**July 22**: Wednesday

e2e test:

Load data, save data and test.

Test h2 database in Jenkins.

getLogs (1000)

change those logs to get logs of stages we are looking at

in 3p-utils,

scan\_third\_parties.groovy

Line 36

run\_hub\_detect

process\_scan\_results

substitute

look into whether there is

jenkins variable that tells about current stage name.

3p-utils does not know which stage it is running in.

How we would read stage logs within 3p-utils ?

Working on finding a way to read stage logs within 3p-utils library for a particular stage rather depending on the pipeline.

Directly run in 3p-utils and sca-utils ?

Because shared libs may not know what stage it is in.

Maybe last 2000 lines of logs come from some other stage.

MEF

env.STAGE\_NAME

**August 4: Tuesday**

Binding

Auto-Binding

**August 10: Monday**

Dependency cleanup

Intelligent Re-push

**August 11**: Tuesday

Take a look at 3p-utils. Is it running with blackduck backup or production ?

Not able to use global methods in the class in a Jenkinsfile.

These global methods are linked to runtime.

Run a unit test

Only call setCurrentLogName() once and not 2 times.

==~ matches the subject with the regex, but the match must be strict.

findAll

Reading logic is not working.

Reading the file from the wrong place.

shell and bash

Problem with exit codes.

Escape the built-in linux shell variable to avoid it being read as a groovy variable.

Mixing java, groovy and Linux together leading to technical errors.

Make logName a local variable rather a global variable.

Creating stateful classes with jenkins shared library

Now there is a concurrency issue due to the global variable.

For mx-staging,

Look at the example that Louis sent.

“load” class

Run the class on its own without the Jenkinsfile.

In linux, we had to add the workspace variable as it was assumed to be at the root on linux.

If I do deleteDir() before loading a file, then file would not be loaded correctly.

**August 13**: Thursday

XPath: standard for describing paths in xml.

Lot of design patterns are made of different ones.

Swing: Common example of observer pattern.

If we define a variable in groovy in a method without using “def”, then this variable will be global even if it is declared in a method.

This behaviour is similar to Javascript.

Now 3p-utils is thread safe.

Do the task of adding badges in the jenkins file itself.

“ccp-test-env-bpm” pipeline.

It should not run production jenkins file.

Ccp

JenkinsfileBPM: make it compatible with test environment as well.

In one of the branches, this already exists somewhere.

In 3p-utils,

JenkinsfileBlackduckBackup

Do we need this file ?

repo-name@tag\_version

Update the clean\_input method in mx-staging.

If comps is a blank line, ignore it.

Groovy unit tests. Look this up.

May need to add gradle or pom file.

See the branches of ccp.

In mx-staging, we need did\_fail() because it calls write\_to\_file() method which is required in the “Staging” stage.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Artifacts | Exit\_code | Result | Failure\_  message | Force  Staging |
| abc:versions-maven-plugin:jar:2.7 | 1 | Unstable | Not empty | True |
| abc:versions-maven-plugin:jar:2.7 | 1 | Fail | Not Empty | False |
| org.eclipse.sisu:sisu-inject:pom:0.3.4  com.google.inject:guice:jar:no\_aop:4.0 | 0 | Pass | Empty | Don’t care |
| io.swagger.core.v3:swagger-annotations:jar:2.1.2  org.springframework.boot:spring-boot-maven-plugin:jar:2.3.0.RELEASE  org.springdoc:springdoc-openapi-maven-plugin:jar:1.0  org.springdoc:springdoc-openapi-ui:jar:1.3.6 | 0 | Pass | Empty | Don’t Care |
| abc:abc:123 | 1 | Fail | Non-empty | False |
| abc:abc:123 | 1 | Unstable | Non-Empty | True |
| abc:abc:123 | 1 | Fail | Non-Empty | False |
| org.codehaus.mojo:versions-maven-plugin:maven-plugin:2.7  junit:junit:4.12 | 1 | Pass | Empty | True or False |
| abc:abc:123  junit:junit:4.12 | 1 | Unstable | Non-empty | True |
| abc:abc:123  junit:junit:4.12 | 1 | Fail | Non-empty | False |
| org.codehaus.mojo:versions-maven-plugin:maven-plugin:2.7  org.codehaus.mojo:abc:eclipse-plugin:2.7  org.codehaus.mojo:xyz:javadoc:2.7 | 1 | Pass | Empty | False |
| org.codehaus.mojo:versions-maven-plugin:maven-plugin:2.7  org.codehaus.mojo:abc:eclipse-plugin:2.7  org.codehaus.mojo:xyz:javadoc:2.7 | 1 | Pass | Empty | True |
| org.codehaus.mojo:versions-maven-plugin:maven-plugin:2.7  org.codehaus.mojo:abc:jar:2.7  org.codehaus.mojo:xyz:zip:4.6 | 1 | Fail | Non-Empty | False |
| org.codehaus.mojo:versions-maven-plugin:maven-plugin:2.7  org.codehaus.mojo:abc:jar:2.7  org.codehaus.mojo:xyz:zip:4.6 | 1 | unstable | Non-Empty | True |
| **?** | 0 | Fail | Non-empty | Don’t care |

