



# TPF Toolkit Task Force

Scriptable Code Coverage

**Matt Gritter**  
TPF Toolkit

© 2017 IBM z/TPF | TPF Users Group Spring Conference | IBM Confidential

IBM z/TPF  
April 4th, 2017

# Disclaimer

Any reference to future plans are for planning purposes only.

IBM reserves the right to change those plans at its discretion.

Any reliance on such a disclosure is solely at your own risk.

IBM makes no commitment to provide additional information in the future.

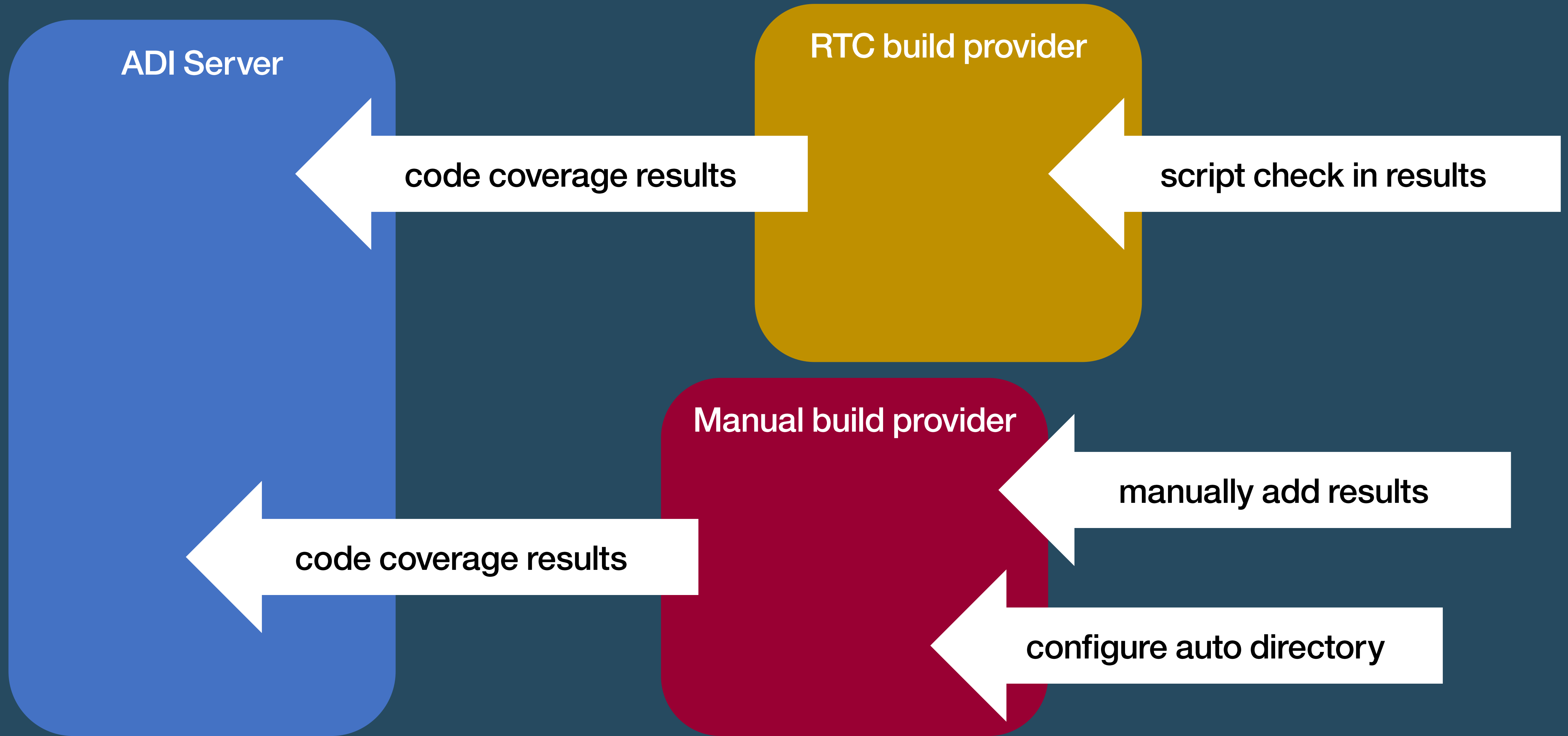
# So what's the purpose?

