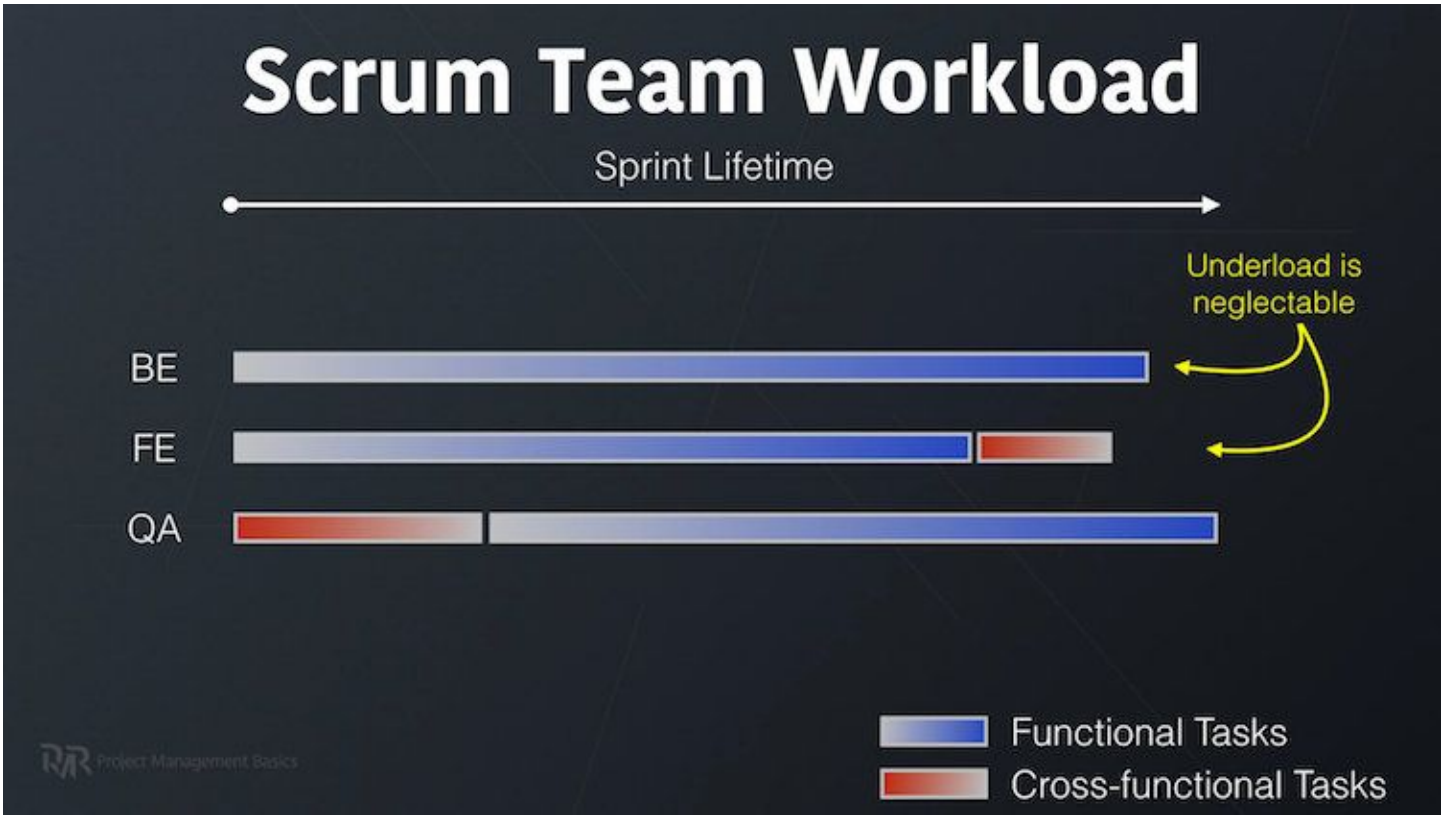
You see, everything in Scrum is aimed at simplifying the project management. Short iterations, relative complexity estimates, cross-functional team, and small increments are here to avoid micromanagement.

If you break one of the fundamental and prescribed pieces of Scrum framework, you will face the consequences.

