SCM and DevOps

Lars Bendix  
bendix@snescm.org

Christian Pendleton  
cpe@praqma.net

Scandinavian Network of Excellence  
in Software Configuration Management  
(sneSCM.org)

Praqma AB  
Malmö  
Sweden

http://cs.lth.se/~bendix/Research/SCMnDevOps/
Preamble

Premise:
- no project can be (successfully) carried out without SCM
- it might not be called SCM
- it might not be carried out by SCM people
- so, from the outside it might seem like SCM is absent in DevOps
SCM needs to change
Revolutions
Presentation objectives

After this presentation you will:

- have refreshed your SCM concepts and principles
- know something about DevOps ”philosophy”
- understand what role SCM has (also) in DevOps
- have a better idea of where (and what) SCM activities are needed in DevOps processes
- know how to generally adapt to DevOps ”philosophy”
- understand how to implement SCM activities in a DevOps context
What is SCM – short version

Software Configuration Management is this cool stuff that will facilitate a team to coordinate people and things so they can carry out changes in an orderly fashion right from idea/conception to production – and retirement – and avoid any chaos and confusion in the process.

SCM will make sure that you know exactly what you have, not just when it is in production but also while you are developing it and SCM will provide a team with all the safety nets they could wish for.

SCM can be done in formal and rigid ways (top-down/waterfall) - or it can be done in more informal and flexible ways (bottom-up/Agile, DevOps).
What is SCM - longer version I

It is about tracking, managing and controlling changes to (re-)establish baselines.

A process for establishing and maintaining consistency and integrity of a product. Provides visibility, control, orderly change.

Provisions for the storing, tracking and updating of all parts.

Tracks requirements (change requests) throughout the life-cycle.

Establishes baselines and performs a standard change management process through to “release management and delivery”.

Configuration management database – Configuration Items, their attributes and the dependencies between them.
What is SCM – longer version II

Part CM:
- Identification (configuration items, baselines, traceability)
- Control (change process, change requests, CCB)
- Status Accounting (recording and reporting information)
- Audit (assesses compliance with requirements before acceptance of change into a baseline)

Part Software:
- Build management
- Process management (adherence to development processes)
- Environment management
- Teamwork (facilitate team interactions and parallel work)
- Defect tracking
What is SCM – longer version III

Roles:
   • Strategic SCM
   • Operational SCM

Triangle:
   • People (education)
   • Processes
   • Tools
DevOps is: “a set of practices intended to reduce the time between committing a change to a system and the change being placed into normal production, while ensuring high quality”

Also includes monitoring and measuring of the software in runtime!
What is DevOps – longer version I

Culture (People, processes and tools)
Automation (Do it once, do it twice, automate)
Lean (Continuous improvement and learning)
Measurement (development, production, business)
Sharing (Collaborate, give feedback, don’t copy)
What is DevOps – longer version II

Goals:

• improved deployment frequency
• lower failure rate of new releases
• shortened lead time between fixes
• faster mean time to recovery

Aims to maximize the **predictability**, **efficiency**, **security** and **maintainability** of operational processes.
What is DevOps – longer version III

How does DevOps differ from Agile/Scrum/XP?

What more is DevOps than “just”: Dev + Ops?

Idea => backlog => development => production => used by people

In order to get as fast as possible from idea to use we take a small (part of an) idea and work continuously on it in a cross-functional way until it is in production and then we monitor what happens when it is used to get feedback that creates new ideas that ……
DevOps Life cycle

Customers

Business Owners

Development/Test

Operations/Production

Continuous innovation, feedback, and improvements

Plan and Measure

Develop and Test

Release and Deploy

Monitor and Optimize
The five stages of DevOps evolution:

- **Stage 0**: Build the foundation (facilitate sharing of ideas, metrics, knowledge, processes and technologies)
- **Stage 1**: Normalize the technology stack (agile, version control, continuous integration)
- **Stage 2**: Standardize and reduce variability (reduce overall complexity, reduce errors from inconsistencies)
- **Stage 3**: Expand DevOps practices (deployments are a huge pain, provide predictability and reliability)
- **Stage 4**: Automate infrastructure delivery (automation of system configuration and provisioning)
- **Stage 5**: Provide self-service capabilities
SCM-related DevOps activities I

Foundational practices:
• Monitoring and alerting are configurable by the team operating the service
• Deployment patterns for building applications or services are reused
• Testing patterns for building applications or services are reused
• Teams contribute improvements to tooling provided by other teams
• Configurations are managed by a configuration management tool
SCM-related DevOps activities II

Various practices:

- Application development teams use version control
- Put application configurations in version control
- Source code is available to other teams
- Deployment patterns for building applications and services are reused
- Put system configurations in version control
- Teams use continuous integration
- Infrastructure teams use version control
- System configuration is automated
- Provisioning is automated
- Success metrics for projects are visible
SCM-related DevOps activities III

Automate system configurations

- Some teams have a goal of automating all change, giving them completely repeatable, rebuildable systems.
- Consistency: Automated tasks follow a set process and thus produce predictable results.
- Documented behavior: Tasks now have a defined way they are supposed to work, so are easier to troubleshoot.

