6th UCAAT User Conference on Advanced Automated Testing

ETSI

Paris, 16-18 October 2018

Organizer: TESTING SOLUTIONS & SERVICES

Jenkins pipelines

Presented by Pierre-Henri Symoneaux

NOKIA

Introduction

- Who am I
  - Pierre-Henri Symoneaux
  - Nokia France
  - SW architecture & development (Cloud Core for 5G Mobile Networks)
- The topic
  - Problems of « classical » Jenkins
  - Jenkins pipelines: What, why and how
  - Based on feedback from real usage

6th UCAAT User Conference on Advanced Automated Testing

ETSI

Paris, 16-18 October 2018

Organizer: TESTING SOLUTIONS &SERVICES

Jenkins

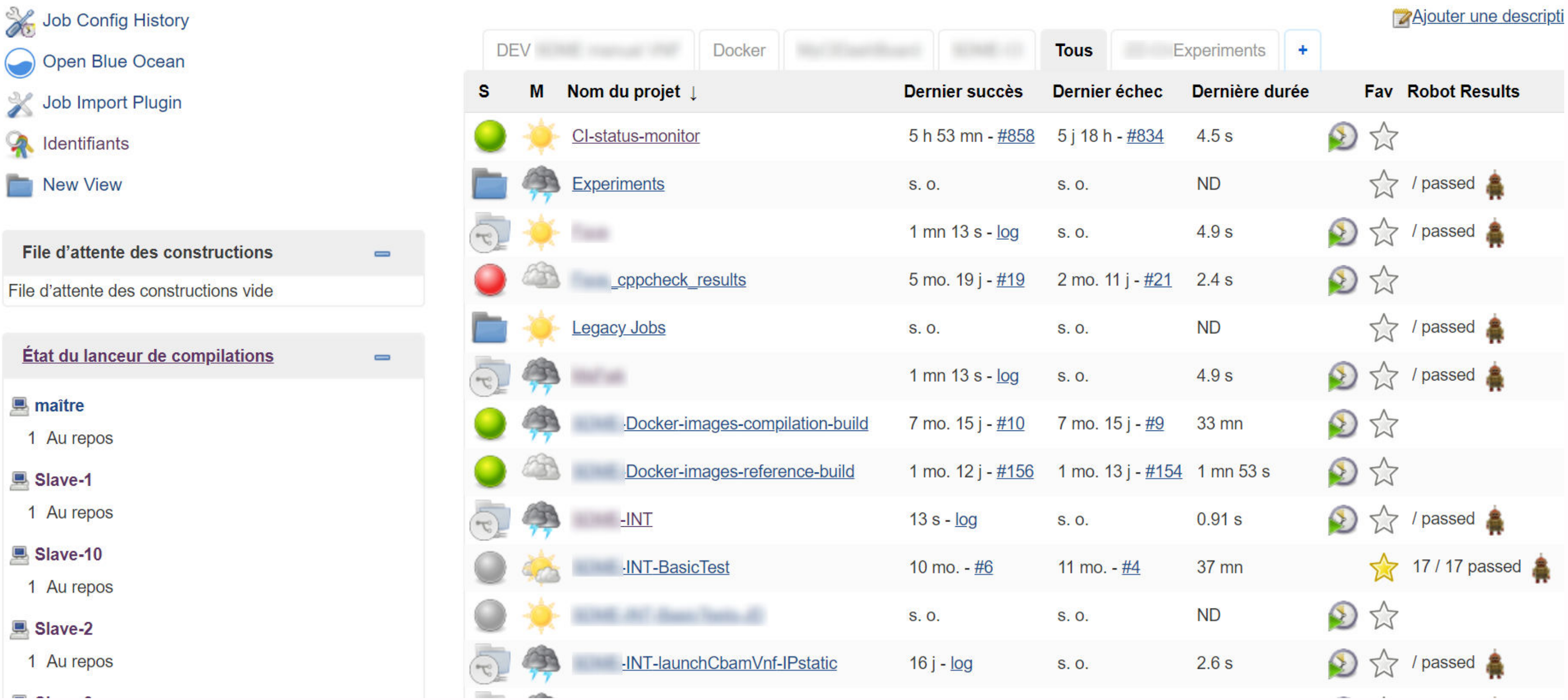
A quick reminder

What is Jenkins

- An open-source automation server
- Extensible with hundreds of plugins
- Distributed – Jobs (may) run in slaves
- Build, test, package, deploy. Automate anything
- Mainly used in software industry
  - Continuous Integration (CI)
  - Continuous Delivery (CD)
  - DevOps
- But not only

6th UCAAT User Conference on Advanced Automated Testing

ETSI The Standards People



- Job Config History
- Open Blue Ocean
- Job Import Plugin
- Identifiants
- New View

File d'attente des constructions

File d'attente des constructions vide

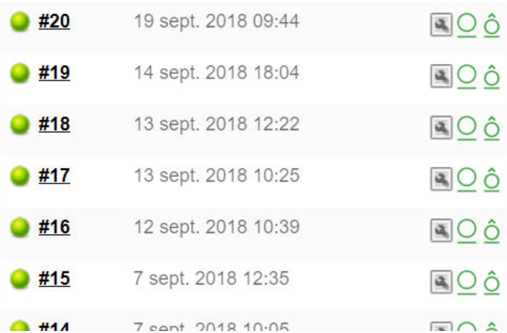
État du lanceur de compilations

- **maître**
  - 1 Au repos
- **Slave-1**
  - 1 Au repos
- **Slave-10**
  - 1 Au repos
- **Slave-2**
  - 1 Au repos

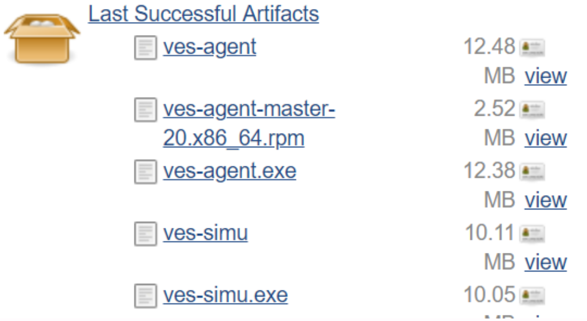
Ajouter une descripti

DEV [redacted] | Docker | [redacted] | [redacted] | **Tous** | [redacted]Experiments | +

S	M	Nom du projet ↓	Dernier succès	Dernier échec	Dernière durée	Fav	Robot Results
🟢	☀	CI-status-monitor	5 h 53 mn - #858	5 j 18 h - #834	4.5 s	☆	
📁	⛈	Experiments	s. o.	s. o.	ND	☆	/ passed
	☀	[redacted]	1 mn 13 s - log	s. o.	4.9 s	☆	/ passed
🔴	☁	[redacted]_cppcheck_results	5 mo. 19 j - #19	2 mo. 11 j - #21	2.4 s	☆	
📁	☀	Legacy Jobs	s. o.	s. o.	ND	☆	/ passed
	⛈	[redacted]	1 mn 13 s - log	s. o.	4.9 s	☆	/ passed
🟢	⛈	[redacted]-Docker-images-compilation-build	7 mo. 15 j - #10	7 mo. 15 j - #9	33 mn	☆	
🟢	☁	[redacted]-Docker-images-reference-build	1 mo. 12 j - #156	1 mo. 13 j - #154	1 mn 53 s	☆	
	⛈	[redacted]-INT	13 s - log	s. o.	0.91 s	☆	/ passed
⚪	⛅	[redacted]-INT-BasicTest	10 mo. - #6	11 mo. - #4	37 mn	★	17 / 17 passed
⚪	☀	[redacted]	s. o.	s. o.	ND	☆	
	⛈	[redacted]-INT-launchCbamVnf-IPstatic	16 j - log	s. o.	2.6 s	☆	/ passed