The foremost is:

Larger Costs of Underloading the Team

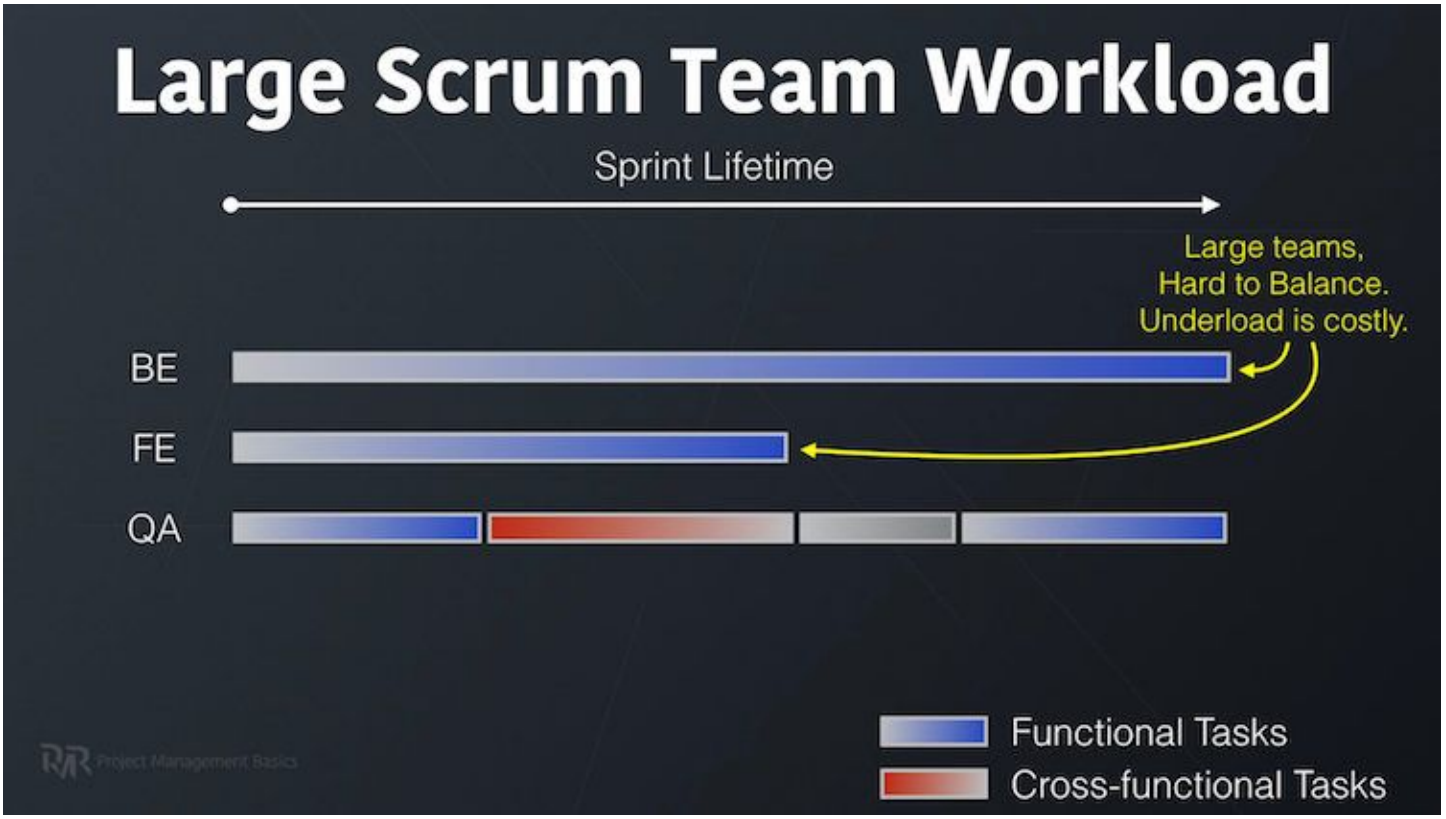
When you have a small cross-functional team, it is easy to control their workload. Even if one or two persons will be underloaded, there is a way to add some work from testing, for example. Also, there are always some polishing left behind.



You should never manage Scrum Team on a level of separate expertise. But you need to understand the dynamics.

In any case, John, your costs of downtime are negligently low.

However, if you try to use a large team in Scrum sooner or later, you will face the resource levelling disaster.