- To enable your QA staff to collect code coverage results when running automated regression or unit test cases
- Provide an easy way to deliver those code coverage collections to the IBM Application Delivery Intelligence (ADI) solution
- Visualize QA trends for your applications
- Quickly understand deficiencies in test coverage
- Understand coverage of new and changed code

# Installation

- Apply TPF APAR to test system and configure service endpoints
- Deploy and configure web application on development system
- Update automation scripts
- Install and configure ADI

# The big picture – ADI



# Demo

- Creating manual build data provider
- Creating analysis collections
- Adding data to a data provider
- Adding builds to a data provider
- Refreshing data provider
- Viewing analysis

# Creating data provider

- Select the Providers icon
- Click the Add icon in the header
- Select the Manual Builds icon
- Add First Build and Data Provider information
- Enable headless collection support if desired
- Manual or automatic collection

# Creating analysis collections

- Select the Analysis Collections icon
- Click the Add icon in the header
- Add Name and Description information
- Add permissions
- Select one or many data providers to include



# Adding data to data provider

- Can manually add code coverage results to builds whenever
- Each provider is given a unique key
- Headless collection will create a folder for the data provider based on the unique key
- Root folder location for code coverage results is customizable
- Results placed in that folder will be consumed on manual refresh or timed intervals
- Result files are deleted after consumed

# Adding builds to data provider

- Select Providers icon
- Select Add Build action icon in associated data provider
- Provide name and date for build
- Provide file change information for build
- Newly collected results are associated with the added build

# Refreshing data provider

- Select Providers icon
- Select Refresh action icon under the associated data provider
- Collects results from the headless collection directory

