Vision of Continuously Assuring Performance

Symposium on Software Performance

David Georg Reichelt[1] Stefan Kühne[1]
Wilhelm Hasselbring[2]

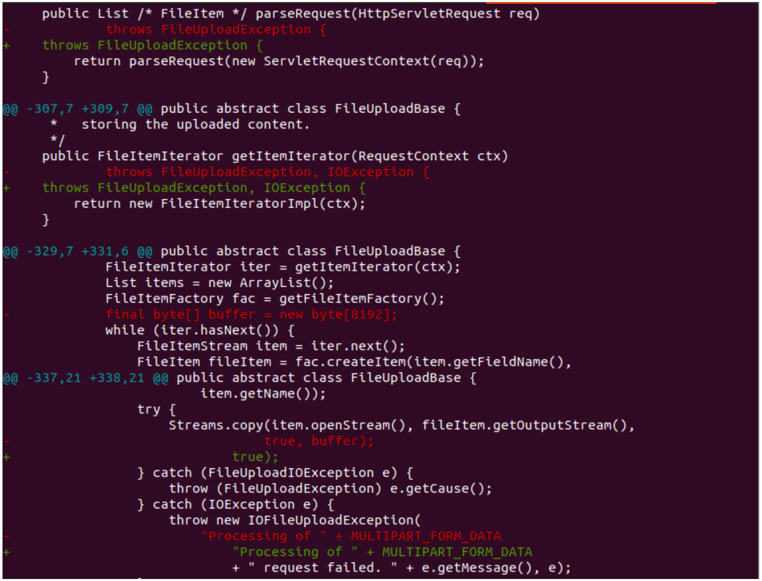
[1]Universität Leipzig, Universitätsrechenzentrum, Forschung und Entwicklung

[2]Christian-Albrechts-Universität zu Kiel, Software Engineering Group

12. November 2020

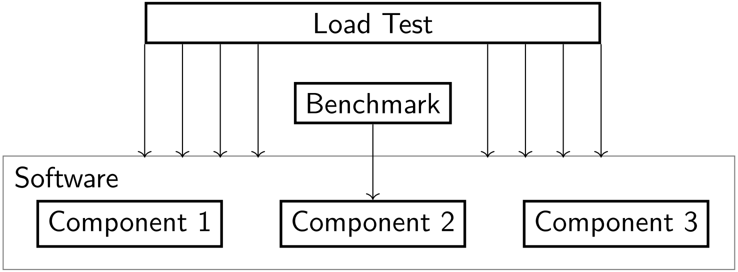
GEFÖRDERT VOM
Bundesministerium
für Bildung
und Forschung

Usual Commit

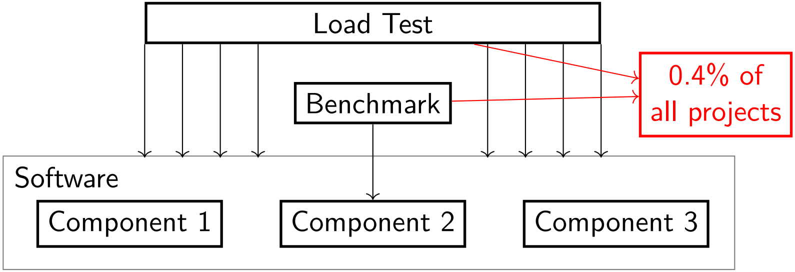


```
     public List /* FileItem */ parseRequest(HttpServletRequest req)
-            throws FileUploadException {
+    throws FileUploadException {
         return parseRequest(new ServletRequestContext(req));
     }

@@ -307,7 +309,7 @@ public abstract class FileUploadBase {
      *   storing the uploaded content.
      */
     public FileItemIterator getItemIterator(RequestContext ctx)
-            throws FileUploadException, IOException {
+    throws FileUploadException, IOException {
         return new FileItemIteratorImpl(ctx);
     }

@@ -329,7 +331,6 @@ public abstract class FileUploadBase {
             FileItemIterator iter = getItemIterator(ctx);
             List items = new ArrayList();
             FileItemFactory fac = getFileItemFactory();
-            final byte[] buffer = new byte[8192];
             while (iter.hasNext()) {
                 FileItemStream item = iter.next();
                 FileItem fileItem = fac.createItem(item.getFieldName(),
@@ -337,21 +338,21 @@ public abstract class FileUploadBase {
                         item.getName());
                 try {
                     Streams.copy(item.openStream(), fileItem.getOutputStream(),
-                            true, buffer);
+                        true);
                 } catch (FileUploadIOException e) {
                     throw (FileUploadException) e.getCause();
                 } catch (IOException e) {
                     throw new IOFileUploadException(
-                        "Processing of " + MULTIPART_FORM_DATA
+                            "Processing of " + MULTIPART_FORM_DATA
                             + " request failed. " + e.getMessage(), e);
```

