



AARHUS UNIVERSITET

# Microservices and DevOps

DevOps and Container Technology

Scientific Process

Henrik Bærbak Christensen

# Scientific Process

*The only difference between  
screwing around and science is  
writing it down!*

- Andy Savage / MythBusters

- **Apply Science in your work!**
  - *(Most likely, you will need what you've written down at the final individual exam)*

- Observation
  - How does system behave?
- Hypothesis
  - Improve/correct by doing X
- Experiment
  - Execute
- Conclusion
  - Hypothesis is

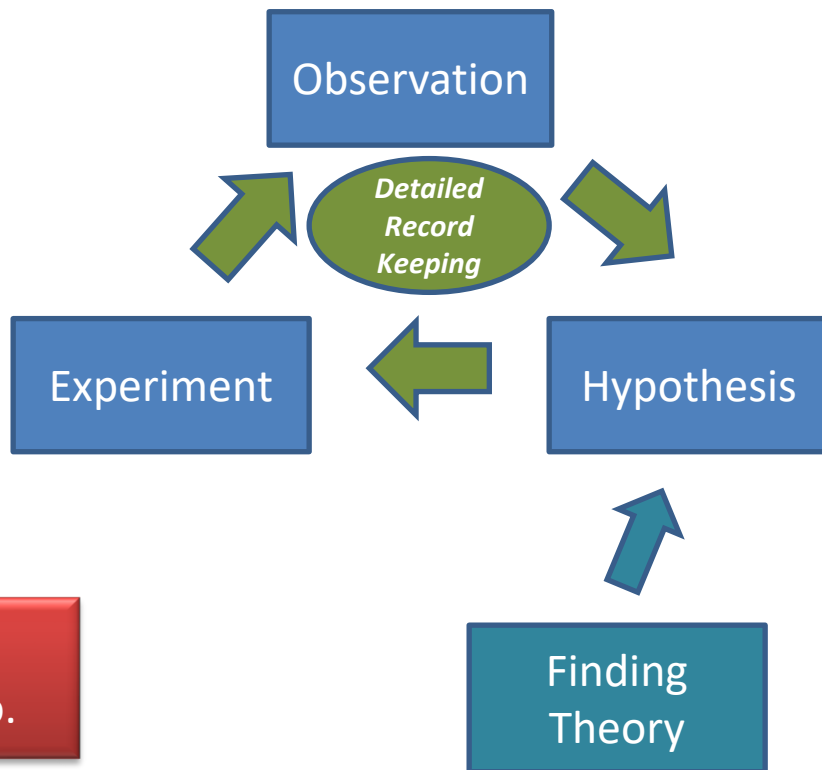
TRUE:  
Refine Hypo.

FALSE:  
Alt. Hypo.

- *Detailed Record Keeping!*

- To avoid getting lost! To track progress! To **be reproducible!**

# Scientific Process



- Keep the *diary/logbook/journal* at hand! Make **notes!**
  - **Goal, hypothesis, experiment, conclude [loop]**

```
File Edit Format View Help
Goal
=====
Can ASM find the proper type?

Experiment

I compiled the ORIGINAL code (not the sprime code) and ran Klaus ASM visitor on the
Datauploader and got:

d:\proj\Akhera-stepstone\Akhera\experiments\knh-2015-03-asm\asm-extractor>java -classpath
lib\asm-5.0.3.jar;build dk.diku.asmextractor.Extractor .\..\..\Mini-Akhera\build\org\n
et4care\Datauploader.class
Class org.net4care.datauploader
Dependson org.net4care.datauploader java.lang.Object 4
Dependson org.net4care.datauploader org.net4care.common.TheLogger 8
Dependson org.net4care.datauploader org.net4care.common.FutureResult 5
Dependson org.net4care.datauploader java.lang.StringBuilder 7
Dependson org.net4care.datauploader java.util.Iterator 4
Dependson org.net4care.datauploader java.io.PrintStream 2
Dependson org.net4care.datauploader org.net4care.Observationspecifics 2
Dependson org.net4care.datauploader org.net4care.common.Net4CareException 1
Dependson org.net4care.datauploader org.net4care.ContextCode 2
Dependson org.net4care.datauploader java.lang.String 8
Dependson org.net4care.datauploader org.net4care.StandardTelemetryObservation 4
Dependson org.net4care.datauploader java.lang.System 1
Dependson org.net4care.datauploader org.net4care.Datauploader 4

d:\proj\Akhera-stepstone\Akhera\experiments\knh-2015-03-asm\asm-extractor>

Thus - it seems that it fully qualifies all types depended upon.

Partial conclusion
=====
Does it help us?

Maybe.

Problem 1: Sprime code cannot compile, so we either have to
A) do all this analysis during the sensing phase
B) do it on the byte-code of the host system

Either way it is ... a long way...

Partial conclusion 2
=====
No - it is not enough.
```

readme - Notepad

File Edit Format View Help

01-06-2015

-----

Goal: Make sonarqube use docker based MySQL image

-----

Experiment

-----

On Rarotonga

Created a MySQLdocker VM which has docker mysql image

```
docker search mysql
docker pull mysql
```

pulls the latest version of mysql

Next, create a host directory for the mysql db files

```
mkdir home/cloud/mysql/db
```

and start the mysql container

```
docker run --name mysql -e MYSQL_ROOT_PASSWORD=wffjPZqaBp7cCeRwD8k8B -v /home/cloud/mysql/db:/var/lib/mysql
```

the logs state that the version is 5.6.24 and it is running on port 3306

Conclusion

-----

**DO OVER - wrong way to do stuff! - instead run it all in the same VM!!!**

RESTART

On the 'Cloudocker' VM

- create the data directory mysql/db in the home folder
 

```
mkdir mysql
mkdir mysql/db
```
- execute the above docker run command
 

```
docker run --name mysql -e MYSQL_ROOT_PASSWORD=wffjPZqaBp7cCeRwD8k8B -v /home/cloud/mysql/db:/var/lib/mysql
```
- verify that mysql is running
 

```
docker logs mysql
```
- start the sonar with the mysql linked
 