Invalid, Required and Ignore\_types.

When failure message is empty => Pass

When exit\_code is 0, then we don’t need to test for force staging.

In Jenkins, We can also replay a file which we load using “load” function.

Client Evolution Service (CES)

Product Evolution Service (PES)

**TPK:** Test Package

TPK - Test package. It is used for migration. It usually contains configuration for 3.1 binaries and for a 2.11 DB. A job can be pushed with this TPK in 3 modes: DEFAULT,  DEPLOY, TESTEXECUTION

* DEPLOY - will just deploy the binaries and the database
* TESTEXECUTION - will run some tests against the database (among which the migration sequence)
* DEFAULT = DEPLOY + TESTEXECUTION

In Jenkins history, some links may get hidden in normal screen zoom in.

It could become visible when there is 150% zoom.

Saw this in ccp-test-env-bpm, lgtm pipeline as well.

**August 17: Monday**

mvn clean test -DuseMavenRepo

Tell maven to stop looking for java directories and look for groovy directories.

With maven, -X -e: more verbose information

(X is capital and e is small)

It never tries to use groovy eclipse compiler.

Something is missing in pom.xml

Build a maven project using groovy.

Trying to switch to gradle.

Made the project to be viewed as a Groovy project.

Initially I just had the gradle project.

Similarly with maven ?

mvn clean install --settings settings.xml

-Pdev-repositories  -U

This installs “maven-repo-local” directory.

This directory contains junit, log4j,

mx-staging, org, xml-apis.

When we don’t specify --settings, then it takes another settings.xml file.

maven-repo-local is a temporary directory to have the artifacts locally.

August 18: Tuesday

Automatically re-try the blackduck operation based on exit code and error messages.

**IMP NOTE**:

Never refactor on a failing test. That is never modify a thing which is already failing.

First make it succeed and then refactor.

Reflection important concept:

Getter and class variable are coupled with each other.

a, getA()

These 2 can be used interchangeably and lead to stack overflow error.

How often does the blackduck client version changes ?

**Wednesday: August 19**

Artifact containing spaces.

For example, “ ”, “ junit:junit:4.12”, etc

3p-utils read/write logs not working for CJE (BPM):

For CJE, the scan\_third\_parties() log written during the scan via ‘tee’ is not retrievable when processing the results.

This behavious is not observed outside of CJE.

On the cje, test elastic search and see if we can reproduce the issue.

CodeQL

**August 20: Thursday**

Organization into packages is very important.

Run shared library with logs

ls -lt

Test pipeline for staging.

Pipeline that stages different things.

Test jenkings pipeline.

Make staging a shared library ?

If this does not work, write

Readme test cases

**Another interesting case**:

abc:def:xyz:abc:def:xyz:abc:def:xyz:123

The job passes for this.

End to end tests.

For these artifacts,

abc:def:xyz:def

abc:def:xyz:abc:def

the job is passing, Why ?

Because did\_fail() returns false.

These 2 artifacts do not get converted into :jar: or :pom:

3p-utils: Read/write logs not working for CJE

Created a new job in cje.

**New task**:

Ownership generation: do not stop the generation of new report if it failed to compute owners of some files.

**Friday**:

Can’t replay a jenkins file if there is a syntax error.

Remove multiple calls to 3p-utils.

Hard code the name of the log file.

The only time we can go into a docker container only when it is running.

Single point of failure.

Standard output could disappear. That is why tee isn’t working.

Linked to standard output. In docker, standard output is handled in a special way.

ci:codegov runs in a docker container.

Does not work in cje docker container.

import java.nio…. worked in Jenkins file

groovy: interpretated language

Something can only fail when we call it.