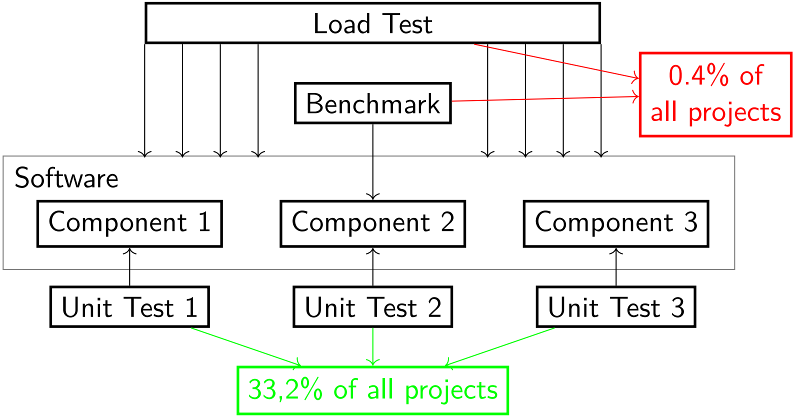
Method: Unit Test Assumption



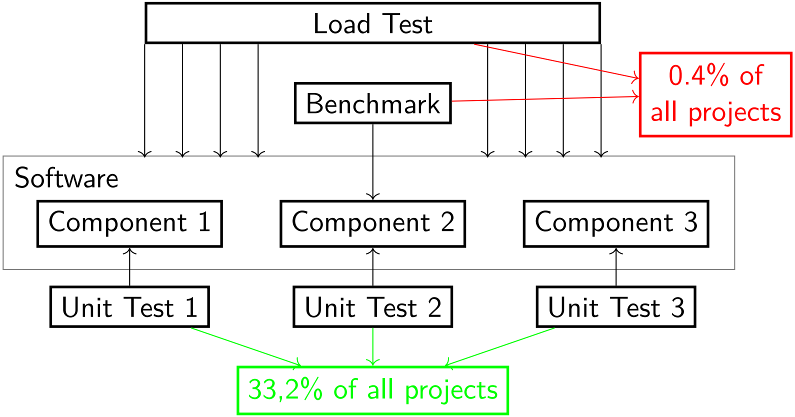
Method: Unit Test Assumption



Method: Unit Test Assumption



Method: Unit Test Assumption



Unit-Test-Assumption:
Performance of relevant use cases of a program correlates with the performance of at least **a part** of its unit tests

