INSTITUTIONEN FÖR DATAVETENSKAP | LUNDS TEKNISKA HÖGSKOLA | PRESENTATIONSDAG 2015-08-28

EXAMENSARBETE Piping for Continuous Delivery
STUDENTER Jonathan Klingberg
HANDLEDARE Lars Bendix (LTH)
EXAMINATOR Ulf Asklund

Aiming for Continuous Delivery

POPULÄRVETENSKAPLIG SAMMANFATTNING **Jonathan Klingberg**

Software companies are transforming their development process into becoming more agile to keep up with a rapidly changing market. This comes with challenges, which Continuous Delivery aims to solve.

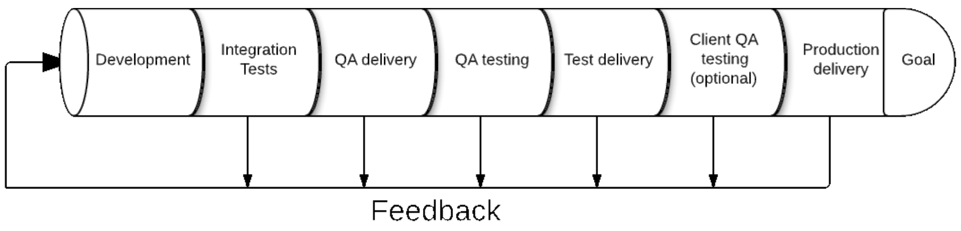
Time to market (TTM), i.e. the length of time it takes from a product being conceived until it's being available for sale, has become more important than ever for software development companies operating in a competitive market. In order to reduce TTM lag, many companies are today transforming their development process by leaving an old-fashioned one, like the waterfall model, behind and instead aiming to become more agile. With this comes the need to have shorter release cycles to get feedback faster and be able to act on it just as fast. The name Continuous Delivery (CD) has emerged from the agile community and it defines the principle of frequent deliveries in a highly automated manner. CD holds principles and practices that can help keep the application in a working state, making releasing software a piece of cake.

This case-study started on a company that suffered from an overheated Delivery Department that did not manage to deliver reliable software as frequent as needed. During the problem analysis it soon turned out that the problem was actually on a completely other level, not directly connected to Deliveries work but rather to the work model in whole. Today's process is based on the waterfall model where the application is kept in a non-working state during long periods of new development that then requires intense fixing close before the release. This is not sustainable if more frequent deliveries are the goal and instead new practices to keep the application in a working state is required.

The current release cycle is about 3 months long and since the customers can't wait so long for an upgrade, this has led to consequences. Each upgrade holds a great number of new features and further also a great risk for breakage. The instability of the deployments has affected the customers' trust which now rather prefer to have their own product variant which includes less features and also a reduced chance of breakage. A release of such variant is also faster than a regular upgrade, making them very popular. This however has led to a variant-inferno where all customers have their own customized product variant, which requires individual support.

By localizing the main barriers in today's process and discussing how Continuous Delivery-principles can be used to address these problems, we will end up in an alternative release management process which is more rapid and further also lives up to the customers' needs. Strategies to help keep a releasable codebase and increase the general product quality are discussed primarily from a configuration management perspective. Release-branching, continuous integration, production-like environments and continuous testing are just some examples of practices being discussed as possible improvements.

By creating a prototype of an improved delivery pipeline based on tools such as Vagrant, Ansible and Jenkins, this can be used as a proof of concept before implemented at scale.



An improved release management process illustrated as an automated delivery pipeline