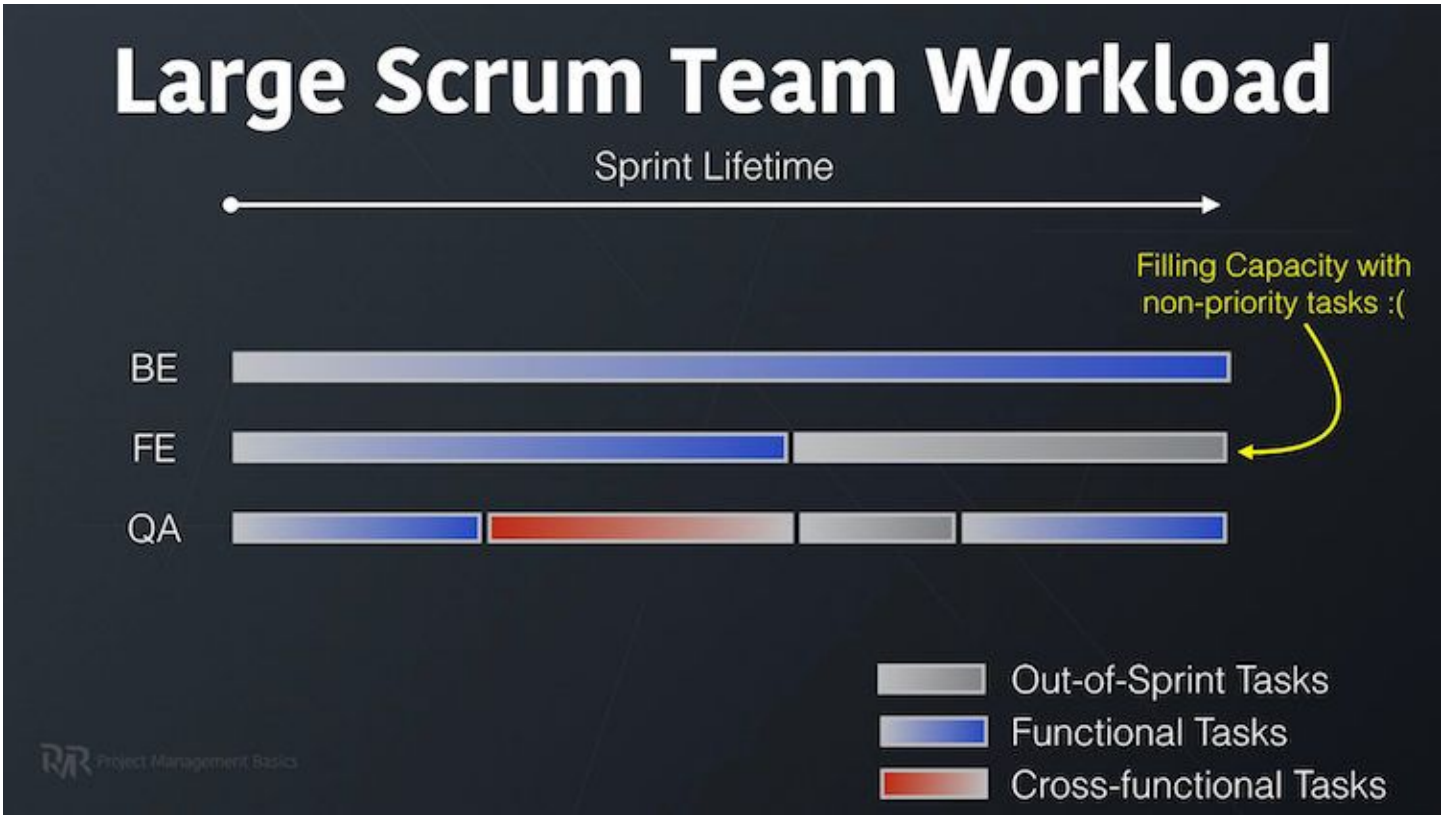
Sooner or later you will face unbalanced workload. Underloaded resources will look very costly.

What will you do in that case?

You will make an obvious mistake:

Breakdown of Increment Consistency

You will want to take some extra work. If you are lucky, you will find some low-priority User Stories that can be done solely by the underloaded expertise.