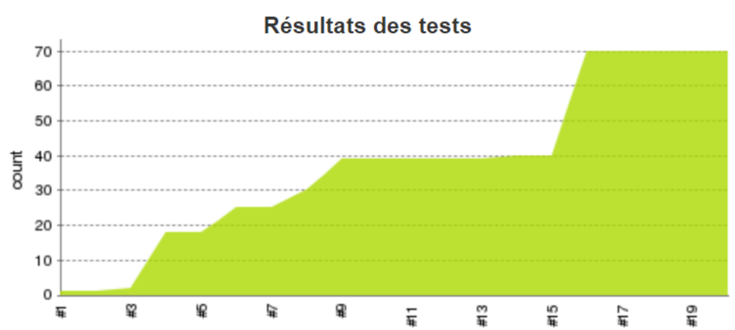


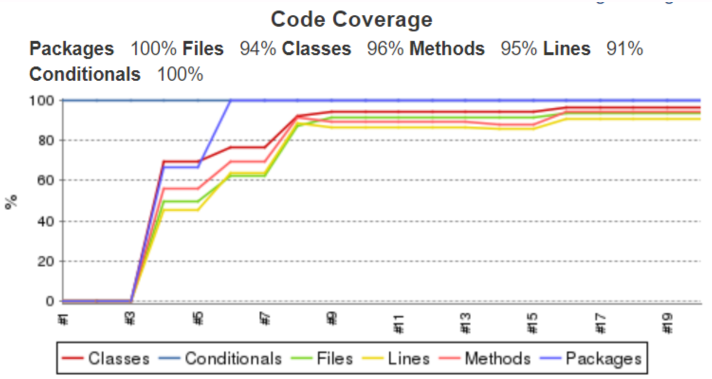
#20	19 sept. 2018 09:44	
#19	14 sept. 2018 18:04	
#18	13 sept. 2018 12:22	
#17	13 sept. 2018 10:25	
#16	12 sept. 2018 10:39	
#15	7 sept. 2018 12:35	
#14	7 sept. 2018 10:05	



Last Successful Artifacts

ves-agent	12.48 MB view
ves-agent-master-20.x86_64.rpm	2.52 MB view
ves-agent.exe	12.38 MB view
ves-simu	10.11 MB view
ves-simu.exe	10.05 MB view





Packages 100% **Files** 94% **Classes** 96% **Methods** 95% **Lines** 91% **Conditionals** 100%

6th UCAAT User Conference on Advanced Automated Testing

ETSI

Paris, 16-18 October 2018

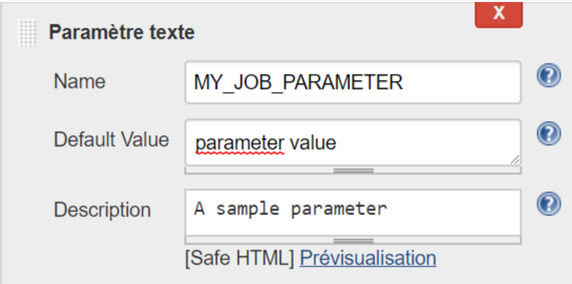
Organizer: TESTING SOLUTIONS &SERVICES

Jenkins – The “old” way

And its issues

Freestyle jobs – the « old » way

- Jobs are fully defined in Jenkins web-UI
  - Input Parameters

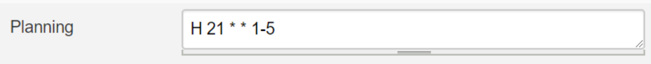


Paramètre texte

Name	MY_JOB_PARAMETER
Default Value	parameter value
Description	A sample parameter

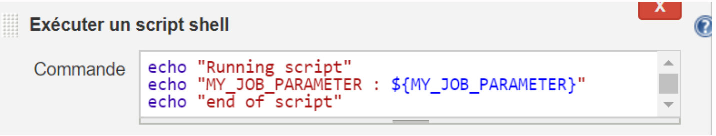
[Safe HTML] Prévisualisation

  - Triggers / scheduling



Planning	H 21 * * 1-5

  - Scripts



Exécuter un script shell

Commande

```
echo "Running script"
echo "MY_JOB_PARAMETER : ${MY_JOB_PARAMETER}"
echo "end of script"
```

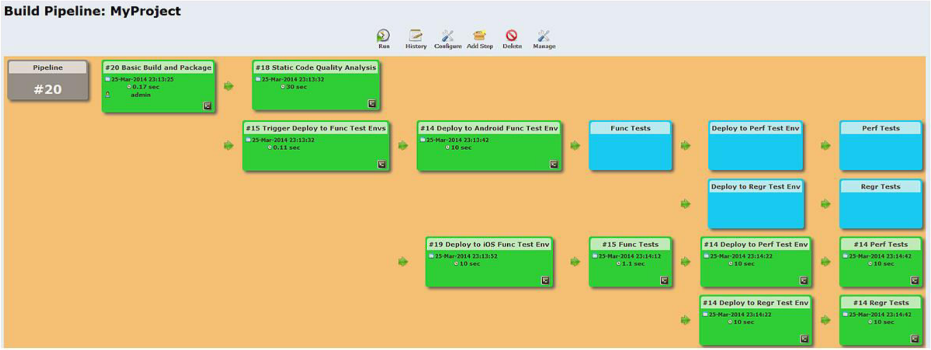
  - Post actions (archive artefacts, publish results & graphs)
  - More …

Problems

- As a job grows, it will become
  - Hard to maintain
  - Hard to understand
  - Hard to troubleshoot
- Hard to track changes in a job
- Cannot review changes in a job before applying
- What if many people perform changes at the same time
- Cannot replay an old job

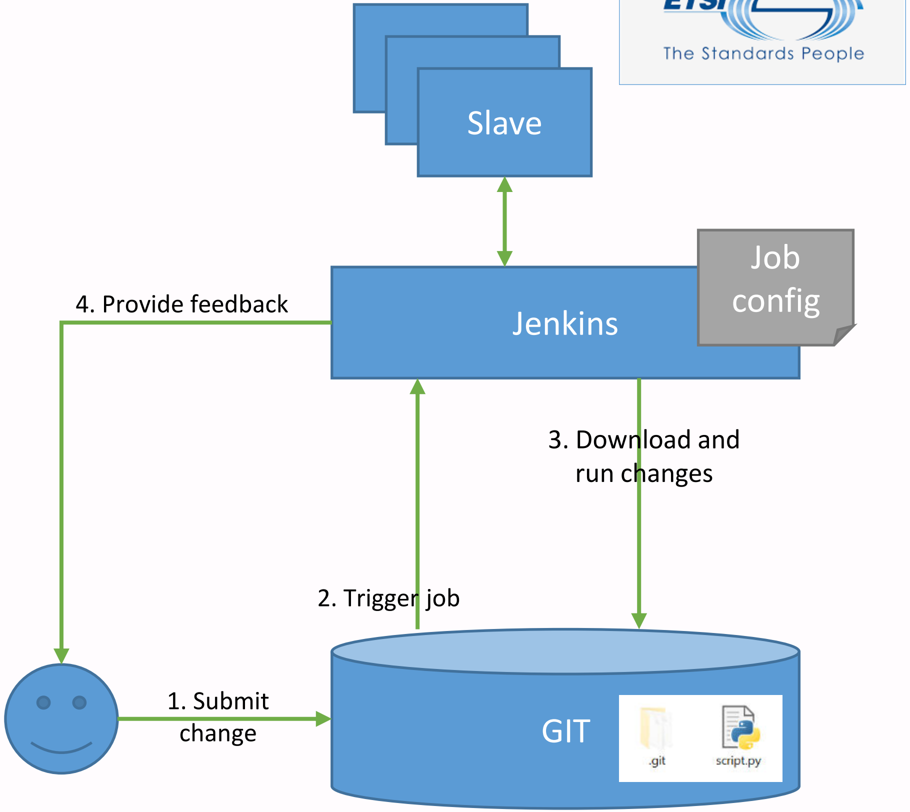
Multijob Pipelines – the « old » way

- Split work into multiple jobs
  - Jobs trigger each other
  - Introduce dependencies between job
  - Better view on each steps
- Problems
  - Increased complexity (hard to maintain)
  - Tracking jobs defininition changes is even harder



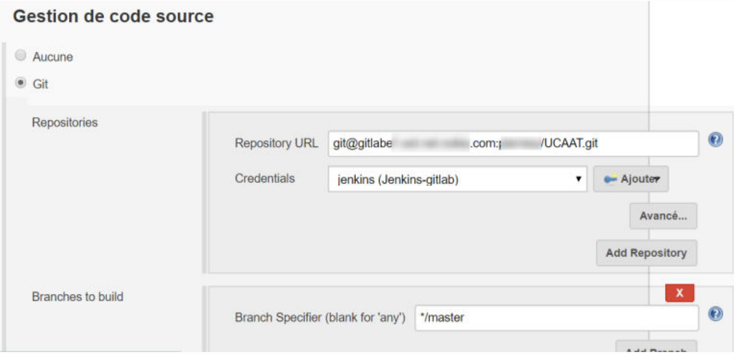
Integration with SCM

- Most of the time, Jenkins is coupled with an SCM (GIT, SVN, …)
  - To store tested code
  - To store testing code
  - Both together
- New changes in SCM can trigger a job
- Keep track of changes in scripts
- Changes can be reviewed before integration



Integration with SCM

Add SCM configuration



Gestion de code source

○ Aucune

● Git

Repositories

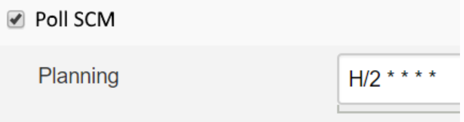
Repository URL: git@gitlabe[redacted].com:[redacted]/UCAAT.git