See #24

Jenkins Groovy sandbox

In-process script approvals.

java.io.File methods will refer to files on the master where Jenkins is running, so not in the current workspace on current slave machine.

Try to use the readFile() method.

Jenkins stores all logs on master only.

Jenkins master-slave.

Jenkins readFile: Read file from workspace.

Cannot use readFile in the shared library.

#26

Try to use readFile in a shared library.

This code works:

import java.nio.charset.StandardCharsets

import java.nio.file.\*

pipeline {

agent any

stages {

stage('Scan Timeout Test') {

steps {

script {

sh '''echo "Running ls before ..."

ls -lt

'''

sh 'ls -lt | tee lslog.txt'

sh '''ls -lt

echo "Trying to read the file..."

cat lslog.txt

'''

}

}

} //end of stage

}

}

def getLogs(logPath) throws IOException {

println "[INFO] Reading Log File: " + Paths.get(logPath).toAbsolutePath().toString()

try {

println Files.readAllLines(Paths.get(logPath), StandardCharsets.UTF\_8);

} catch(IOException e){

println "[ERROR] Failed to read file '"+logPath+"': "+e.getMessage()

throw e

}

}

Here we are not using getLogs() method.

This works when we use readFile() method of Jenkins:

import java.nio.charset.StandardCharsets

import java.nio.file.\*

pipeline {

agent any

stages {

stage('Scan Timeout Test') {

steps {

script {

sh '''echo "Running ls before ..."

ls -lt

'''

sh 'ls -lt | tee lslog.txt'

sh '''pwd

ls -lt

'''

getLogs("$WORKSPACE/lslog.txt")

}

}

} //end of stage

}

}

def getLogs(logPath) throws IOException {

println "Reading $logPath..."

def text = readFile logPath

println text

// println "[INFO] Reading Log File: " + Paths.get(logPath).toAbsolutePath().toString()

// try {

// println Files.readAllLines(Paths.get(logPath), StandardCharsets.UTF\_8);

// } catch(IOException e){

// println "[ERROR] Failed to read file '"+logPath+"': "+e.getMessage()

// throw e

// }

}

Here, I have used readFile in jenkins file.

I have to use it in the shared library.

<https://www.jenkins.io/doc/book/pipeline/shared-libraries/#accessing-steps>

Pass in the “steps” variable from the jenkins file to the shared library.

**August 24**: Monday

Pluralsight for implementing producer and consumer.

Linux uses slave, groovy and java use master.

Read file using Linux.

Currently we have a more groovy way of doing things.

Return result of of a command as a variable and we also want to return an exit code.

Want getLogs() to use sh to read the log file.

Return log file to groovy.

Stack overflow:

How to do I get the output of a shell command executed

Change getLogs.

Leave tee as it is.

Catch stdout with return stdout as true.

Change it to true. Split on \n.

Linux commands run on slave.

Java and groovy command run on master

Log file is written on slave, therefore we cannot find it on master.

We write the file using linux and also work with reading the file using linux.

Return stdout as a string.

cat fileName

This will simulate reading the file using groovy.

**August 25**: Tuesday

\r?\n

Split on \r, otherwise split on \n

If we use “tee” command, then it will write on the slave.

Want everything on master or on slave.

Alternative: never write to a log, instead pass it to next part of a shared library.

Linux works on the slave node.

Interesting Groovy syntax:

referenceVariable**?.**method()

Do not execute method() if referenceVariable is null.

extractFolder() method in find\_unknown\_components.groovy file.

Make the 3p-utils work on ci:codegov along with the tests and updated getLogs() method.

validateLogReading() test: can be removed.

In scan\_third\_partiesTest.groovy

Currently I am working on making the 3p-utils pipeline green on ci:codegov. I made the changes in scan\_third-parties for reading the file using linux and

Linux working directory is slave.

If we mix linux, groovy and java then we have a problem.

We write the file using “tee” command and read the file using “cat” command.

Write a wiki page.

LDAP configuration

**Thursday**: August 27

System test, behavioural test, smoke test

Quality gates

Test package

Look at the find\_unknown\_components now.

Need to add gradle in the cje.

Interesting in Git:

Cannot push branches which have “bugfix/” directly. Can only push commits through pull requests.

Workaround such as copying files to master or tricking Groovy so that it can access the slave.

Pain points

What is wasting time and blocking progress.

Topics I am currently working on.