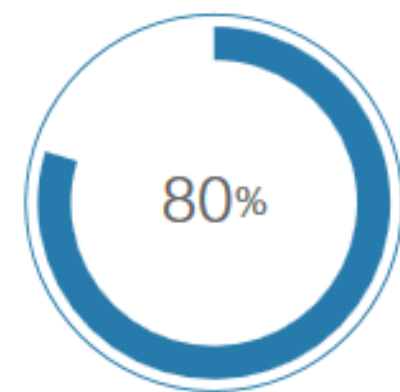


## < EBUD-Retirement-Calculator

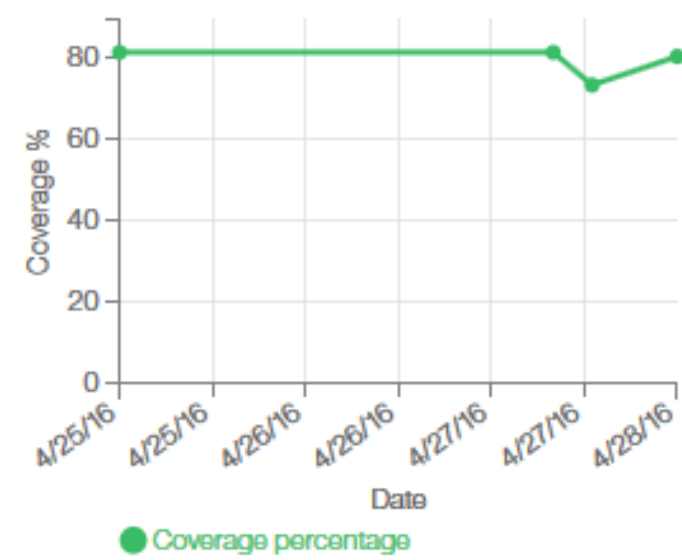
Last Update Today at 11:23

### Manual Test Cycles : Test Execution Cycle 04

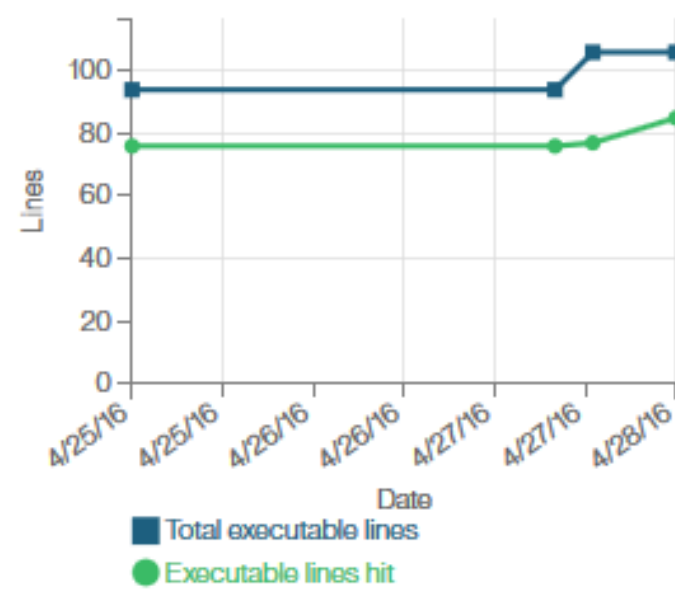
Code Coverage



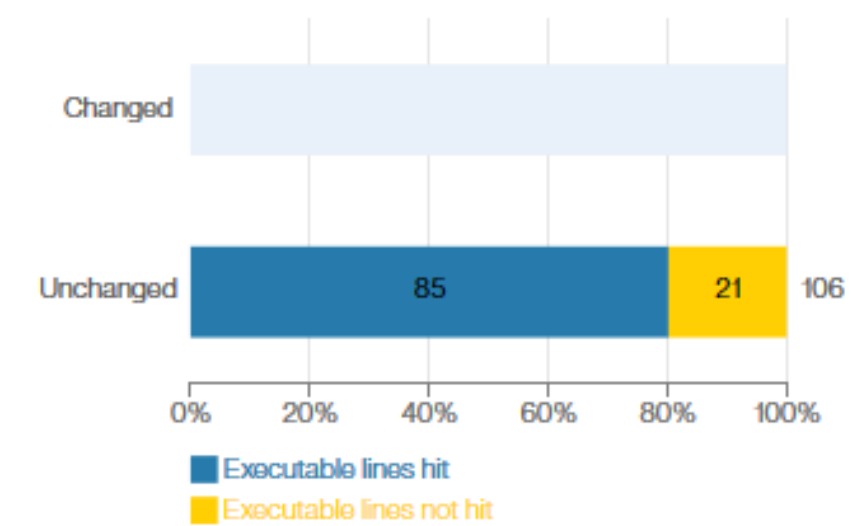
Code Coverage Percentage Trend



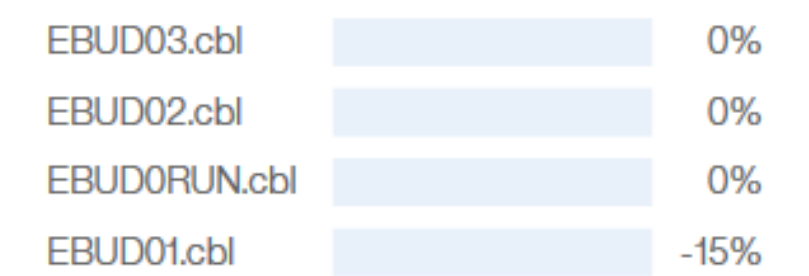
Executable Lines Covered Trend



Coverage Changed vs Unchanged Code



Biggest Coverage Drops



Files with Lowest Coverage



## < Test Execution Cycle 03 ▾









EBUD-Retirement-Calculator | Manual Test Cycles

Date of Build  
Apr 27, 2016, 1:00:00 PM

Last Coverage Data Update  
Apr 27, 2016, 9:17:15 AM

Code Coverage  
73%

Exec. Line Changes  
18%

Name	Code Coverage	Coverage Change	Executable Lines Changed	Change Percentage	Historical Tests	Minimal Tests	Warnings
▶ EBUD01.cbl	 65%	-16% ↓	 19 of 55	35%	3	2	⚠
▶ EBUD02.cbl	 77%	0%	 0 of 13	0%	2	1	
▶ EBUD03.cbl	 78%	0%	 0 of 23	0%	2	1	
▶ EBUD0RUN.cbl	 87%	0%	 0 of 15	0%	3	1	



## < Test Execution Cycle 03 ▾

EBUD-Retirement-Calculator | Manual Test Cycles

Date of Build  
Apr 27, 2016, 1:00:00 PM

Last Coverage Data Update  
Apr 27, 2016, 9:17:15 AM

Code Coverage  
73%

Exec. Line Changes  
18%

Name	Code Coverage	Coverage Change	Executable Lines Changed	Change Percentage	Historical Tests	Minimal Tests	Warnings
EBUD01.cbl	65%	-16% ↓	19 of 55	35%	3	2	
<b>Executable Lines:</b>		<b>Executable Lines Changed:</b>		<b>Historical Tests to Run:</b>		<b>Minimal Tests to Run:</b>	
Total	55	Added	12	TESTA	TESTB		
Hit	36	Updated	7	TESTB	TESTA		
		Deleted	0	TESTC			
EBUD02.cbl	77%	0%	0 of 13	0%	2	1	
EBUD03.cbl	78%	0%	0 of 23	0%	2	1	
EBUD0RUN.cbl	87%	0%	0 of 15	0%	3	1	