Credentials: jenkins (Jenkins-gitlab)

Branches to build

Branch Specifier (blank for 'any'): */master

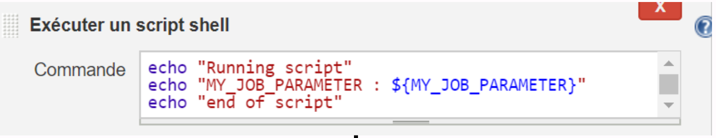
Add new trigger: Poll SCM for changes



☑ Poll SCM

Planning: H/2 * * * *

Update script to use files from SCM

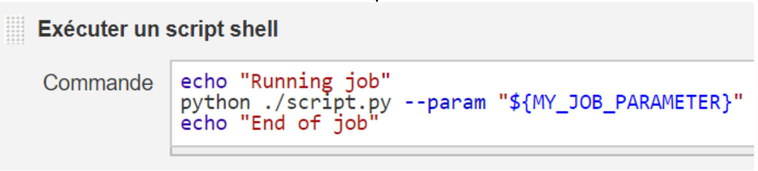


Exécuter un script shell

Commande

```
echo "Running script"
echo "MY_JOB_PARAMETER : ${MY_JOB_PARAMETER}"
echo "end of script"
```

Becomes



Exécuter un script shell

Commande

```
echo "Running job"
python ./script.py --param "${MY_JOB_PARAMETER}"
echo "End of job"
```

Problems

- Old version can be re-executed: But only with current job definition
- Job definition still in Jenkins
- What if breaking changes are introduced
  - Eg: `python ./script.py --param ${…} --newparam ${…}`
  - Or a new script is invoked
    - → Job needs an update
    - → Cannot run old versions anymore (incompatibility introduced)
- What about execution environment ? (eg: migrate from python 2.7 to python 3.6)

6th UCAAT

User Conference on Advanced Automated Testing

ETSI

Paris, 16-18 October 2018

Organizer: TESTING SOLUTIONS & SERVICES

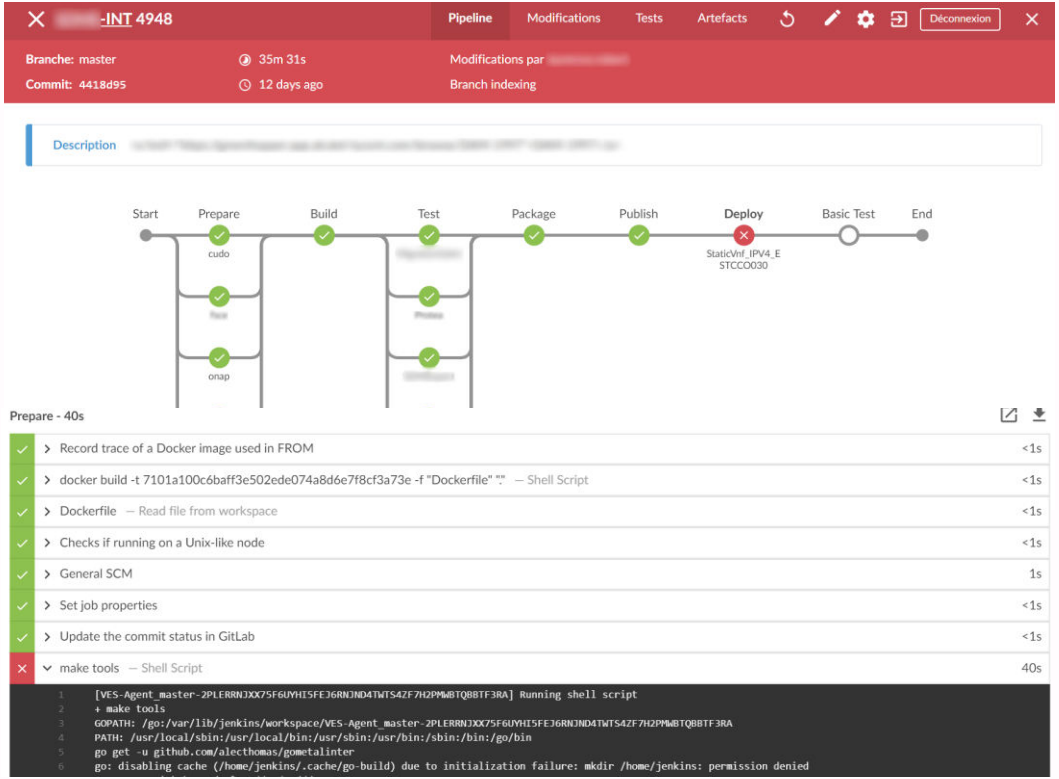
Jenkins Blue Ocean

A new way to write pipeline

NOKIA

What is Blue Ocean

- A Jenkins plugin
- Appeared in 2016 – Still in early stage
- Rethinks user experience
  - New UI (classical UI still available)
  - New syntax: The whole job is a script
  - Pipeline graphical editor
- Designed for pipelines
  - Sophisticated pipeline visualization
  - Pinpoint precision
- CI / CD as code
- Modular with shared pipeline libraries
- First class integration with Docker



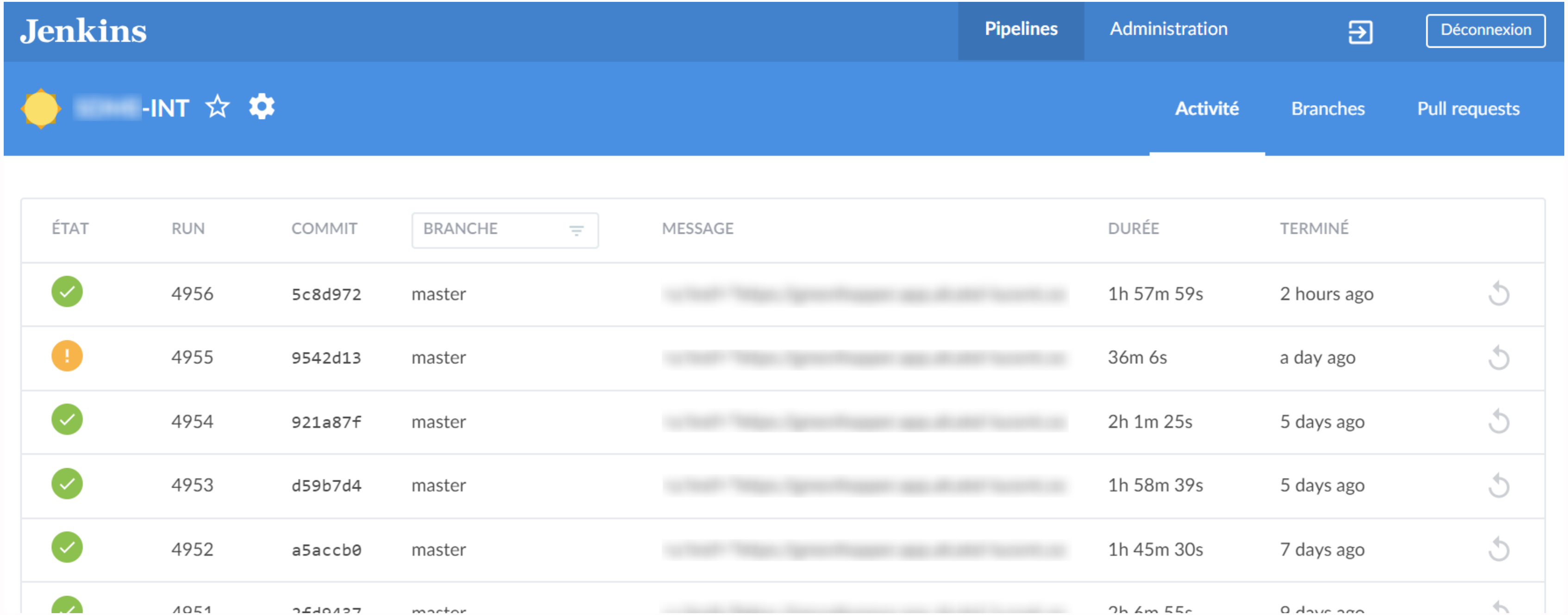
Branche: master
Commit: 4418d95
35m 31s
12 days ago
Modifications par [redacted]
Branch indexing

Description [redacted]

