Outline

- Types of testing - overview of testing categories and misconceptions
- How to test now
- Dream system vs What we have now
  - Big
  - Fast
  - Lasts “forever”
  - Data
  - Import (automatic & non-automatic, with metrics) (5 minutes)
  - Access (to logs, via clients, to DB via psql, Managed Repo, configs)
  - LDAP
  - Apps
  - Server-performance testing & tools
  - Versioning (apps, server and dependencies)
  - Continuous integration *
  - Integration tests, robot tests*
  - How others do it - Google CI workflow

Types/methods of testing

- Manual vs automation
  - Recommended to run the manual testing first
  - The automation should be approached only in certain cases, depending on costs
- Black box vs White box
  - https://www.tutorialspoint.com/software_testing/software_testing_methods.html
  - Depending on the knowledge of the tester
  - We are usually in the grey zone

Misconceptions

- Testing is Too Expensive
- Testing is Time-Consuming
- Only Fully Developed Products are Tested
- Complete Testing is Possible
- A Tested Software is Bug-Free
- Missed Defects are due to Testers
- Testers are Responsible for Quality of Product
- Test Automation should be used wherever possible to Reduce Time (manual first, must be stable, changing requirements make impossible)
- Anyone can Test a Software Application
- A Tester's only Task is to Find Bugs

How to test now – PR review

- Check that the PR was included in the build, using https://github.com/snoopycrimecop/openmicroscopy
- Use provided CI servers (e.g. eel) preferably over localhost (short tutorial about how to use eel)
- Use real data over fake files
- If forced to start from fresh DB, do not use root/system group for your manual test, create at least one group with one ”normal” user, or use https://github.com/openmicroscopy/management_tools/blob/master/ci/config/create_users script to have a decent complexity in your DB (download it and use “bin/omero load create_users” command)
- Insist on developer putting some testing steps into the header of the PR, but try to think about other areas which the PR might influence and consult/test them
- Write ”everything” down, mainly in cases where a second re-test on the same PR is probable (your setup and commands/clicks used during the test as well !)
- try to be widespread in your testing
- read the comments history on the PR
- If clearly overburdened by the complexity, ask for the testing to be relisted and for other colleague to test the PR
- If overburdened by the workload, do make a remark at the end of the day on the PR, indicating that you will do the testing tomorrow or that you are unable to manage the load (**do not simply do nothing !)**
- Always check automatic tests failures – was this PR a culprit ?
- Learn how to establish LDAP (using docker and scripts) which you can manipulate if necessary, see further slides

eel workflow

- ssh your-lifesci-ldap-name@eel.openmicroscopy.org
- cd /home/hudson/
- ls
- cd OMERO-DEV-merge-deploy/ # or other server as needed
- eval $(bash /home/hudson/ice/ice-multi-config.sh ice35)
- bin/omero login # should work now
- bin/omero config get # to get the name of the DB and path to ManagedRepo
- psql -U omero –l # to verify the DB list
- psql -U omero OMERO-DEV-merge-deploy # or other DB as appropriate
- SELECT….. etc. to use sql queries
- #from another terminal, to copy logs to your local machine
- scp -r your-lifesci-ldap-name@eel.openmicroscopy.org:/home/hudson/OMERO-DEV-merge-deploy/var/log .
- #or, being inside eel, inspect the logs
- cat var/log/Blitz-0.log | grep ERROR

How to test now – bigger testing

- Study the testing sheet, especially “concentrate on” column
- Double-check server address, and your supposed username and group you are supposed to work in
- Write everything down, preferably into the new tab Tester’s report, giving as much detail and data links as possible
- Store screenshots on squig
- For load, stress and pressure testing, note also times when problems occurred

Prep of larger testing (using g.sheet)