Best next steps for the sprint.

Usability

Functional tests for blackduck. Enhance 3p-utils pipeline.

When we move 3p-utils to cje, everything breaks.

Works for scan-third-parties.

Falls apart for find-unknown-components.

Jenkins master/slave problem.

cje cannot read the workspace.

Brainstorming meeting. Look at the problem and think of different solutions.

Cannot run gradle in the codegov cje.

Faster feedback while testing jenkins.

MDS3P-1201: Add a cje link to this.

(CPS)

Continuation Passing Style

Restore the old version.

Add a ReadMe in staging repo.

**September 2**: Wednesday

In mx-staging,

Number of allowed colons = 2, 3 or 4

**Standup**:

In mx-staging, I added a file for test cases.

did\_fail()

I added a call to did\_fail in the basic set up rather in the if block because if we add it in the if block then it is always true.

3p-utils: 3rd party utilities

Shared library: package that we import in a jenkins pipeline.

sca-utils: static code analysis utilities

find unknown components: way to ensure that where the blind spots are.

**September 3**:

Business as usual involves bug fixes.

Find\_unknow\_components has Only 1 input, ignore\_patterns

One more property.

Functional failure if number of unknown components exceed a threshold.

Warning sign showing unexpected components for find\_unknown\_components.

maybeAddBadge() function

If exit code ==3 // functional failure

If exceed a threshold, add error badge else warning badge.

Add “unknown\_components\_threshold” parameter.

Quality control check

Jars owned by Murex.

murex-known-jars.txt

(regex names of jars)

ts-….

(time series)

Unexpected jars are not seen by Blackduck.

notices\_jars\_and\_zips.log

These are seen by Blackduck

Jenkins convention,

Mandatory to have the groovy files in “vars” folder.

In IDE itself, we can do a diff.

Master-slave, Docker

Java and groovy have access to disk of master.

We tell jenkins about the working directory.

Pull out the code as a java 3rd party.

Have a src/main/java

**Send jar to nexus**.

Build a standalone tool.

maven:dependency:get

java -jar

see diff() method in find\_unknown\_components.groovy

take extractFolder() function and parse, etc and all stuff

pull into a standalone java project.

Name it as Archive analyzer

computeActualFilesDetected()

everything this calls will go into the java library.

computeActualFilesDetected(): Change the implementation.

Return an object.

All this can be done in native java.

Take chunks of the file.

writeSourcesLogFile()

Copy into a new maven project.

Unzip

zip.stream()

print to stdout

minimum viable component (mvc)

Follow the template similar to diff() method.

Help james to setup 3p-utils.

Jenkins groovy is a bit different from normal groovy.

**Archive analyzer utility**:

Write a jar

java -jar archive-analyzer-<version>.jar <path-to-archive-or-directory>

**Friday**:

In groovy, everything is detected at runtime.

Some errors are not detected in an IDE.

Strange highlighting

Gradle is maven with some extra bits.

Gradle connects to the maven repository.

**Start from the command line**.

I have created a class ArchiveAnalyzer.

How to call the jar ?

Everything is done inside the 3p-utils library.

Do Simplest possible technical validations first.

**Simplest**:

Read and write the contents of the jar and write to a file.

Try to pull it in a shared library.

Take scan\_third\_parties and duplicate it.

Shared library test.

Can copy and paste if I want.

Then,

Test it again on cje.

Then work on the properly solution.

Step by step.

Help james,

Setup an integration test that call Linux.

Exit code is 3 for functional failure.

Pipeline that calls 3p-utils and calls find\_unknown\_components.

Go to nexus and pull down the jar.

mx-dst

JAR (Java Archive File)

Build mx-staging using this command:

mvn clean install --settings settings.xml

-Pdev-repositories  -U

In mx-staging, the exit code may not matter.

Start with positive cases in ifs as people understand positive cases better.

Real deployment is done somewhere else.

**IMPORTANT NOTE**: In mx-staging,

If something does not deploy, then pipeline should fail even if the artifact has one of the ignore types.

Take the last version from the jenkins.

To run the jar,

java -cp archive-analyzer-1.0.jar src.main.groovy.ArchiveAnalyzer

To add java nature, I updated .project and .classpath files.

<https://stackoverflow.com/questions/22667481/why-cant-i-add-a-groovy-nature-to-a-java-project>

**Monday: September 7**

In mx-staging, FORCE\_STAGING is a boolean parameter. In jenkins file we need to check it as a string “true” or “false” to use it in an if block.

