@ExpertsLiveAT
#ExpertsLive

CI/CD

DevOps and Ops without Dev ;-)

Michael Pambalk-Rieger, CEO
www.timewarp.at
@timewarpit

Experts Live Austria

Who am I

- Founder and owner of TIMEWARP IT Consulting GmbH (since 2005) – www.timewarp.at
- Working in IT since 24 years
- Co founder of the **S**ystem **C**enter **U**ser **G**roup **A**ustria which is now expertslive Austria (merged with 2 other user groups)

What does CI/CD mean?

- CI (continuous integration)
  - Automatically building and unit/integration testing an entire application frequently, ideally on every source code check-in.
- CD (continuous delivery)
  - Releasing new changes to production at an point of time by clicking on a button
- CD (continuous deployment)
  - Automatically deploy every build to production after it passes its automated tests (best case)

Do I need CI/CD ?

- Do you want more deployments as you have right now?
- Is it a pain to plan and deploy an application release?
- Do you experience buggy and unstable services after deployment of a new version?
- Does it take days, weeks or months to build and release applications in your organisation?

Benefits of using CI/CD

- Lower release cycle times (from maybe months to minutes or hours)
- Maximize quality of delivered artifacts
- Deploying small packages with less debugging
- Reducing stress during deployment ;-)
- Large dev teams has the most benefit
- Standardizing testing and deployment

Source control (gitlab)

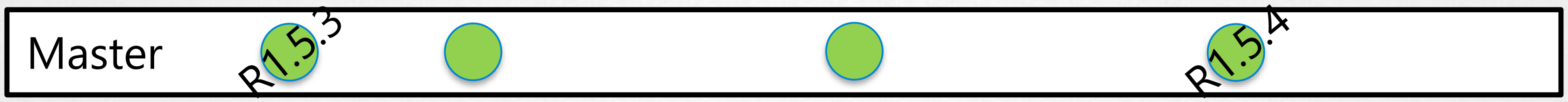
- No project is to small for a source control
- It maintains all states and historic changes
- Include everything needed (also deploy scripts)
- Write understandable commit message's
- Don't commit code which doesn't build
    - us pre commit hook's
- Done add sensitive information and commit it to SC

Different branching concepts

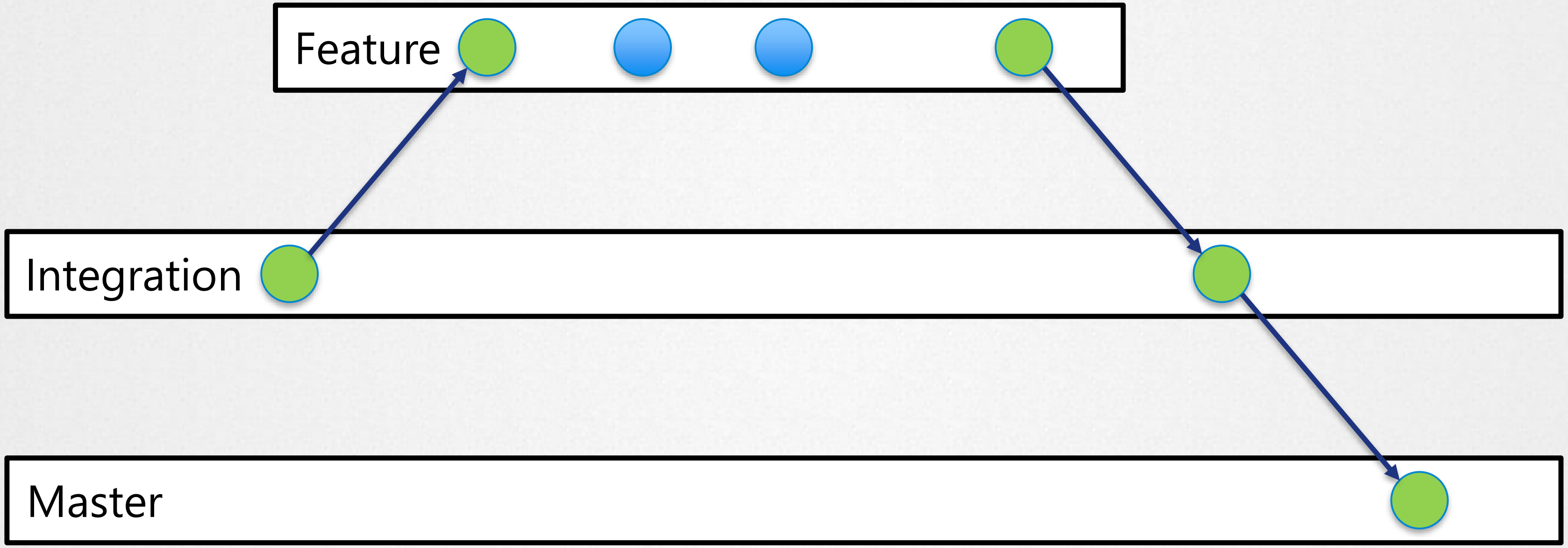
- Branching (and TAGGING)
  - Multi long running branch strategy
  - Single branch strategy
  - Feature branch strategy (combined with long running branches)
- Deployment
  - Staged deployment with manual integration
  - Real time deployment to production with every check-in
  - Deploying artifacts to production only or build and deploy to production code as needed