- Select the sheet nearest to your topic and setup from
- A. G-drive > OME Docs > Testing or
- B. G-drive > OME Docs > Testing > sprints
- Example: https://docs.google.com/spreadsheets/d/1IrRaJ60utV_IHqXa_zxQhkz3RKNnbW6sw8B68xTYItA/edit#gid=1254367109
- Being inside the g.doc folder containing the sheet, copy it by right-click > Copy
- Open the copied sheet and save it under a new name which describes your testing topic
- Exchange the links to the scenarios in “scenario name” column
- Note https://docs.openmicroscopy.org/internal/testing_scenarios/index.html
- Reformulate the “concentrate on” column (specifying whether or not the whole scenario is to be tested, or part, or some additional steps not included in scenario)
- Specify the clients to use, OS and browser types
- Specify the users and groups to be used during testing for each tester
- Pick the names of the testers, checking their availability in the calendar
- Try to speak to testers prior to testing, to make sure they are ready and capable of performing the testing with the prescribed setup

What to do when scenario/workflow is outdated/ imprecise

- As appropriate either/or, or both:
  - Open a PR on https://github.com/openmicroscopy/ome-internal
  - Improve the hints on https://trello.com/b/gUTby8cp/omero-release-template in OM|ERO: Release review column

Dream testing system
(OME Docs > Testing > Dream testing system)

Big

- current reasonable research scales— ~1-100TB.
- emerging (flagship) projects: 1-2 orders of magnitude more
- In-place import : how far will it get us ?
- eel has 150 GB ?

Fast

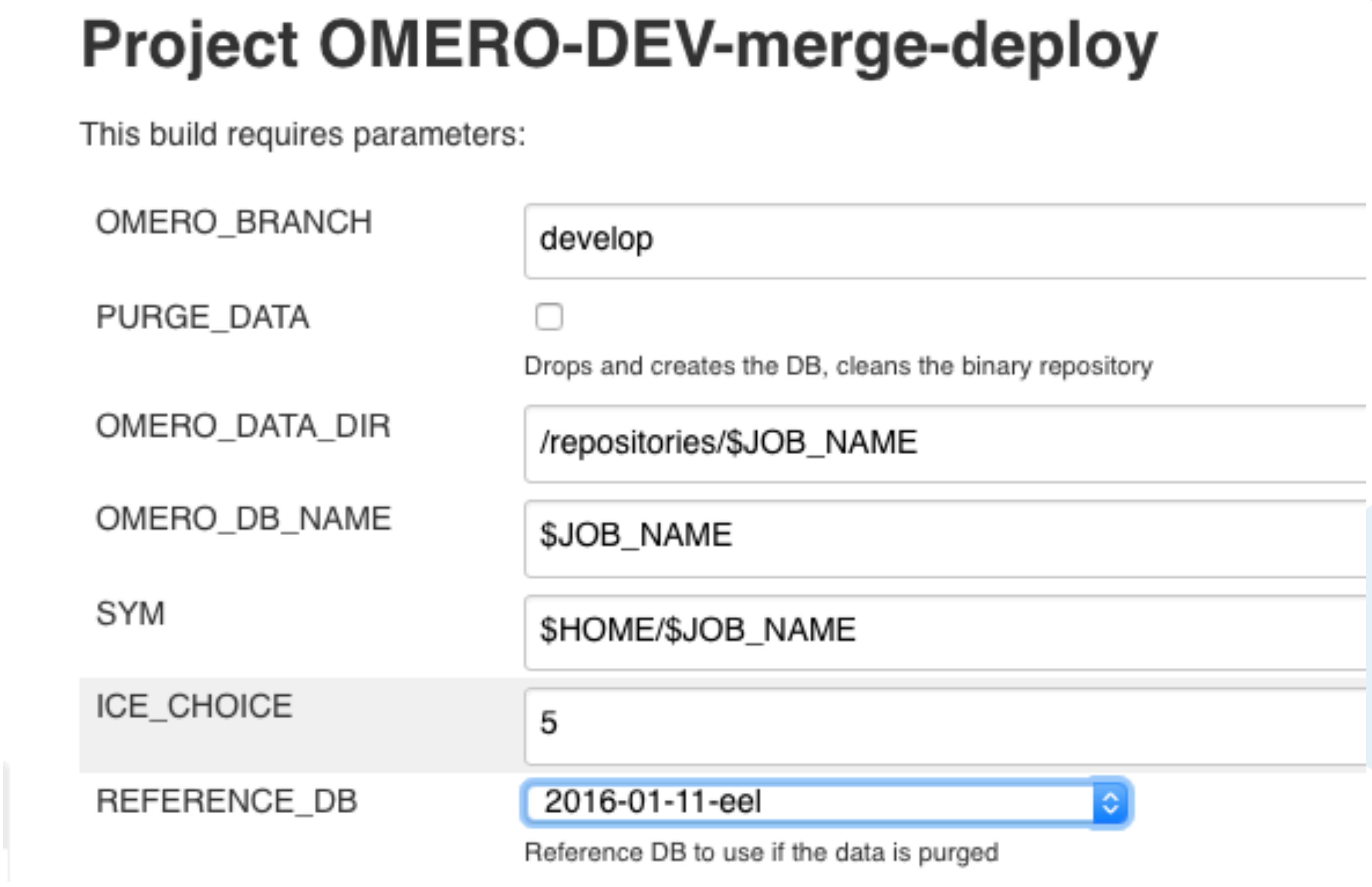
- Not specified in numbers, but the current speed of nightshade server is acceptable, the current speed of eel server is not
- When testing system is constantly slow, then any feedback on the “sudden slowing down” due to cause x is not possible