Prepare - 40s

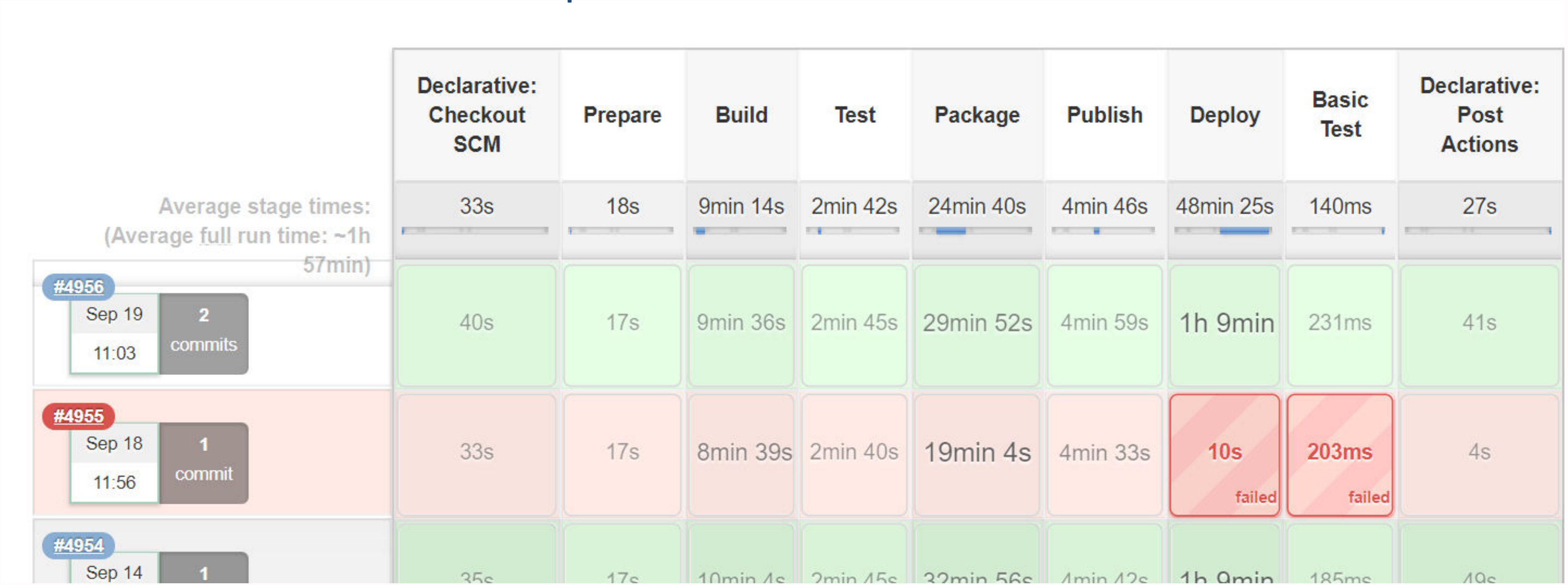
✓	Record trace of a Docker image used in FROM	<1s
✓	docker build -t 7101a100c6baff3e502ede074a8d6e7f8cf3a73e -f "Dockerfile" "." — Shell Script	<1s
✓	Dockerfile — Read file from workspace	<1s
✓	Checks if running on a Unix-like node	<1s
✓	General SCM	1s
✓	Set job properties	<1s
✓	Update the commit status in GitLab	<1s
✕	make tools — Shell Script	40s

```
[VES-Agent_master-2PLERRNJXX75F6UYHI5FEJ6RNJND4TWTS4ZF7H2PMWBTQBBTF3RA] Running shell script
+ make tools
GOPATH: /go:/var/lib/jenkins/workspace/VES-Agent_master-2PLERRNJXX75F6UYHI5FEJ6RNJND4TWTS4ZF7H2PMWBTQBBTF3RA
PATH: /usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/go/bin
go get -u github.com/alecthomas/gometalinter
go: disabling cache (/home/jenkins/.cache/go-build) due to initialization failure: mkdir /home/jenkins: permission denied
```



ÉTAT	RUN	COMMIT	BRANCHE	MESSAGE	DURÉE	TERMINÉ	
✓	4956	5c8d972	master	[redacted]	1h 57m 59s	2 hours ago	↻
!	4955	9542d13	master	[redacted]	36m 6s	a day ago	↻
✓	4954	921a87f	master	[redacted]	2h 1m 25s	5 days ago	↻
✓	4953	d59b7d4	master	[redacted]	1h 58m 39s	5 days ago	↻
✓	4952	a5accb0	master	[redacted]	1h 45m 30s	7 days ago	↻
✓	4951	2fd9437	master	[redacted]	2h 6m 55s	9 days ago	↻

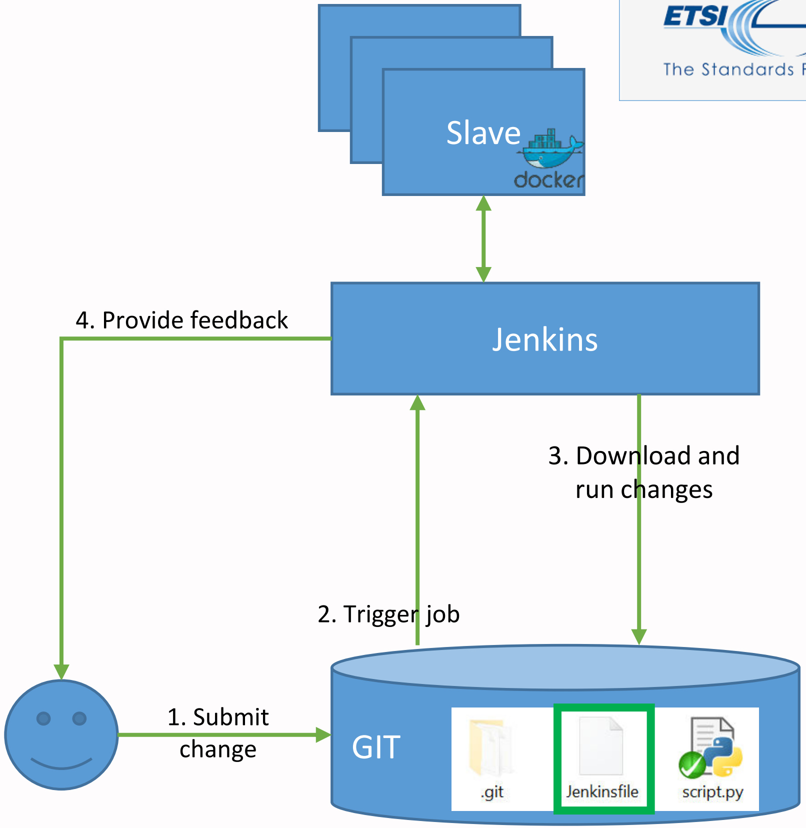
Classical Jenkins UI is also updated



	Declarative: Checkout SCM	Prepare	Build	Test	Package	Publish	Deploy	Basic Test	Declarative: Post Actions
Average stage times: (Average full run time: ~1h 57min)	33s	18s	9min 14s	2min 42s	24min 40s	4min 46s	48min 25s	140ms	27s
#4956 Sep 19 11:03 2 commits	40s	17s	9min 36s	2min 45s	29min 52s	4min 59s	1h 9min	231ms	41s
#4955 Sep 18 11:56 1 commit	33s	17s	8min 39s	2min 40s	19min 4s	4min 33s	10s failed	203ms failed	4s
#4954 Sep 14 1	35s	17s	10min 4s	2min 45s	32min 56s	4min 42s	1h 9min	185ms	49s

Pipeline as code

- With blue ocean, Job/Pipeline definition is also stored in SCM
- Pipeline can run in a dedicated Docker container
- Each Git branch will automatically have its own job



6th UCAAT User Conference on Advanced Automated Testing

ETSI

Paris, 16-18 October 2018

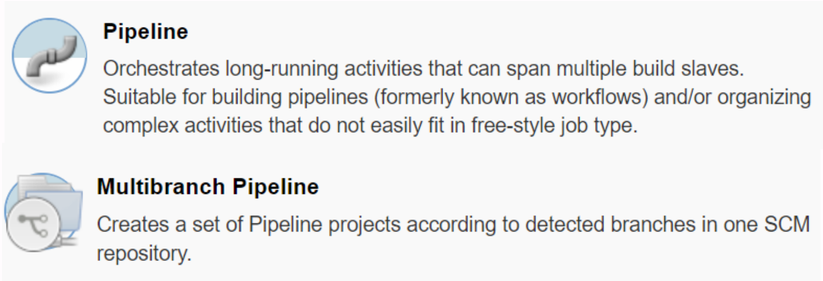
Organizer: TESTING SOLUTIONS & SERVICES

Jenkins Blue Ocean

Creating a pipeline

Setting up the pipeline

- Install **blueocean** plugin
- Create a new job
  - Choose type of job
    - Pipeline
    - Multibranch Pipeline
- Let’s choose Multibranch Pipeline



Pipeline

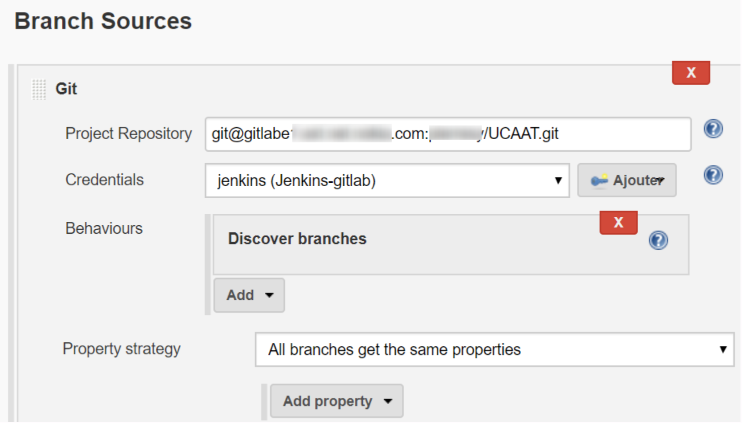
Orchestrates long-running activities that can span multiple build slaves. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

Multibranch Pipeline

Creates a set of Pipeline projects according to detected branches in one SCM repository.

