# The Convergence of Scrum and DevOps

Agile emphasizes team interactions, culture, and values, while DevOps emphasizes delivery pipelines and flow. But, Agile is also concerned with automation, and DevOps is also concerned with communication and culture.

Bringing Together Scrum and DevOps

Imagine a world where cross-functional, businessoriented teams deliver working software continuously and where work flows seamlessly between all the right stakeholders in real time. Add to that where value is clearly understood, measured and reported on. This vision is the reality for many smaller startups that have blended business and technology with customer and market to work in a holistic way of delivering customer value.

Imagine a world where cross-functional, businessoriented teams deliver working software continuously and where work flows seamlessly between all the right stakeholders in real time. Add to that where value is clearly understood, measured and reported on. This vision is the reality for many smaller startups that have blended business and technology with customer and market to work in a holistic way of delivering customer value. Scrum Teams are accountable for delivering software and operations teams are responsible for putting the environment in place that enables them to do it in a secure, safe, high quality way.

CrossFunctional Teams With All the Right Skills Deliver “Done” Software, Faster

Cross-functional teams need to have the right skills to get the work done

This doesn’t mean that an individual can only be on a single team. There are many instances where someone with operations expertise can sit on multiple teams (acting almost like a “service” to the team), bringing their expertise and knowledge to the team(s), enabling better delivery while helping to educate other team members so that they can broaden their skillsets.

== Nota Pessoal abaixo, do próximo destaque: Convidar o Noronha, Heron, Thai para a review da Squad

Develop all the skills they need to deliver value. When teams don’t have all the skills they need, they will need support from specialists.

Develop all the skills they need to deliver value. When teams don’t have all the skills they need, they will need support from specialists. Over time, they should strive to learn from these specialists so that they increase their independence.

== Nota Pessoal abaixo, do próximo destaque: Esse exemplo da necessidade de aprender com o time de segurança em específico é muito bom, a questão é , o que podemos aprender com otras áreas para automatizar esse processo

== Nota Pessoal abaixo, do próximo destaque: Esse exemplo da necessidade de aprender com o time de segurança em específico é muito bom, a questão é , o que podemos aprender com otras áreas para automatizar esse processo ?

DevOps and Scrum Succeed When Impediments Are Removed

Queues break agility because they remove the ability of the team to be in control of the situation. When support organizations are connected to Agile teams, via queues, it is very likely that the team as a whole will be less effective as they wait for their item in the queue to reach the top.

Helping grow team capabilities means:

Empowering teams to “do” what needs to be done to release the product.

Reducing external dependencies through self-service automation.

Supporting teams with shared technical services, available without queuing.

By concentrating on supporting Scrum Teams with experts who can coach or mentor teams on a particular technology or approach, you both remove a bottleneck and increase the skill of the team, making them more flexible and Agile.

Simplifying and modularizing legacy applications. Look to refactor the legacy systems and supporting infrastructure such as environments and tests to better support Agility. That means focus on reducing the overhead of testing, provider better support for decoupling, and spread legacy knowledge throughout the teams rather than

Simplifying and modularizing legacy applications. Look to refactor the legacy systems and supporting infrastructure such as environments and tests to better support Agility. That means focus on reducing the overhead of testing, provider better support for decoupling, and spread legacy knowledge throughout the teams rather than leaving it in one specialized area.

Some good tips for automation include:

Simple processes are best. Remove complexity and over-engineered processes.

• Start from the back and the front and work toward the middle. Unit testing and release deployment areas are perhaps the best places to start providing numerous opportunities to create scripts.

Automation should be part of the natural work of the Scrum Teams. It is easy to think that there should be a separate team building automation and creating standardized infrastructure. Experience shows that these centralized tools teams tend to build for everyone and serve no one. Focus on each Scrum Team building the right stuff for their own

Automation should be part of the natural work of the Scrum Teams. It is easy to think that there should be a separate team building automation and creating standardized infrastructure. Experience shows that these centralized tools teams tend to build for everyone and serve no one. Focus on each Scrum Team building the right stuff for their own use.

• Reward people for creating value, not merely keeping busy. Focus on improving customer outcomes and removing waste, not on keeping people as busy as possible. While labor is expensive, no one benefits when busy people deliver the wrong solution. And keeping critical people with scarce skills as busy as possible often simply means that many others will end up waiting.

Change starts with a customer who could be better served, and a product delivery organization that wants to make them more satisfied.

Step 1: Integrate Your Approach to Delivering Software

many organizations find their approach is fragmented into specialist groups who are aligned around reducing cost and risk, instead of aligned around delivering customer value.

Development teams and support teams are separated and projects will even put different groups of people working on the same systems and business areas.

Thus, the first step encourages the organization to stop this madness by taking a holistic productcentric approach to delivering business value.

Building teams that are aligned to products and the customer. By aligning to a product or customers, that means that the Product Backlog will include new and maintenance requests.

Taking responsibility for the complete value chain, not just the development bit.

to be effective and deliver real business value, they must be clearly responsible for the end-to-end process of delivering value as a team, breaking down the silos of ownership and individualism.

Restructuring the portfolio around products/customers. The first step is approaching work on items in the context of the business, and restructuring around business outcomes and products, not around processes or specialist skills.