Long-lasting setups

- Necessary for
  - Repetitions and retests on the same data
  - Building up systems with specific data (possibly imports take long, and the data in question are not contained in any DB yet)
  - Catching bugs which appear only via long usage
  - Catching performance issues connected with accumulation of unnecessary files in the ecosystem
  - Lack of long-lasting setups typically results in (mis)using production servers for debugging
  - Atm we are using nightshade and demo for these purposes

Data-flexible

- Switching (switching DBs & original data at will)
- Reverting (quickly getting DB and original data into a state before experiment)
- Merging (missing feature in OMERO, merging 2 DBs)
- Snapshotting (capture the state of the DB quickly just after experiment)
- Duration (see previous slide)
- Searching (missing search for Metadata in OMERO, hard to search for metadata in a filesystem)
- **What we have now:**
- Atm we have the CI widget enabling to switch DBs and purge data, e.g. https://ci.openmicroscopy.org/job/OMERO-DEV-merge-deploy/build?delay=0sec ,See screenshot on next slide
- DBs are stored (OME Docs > Testing > CI setups and DB structures)



Project OMERO-DEV-merge-deploy

This build requires parameters:

OMERO_BRANCH	develop
PURGE_DATA	☐ Drops and creates the DB, cleans the binary repository
OMERO_DATA_DIR	/repositories/$JOB_NAME
OMERO_DB_NAME	$JOB_NAME
SYM	$HOME/$JOB_NAME
ICE_CHOICE	5
REFERENCE_DB	2016-01-11-eel Reference DB to use if the data is purged

Import

- Automatic
- Non-automatic
- Metrics on import
- Atm we have the IDR import machinery (we learn more in Screen importing workshop)
- Atm we have the script of Josh for minimal automatic import of selected files to OMERO
- Atm we have Balaji’s script in Matlab ? (for metrics on import)

Access

- Via clients
- Logs
- Managed Repo
- Config
- Atm we have some access on eel
- Atm we have possible access to the dockers, but need to find out during training how