Single branch with TAGs

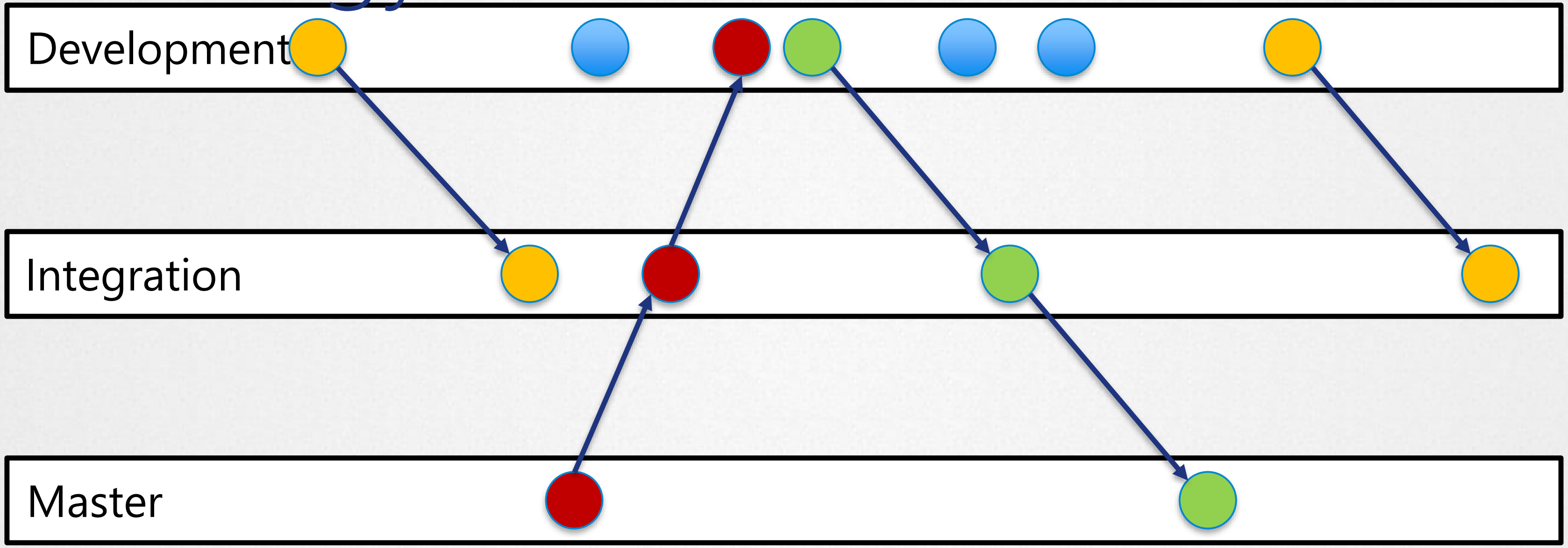
- Not recommended but also working
- Use TAGs for deployed version at production



Featue branching



Multi long running branch strategy



Merge from dev to int to prod only!! (excep hotfixes)

CI is a kind of work, not a tool

- Never commit code which doesn't build
- Before committing run tests locally
- Don't go into the weekend with a broken build
- Tests that fail should be fixed, not removed ;-)
- Notifications are key (slack, teams, email, ...)

Deployment Artifacts?

- Deploy to production exactly what is tested
- Docker image is a good sample
- Control who has the right to write to the repo (build system only)
- It is easy to share

Test types

- Unit tests
  - kind of White Box Testing (internal view)
  - Test the unit of code
  - written by the developer who has written the unit of code
- Integration tests
  - Kind of Black Box Testing (external view)
  - Tests integration between software modules
  - verifies that your code works with internal and external dependencies correctly
- End 2 end
- And many more ….