Setting up the pipeline

- Setup source SCM with branch discovery
- Setup branch scanning



Branch Sources

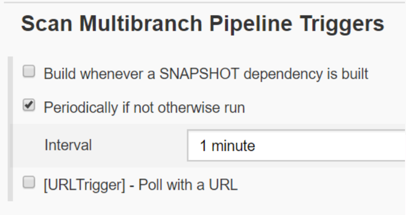
Git

Project Repository: git@gitlabe[redacted].com:[redacted]/UCAAT.git

Credentials: jenkins (Jenkins-gitlab)

Behaviours: **Discover branches**

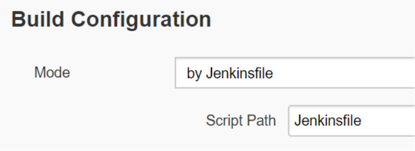
Property strategy: All branches get the same properties



Scan Multibranch Pipeline Triggers

- □ Build whenever a SNAPSHOT dependency is built
- ☑ Periodically if not otherwise run
  - Interval: 1 minute
- □ [URLTrigger] - Poll with a URL

- Set path to pipeline file
- Prepare your Jenkinsfile
- Commit and push it to SCM



Build Configuration

Mode: by Jenkinsfile

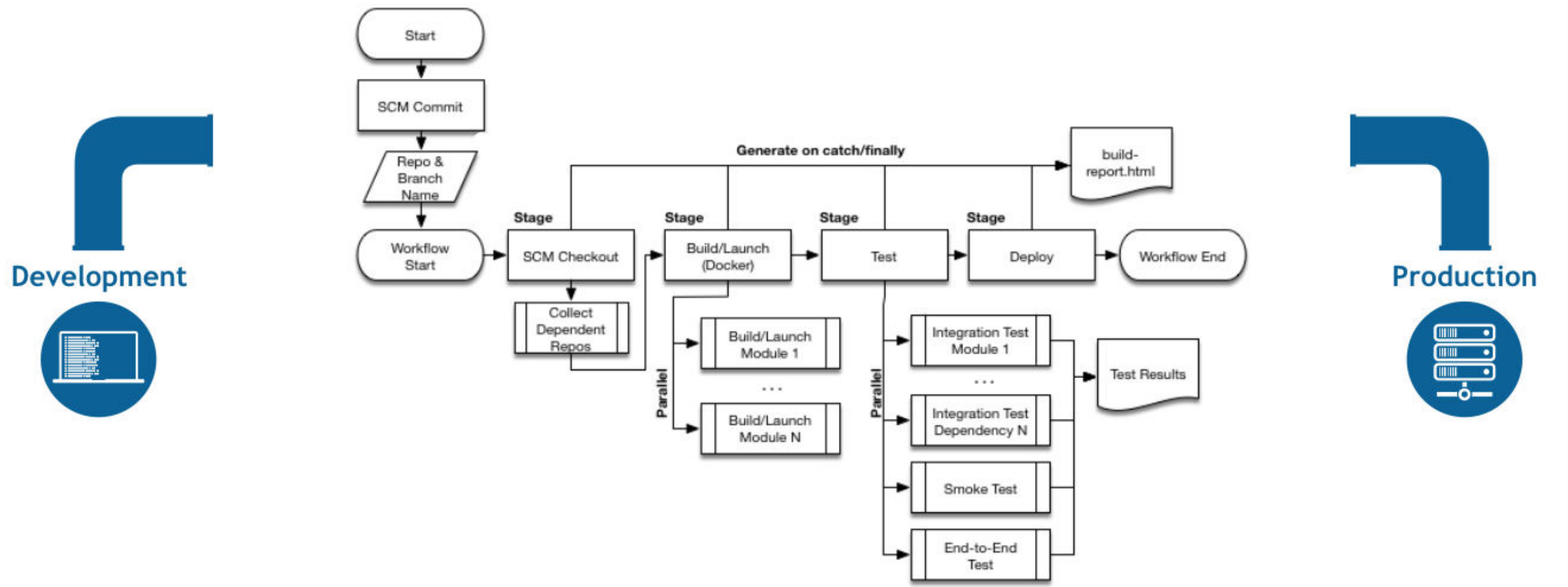
Script Path: Jenkinsfile

Jenkinsfile is the default name

The Jenkinsfile

- A text file
- Checked into SCM
- Declarative pipeline syntax
- DSL base on Groovy language
- Structure documented at https://jenkins.io/doc/book/pipeline/syntax/
- A single source of truth for the pipeline
  - Can be viewed and edited by multiple members of the project

Anatomy of a pipeline



Development

Start

SCM Commit

Repo & Branch Name

Workflow Start

Stage

SCM Checkout

Collect Dependent Repos

Stage

Build/Launch (Docker)

Parallel

Build/Launch Module 1

. . .

Build/Launch Module N

Stage

Test

Parallel

Integration Test Module 1

. . .

Integration Test Dependency N

Smoke Test

End-to-End Test

Test Results

Stage

Deploy

Workflow End

Generate on catch/finally

build-report.html

Production

Structure of a jenkinsfile

- A Jenkinsfile has 6 main sections
  - **Agent :** Specifies where the entire pipeline will run
  - **Options** : Global options
  - **Parameters**: Input parameters
  - **Environment** : Global environment variables
  - **Stages :** Sequence of stage definitions
  - **Post :** Steps to be run at the end of pipeline

```
pipeline {
     agent {/*...*/}
     options {
          timeout(time: 1, unit: 'HOURS')
          retry(3)
     }
     parameters {
          string(name: 'MY_JOB_PARAMETER’,
                 defaultValue: '<NONE>’,
                 description: 'Job parameter’
                 )
     }
     environment {
          MY_ENV_VARIABLE = "foobar"
     }
     stages {/*...*/}
     post {/*...*/}
}
```

Agent

- Defines where to run the pipeline
  - In any slave
  - In a slave with a given label
  - In docker container
    - Either from an image
    - Or built from a Dockerfile
- Docker makes managing running environments a piece of cake

```
agent any

agent {
      label "slave-with-python2.7"
}

agent {
   docker {
      image: "python:2.7"
   }
}

agent {dockerfile true}
```

Post

- Perform steps at end of pipeline
  - Archive artifacts
  - Publish result
  - Send an email
  - Etc …
- Actions can be conditionned by pipeline status
  - Always
  - Changed
  - Fixed
  - Regression
  - Aborted
  - Failure
  - Success
  - Unstable
  - Cleanup

```
post {
    always {
        archive "build/*.exe"
        deleteDir()
    }
    failure {
        echo "Failure"
    }
    success {
        echo "Success"
    }
    unstable {
        echo "Unstable"
    }
}
```

Stages

- Each stage has either
    - A sequence of steps
    - A list of parallel stages
- Can have conditional switch
- Can have their own environment variables
- Can have their own agent

```
stages {
        stage("Stage-1") {
                steps {
                        echo "Welcome in stage 1"
                        sh "python script.py"
                        sh “./script.sh"
                }
        }
        stage('Stage-2') {
                when {
                                branch "master"
                }
                environment {
                                MY_VARIABLE = "My-Value"
                }
                parallel {
                        stage("Sub-stage-1") {
                                steps {
                                                echo "sub stage 1"
                                }
                        }
                        stage("Sub-stage-2") {
                                steps {
                                                echo "sub stage 2"
                                }
                        }
                }
        }
}
```