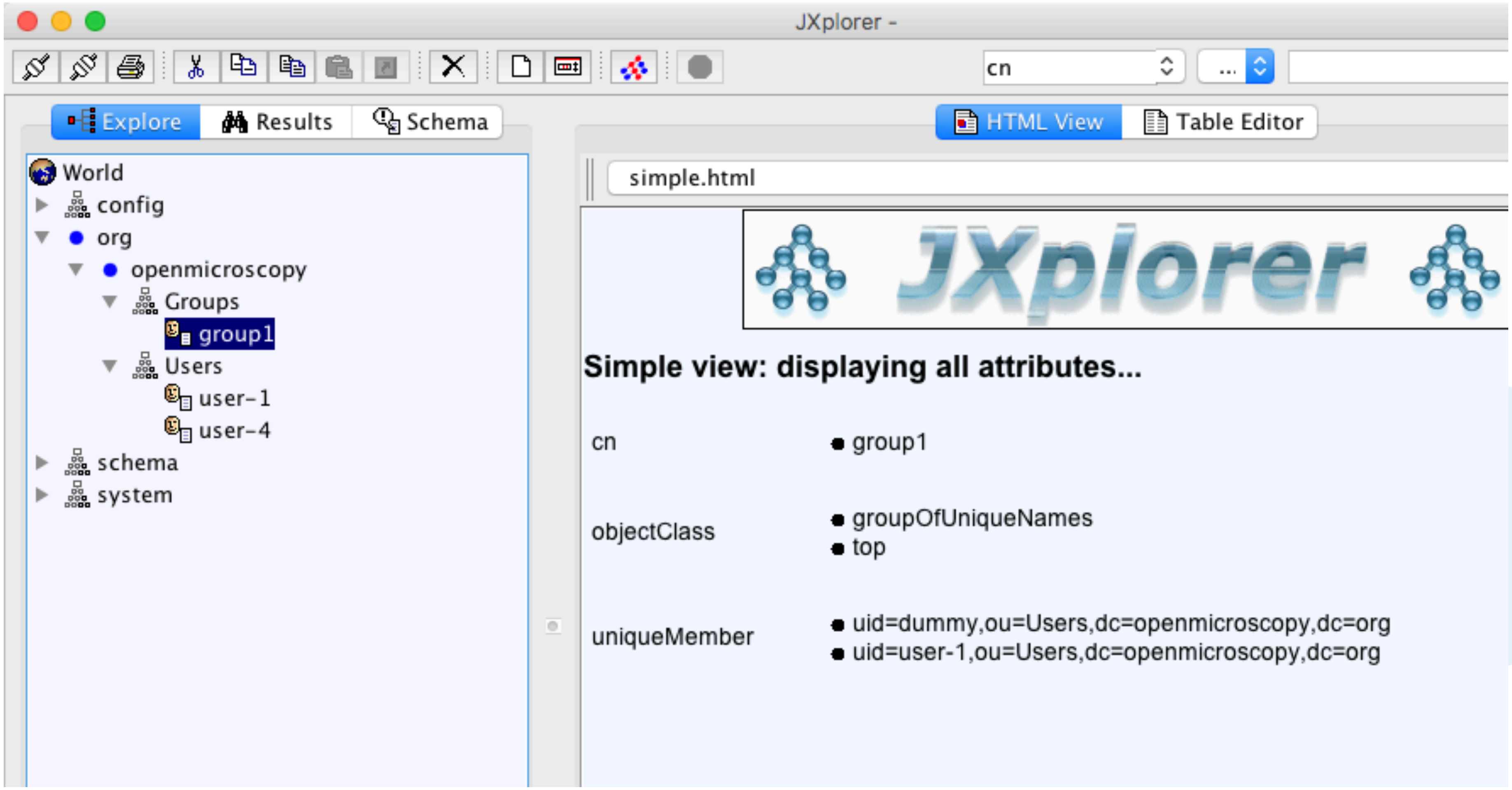
LDAP

- LDAP is almost a default with our users. But atm we have no LDAP server in our hands.
- Docker training should reinforce the workflow about how to establish our own LDAP server (but some basic workflow is going to be given here)
- Permissions on beluga (running docker on beluga) unclear -> computational resources workshop
- https://github.com/openmicroscopy/apacheds-docker#installation
- 1- start an ApacheDS Docker container on localhost (in future, minke/ beluga/xxx -> need to clarify docker perms on those servers) with the correct port-forwarding
- 2- use the scripts to create your LDAP tree on this LDAP server https://docs.openmicroscopy.org/internal/instructions/ldap-dev-configuration.html
- 3- set up your OMERO to point at this LDAP