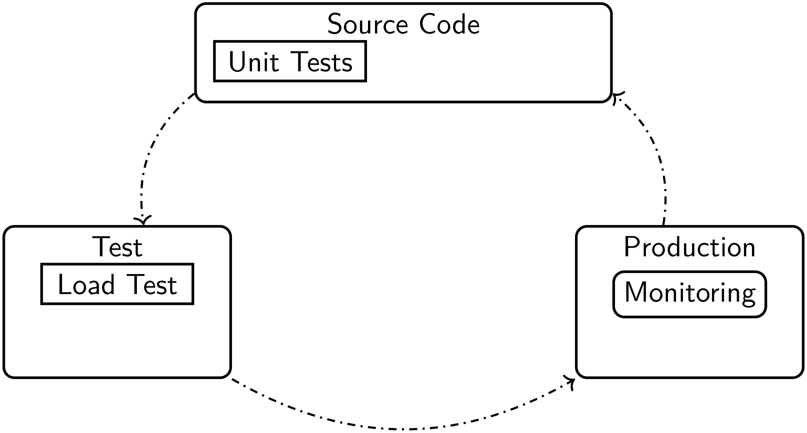


Figure: Approach of *PermanEnt*

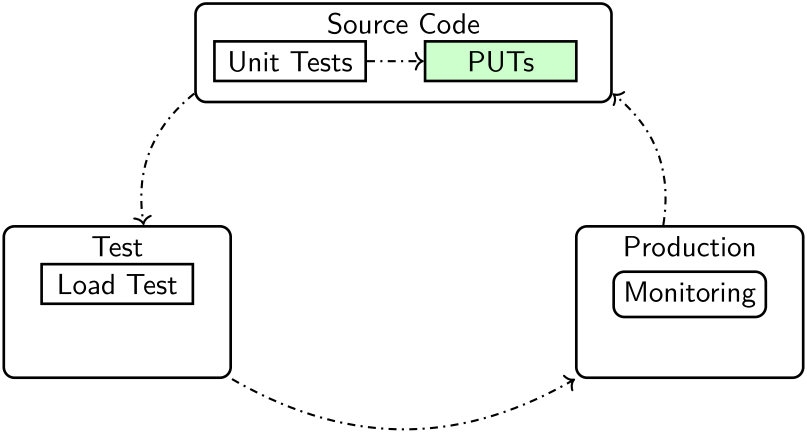


Figure: Approach of *PermanEnt*

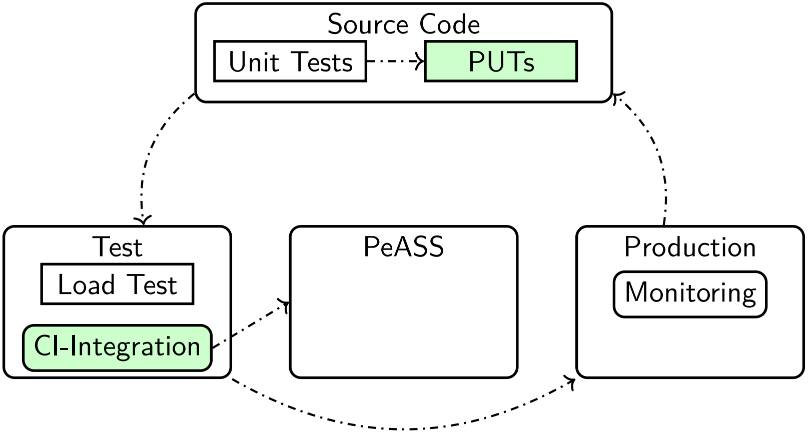


Figure: Approach of *PermanEnt*

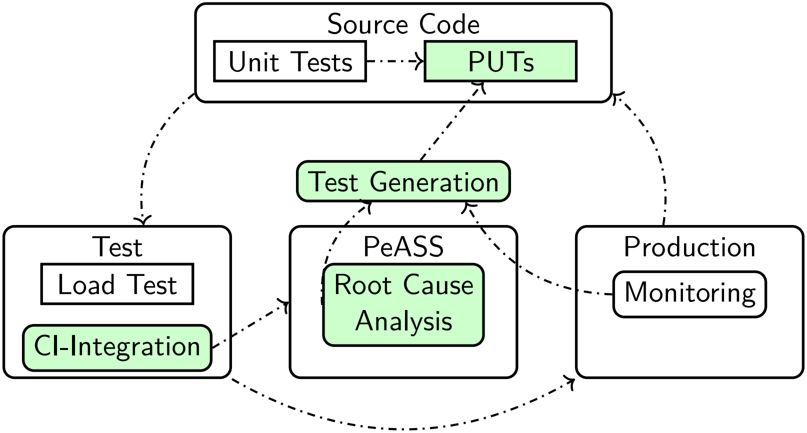


Figure: Approach of *PermanEnt*

CI-Integration

- goal: identify performance changes *correct* and *fast*

CI-Integration

- goal: identify performance changes *correct* and *fast*

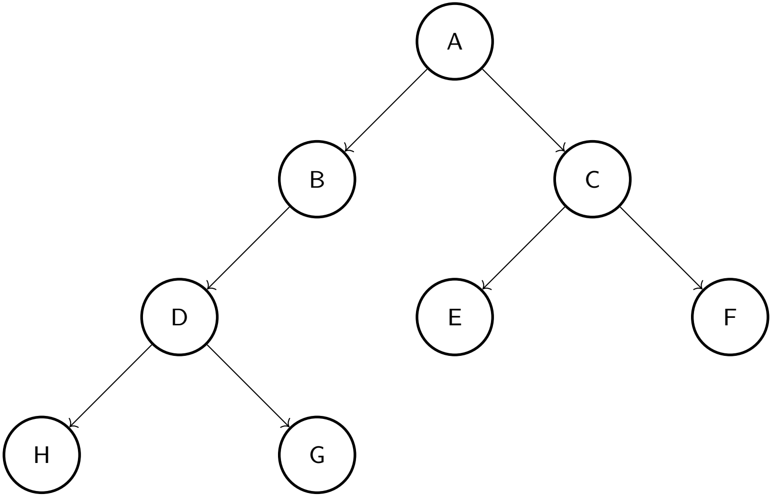
- plugin implementation
- configuration detection
  - measurement configuration ⇒ VMs, iterations, ...
  - analysis configuration ⇒ statistical test, significance level
- measurement isolation
  - isolation of measurements (cgroups)
  - parallel measurements (Bulej et al., 2019)