## < EBUD01.cbl ▾

EBUD-Retirement-Calculator | Manual Test Cycles | Test Execution Cycle 03

Code Coverage  
65% (36/55)

Coverage Change  
-16% ↓

Added  
12

Executable Lines Changed  
7

Deleted  
0

Name	Code Coverage	Coverage Change	Executable Lines Changed	Change Percentage	Historical Tests	Minimal Tests	Warnings
▶ A000-MAINLINE	100%	0%	0 of 4	0%	3	1	
▶ END-OF-SECTION	0%	0%	0 of 1	0%	0	0	
▶ A100-VERIFY-INPUT-DATE	100%	0%	0 of 8	0%	3	2	
▶ A200-CALL-DAY-DIFFERENCE-PROG	67%	0%	0 of 12	0%	2	1	
▶ A300-CALCULATE-RETIREMENT	48%	-32% ↓	19 of 27	70%	2	1	





## < EBUD01.cbl ▾

EBUD-Retirement-Calculator | Manual Test Cycles | Test Execution Cycle 03

Code Coverage **65% (36/55)** Coverage Change **-16%** ↓ Added **12** Executable Lines Changed **7** Deleted **0**

Name	Code Coverage	Coverage Change	Executable Lines Changed	Change Percentage	Historical Tests	Minimal Tests	Warnings
▶ A000-MAINLINE	<div style="width: 100%;"><div style="width: 100%;"></div></div> 100%	0%	<div style="width: 25%;"><div style="width: 25%;"></div></div> 0 of 4	0%	3	1	
▶ END-OF-SECTION	<div style="width: 0%;"><div style="width: 0%;"></div></div> 0%	0%	<div style="width: 0%;"><div style="width: 0%;"></div></div> 0 of 1	0%	0	0	⚠
▶ A100-VERIFY-INPUT-DATE	<div style="width: 100%;"><div style="width: 100%;"></div></div> 100%	0%	<div style="width: 12.5%;"><div style="width: 12.5%;"></div></div> 0 of 8	0%	3	2	
▶ A200-CALL-DAY-DIFFERENCE-PROG	<div style="width: 67%;"><div style="width: 67%;"></div></div> 67%	0%	<div style="width: 8.3%;"><div style="width: 8.3%;"></div></div> 0 of 12	0%	2	1	⚠
▾ A300-CALCULATE-RETIREMENT	<div style="width: 48%;"><div style="width: 48%;"></div></div> 48%	<b>-32%</b> ↓	<div style="width: 70%;"><div style="width: 70%;"></div></div> 19 of 27	70%	2	1	⚠

Executable Lines:		Executable Lines Changed:		Historical Tests to Run:		Minimal Tests to Run:	
Total	27	Added	12	TESTA	TESTA		
Hit	13	Updated	7	TESTC			
		Deleted	0				





## < Test Execution Cycle 05 ▾

EBUD-Retirement-Calculator | Manual Test Cycles

Date of Build

May 6, 2016, 11:00:00 AM

Last Coverage Data Update


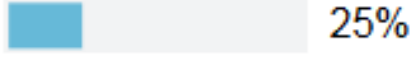

May 18, 2016, 1:24:09 PM

Code Coverage

80%

Exec. Line Changes

0%

Test	Code Coverage	Files Tested by Current Build	Executable Lines Covered	Files Tested by Previous Builds
TESTA	 69%	EBUD01.cbl EBUD02.cbl EBUD03.cbl EBUD0RUN.cbl	74 / 106	EBUD01.cbl EBUD02.cbl EBUD03.cbl EBUD0RUN.cbl
TESTB	 25%	EBUD01.cbl EBUD0RUN.cbl	18 / 70	EBUD01.cbl EBUD0RUN.cbl
TESTD	 70%	EBUD01.cbl EBUD02.cbl EBUD03.cbl EBUD0RUN.cbl	75 / 106	EBUD01.cbl EBUD02.cbl EBUD03.cbl EBUD0RUN.cbl

**Actions**

- [View by Source Files](#)
- [View by Tests](#)
- [Compare Builds](#)

**Filter**

Changed Files Only

Filter Coverage Percentage

0%
100%

Clear Filter



## < Compare Build ▾

EBUD-Retirement-Calculator | Manual Test Cycles

### Test Execution Cycle 04

Date of Build: Apr 28, 2016, 12:00:00 AM  
 Last Coverage Data Update: Apr 27, 2016, 9:37:17 AM  
 Code Coverage: 80%  
 Exec. Line Changes: 0%