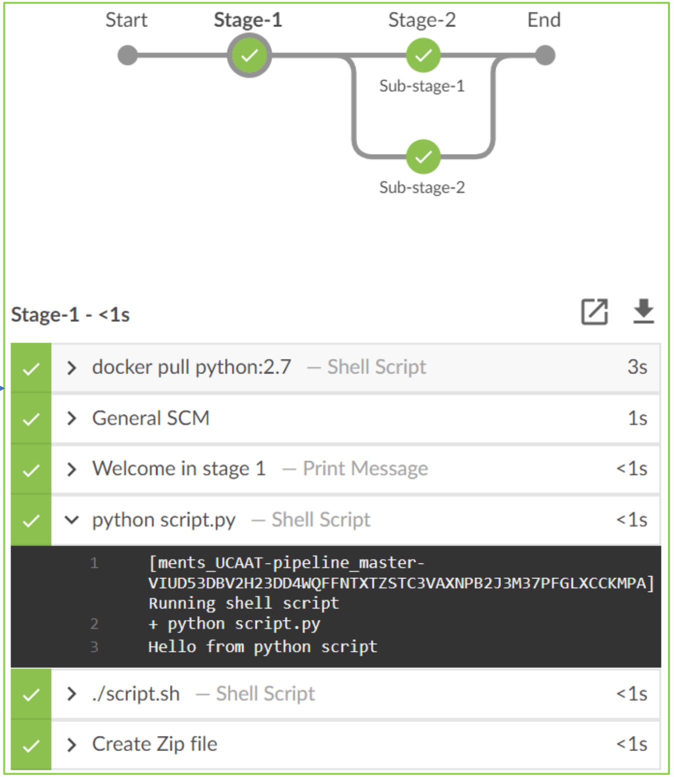
Steps

- A step is a single action
- Jenkins plugins come with their own steps
- Run sequentially in a stage
- Each step has its log output
- Full list available at https://jenkins.io/doc/pipeline/steps/

```
steps {
     addBadge icon: 'computer.png', text: env.NODE_NAME
     echo "Welcome in stage 1"
     sh "python script.py"
     sh "./script.sh"
     zip zipFile: "compressed.zip", dir: "."
}
```

```
post {
     always {
          junit "build/testresults.xml"
          checkstyle pattern: 'build/checkstyle.xml'
          cobertura coberturaReportFile: 'build/coverage.xml'
          sloccountPublish pattern: 'build/sloccount.scc'
          archive "build/*.exe,build/*.rpm"
          deleteDir() // Delete workspace
     }
}
```

```
stages {
      stage("Stage-1") {
            steps {
                  echo "Welcome in stage 1"
                  sh "python script.py"
                  sh “./script.sh“
                  zip zipFile: "compressed.zip", dir: "."
            }
      }
      stage('Stage-2') {
            when {
                        branch "master"
            }
            environment {
                        MY_VARIABLE = "My-Value"
            }
            parallel {
                  stage("Sub-stage-1") {
                        steps {
                                    echo "sub stage 1"
                        }
                  }
                  stage("Sub-stage-2") {
                        steps {
                                    echo "sub stage 2"
                        }
                  }
            }
      }
}
```



Stage-1 - <1s

✓ › docker pull python:2.7 — Shell Script 3s

✓ › General SCM 1s

✓ › Welcome in stage 1 — Print Message <1s

✓ ⌄ python script.py — Shell Script <1s

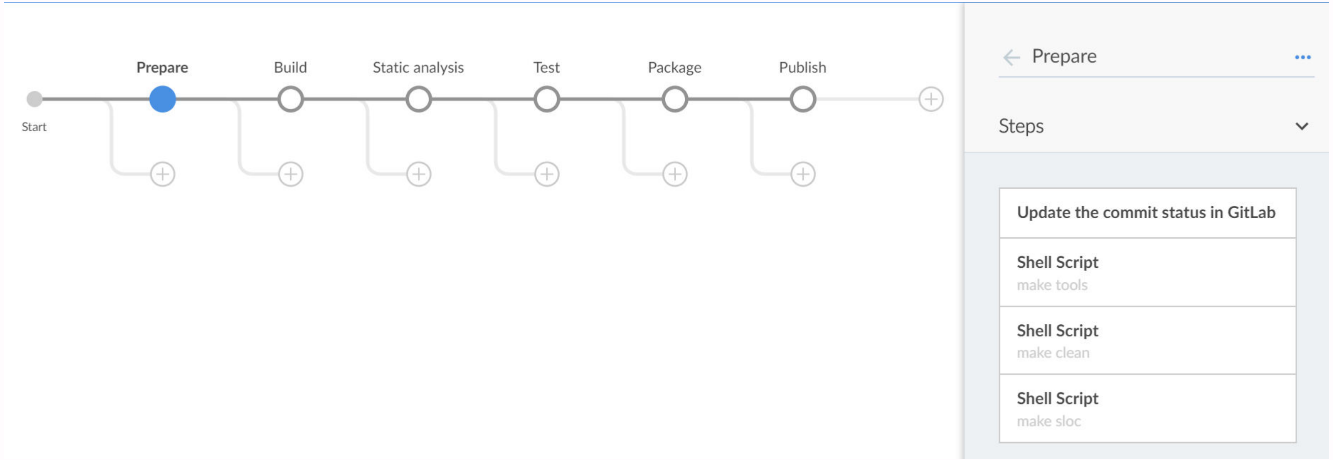
```
[ments_UCAAT-pipeline_master-
VIUD53DBV2H23DD4WQFFNTXTZSTC3VAXNPB2J3M37PFGLXCCKMPA]
Running shell script
+ python script.py
Hello from python script
```

✓ › ./script.sh — Shell Script <1s

✓ › Create Zip file <1s

Pipeline editor

- Graphical tool
- Edit Jenkinsfile
- Makes it less difficult
- Not as powerfull as text edition (yet?)



Update the commit status in GitLab

Shell Script
make tools

Shell Script
make clean

Shell Script
make sloc

6th UCAAT User Conference on Advanced Automated Testing

ETSI

Paris, 16-18 October 2018

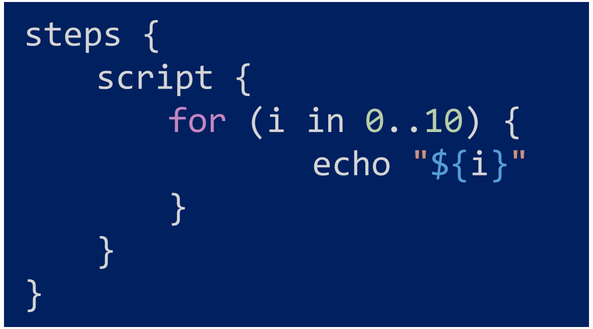
Organizer: TESTING SOLUTIONS &SERVICES

Jenkins Blue Ocean

Advanced scripting

The **script** step

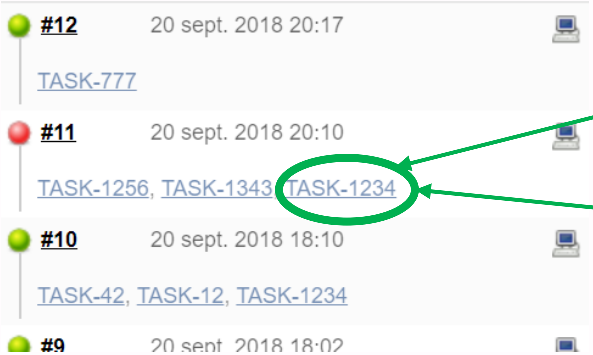
- Takes a block of groovy script
- Mostly an “escape hatch”
- Has access to Jenkins’ internal functions
- Has access to Java/Groovy standard library
- Run in a sandbox
- Big scripts should go into a shared library



```
steps {
    script {
        for (i in 0..10) {
                echo "${i}"
        }
    }
}
```

Example

- Extract JIRA task ID from change's comment
- Display the link in jenkins's job history