Existing Approaches

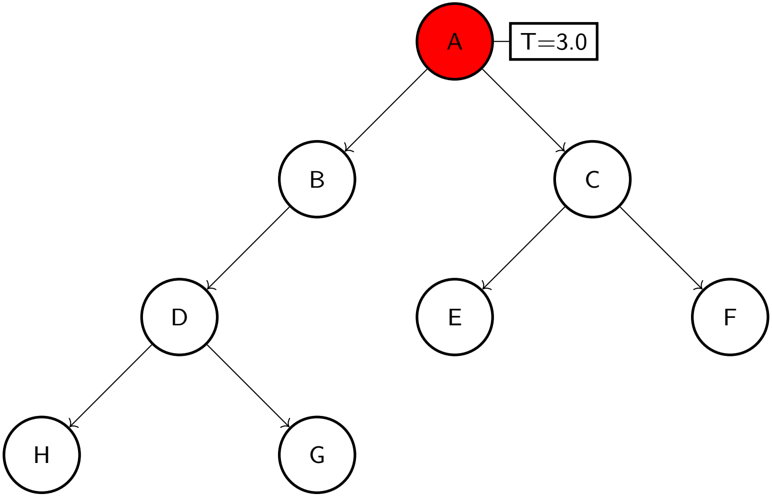
- measurement per level (Heger et al. 2013)
- complete monitoring and analysis of anomaly score (Marwede et al. 2009)

- correlation with code patterns, e.g. introducing locks (Chen et al., 2019)
- correlation with architecture patterns, e.g. excessive messaging (Wert et al., 2013)

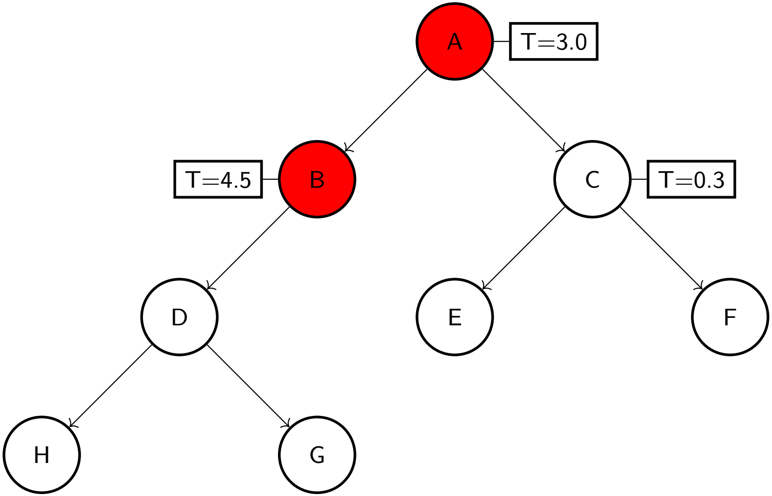
Existing Approaches - Measurement per Level


$$\begin{tikzcd}
 & & & A \arrow[dl] \arrow[dr] & \\
 & & B \arrow[dl] & & C \arrow[dl] \arrow[dr] & \\
 & D \arrow[dl] \arrow[dr] & & E & & F \\
H & & G & & &
\end{tikzcd}$$

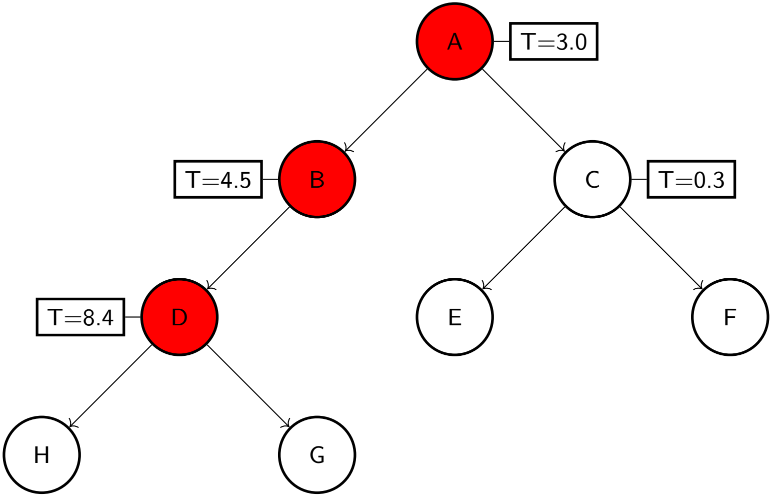
Existing Approaches - Measurement per Level