### Test Execution Cycle 03

Date of Build: Apr 27, 2016, 1:00:00 PM  
 Last Coverage Data Update: Apr 27, 2016, 9:17:15 AM  
 Code Coverage: 73%  
 Exec. Line Changes: 18%



Warnings	Name	Build Name	Language	Code Coverage	Coverage Change	Executable Lines			Historical Tests to Run		
						Hit	Total	Deleted			
	EBUD01.cbl	Test Execution Cycle 04	COBOL	<div style="width: 80%;"><div style="width: 80%;"></div></div> 80%	15% ↑	44	55	0	0	0	TESTA TESTB TESTC TESTD
		Test Execution Cycle 03	COBOL	<div style="width: 65%;"><div style="width: 65%;"></div></div> 65%	-16% ↓	36	55	12	7	0	TESTA TESTB TESTC
	EBUD02.cbl	Test Execution Cycle 04	COBOL	<div style="width: 77%;"><div style="width: 77%;"></div></div> 77%	0%	10	13	0	0	0	TESTA TESTC TESTD
		Test Execution Cycle 03	COBOL	<div style="width: 77%;"><div style="width: 77%;"></div></div> 77%	0%	10	13	0	0	0	TESTA TESTC
	EBUD03.cbl	Test Execution Cycle 04	COBOL	<div style="width: 78%;"><div style="width: 78%;"></div></div> 78%	0%	18	23	0	0	0	TESTA TESTC TESTD
		Test Execution Cycle 03	COBOL	<div style="width: 78%;"><div style="width: 78%;"></div></div> 78%	0%	18	23	0	0	0	TESTA

