On your best DevOps behavior

What kind of behavior is needed to 'do' DevOps?

DevOps is cool. And hot. It's one of the hottest IT trends in town. DevOps is a professional movement based on an emerging set of technical, organizational and cultural insights for fast delivery of resilient IT services, leading to a healthy workforce and bottom line results. Although much of the DevOps guidance can be applied in almost any area of IT, it's usually very much focused on the collaboration between Development and Operations. And as such, highly relevant for IT service management. Approaches such as highly-automated continuous integration, continuous delivery and continuous deployment help to speed up the deployment of releases into production while at the same time improving availability, reliability and security. But this is all in vain if people don't collaborate effectively. So, what constitutes good DevOps behavior?

The Three Ways of DevOps

The thinking in DevOps is structured around high-level guiding principles called The Three Ways of DevOps:

- Fast flow of work from development to operations
- Fast, good and frequent feedback
- Continual learning and experimentation.

The First Way enables fast flow of work from development to operations to the customer. In order to maximize flow, work is made visible, batch sizes and intervals of work are reduced, quality is built in by preventing defects from being passed to downstream work centers, and the global goals are constantly optimized.

The Second Way enables the fast and constant flow of feedback at all stages of the value stream. Feedback is amplified to prevent problems from reoccurring, or to enable faster detection and recovery. Quality is therefore created closer to the source and knowledge is generated or embedded where needed — this enables the creation of safer systems of work that address problems before the risk of disasters.

The Third Way enables the creation of a generative, high-trust culture that supports a dynamic, disciplined, and data-driven approach to experimentation and safe-to-fail risk-taking. Safe systems of work are created by continually shortening and amplifying feedback loops. This enables organizational learning from successes and failures. Additionally, systems of work are designed to transform local discoveries into global improvements, enabling others to benefit from lessons learned.

Effective behavior

Based on lessons learned in various The Phoenix Project business simulations (experiential learning in which a team learns by playing a game with business, Dev and Ops roles), the following behavior patterns were deemed to be crucial:

- Understand business context – IT asked the business about the various projects and their significance in terms of impact on the business
- Set right business priorities – the business roles were very conscious of the financial and reputational consequences of their actions, and when confronted by IT with "not enough capacity", they choose the most profitable combination of work items
- Visualize value stream(s) – the sequence of activities and handoffs from role to role was mapped, creating an overview of who does what and when
- Visualize work in progress – IT created Kanban boards to map the progress of work items
- Provide feedback/feedforward – there was regular communication back and forth between the roles
- Identify & elevate bottlenecks – when the weakest link in the chain was evident, other roles helped out
- Limit WIP (pull > push) – various roles learned to pull work only when there was enough capacity
- Working with the Agile / Minimal Viable Prduct concept of small batch sizes – the team split a project up into smaller parts, enabling the business to derive value earlier
- Focus on waste – the Lean discipline recognizes 7 or 8 kinds of waste and several of these where addressed, e.g. overproduction when work was executed before it was needed
- Develop cross-functional teams – various roles trained each other so that they could help each other when there was not enough capacity
- Shift-left testing – the test team engaged with various roles as early as possible instead of waiting for the last link in the chain
- Use Andon cord to stop & swarm, & solve problems – the Andon cord principle of stopping the production line was used once when something when wrong: people gathered and discussed the issue before resuming work – this is only possible when people feel safe to take such a major step
- Continually learn & experiment – this was done after each round of the simulation
- Foster collaborative culture – there was an open, collaborative atmosphere – when money was lost due to omissions by some team members, they were not blamed

These are good examples of the kind of behavior that contributes to the healthy, generative culture that characterizes organizations that have adopted DevOps successfully.