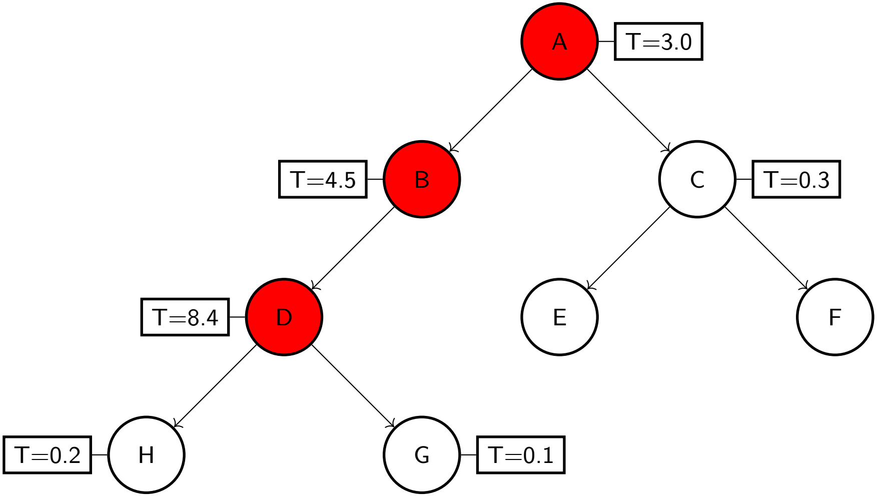
Existing Approaches - Measurement per Level



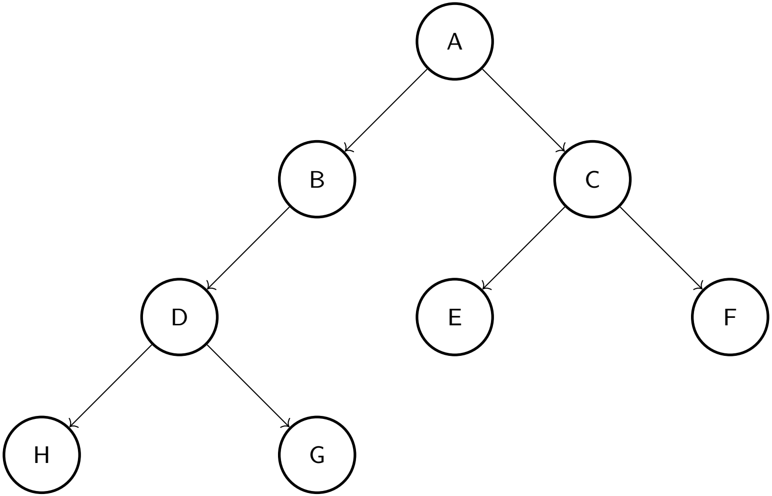
Existing Approaches - Measurement per Level



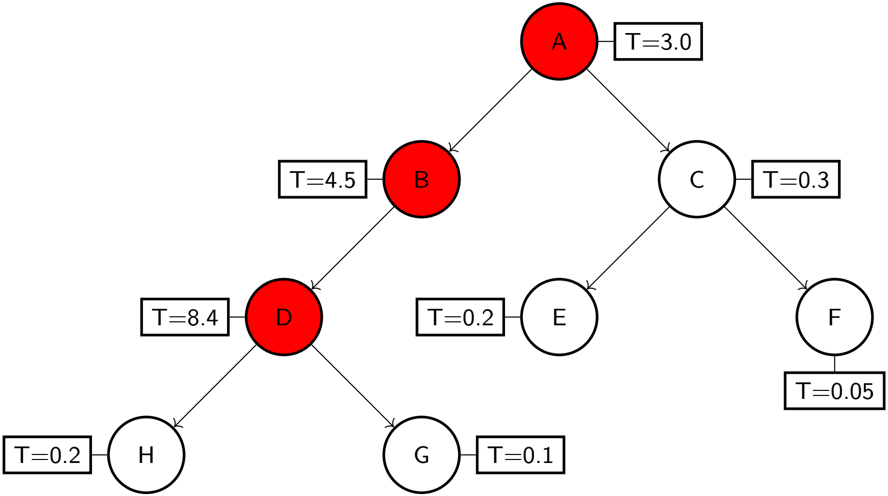
Existing Approaches - Measurement per Level