In 3p-utils, I have the jar and am facing the problem of main class not being found when I run the jar.

To create the groovy class, I had to add java nature to the project.

Never test on production environment.

**groovy -cp xyz.jar ArchiveAnalyzer abc**

**.project file**

**In mx-staging,**

The type must be the 3rd value. That is between 2nd and 3rd colon.

For example, abc:xyz:maven-plugin:1.1 and not

abc:maven-plugin:1.1

groupId:artifactId[:packaging[:classifier]]:version.

**Tuesday, September 8**:

I resolved the issue with main method. For this I had to add groovy nature and java nature in the .project file.

Compilation issues with groovy in 3p-utils.

mx-staging, found a case for a maven-plugin which should fail as nothing is staged.

Run in 3p-utils

Maven archetype plugin.

To specify a tag in the branch specifier, we use tags/name\_of\_tag

**Wednesday: September 9**

Link between tools and jira.

Managed-interfaces from jenkins configuration (seed job) should map to “Managed-interfaces” in the Jira ticket.

TPS-Approval does a bom scan. Creates a fake maven project and scans it.

In 3p-utils, I am currently working on creating the archive analyzer jar to be used with find unknown components.

I had to convert groovy syntax from find unknown component to java syntax and I have a few minor issues with groovy closures that I am not able to convert to java code.

Cje docker containers. Depends on which container we are using.

Move the archive-analyzer project to tmt.

In the pom.xml file, Maven shade plugin

“it”: built in iteration variable in groovy.

Build your own method. Just print stuff.

Convert stuff inside forEach() into a method.

Split out what the loop is doing.

Pass in zipEntry to a method.

Pull all of it in a method.

2nd: print out the name of the files initially.

When the looping part is working, then do the complicated logic of unzipping.

Use the dummy zips in 3p-utils.

Recursively list files in java.

Files.walk

.filter

.forEach

Can also convert the unit tests into my library.

Change “data” to “Results”

Start with a new tool. First figure out how to iterate over zips, then iterate over sub-directories.

zip.stream().forEach(entry -> processZipEntry(entry))

This command works:

java -cp archive-analyzer.jar com.murex.tmt.archiveanalyzer.ArchiveAnalyzer D:\archive-analyzer\archiveanalyzer\src\main\java\com\murex\tmt\archiveanalyzer\dummy.jar

BOM Scan: Analyze the project at the POM level

Signature Scan: Analyze the project at the compiled JAR level.

**Thursday:** September 10

I was able to make the jar work by passing in the dummy jar.

admin, admin690

Make the jar call from sh command both in normal jenkins ci:codegov and cje.

Case with recursive directories.

D:\a\b\c\d.zip

source.eachFileRecurse

list all files recursively in a directory.

Files.walk()

Walking the file tree.

Visitor Design Pattern

Java-how to list all files in a directory: mkyong

Push the jar to nexus. repo-dev

repo-dev/nexus

mx-dst repository: use this one

artifact upload tab, creat the gav parameters.

**Friday**: September 11

To test the case of recursive directory structure, we can use

java -jar archive-analyzer.jar D:\archive-analyzer\archive-analyzer\src\main\java\com\murex\tmt

java -jar archive-analyzer.jar D:\archive-analyzer\archive-analyzer\src\main\java\com\murex\tmt

**September 14: Monday**

+ java -Xloggc:garbage-collection.log -jar archive-analyzer-1.0.1.jar /var/jenkins\_home/workspace/fix-3p-utils//murex-known-jars.txt

no main manifest attribute, in archive-analyzer-1.0.1.jar

To resolve this, add the Main-Class property in the manifest file.

Error: A JNI error has occurred, please check your installation and try again

Exception in thread "main" java.lang.UnsupportedClassVersionError: com/murex/tmt/ArchiveAnalyzer has been compiled by a more recent version of the Java Runtime (class file version 55.0), this version of the Java Runtime only recognizes class file versions up to 52.0

To resolve this, compile the java class using jdk version 1.8 instead 11.

Compiler compliance level: 1.8

I am able to now call archive-analyzer jar from jenkins. Currently I am printing logs.

The archive-analyzer jar now works.

Abandon groovy and do in a java or linux way.

Use writeFile instead of file.write

Found a difference between scan\_third\_parties and find\_unknown\_components

writeFile: write file to workspace.