Removing queues from the system. When a Scrum Team is not able to do the work themselves, don’t make them wait for someone else to get help; make the help available whenever the Scrum Teams need it.

Provide self-service capabilities wherever possible, or staff the “expert” positions with enough people that there are enough of them to help teams whenever they need it. Either way, a strong focus on removing impediments to flow is crucial.

Provide Operations Services to the Scrum Teams. Having separate development and operations teams may be inevitable, but have team members available to be a service to the Scrum Team. Take part in the Daily Scrum, be assigned to a set of Scrum Teams, understand what is in the Product Backlog and Sprint Backlog. Be embedded with the teams you are a part of so that you can help them prioritize as much as you prioritize yourself.

Step 2: Add Feedback Loops

Adding operational and lean/business analytics to the process provides a foundation for the Product Owner to make the right decisions about their product.

What this means for Scrum Teams is:

Releasing software more frequently during Sprints. The Sprint Review is NOT a phase gate; software can be released as fast as required to production, and many times throughout a Sprint.

Building success measures into Product Backlog Items.

== Nota Pessoal abaixo, do próximo destaque: Adicionar no nome da história os cifrões do valor agregado da história para o cliente

By looking at Product Backlog Items in terms of how success could be measured not only ensures that when the team implements the item they will have some instrumentation in place, but it also validates the item itself.

== Nota Pessoal abaixo, do próximo destaque: Como definir o que é valor para um cliente? Seria: impacto? utilidade?

Making Sprint Reviews more visible and transparent to the business.

By adding metrics to the review, you can focus the review on the things that matter.

== Nota Pessoal abaixo, do próximo destaque: Avaliar, criar no Metbase métricas que exponham o que foi entregue em uma Sprint. Por exemplo: ao entregar o CRUD mapear quantos clientes estão criando contas através deles?

What this means for Operations is:

Building shared dashboards that show success in delivering customer outcomes. Too often, operations and development groups are using different dashboards to report progress, success or failure. By putting in place a dashboard that is consistent for both groups you build a communication environment that allows effective collaboration.

== Nota Pessoal abaixo, do próximo destaque: Construir uma monitoria para a Apolo, que inclui métricas de negócios

Being part of the Scrum Team(s), Daily Scrum, Sprint Review and Sprint Retrospective.

Scrum is an empirical approach; this requires constant testing of ideas through transparency, but tests are fundamentally flawed when the right stakeholders are not in the room.

By making operations professionals a part of the team and involving them in the various Scrum Events, insights can be gathered, allowing the future work in the backlog to be refined and plans changed.

Step 3: Grow a Culture of Continual Experimentation and Learning

The work of the team is based on clearly measurable outcomes with each “experiment” or Sprint working toward learning. Supporting these Scrum Teams is infrastructure and a community that is lightly coupled to the work via self-service capabilities, team members and mentors who help the team deliver work and improve.

To do this, teams need to:

Put in place evidence-based measures to enable teams to improve

Evidence Based Management (EBMgt™)10 provides an approach that looks to three groups of value-based measures in areas of current value, ability to innovate and time to market.

Empower the Product Owner to implement changes and make product decisions.

No need for complex reporting and sign-off committees. The Product Owner is accountable for not just the development of the product, but also its production usage.

"Evidence Based Management (EBMgt™)." Scrum.org. https://www.scrum.org/resources/evidence-basedmanagement.

== Nota Pessoal abaixo, do próximo destaque: Buscar na página Resources, artigos de qualidade sobre Scrum

What is true is that each organization will have multiple models running because their products/customer situations will require different support needs. And, organizations that think that optimizing the efficiency of operations is more important than delivering value will never become Agile.

Does the amount of complexity and the unknown make any shared model impossible to operate?

Does it make sense for these specialist skills to be learned by more people? Agility requires generalists and by supporting the transfer of skills to Scrum Team members, operations becomes less of a bottleneck and more a part of the team. It also ensures long term flexibility as more people have more skills.

Accountability and Responsibility

Traditional approaches to development and operations have created different areas of accountability and responsibility. The DevOps movement has focused on removing/changing this. However, without shared measures, objectives and a clear connection to the customer, it is impossible for this to be successful.

It is therefore important to align teams and teams of teams around the customer and the outcomes they seek. The container of these outcomes is the product which might map across several applications and systems. To be successful with this change in orientation, organizations need to have:

One shared Product Backlog.

Today most organizations have several backlogs with each team having their own work lists and priorities. To create shared accountability and responsibility, it is fundamental to have ONE shared, visible, Product Backlog. This backlog should be managed by one Product Owner who is accountable for ensuring that value is delivered.

A single, outcome-based dashboard that everyone works toward.

If you want everyone to walk to the same drum beat, it is important that everyone knows what that drum beat is. By building a shared dashboard, Scrum Teams and operations will work to the same transparent measures.

A Product Owner who is accountable for what is delivered to customers.

It is too easy for Product Ownership to be focused on delivering new stuff, but only lightly responsible for the operational systems, cost and quality.

Self-Organizing and Empowerment

Healthy, high-performing teams decide how they are organized, how they work, and what tools and processes they use.