Existing Approaches - Complete Monitoring


$$\begin{tikzcd} & & & A \arrow[dl] \arrow[dr] & & \\ & & B \arrow[dl] & & C \arrow[dl] \arrow[dr] & \\ & D \arrow[dl] \arrow[dr] & & E & & F \\ H & & G & & & \end{tikzcd}$$

Existing Approaches - Complete Monitoring



Problem - Complete Monitoring

Old Version

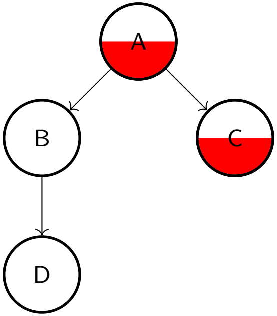
$$\begin{tikzcd}
 & A \arrow[dl] \arrow[dr] & \\
B \arrow[d] & & C \\
D & &
\end{tikzcd}$$

New Version

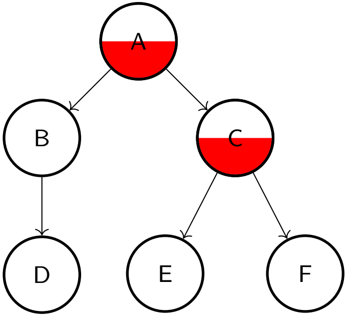
$$\begin{tikzcd}
 & A \arrow[dl] \arrow[dr] & & \\
B \arrow[d] & & C \arrow[dl] \arrow[dr] & \\
D & E & & F
\end{tikzcd}$$

Problem - Complete Monitoring

Old Version



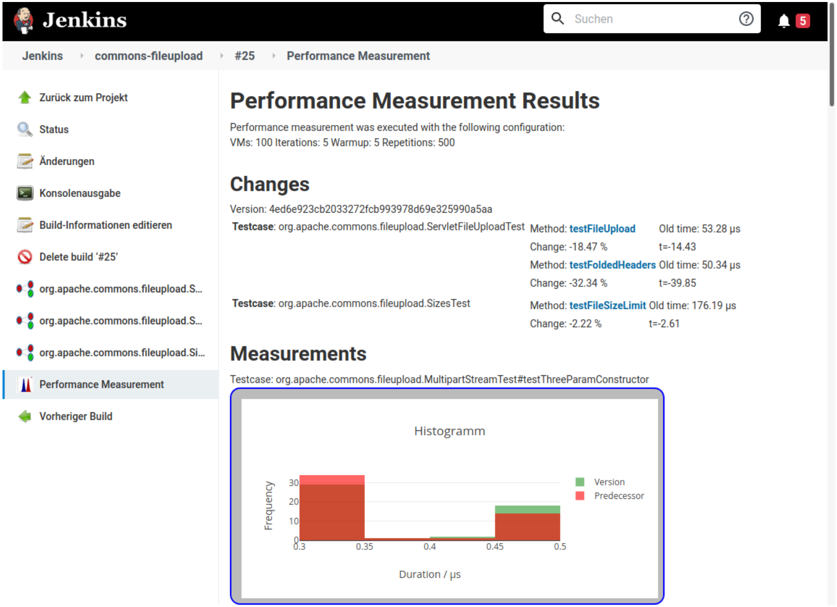
New Version



Solutions

- partial call tree measurement
- adaptive monitoring
- calculating performance difference with *call tree change lag*

Demo (hopefully ;))



Jenkins

Suchen

5

Jenkins › commons-fileupload › #25 › Performance Measurement

- Zurück zum Projekt
- Status
- Änderungen
- Konsolenausgabe
- Build-Informationen editieren
- Delete build '#25'
- org.apache.commons.fileupload.S...
- org.apache.commons.fileupload.S...
- org.apache.commons.fileupload.Si...
- Performance Measurement
- Vorheriger Build

Performance Measurement Results

Performance measurement was executed with the following configuration:
VMs: 100 Iterations: 5 Warmup: 5 Repetitions: 500