Someone will be doing work that does not pursue the Sprint goal.

But it is a self-deception. Everything should be a decent quality in Scrum. You can not make those guys to do the coding and testing equally well.

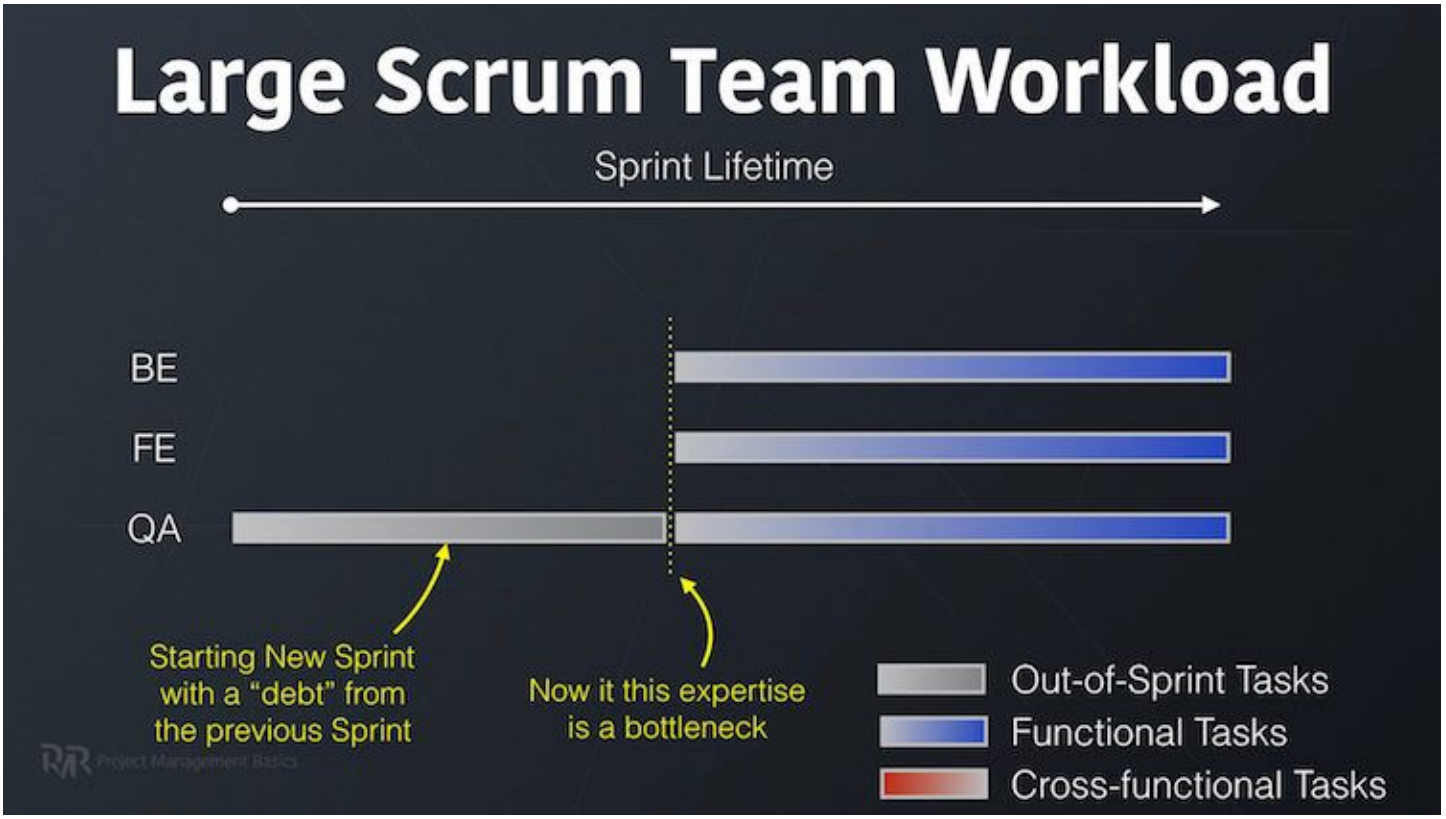
It is a risk to put that work into increment without proper testing. But wait, we don't have the capacity to test that.

So, what can we do? One day you will say: "OK, guys let's just put this part untested. We do not plan to deliver this increment to market anyway."