Build pipeline with c# web app

Source Code Dev team

SC Version control

Gitlab

Build System & tools

Gitlab-Runner

Dotnet build

Unit tests

Dotnet test

Integration tests

Dotnet test

Artifacts

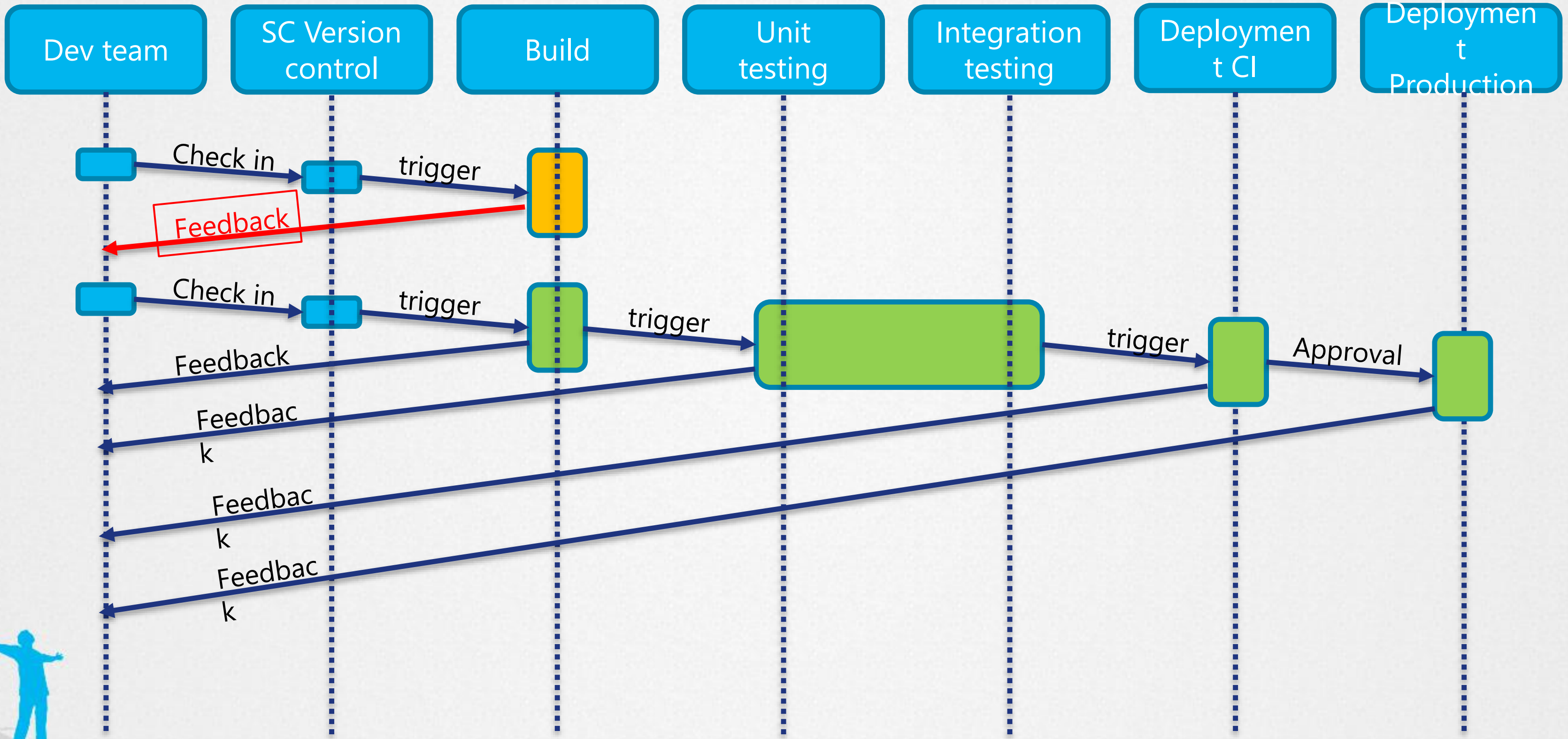
Artifact repository

Deployment tools

Dotnet publish

Environment

Pipeline (CDeploy & CDelivery)



Live demo web application

- Multi long running branch strategy used
- C# asp.net core application with CI and continuous delivery
- Gitlab-ci.yml?
- Merge with gitlab
- Deploy to production

Does CI/CD also work for ops only?

- It guarantees standardized testing and deployment
- It secures the access to the environments
    - No access to target system needed
- You have a history of every change
- Only valid configurations are going to production
- Notification of every change on production
- Delegate high level changes to lower trained people

Prepare environment

- Best results with CI/CD if environment preparation is also automated
- Do expect the environment prepared (works) as designed

Live demo DNS infrastructure

- Master branch only
- Continuous deployment instead of continuous delivery
- Realtime infrastructure preparation with ansible
- New servers are added during deployment

Tools and systems

- Gitlab.com
  - GitLab is a single application for the entire DevOps lifecycle that allows teams to work together better and bring more value to your customers, faster.
- Ansible.com
  - Automation for everyone
- Visual studio IDE
- Azure virtual machines

Ein Danke an unsere Sponsoren!

SecureGuard

Hewlett Packard Enterprise

e-shelter

rubrik

ACP

iQSol Security made in Austria.

INGRAM MICRO

TIMEWARP INFORMATION TECHNOLOGIES

au2mator

code4ward.net

royal applications

NiCE IT Management Solutions

Powershell Usergroup Austria