### Actions

- View by Source Files
- View by Tests
- Compare Builds

### Filter

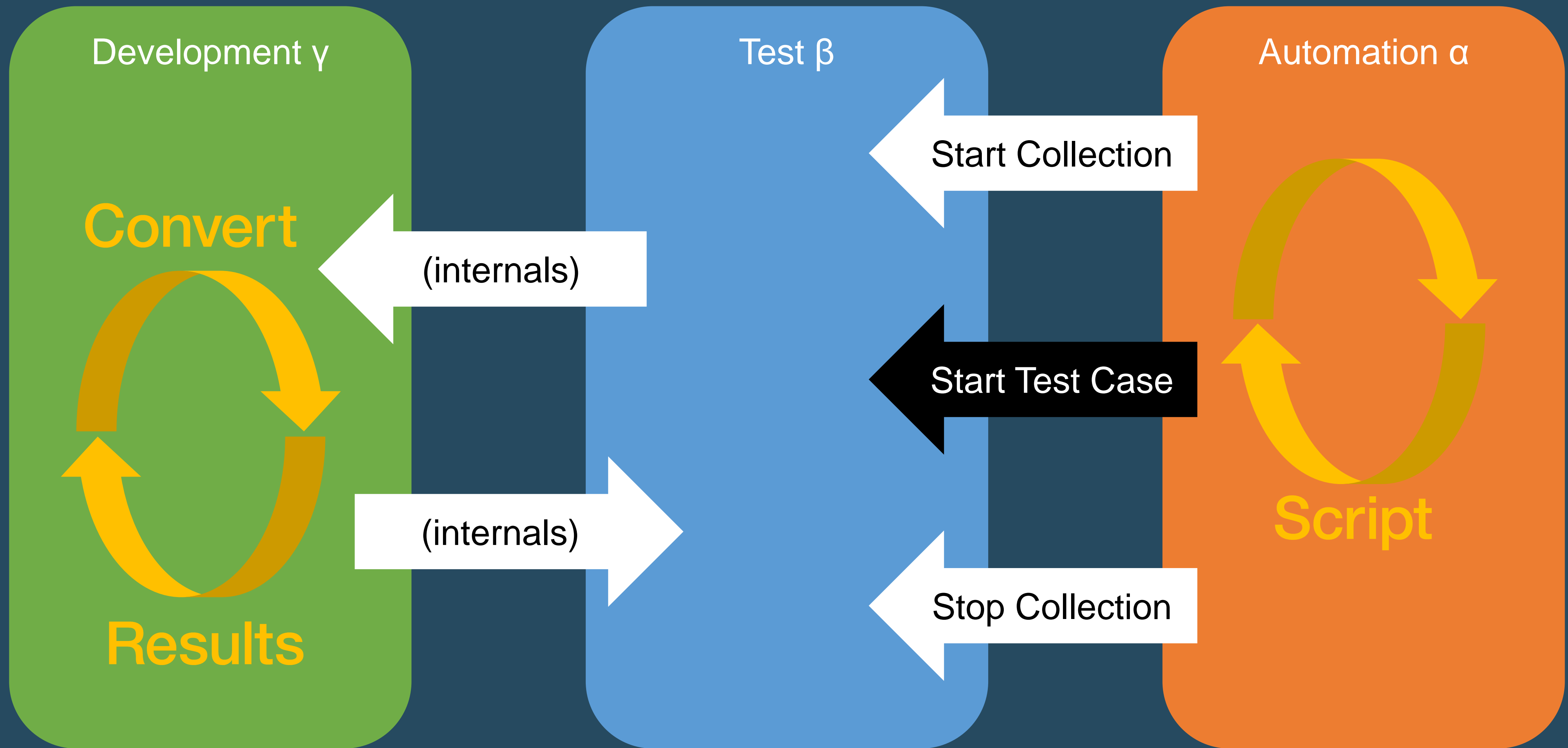
Changed Files Only

Filter Coverage Percentage

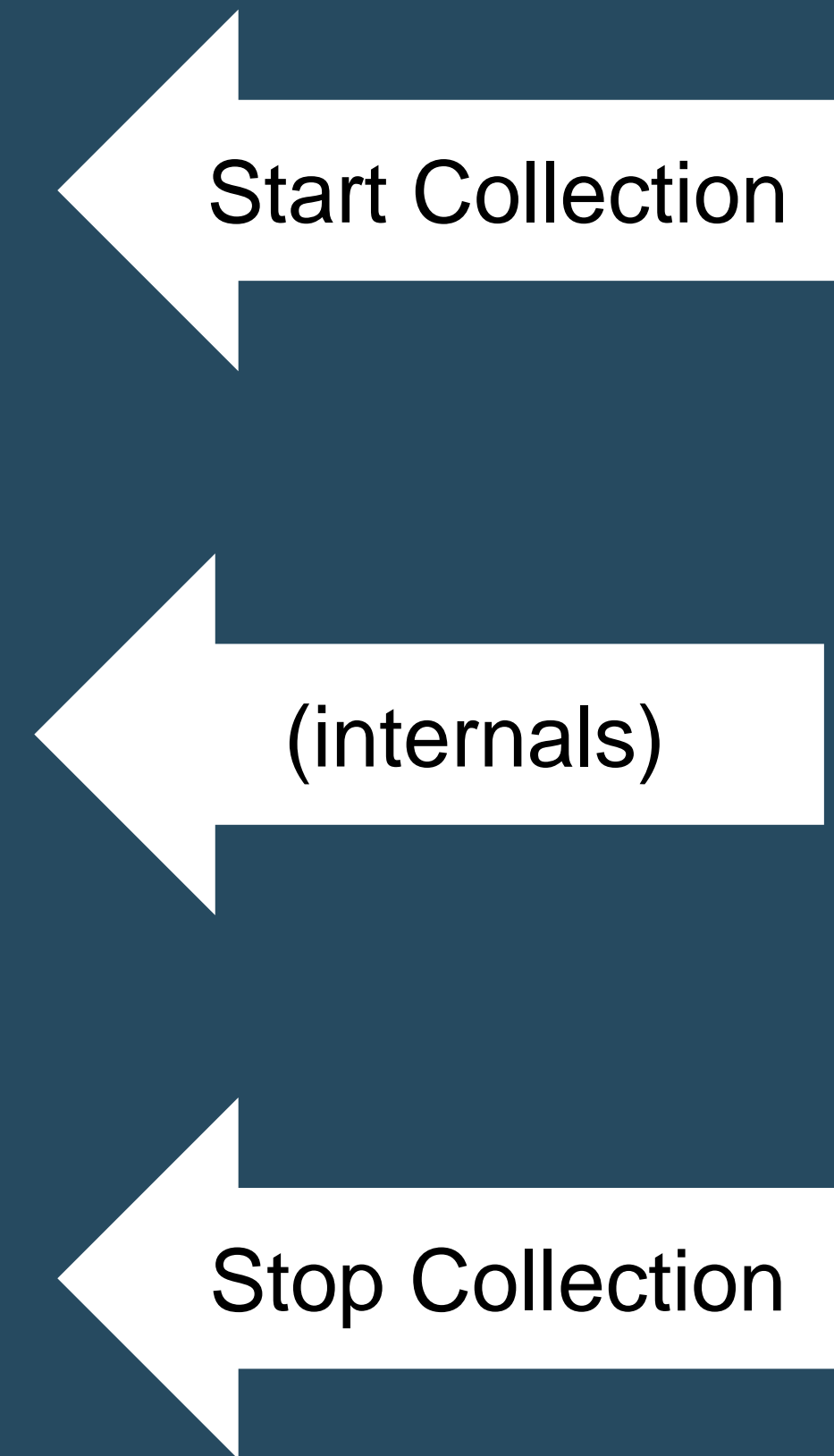


Clear Filter

# The big picture – Code Coverage



- z/TPF services are implemented via the REST service infrastructure available on the z/TPF system
- Use any test infrastructure that supports making REST calls
  - CURL
  - JUnit w/ JAX-RS implementation
  - Etc...
- Additional services deployed to an application server on the development system to format the results and access source code for additional analysis in ADI
- OpenAPI descriptor provided in the TPF APAR to identify development system endpoint for REST calls, host and base path information will need to be customized and deployed



# Start collection

## Request:

Array of program names

Subsystem name

Terminal

User Token

## Response:

JSON object with fields needed for  
Stop collection service

-- or --

Error message

# Stop collection

## Request:

- \* Workstation name
- \* Session name
- \* Timestamp

Results output path

Test case name

Enable source code collection

Path substitutions

Linux endpoint

## Response:

ACCEPTED status code

\* - values returned from Start collection service

# Demo

- Calling START – JAX-RS
- Calling STOP – JAX-RS
- Calling START – CURL
- Calling STOP - CURL

# START – JAX-RS

- Construct Client object
- Construct WebTarget object
- Build START JSON request
- Send Request / Get Response
- Parse response for STOP information
- Close response object
- See example slides



# STOP – JAX-RS

- Construct Client object
- Construct WebTarget object
- Build STOP JSON request
- Send Request / Get Response
- Close response object
- See example slides

# START – CURL

```
curl -X POST -H "Content-Type:application/json"  
-d '{"programs":[{"QDB0"}],"subsystem":"BSS","terminal":"*","userToken":"mrgritte"}'  
"http://9.57.13.36:81/tpf/tools/ccv/services";
```

# STOP – CURL

```
curl -X PUT -H "Content-Type:application/json"  