We can live with that. At least for a short period of time.

Is there a way out?

A solution is on the surface. Postpone the testing of that piece until the next iteration. So, now are getting into the debts. In this case, we have a testing debt for a low priority User Story.



If you have an option to keep the "non-priority" work out of your main product it will not hurt too much. Otherwise, you will have to pay the debt.

Does it make this User Story more important now? No.

In the long run, the debts will accumulate. There are two common aftermaths:

- Stabilisation before shipping the product. So, we will have to pay back the debt. It can take one or several extra sprints. Before that, your product is not shippable, even potentially.
- Further disbalance of the workload. While testing, fixing and preparing the product for the market, part of your team will be idle. So, either you accept the loss of their precious time. Or, you find some "simple" work to do. And you are in a vicious circle. Welcome to the world of Critical Path in Scrum.

Little Amount of Risk

We talked about risks, John. A Larger team can deliver significant increment. Likewise, a larger team can fail to deliver that chunk of work at all. Messing with shippability of the product for long.

John took a few minutes to think it over. While sipping his team, he was nurturing the project work without explicitly defined dependencies. How efficient self-organisation can be? There is one missing link left.

Which one is it?

John, it’s time to discuss Story Points.

John: Why Story points?

People are better in comparing things rather than estimating. You see, John, we can hardly calculate how many inches in that TV. Nevertheless, we can easily say which one is bigger.

And if we know the dimensions of at least one of them, you can approximate the dimensions of the rest.

Is that all?

No.

In Scrum we don’t separate different expertise, we treat the team as a whole. Everyone is responsible for each User Story.

All the tasks are the team responsibility. Not personal.

In theory.

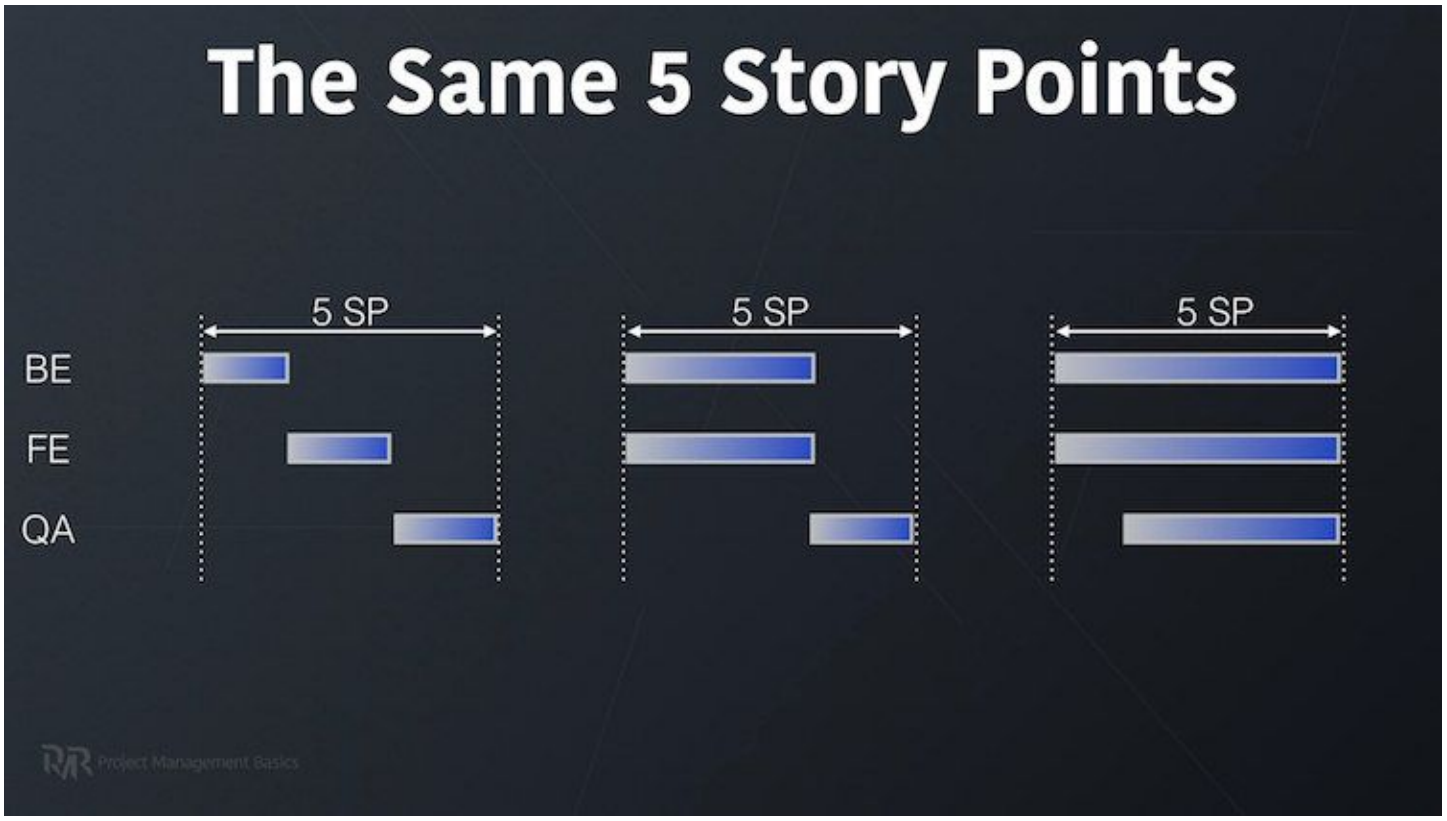
What is it in for you?

