March 2017

# Continuous Software Engineering

Yvonne Dittrich, Lektor, IT University of Copenhagen
Jacob Nørbjerg, Lektor, Copenhagen Business School
Lars Bendix, Lektor, Lund Tekniska Høgskolan

With the advance of the deployment of software as a service and the possibility to update software on mobile phones, embedded computers and personal computers through digital networks, the frequency of updates of software increases. Internet services like Google and Amazon are deploying new software several times a day. The industry talks about 'cloud cadence', and in the beginning academic discussion the term 'continuous software engineering' has been established. New ways of cooperation between system operators and development teams are discussed under the terms 'DevOps' – developers are tightly cooperating with systems operators in case of a service.

Continuous deployment is seen as a strategic business advantage, as it allows shipping development results quicker and harvesting the return on investment earlier. This is most relevant for service industries like banking, insurance and online games. However, also software products, which are more and more provided as a service as well as in a shipable version, technically embedded software and mobile applications are observed to adopt continuous software engineering to improve quality.

The proposed research project sets out to develop processes, methods, tools and techniques to support the Danish industry in this rapid change. To this end we will deploy an empirical research approach, where researchers and practitioners together experiment with new techniques, tools and methods. We aim at recruiting a number of companies who have started or want to work with continuous deployment and continuous software engineering.

We plan to apply for external funding from the Innovation Foundation with a deadline June 2017. The project would start 3 to 6 months later.

We currently are in contact with several companies. The following focus areas transpired from the bilateral discusions:

1. **Continuous software development processes** We see a diversification of rhythms of different activities in the software development process in organisations that work with continuous deployment. Whereas in more traditional development the release schedule provided a common clock, continuous deployment allows for parallel teams to have independent development schedules. Further, different software development activities follow independent rhythms This requires new kinds of coordination mechanisms. This also relates to cooperation with business departments, and with current and future users.
2. **Continuous Quality management:** The possibility to deploy a change, update or bug fix of software within short term requires a development environment that automates integration, build, test and deployment. Besides massive, often parallelised test automation, sophisticated configuration management strategies need to be developed and automated. Not only source code, but also database schemes and binaries need to be taken under configuration control for incremental update and rollback of changes. This in turn has implications on the architectural design.
3. **Continuous Integration Tooling & Architecture:** Coordination in continuous development requires new coordination mechanisms and in turn continuous integration, continuous testing and continuous delivery/deployment tools. The flexibility required though will have repercussions on software architecture: Massive incremental change puts additional requirements on the technical design of software. Deployability becomes a software quality criterion. The use of micro-services to this end in turn requires again different coordination and different testautomationstrategies.