LDAP for OMERO via docker setup example

- docker pull openmicroscopy/apacheds
- docker run --name test_ldap -d -p 10389:10389 openmicroscopy/apacheds
- cd /management_tools #cd to the cloned mngmt tools repo on your machine to be able to run scripts
- export LDAPHOST=localhost
- ./search
- ./initialize
- ./user user-1 add # note: the users will have the usual pwd, just like on eel
- ./group group1 add
- ./user user-1 in group1

JXPlorer view of LDAP DB



- World
  - config
  - org
    - openmicroscopy
      - Groups
        - group1
      - Users
        - user-1
        - user-4
  - schema
  - system

Simple view: displaying all attributes...

cn	● group1
objectClass	● groupOfUniqueNames ● top
uniqueMember	● uid=dummy,ou=Users,dc=openmicroscopy,dc=org ● uid=user-1,ou=Users,dc=openmicroscopy,dc=org

Connect your LDAP to OMERO

- https://github.com/openmicroscopy/management_tools/blob/master/ci/config/common/devldap.omero # configure your omero server to point to the LDAP – exchange “minke…” to localhost as appropriate and use bin/omero load <filename>
- bin/omero load ~/ldap.omero # replace ”ldap.omero” with correct filename
- bin/omero user list
- bin/omero ldap create user-1
- bin/omero user list
- bin/omero group list # group should be synced as well because user-1 is a member of group1 in ldap

Apps

- Installation
- Versioning
- Performance (of the app but also burden on server)
- Atm we have https://github.com/openmicroscopy/management_tools/blob/master/ci/config/WEB-DEV-merge-deploy/requirements.txt

Server-performance testing

- 1. general performance numbers (speed and CPU utilisation numbers)
- 2. identification of classes and methods responsible for performance changes
- Used tools ad 1:
  - CLI + ”time” command run manually + bash scripts run locally
  - Docker images spun up on a remote server + bash scripts run automatically https://gist.github.com/joshmoore/b4e69fa232e9d769dd77b6ad202232e3
  - Check_MK monitoring tool for demo server (CPU utilisation)
- Used tools ad

  - Eclipse plugin: JVM Explorer (splits things into Threads, no way around

Versioning

- Apps
- Server
- Dependencies
- Atm…
- 1. We need a widget or a more simple way to find out what versions of the whole stack we are working at any given time.
- 2. We need a way to deploy any set of versions we like (keeping in mind the “microservices” approach will make this even more complex than now).

Continuous integration

- Server – merge
- Server – latest
- Web
- Apps – merge, latest ?