Behind the Scenes of Story Points

Again, resource levelling.

You don’t need to deal with the workload. The team has enough space to self-organise. There is enough buffer to accommodate dependencies and sequence of different tasks.

You see, 5 Story Points may look like anything of these:



Behind the scene of the Story Points. Don’t try to optimise it too much…

Can we optimise it?

Yes, but be ready to face **resource levelling disaster** we discussed earlier.

Complexity is not Linear

One important aspect, John.

Complexity is not linear.

As don’t go into the details of estimating and dependency analysis for each User Story. You don’t really do risk analysis or anything, there is a particular grade of complexity.

Usually it follows modified Fibonacci sequence: 0.5, 1, 2, 3, 5, 8, 13, 20, 40, ect. story points

Sometimes we may use other categories, but all of them point to one problem.

Poorly defined requirements impose a significant risk for Sprint success.

Less we know about User Story, fewer chances it will get into a Sprint. It forces us to decompose and clarify requirements.

John came from the world of enterprise where internal politics is a part of the work. Answering the following question was tough.

How can you persuade a person who knows the true nature of a human being?

John: Scrum Master – Authority That Changes Organisation. How?

It is never easy.

The best case is when an organisation is just forming up and want to be Agile.

It is harder when a company intends to transform into agile one. The best chances are when it is small and has a flat hierarchy. It means that a Scrum Master will be one or two steps from executives and technical managers.

It is almost impossible to do in old and formed corporations with a strong hierarchy. Ranks also hurt.

In any case, a Scrum Master should have authority throughout the whole chain of command.

But that is not all.

The organisation should be willing to transform as well. There will always be opposition. Somehow it should find it's new place in a flat agile world. Otherwise, you have to get rid of the opposition.

But what can a Scrum Master do?

You can start from your project, your team and your Product Owner. Once you are good on that, switch to transforming people you depend upon. Then extend your reach.

It is a weak spot of Scrum, John.

You see, it is not a matter of doing Scrum well. It is all about influence, negotiation skills, and authority.

Several Sprints Later…

It was a party to celebrate the successful release of the product. It was a hard party after the hard work. Close to the end me and John when outside for some fresh air.

And we had a bit of honest talk.

Dmitriy: John, so what are your feelings so far?

It seems that John already thought through the answer long ago.

It is not something really new. We used the similar approaches though did not call it Scrum.

It is the fundamental rule of any complex work. That is why "projects" appeared as they are today.

We break an epic endeavour into projects. We break a project into phases. Phases are decomposed into deliverables or increments if you like.

Scrum simplified the world around it. They say everything else doesn't create value for the customer. So, there is no need to do it.

But here is a problem…

Dmitriy: The World is Not Agile Yet, isn't it?

Yeah, Dmitriy, the world is not agile by its nature.

There is nothing in Agile that resonates with core human driving powers.