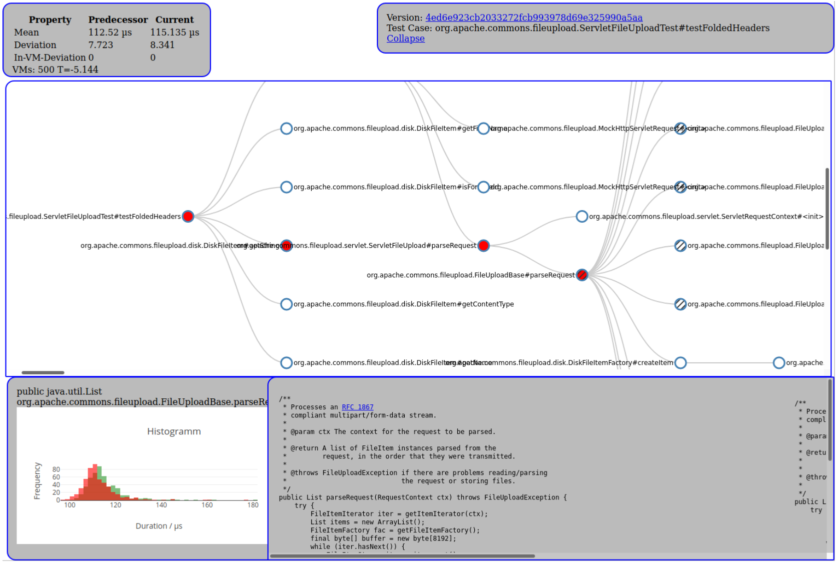
Changes

Version: 4ed6e923cb2033272fcb993978d69e325990a5aa

Testcase: org.apache.commons.fileupload.ServletFileUploadTest	Method: **testFileUpload**	Old time: 53.28 µs
	Change: -18.47 %	t=-14.43
	Method: **testFoldedHeaders**	Old time: 50.34 µs
	Change: -32.34 %	t=-39.85
Testcase: org.apache.commons.fileupload.SizesTest	Method: **testFileSizeLimit**	Old time: 176.19 µs
	Change: -2.22 %	t=-2.61

Measurements

Testcase: org.apache.commons.fileupload.MultipartStreamTest#testThreeParamConstructor



Property	Predecessor	Current
Mean	112.52 µs	115.135 µs
Deviation	7.723	8.341
In-VM-Deviation	0	0

VMs: 500 T=-5.144

Version: 4ed6e923cb2033272fcb993978d69e325990a5aa
Test Case: org.apache.commons.fileupload.ServletFileUploadTest#testFoldedHeaders
Collapse

.fileupload.ServletFileUploadTest#testFoldedHeaders
org.apache.commons.fileupload.disk.DiskFileItem#getFi
org.apache.commons.fileupload.disk.DiskFileItem#isFor
org.apache.commons.fileupload.disk.DiskFileItem#getString
org.apache.commons.fileupload.servlet.ServletFileUpload#parseRequest
org.apache.commons.fileupload.disk.DiskFileItem#getContentType
org.apache.commons.fileupload.disk.DiskFileItem#getName
org.apache.commons.fileupload.MockHttpServletRequest#<init>
org.apache.commons.fileupload.servlet.ServletRequestContext#<init>
org.apache.commons.fileupload.FileUploadBase#parseRequest
org.apache.commons.fileupload.disk.DiskFileItemFactory#createItem
org.apache.commons.fileupload.FileUploa
org.apache.

public java.util.List
org.apache.commons.fileupload.FileUploadBase.parseRe

Histogramm
Frequency
80
60
40
20
0
100
120
140
160
180
Duration / µs

```
/**
 * Processes an RFC 1867
 * compliant multipart/form-data stream.
 *
 * @param ctx The context for the request to be parsed.
 *
 * @return A list of FileItem instances parsed from the
 *         request, in the order that they were transmitted.
 *
 * @throws FileUploadException if there are problems reading/parsing
 *                             the request or storing files.
 */
public List parseRequest(RequestContext ctx) throws FileUploadException {
    try {
        FileItemIterator iter = getItemIterator(ctx);
        List items = new ArrayList();
        FileItemFactory fac = getFileItemFactory();
        final byte[] buffer = new byte[8192];
        while (iter.hasNext()) {
```

```
/**
 * Proce
 * compl
 *
 * @para
 *
 * @retu
 *
 *
 * @thro
 *
 */
public L
    try
```

Summary

- goal: performance measurement of unit tests
- CI-Integration
- Root Cause Analysis

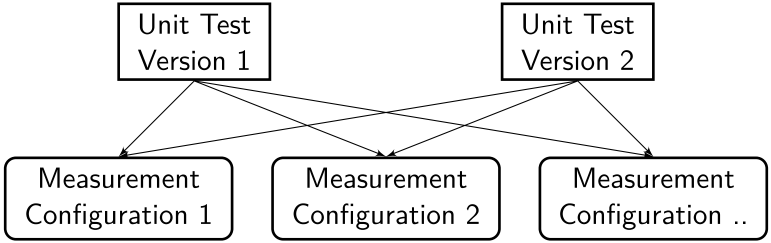
- prototype: https://github.com/DaGeRe/peass-ci

Thanks for your attention!