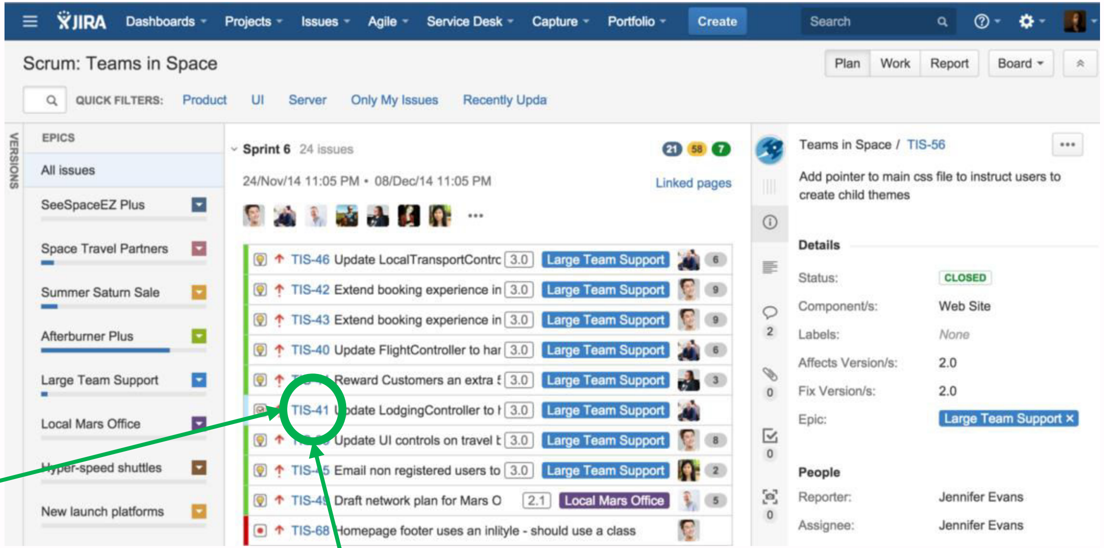


#12 20 sept. 2018 20:17
TASK-777

#11 20 sept. 2018 20:10
TASK-1256, TASK-1343, TASK-1234

#10 20 sept. 2018 18:10
TASK-42, TASK-12, TASK-1234

#9 20 sept. 2018 18:02



Scrum: Teams in Space

QUICK FILTERS: Product UI Server Only My Issues Recently Upda

EPICS: All issues, SeeSpaceEZ Plus, Space Travel Partners, Summer Saturn Sale, Afterburner Plus, Large Team Support, Local Mars Office, Hyper-speed shuttles, New launch platforms

Sprint 6 24 issues 21 58 7

24/Nov/14 11:05 PM • 08/Dec/14 11:05 PM

TIS-46 Update LocalTransportContrc 3.0 Large Team Support 6
TIS-42 Extend booking experience in 3.0 Large Team Support 9
TIS-43 Extend booking experience in 3.0 Large Team Support 9
TIS-40 Update FlightController to har 3.0 Large Team Support 6
TIS-4 Reward Customers an extra 3.0 Large Team Support 3
TIS-41 Update LodgingController to l 3.0 Large Team Support
TIS- Update UI controls on travel t 3.0 Large Team Support 8
TIS-5 Email non registered users to 3.0 Large Team Support 2
TIS-4 Draft network plan for Mars O 2.1 Local Mars Office 5
TIS-68 Homepage footer uses an inlityle - should use a class

Teams in Space / TIS-56

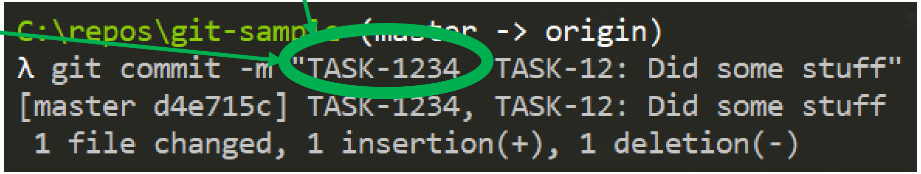
Add pointer to main css file to instruct users to create child themes

Details

Status: CLOSED
Component/s: Web Site
Labels: None
Affects Version/s: 2.0
Fix Version/s: 2.0
Epic: Large Team Support ×

People

Reporter: Jennifer Evans
Assignee: Jennifer Evans



```
C:\repos\git-sample (master -> origin)
λ git commit -m "TASK-1234 TASK-12: Did some stuff"
[master d4e715c] TASK-1234, TASK-12: Did some stuff
 1 file changed, 1 insertion(+), 1 deletion(-)
```

Example

```
script {
    def issues = currentBuild.changeSets
        .collect { c -> c.getItems() }.flatten() // Build a single list with all changesets
        .collect{ c -> c.getMsg() } // Transform the list into a list of commit message
        .collect { msg -> msg.split(':')[0].split(',').collect { it.trim() } } // Extract issues id from
        each commit message
        .flatten() // Merge into a single list
        .findAll { task -> task ==~ /TASK-[0-9]+/ } // Keep only valid issue names
        .unique() // Remove duplicates

    currentBuild.description = issues
        .collect { "<a href=\"https://your-jira.server.com/browse/${it}\">${it}</a>" }
        .join(", ")
}
```

6th UCAAT

User Conference on Advanced Automated Testing

ETSI

Paris, 16-18 October 2018

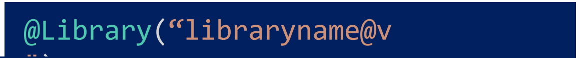
Organizer: TESTING SOLUTIONS & SERVICES

Don’t repeat yourself

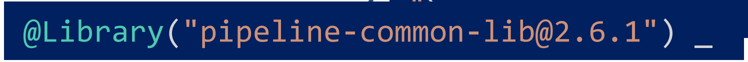
Introduction to shared pipeline libraries

NOKIA

Shared pipeline librarie

- Store subset of pipeline code in separate SCM repository
- Share this code between multiple projects
- Create custom steps
- Avoid script sandbox restriction (a shared library is trusted)
- Imported in Jenkinsfile by 

```
@Library(“libraryname@v
```

  - EG: 

```
@Library("pipeline-common-lib@2.6.1") _
```

  - Version can be the branch name, a tag, or a revision ID
- Check documentation at https://jenkins.io/doc/book/pipeline/shared-libraries/

Why

- Jenkinsfile gets bigger and bigger
- Some parts are common to many projects
- Implements complex steps
- Import and use java libraries
- DRY (Don’t Repeat Yourself)

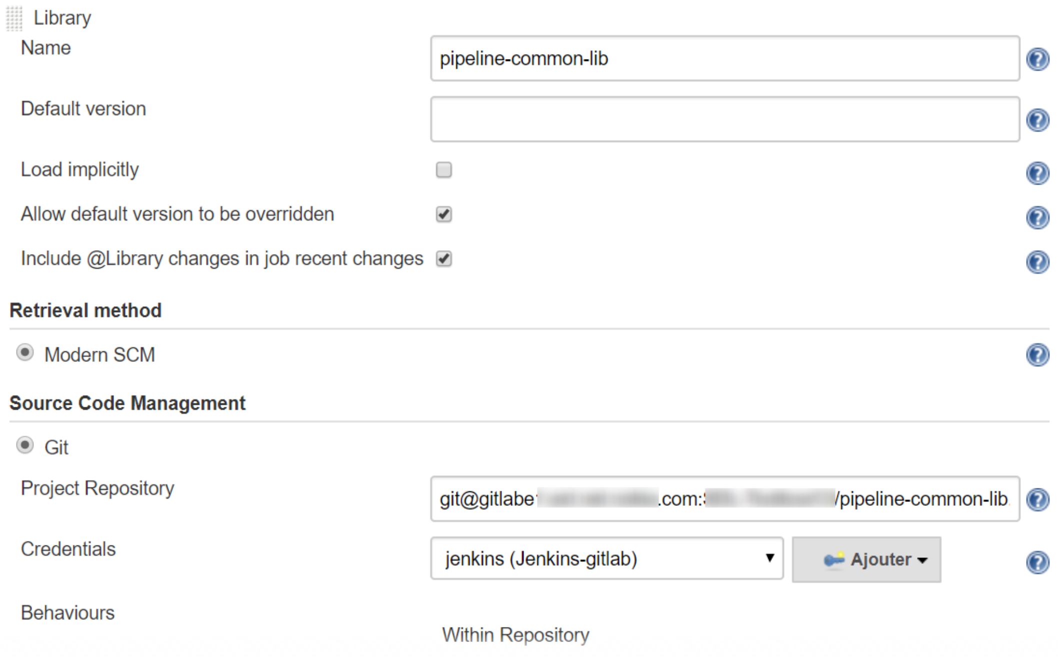
Let’s refactor the JIRA link script

- Create a new GIT repository which will hold the library code
- Create a file **./vars/linkToJira.groovy** with the following content

```
def call(prefix, baseUrl) {
     def issues = currentBuild.changeSets
          .collect { c -> c.getItems() }.flatten() // Build a single list with all changesets
          .collect{ c -> c.getMsg() } // Transform the list into a list of commit message
          .collect { msg -> msg.split(':')[0].split(',').collect { it.trim() } } // Extract issues id from each commit
          message
          .flatten() // Merge into a single list
          .findAll { task -> task ==~ /${prefix}-[0-9]+/ } // Keep only valid issue names
          .unique() // Remove duplicates
     currentBuild.description = issues.collect { "<a href=\"${baseUrl}/${it}\">${it}</a>" }
          .join(", ")
}
```

Let’s refactor our JIRA script

- Update Jenkins system config
  - Add a Global Pipeline Library
  - Name your library
  - Setup the GIT repos