John was talking about Core Human Drives to acquire, bond, learn, defend and feel

A Scrum team may desire to learn and feel the excitement of being busy. But these are not lasting drives.

On the other hand, the corporate world is full of drives. Promotion, politics, departments, constant fight for superiority, endless strive to achieve goals.

All of that intensified with bonuses and recognition. At least in theory.

Dmitriy: But you see that Scrum works, don't you?

Yes, but…

The efficiency of Scrum is directly related to the motivation of the team.

It is scary.

One demotivated slacker puts the whole project at risk.

There are not many options to mitigate that. Particularly in terms planned releases.

Everything is so tightly packed with little margin for unexpected. Or failures.

Like the one with new expertise that the team did not have. Crucial User Story was on the list for half a year. Yet, only during the Sprint, it appeared that we can not do it.

A month and a half of recruitment and still it is not done.

Could we foresee it?

Only if I was analysing the backlog from a technical perspective…

Dmitriy: What Kind of Team Could Have Managed that?

A team of engaged and motivated people with little life outside the work.

Just kidding.

Do you?

You see, we just formed the team and tried to implement everything applying available expertise.

We never really thought of other options.

Dmitriy: What was the Most Difficult Moment for You?

It was during the fourth sprint. Several stakeholders from my side demanded to include several User Stories asap.

They wanted me to cancel the Sprint three times.

The mess was caused by cross department planning.

One required features in the product for a marketing campaign. Others were to show it at the conference. The third one… well, it was her problem.

It was painfully difficult to prove that cancelling the Sprint is not a good idea. And not very convincing from my side. They were ready to double the team to meet deadlines.

While being agile, I find it hard to plan anything with third parties.

Dmitriy: Do You See a Solution?

It is not in the spirit of Agile.

I hardly believe in a project life cycle with is iterations only.

We can make a sprint to plan the project. Create a high-level estimate, a roadmap, plan some sprint ahead.

But it would make sense if we had several Scrum teams that we need to sync up.

Otherwise, why bother with Scrum.

Dmitriy: So You Learned Some Hard Truth About Scrum, don't you?

Scrum doesn't make development cheaper or faster or more predictable.

It does make the process straightforward and transparent.

But the agility of the project…

I doubt it.

I can imagine one project out of hundred, where requirements actually change within two or three weeks.

But let's face it.

Two months of work is like three or four Sprints. Can we define requirements and scope for four Sprints ahead?

Easily.

Even if we need to adjust the requirements in the process, do you often see an 180-degree change? In any case, it can happen is Scrum as well and you will need to cancel the sprint and plan again.

However!

For those two months, I can make accurate estimates. I can level the workload efficiently. I can track risks.

And the best part?

I can dedicate a lot of time to define requirements for the next two months of work in parallel.

I did not push that part further. He was completely right. Only a small number of projects can benefit from Scrum to full extent. And most of them are related to software development. In all other cases, it will just work. With no extra advantages.

Conclusion

If you ask me, I would never start a project with Sprint Planning. Subjectively, there are about 2% of projects where I would make a decision to use Scrum framework only.

In all the rest cases, I will start with a proper analysis of a project life cycle and project planning. I mean planning project as a whole. Even if at some point I will see that Scrum will be a suitable methodology to use, it will never exclude a possibility of using any other appropriate tools and techniques.

It is simple as that.

Plan before you act.

P.S. Did we stop using Scrum with John? No. He was either hoping that it will fly one day or was reluctant to break the working – somehow – process.

Want more? Check the next page!

Become PM Basics Member (free of charge)

Join Now

When you subscribe to PM Basics, you'll get instant access to a set of proven project management content:

- **Mini-Course:** How to Make Your Risk Management Efficient and Stakeholders Happy in 5 Days
- **Mini-Course:** How to Manage Project Scope in a Structured Way
- **Mini-Course:** How to Become a Software Project Manager
- **Book**: How to Become a Leader on a Project
- **Book:** Complete Guide to the Basics of Project Risk Management
- **Book:** Project Charter and Its Benefits You Need to Know About
- **Book:** 3 Motivational Theories Each Project Manager Must Know
- **Book:** How to Manage Stakeholders Engagement in a Strategic Way
- **Book:** How to Create Realistic Project Schedule
- **Book:** Ultimate Guide on How to Create a Robust Work Breakdown Structure