Logs

Regex to pull them out.

Call java from groovy and return java from groovy.

“List of files”

Use of regex.

Build result object.

Then do a diff

Diff on 2 log files.

Build an UnknownComponentResult object.

Parse the file.

Link all the pieces together.

**September 15, Tuesday**:

\r?\n

To be compatible with both Linux and Windows system.

In git, everything is considered as a codeline.

**Wednesday**:

I fixed the issue with reading and writing the file to build the result object.

And then I fixed an issue

The diff tool returns an exit code of 3.

List of regular expressions of jars that we are meant to ignore.

Analyze the jar with the passed in argument in the jenkins file.

In cje, with diff tool, getting

[STACKTRACE] java.io.FileNotFoundException: /jenkins/workspace/test-3p-utils/murex-known-jars.txt (No such file or directory)

getUnexpectedResultsCount: Convert the logic from groovy into shell scripting.

**Very Important point**:

<https://php.developreference.com/article/11779601/How+to+declare+a+new+variable+inside+a+shell+block+in+groovy+script%3F>

If I access a groovy variable inside a shell block, the access is Read-Only.

I can't edit the value of the groovy variable, even temporarily within the shell block.

If I do want to do that, I can assign the groovy variable to a shell variable, manipulate the shell variable value, save the modified value in a file and when the shell block ends, read the file into the original groovy variable.

Writing data to a file and then reading it from somewhere else is highly useful.

Important technique.

To create a variable in a shell script and then in order to access it, we need to escape the $

**Thursday**:

Write documentation for the things which led to failure on cje.

Type of mistakes that will help us to fix all of the pipelines.

Java how to create and write to a file: mkyong

Focus on unzipping.

Run jar locally.

**Monday**: September 21

Locally the jar.

Write unit tests for Archive Analyzer.

Gareth is ooo on Thursday and Friday.

Write the list of cje problems that I found.

Take the unit tests of 3p-utils and run in archive analyzer.

Add the documentation.

Add in the settings.xml rather than depending on an external one.

With unit tests, we do the simplest possible thing.

Read the file as a resource rather from the disk.

Famous problem with file readings.

Test when the source path is a directory or it is a file.

Fat archive file: where we have nested directories and jars.

End to end test

3p-utils

src/test/resources

What is a site useful for in a maven project ?

Read the resource in as a directory.

With java, we can use forward slashes in windows.

Write a jenkins pipeline that just tests archive analyzer.

Smoke tests in 3p-utils.

Most of the real tests will be in archive analyzer.

org.junit.jupiter.api.Test: Junit 5

org.junit.Test: Junit 4

**September 22**:

ci:codegov, to setup jenkins pipeline, git repo link: ssh..

cje:codegov, http…

What is the common maven version for cje and ci:codegov?

2 times when we read resource files in java:

When we run tests, that is file in the resources and when we read a resource file at runtime, load in some hardcoded settings. At runtime we don’t sync the code.

Windows: Maximum path of 255 characters.

System.lineSeparator(): gives end of line separtor for the Operating System.

\r for iOS.

Test when the input is an invalid path, jar, war.

Absolute path in a unit test. File class, getAbsolute() and turn it into an absolute path.

Further work:

Given an archive, check whether it is a murex archive, open source.

Middle aged man (maml)

When writing to a file, using resultFile.write(System.lineSeparator()); is highly helpful.

Convert a path to an absolute path:

FileSystems.*getDefault*().getPath("src/test/resources/dummy-source").normalize().toAbsolutePath().toString();

**September 23**:

Javascript running on Windows.

Read the log files. Assert the contents of the file.

Separate assertions for both the files.

Create a buildString() method.

Is there is a way to do, If(isCje) ?

Or check whether gradle exists or not.

Linux does command exist.

<https://stackoverflow.com/questions/7522712/how-can-i-check-if-a-command-exists-in-a-shell-script>

Update 3p-utils.

mvn clean install -DuseDevMavenRepo

(using the default Murex settings.xml)

Inside main, convert the absolute into relative path.

No need to do “clean” with mvn when the workspace is empty.

3p-utils-settings.xml or take it from blackduck reporting.

MX.3 setups

Create a version 0.0.1 and then in 3p-utils execute the jar.

Call the ccp using my branch of 3p-utils.

Final proof by running it in ccp.

Takes time to learn new technologies.

What can I learn from that?

Stabilise pipelines and user friendly.