Library

Name: pipeline-common-lib

Default version:

Load implicitly ☐

Allow default version to be overridden ☑

Include @Library changes in job recent changes ☑

Retrieval method

● Modern SCM

Source Code Management

● Git

Project Repository: git@gitlabe[redacted].com:[redacted]/pipeline-common-lib

Credentials: jenkins (Jenkins-gitlab) ▾

Behaviours

Within Repository

Let’s refactor our JIRA script

- Update the calling Jenkinsfile
    - Load the library on the first line of Jenkinsfile

```
@Library("pipeline-common-lib@master") _
```

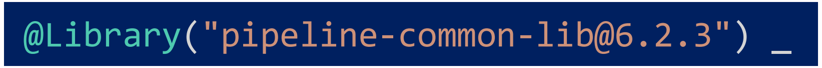
- Call the **linkToJira** step somewhere in a stage’s steps block

```
steps {
    linkToJira "TASK", "https://your-jira.server.com/browse"
}
```

Version your lib

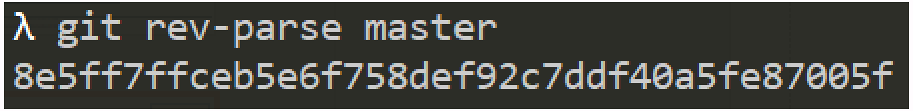
- Avoid importing the master branch of a library
- Add versioning to it with a git tag with `git tag 6.2.3 && git push`

  And import it

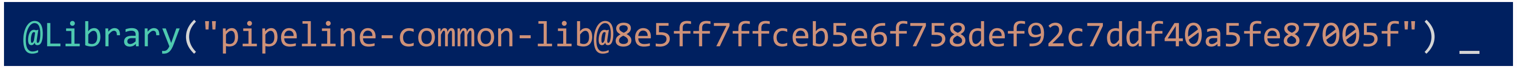


```
@Library("pipeline-common-lib@6.2.3") _
```

- Or directly import a git revision



```
λ git rev-parse master
8e5ff7ffceb5e6f758def92c7ddf40a5fe87005f
```



```
@Library("pipeline-common-lib@8e5ff7ffceb5e6f758def92c7ddf40a5fe87005f") _
```

6th UCAAT
User Conference on Advanced Automated Testing
ETSI
Paris, 16-18 October 2018
Organizer:
TESTING SOLUTIONS &SERVICES

Conclusion

Why using Blue Ocean

- Enable good practices
- Keep your whole pipeline in an SCM (eg: GIT)
  - Stored alongside with testing scripts and / or tested code
  - Can be passed through the code review process
  - Track changes
  - Development made easier
  - Branches can be forked easily
  - Old versions can easily be relaunched
- Easy use of docker
- Share and reuse common parts accross projects
- Flexibility
- Understandability

Why not

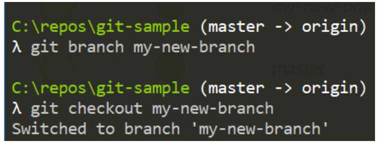
- The learning curve
- Blue ocean is still under heavy development
- Groovy language
- The cost of rewriting existing freestyle jobs
- Still hard for now to test Jenkinsfile without having to submit it

QUESTIONS ?

Backup slides

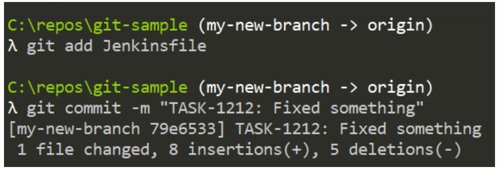
Review workflow

- Fork master branch into a new one
- Make your changes and commit / push them



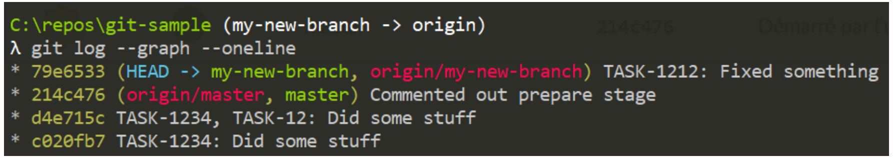
```
C:\repos\git-sample (master -> origin)
λ git branch my-new-branch

C:\repos\git-sample (master -> origin)
λ git checkout my-new-branch
Switched to branch 'my-new-branch'
```



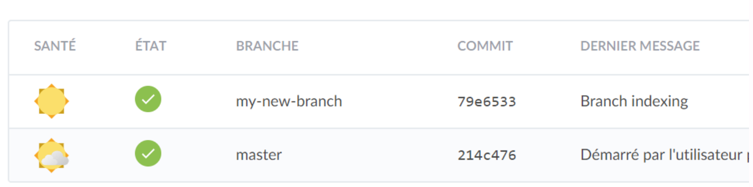
```
C:\repos\git-sample (my-new-branch -> origin)
λ git add Jenkinsfile

C:\repos\git-sample (my-new-branch -> origin)
λ git commit -m "TASK-1212: Fixed something"
[my-new-branch 79e6533] TASK-1212: Fixed something
 1 file changed, 8 insertions(+), 5 deletions(-)
```



```
C:\repos\git-sample (my-new-branch -> origin)
λ git log --graph --oneline
* 79e6533 (HEAD -> my-new-branch, origin/my-new-branch) TASK-1212: Fixed something
* 214c476 (origin/master, master) Commented out prepare stage
* d4e715c TASK-1234, TASK-12: Did some stuff
* c020fb7 TASK-1234: Did some stuff
```

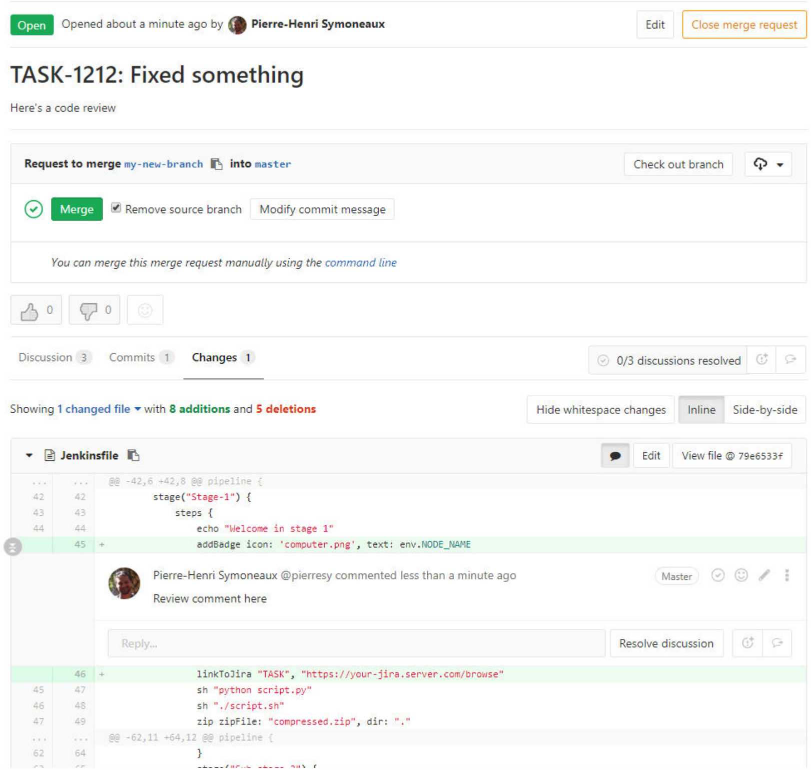
- New Job is automatically created and run



SANTÉ	ÉTAT	BRANCHE	COMMIT	DERNIER MESSAGE
☀	✓	my-new-branch	79e6533	Branch indexing
⛅	✓	master	214c476	Démarré par l'utilisateur

Review workflow

- You can ask for a peer to review your changes
  - Pull Request (Github)
  - Merge Request (Gitlab)
  - Working with Gerrit also possible



Open Opened about a minute ago by Pierre-Henri Symoneaux

TASK-1212: Fixed something

Here's a code review

Request to merge `my-new-branch` into `master`

You can merge this merge request manually using the command line

Showing 1 changed file with 8 additions and 5 deletions

Jenkinsfile

```
@@ -42,6 +42,8 @@ pipeline {
        stage("Stage-1") {
            steps {
                echo "Welcome in stage 1"
+               addBadge icon: 'computer.png', text: env.NODE_NAME
```

Pierre-Henri Symoneaux @pierresy commented less than a minute ago

Review comment here

```
+               linkToJira "TASK", "https://your-jira.server.com/browse"
                sh "python script.py"
                sh "./script.sh"
                zip zipFile: "compressed.zip", dir: "."
@@ -62,11 +64,12 @@ pipeline {
                }
```