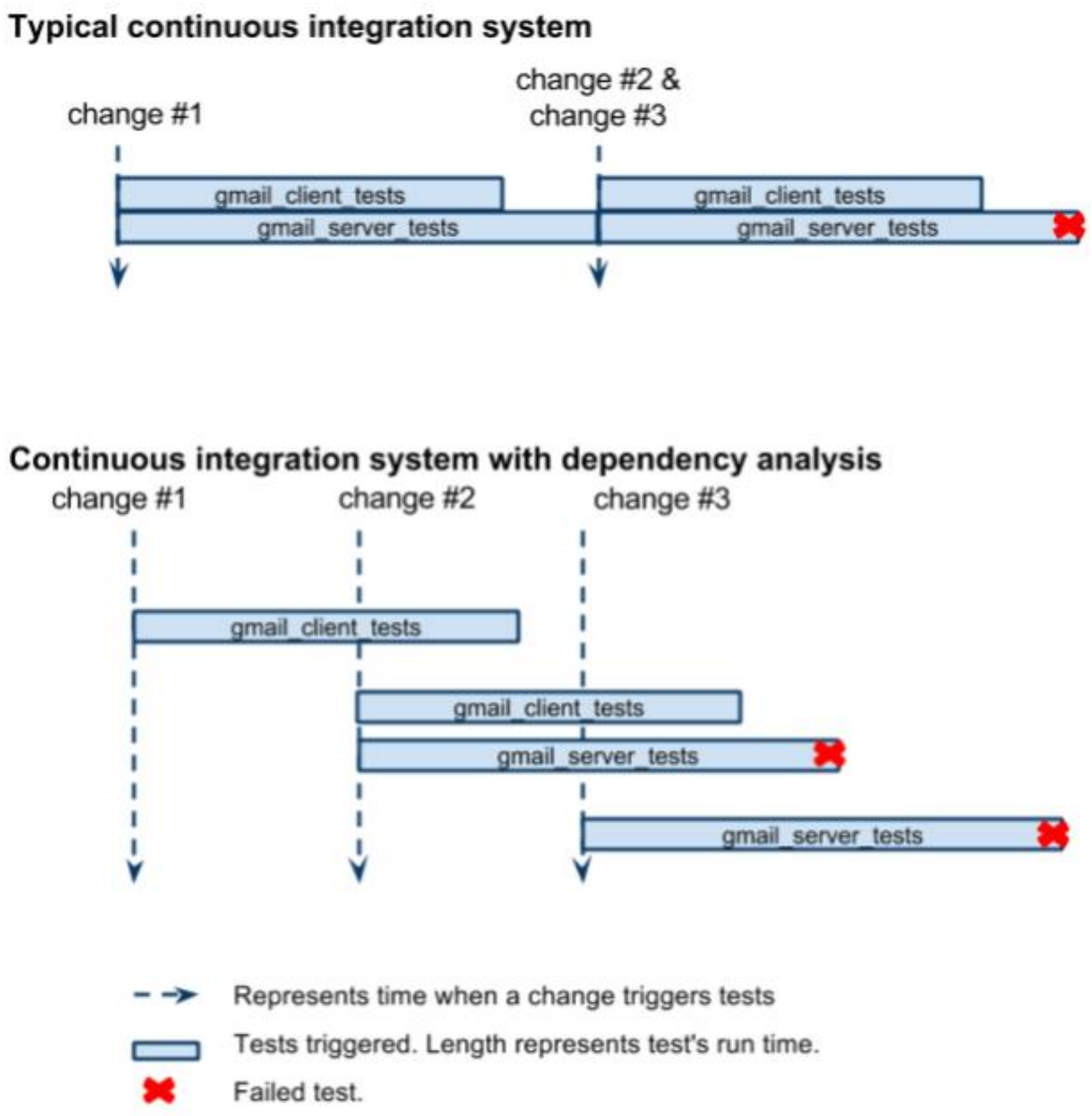
Automatic tests

- Integration tests (why so flaky ? See google example next slide)
- Robot tests (flaky and low coverage)
- What else could we use ? The following tools can be used for automation testing:
- HP Quick Test Professional
- Selenium
- IBM Rational Functional Tester
- SilkTest
- TestComplete
- Testing Anywhere
- WinRunner
- LaodRunner
- Visual Studio Test Professional
- WATIR

Google CI

- https://testing.googleblog.com/2011/06/testing-at-speed-and-scale-of-google.html
- Do not run all the tests every time
- Have a **schema** defined -> **graph** which enables **tracing which tests are influenced by a particular change**
- Run only the influenced tests. Run the tests immediately after a PR is opened (or otherwise the change in code occurs)
- Helps to spare time – no need to “get the build green” with no clear sign from where the problem comes

Compare typical and Google's CI



From https://testing.googleblog.com/2011/06/testing-at-speed-and-scale-of-google.html

Thank to Funders

HORIZON 2020

MULTI MOT

bbsrc
biotechnology and biological sciences research council

CORBEL

GLOBAL BIOIMAGING
growing collaboration

EURO-BIOIMAGING

wellcome

OME Consortium