David Georg Reichelt
Universitätsrechenzentrum
Universität Leipzig
david_georg.reichelt@uni-leipzig.de

GEFÖRDERT VOM
Bundesministerium
für Bildung
und Forschung

Configuration Detection



Unit Test Version 1
Unit Test Version 2
Measurement Configuration 1
Measurement Configuration 2
Measurement Configuration ..

Figure: Approach for Measurement Calibration

Configuration Detection

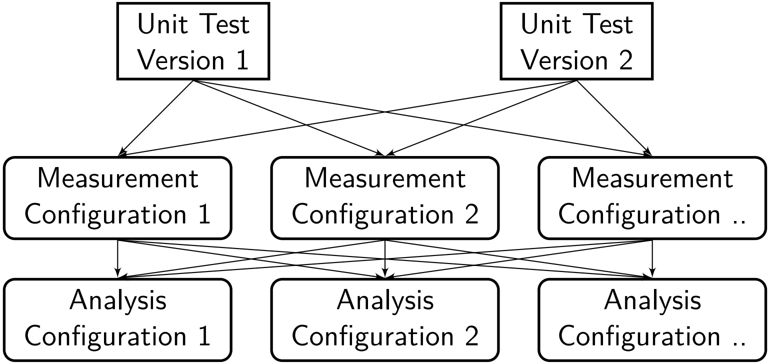

Unit Test Version 1
Unit Test Version 2
Measurement Configuration 1
Measurement Configuration 2
Measurement Configuration ..
Analysis Configuration 1
Analysis Configuration 2
Analysis Configuration ..

Figure: Approach for Measurement Calibration

Configuration Detection

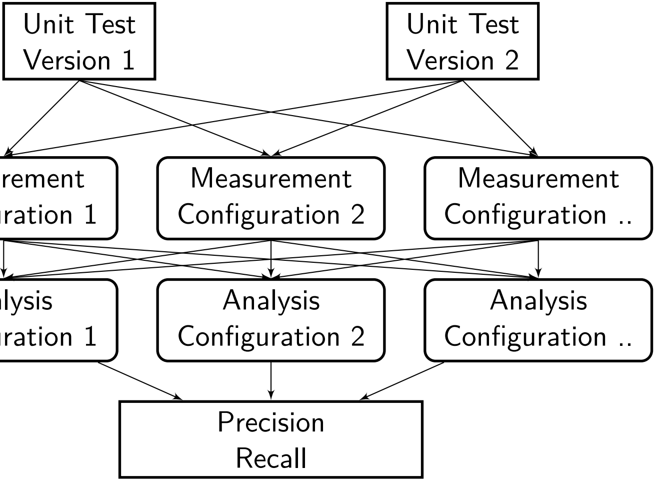

Unit Test Version 1
Unit Test Version 2
Measurement Configuration 1
Measurement Configuration 2
Measurement Configuration ..
Analysis Configuration 1
Analysis Configuration 2
Analysis Configuration ..
Precision Recall

Figure: Approach for Measurement Calibration

Configuration Detection

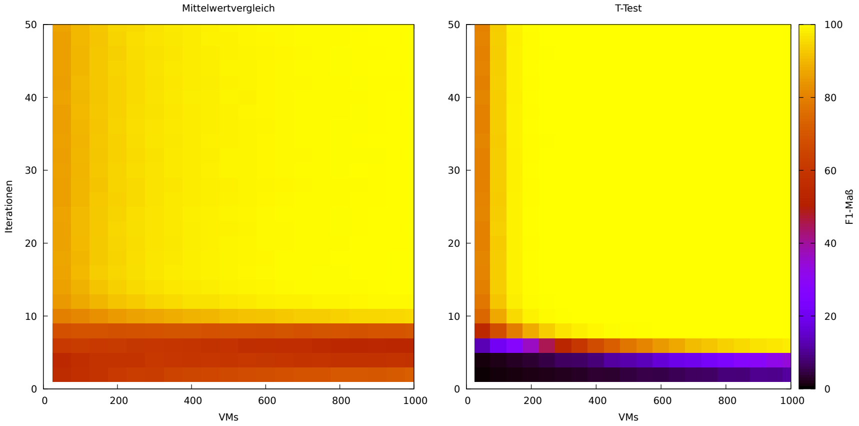


Figure: F1-Measure Example for Artificial Test Pair with 0,3% performance difference