-d '{"workstationName":$workstationName,  
"session":$session,  
"timestamp":$timestamp,  
"linux_path_for_results":"/IBM/ADI/headless-CC-files/CAWDVE-files",  
"testcase_name":"MyTest",  
"collect_source":true,  
"directorySubstitution":"/ztpf/bld:/ztpf/cur",  
"linux_url":"http://linuxtpf.pok.ibm.com:8080/toolkit/convert/tpf_code_coverage"}'  
  
"http://9.57.13.36:81/tpf/tools/ccv/services";
```



# THANK YOU

Questions or comments?

**Matt Gritter**  
TPF Toolkit

© 2017 IBM z/TPF | TPF Users Group Spring Conference | IBM Confidential

IBM z/TPF  
April 4th, 2017

IBM, the IBM logo and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at “[Copyright and trademark information](#)” at [www.ibm.com/legal/copytrade.shtml](http://www.ibm.com/legal/copytrade.shtml).

## Notes

Performance is in Internal Throughput Rate (ITR) ratio based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput improvements equivalent to the performance ratios stated here.

All customer examples cited or described in this presentation are presented as illustrations of the manner in which some customers have used IBM products and the results they may have achieved. Actual environmental costs and performance characteristics will vary depending on individual customer configurations and conditions.

This publication was produced in the United States. IBM may not offer the products, services or features discussed in this document in other countries, and the information may be subject to change without notice. Consult your local IBM business contact for information on the product or services available in your area.

All statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.

Information about non-IBM products is obtained from the manufacturers of those products or their published announcements. IBM has not tested those products and cannot confirm the performance, compatibility, or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Prices subject to change without notice. Contact your IBM representative or Business Partner for the most current pricing in your geography.