Application configurations are in version control

- Should be versioned, auditable, contain history, and ideally, the reasons why they changed.
- Separating code from configuration data allows for more rapid deployments, updates and validation.
Fve metrics define SDO performance

Software delivery and operational performance:

**PERFORMANCE METRICS**

**SOFTWARE DEVELOPMENT**
- Lead Time
- Deployment Frequency

**SOFTWARE DEPLOYMENT**
- Change Fail
- Time to Restore

**SERVICE OPERATION**
- Availability
TECHNICAL PRACTICES

- Deployment automation
- Continuous integration
- Trunk-based development
- Loosely coupled architecture
- Version control

CONTINUOUS TESTING

MONITORING & OBSERVABILITY

DATABASE

SECURITY

CLOUD INFRASTRUCTURE

Westrum organizational culture

SDO PERFORMANCE
- Software delivery performance

AVAILABILITY

- Deploy pain
- Burnout

Organizational performance
TECHNICAL PRACTICES

- Deployment automation
- Continuous integration
- Trunk-based development
- Loosely coupled architecture
- Version control

CONTINUOUS TESTING

- Monitoring & observability
- Database
- Security

Frequent commits
No long-lived feature branches
“...it’s easy to modify or replace any individual component or service”
Test & infrastructure managed

Test data is available

Get feedback from automated tests in less than ten minutes

Continuously *review and improve* test suites to better find defects and keep complexity and cost under control
Different roads to DevSCMOps

“Zero” ➔ DevOps&SCM

”Hardware” ➔ DevOps&SCM

“Zero” ➔ DevOps&SCM ⇐ “Hardware”
How to organize SCM?

Company (usually start-up) with only one team (3-8 people):
  • SCM as a service / consultant + ”deputy”

Company with 3-4+ teams:
  • SCM “team”
  • SCM on teams
Discussion questions I

**Why** do you use version control?
**Discussion questions I**

**Why** do you use version control?
- to store
- to share
- to diff
- to track changes
- to roll back (if needed)
- to remove fear of making changes (can revert)
- to avoid “it works on my computer” (sharing)
- Wayne Babich
- to conserve old states (if sued?)
- but – how to revert a database with one month of data?
Discussion questions II

Concrete examples:

• What branching strategies are there and what are their pros and cons?
• Why and what should be automated?
• Why should you worry about configuration items and a CMDB (git)?
• What are quality gates and where and when to use them?
• Why are concepts like baselines and BoM also useful for DevOps?
• Why should you have a change process with CRs and a CCB?
• Why is “build equivalence” important (if you are Huawei)?
• How do you handle multiple versions of libraries?
• Do you know which libraries you use – and where?
Famous last words

This is the last time that I will deploy (someone else will do it in six month) and I can remember how I did it six month ago, so it is much faster/easier to do it by hand.

This is the last time that someone comes to me to ask for a new machine with a test environment, so it is much faster/easier to set it up by hand.

Checklists? A codified representation of things we have previously screwed up!

Automation? A quicker and safer way of doing things that we (or others) have to do often!
What is SCM – short version

Software Configuration Management is this cool stuff that will facilitate a team to coordinate people and things so they can carry out changes in an orderly fashion right from idea/conception to production – and retirement – and avoid any chaos and confusion in the process.

SCM will make sure that you know exactly what you have, not just when it is in production but also while you are developing it and SCM will provide a team with all the safety nets they could wish for.

SCM can be done in formal and rigid ways (top-down/waterfall) - or it can be done in more informal and flexible ways (bottom-up/Agile, DevOps).
Conclusions

DevOps change requests to SCM:
• help us ;-)
• handle many small changes – roughly one day to one week
• keep us in the flow – once we stop we are dead
• make it fast – we hate to wait
• KISS – because we will do it (as instructed)
Final words

SCM as a discipline is invariant (is published)

SCM as a role will change (will not perish)