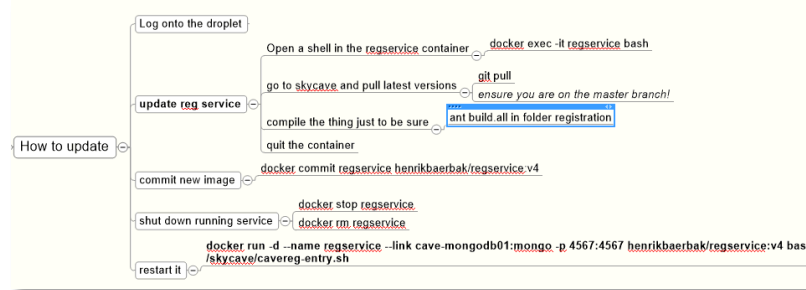
DID NOT MAKE IT WORK; perhaps look at the link

<https://github.com/SonarSource/docker-sonarqube/blob/master/recipes.md>

SonarQube setup with CloudArch Auto e

# Example: Guides (Conclusions)

- Update the Cave Subscription service (2016 version)



- All that Gradle stuff...

TheData (D:) > Dropbox > work > guides > Gradle					Search Gradle
Name	Date modified	Type	Size		
ImportGradleProjInIntelliJ	30-04-2018 13:55	File folder			
AvoidGradleOutputInShell	07-05-2019 16:23	Text Document	1 KB		
BuildRunnableJAR	29-05-2019 15:26	Text Document	3 KB		
CreateBinTrayLibrariesAndUseThem	11-10-2018 13:22	Text Document	2 KB		
GetSystemIntoCLitool	13-12-2017 09:39	Text Document	1 KB		
GradleWrapperUpdate	17-05-2018 13:46	Text Document	1 KB		
HangingDaemons	28-08-2018 13:13	Text Document	1 KB		
IncludeLocalJarLikeMinidraw	02-10-2017 20:24	Text Document	1 KB		
InitNewProjectUsingGradle	17-05-2018 13:47	Text Document	1 KB		
IntelliJGradle	30-08-2017 13:50	Text Document	1 KB		
MultiProjectGradle	19-12-2017 16:09	Text Document	1 KB		
MultiProjectTestDependenciesGradle	14-03-2018 14:07	Text Document	1 KB		
ReadResourceFilesGradleStructure	13-05-2019 13:50	Text Document	2 KB		
ResourceHandling	01-06-2018 16:30	Text Document	1 KB		
RunningApplications	07-07-2019 13:41	Text Document	1 KB		

```

CreateBinTrayLibrariesAndUseThem - Notepad
File Edit Format View Help
Bintray library publishing and retrieval
=====

From programmingkata/upload-maven-central:
Overview of usage:
----

For Gradle, add the following to the build.gradle

repositories {
    jcenter()
    maven {
        url 'https://dl.bintray.com/henrikbaerbak/maven'
    }
}

As I publish to JCenter the 'maven' section is probably not even required.
Dependencies follow the normal scheme

compile 'henrikbaerbak:minidraw:1.11'

For Ivy, you have to fiddle with the ivysettings.xml file, which is
tedious; but have a look in the programmingkata/upload-maven-central
kata. You have to set the resolvers!

<!-- The henrikbaerbak Bintray modules at CS, Aarhus university -->
<resolvers>
  <bintray name="hbc" subject="henrikbaerbak" repo="maven"/>
</resolvers>

overview of upload:
----

1. Gradlify the library code and prepare for publishing. In 'build.gradle'
2. update to gradlew version 4.6
3. plugins 'java-library' and 'maven-publish'
4. define sourceJar and javadocJar
5. define maven publication details
  
```

- Detailed record keeping saves your sanity!
  - Christmas vacation: MQ update lead to total reinstall
    - Guides helped me reestablish full working environment
  - Extending Mongo DB disks
    - Involves repeating 10-15 steps in three differently configured machines
    - *And a wrong step may crash 6 TB data* 😞

# Test-Driven-Development

- Is basically a natural science process

But there is more. When we embrace hypothesis-driven development, we can learn how technology works together, or not, and what our customers need and want. We also complement the test-driven development (TDD) principle. TDD encourages us to write the test first (hypothesis), then confirm our features are correct (experiment), and succeed or fail the test (evaluate). ***It is all about quality and delighting our users, as outlined in principles 1, 3, and 7 of the Agile Manifesto:***

- Our highest priority is to satisfy the customers through early and continuous delivery of value.
- Deliver software often, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
- Working software is the primary measure of progress.

Reed, 2019