This presentation and the claims outlined in it were reviewed for compliance with US law. Adaptations of these claims for use in other geographies must be reviewed by the local country counsel for compliance with local laws.

```
public class TPFUGExamples {  
  
    private String workstationName = null;  
    private String session = null;  
    private String timestamp = null;  
  
    public final static String TPF_HOST = "http://9.57.13.36:81";  
  
    @Test  
    public void startCodeCoverageCollection() {  
        // Construct Client and WebTarget objects to point to service uri  
        Client client = ClientBuilder.newClient();  
        WebTarget target = client.target(TPF_HOST)  
            .path("tpf")  
            .path("tools")  
            .path("ccv")  
            .path("services");  
  
        // Construct JSON Object Builder to create request entity  
        JsonObjectBuilder entityBuilder = Json.createObjectBuilder();  
  
        // First is an array of programs, need additional builder  
        JsonArrayBuilder programs = Json.createArrayBuilder();  
        programs.add("QDB0");  
        // Add additional programs as needed  
  
        // Add program list to entity  
        entityBuilder.add("programs", programs);  
  
        // Add subsystem to entity  
        entityBuilder.add("subsystem", "BSS");  
    }  
}
```

```
// Add terminal to entity - optional
entityBuilder.add("terminal", "*");

// Add usertoken to entity - optional
entityBuilder.add("userToken", "mrgritte");

// Retrieve JSON object from builder
JsonObject jsonRequest = entityBuilder.build();

// Retrieve String entity from JSON
String entity = jsonRequest.toString();

// Prepare Response object
Response response = null;

// if request returns with error, exception will be throw so surround
// with try-catch block
try {
    response = target.request(MediaType.APPLICATION_JSON)
        .post(Entity.entity(entity, MediaType.APPLICATION_JSON));
    int status = response.getStatus();
    // Do something with other unexpected responses
    if (status != 200) {
        throw new Exception(response.getStatusInfo().getReasonPhrase());
    }
} catch (Exception e) {
    // do something with the error cases
    e.printStackTrace();
} finally {
    // always close the response object before continuing
    if(response != null)
```

```
        response.close();
        // also close out client if no longer being used
        client.close();
    }

    // out of the try block so first check if response is null in case
    // something happened
    if (response != null) {
        String jsonResponse = response.readEntity(String.class);
        StringReader stringReader = new StringReader(jsonResponse);
        JsonReader responseReader = Json.createReader(stringReader);
        JsonObject responseObject = responseReader.readObject();

        // close readers now
        stringReader.close();
        responseReader.close();

        // Read out the passed strings for use in stop
        this.workstationName = responseObject.getString("workstationName");
        this.session = responseObject.getString("session");
        this.timestamp = responseObject.getString("timestamp");
    }
}
```



```
public void stopCodeCoverageCollection() {
    // Construct client and target objects
    Client client = ClientBuilder.newClient();
    WebTarget target = client.target(TPF_HOST)
        .path("tpf")
        .path("tools")
        .path("ccv")
        .path("services");

    // Construct JSON Object Builder to create request entity
    JsonObjectBuilder entityBuilder = Json.createObjectBuilder();

    // add workstation name
    if (this.workstationName != null) {
        entityBuilder.add("workstationName", this.workstationName);
    } else {
        return;
    }

    // add session
    if (this.session != null) {
        entityBuilder.add("session", this.session);
    } else {
        return;
    }
}
```

```
// add timestamp
if (this.timestamp != null) {
    entityBuilder.add("timestamp", this.timestamp);
} else {
    return;
}

// add path to place results on linux
entityBuilder.add("linux_path_for_results", "/IBM/ADI/headless-CC-files/CAWDVE-files");

// add testcase name
entityBuilder.add("testcase_name", "MyTest");

// add collect source boolean
entityBuilder.add("collect_source", true);

// add source path substitution string - optional
entityBuilder.add("directorySubstitutions", "/ztpf/bld:/ztpf/cur");

// add linux endpoint url - the root location where you deployed the
// linux web application
// optional - configurable on TPF system
entityBuilder.add("linux_url", "http://linuxtpf.pok.ibm.com/toolkit");

// get JSON object from builder
JsonObject jsonRequest = entityBuilder.build();

// get String entity from JSON object
String entity = jsonRequest.toString();
```

```
// Prepare response object
Response response = null;

// Catch any errors
try {
    response = target.request().put(Entity.entity(entity, MediaType.APPLICATION_JSON));
    int status = response.getStatus();
    // Check for any unexpected status codes
    if (status != 202) {
        throw new Exception(response.getStatusInfo().getReasonPhrase());
    }
} catch (Exception e) {
    e.printStackTrace();
} finally {
    // close response object
    if(response != null)
        response.close();
    // close client object
    client.close();
}

// done once 202 received
}
```