



Validating Integrity Setup

PTC®

Agenda

In this Session

Automated Setup Testing with Integrity STeF

Automated Test Execution with Integrity's ATEF

Not in this Session

Validating Integrity Setup

PTC®

Challenges

How can we test that?

3

Validating Integrity Setup

PTC®

How do you validate Integrity Setup today?

4

Setup Testing Framework for Integrity

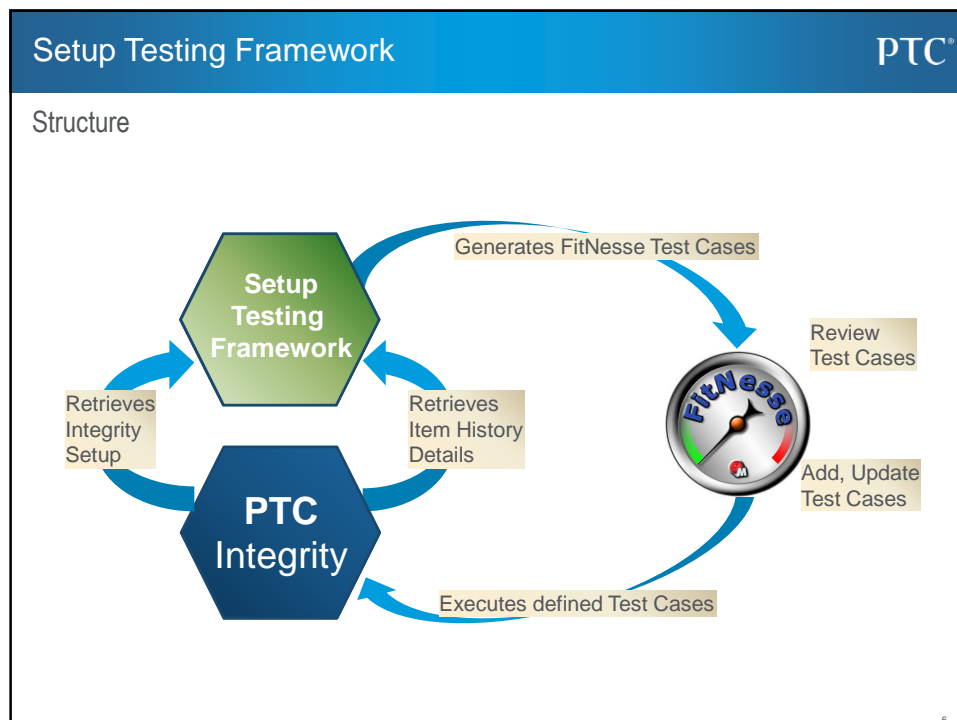
PTC®

Key Features

STeF

- 1 Setup Testing Framework allows you to
 - Define Test Cases
 - Execute them step-by-step, or fully automated
 - Track results
 - Provide a test summary and statistical information
 - Test protocol
- 2 Setup Test Case Retriever
 - Captures Item History entered
 - Automatically creates test cases for them
 - Allows further tailoring

5




Setup Testing Framework


PTC®

Components

- **FitNesse**
 - Web Based Functional Integration Test Tool
 - Open Source Project
 - Easy to Install
 - Many (still unused) Features


<http://www.fitnesse.org/>

- **Setup Testing Framework**
 - Custom developed Testing Framework
 - Utilizing FitNesse Techniques
 - Written in Java
 - Easy to Configure
 - Extendable
 - Can be integrated in Integrity



7

PTC®


Writing and Executing Tests for Integrity

Within FitNesse

8

Test Cases in FitNesse

PTC®


[FrontPage](#) > [IntegrityTesting](#) > [TestSuite](#)
RequirementDocument

Test

Edit

Add

Tools

In this scenario we create a requirement document, and create a trace to a test case afterwards. Then, we check for the suspect flag, which should be unset. Afterwards we perform a significant update, and recheck the suspect flag again. Now it should be flagged as suspect. Finally we use a trigger to clear the suspect flag again.

1. Initialization - Find Items already in Integrity

Search for the project item with Summary "Project for Release 1" and return the ID, lookup also the Test Case previously created.

Integrity	localhost:7001				
Type	Summary	Text	State	find Issue?	Value?
Project	Project for Release 1		Defined	\$IDP=	>0
Test Case		2020 TC2		\$IDTC2=	>0

2. Create Requirement Document and Requirement


Create a simple requirement document.

Integrity	localhost:7001									
Module	Command	Type	Parent ID	Document Short Title	Project	Text	Category	Result?	Message?	Value?
im	createsegment	Requirement Document		The Requirement Title	/Projects/Release1			0		\$IDRD=
im	createcontent	Requirement	\$IDRD			Any Requirement Text	Functional Requirement	0		\$IDR1=

9

Edit Test Cases in FitNesse


PTC®


[FrontPage](#) > [IntegrityTesting](#) > [TestSuite](#)
RequirementDocument

Rich Text Editor

Normal

B / ABC



In this scenario we create a requirement document, and create a trace to a test case afterwards. Then, we check for the suspect flag, which should be unset. Afterwards we perform a significant update, and recheck the suspect flag again. Now it should be flagged as suspect. Finally we use a trigger to clear the suspect flag again.

1. Initialization - Find Items already in Integrity

Search for the project item with Summary "Project for Release 1" and return the ID, lookup also the Test Case previously created.

Integrity	\$(HOST)				
Type	Summary	Text	State	find Issue?	Value?
Project	Project for Release 1		Defined	\$IDP=	>0
Test Case		2020 TC2		\$IDTC2=	>0

2. Create Requirement Document and Requirement

Create a simple requirement document.

Integrity	\$(HOST)									
Module	Command	Type	Parent ID	Document Short Title	Project	Text	Category	Result?	Message?	Value?
im	createsegment	Requirement Document		The Requirement Title	/Projects/Release1			0		\$IDRD=
im	createcontent	Requirement	\$IDRD			Any Requirement Text	Functional Requirement	0		\$IDR1=

Plain Text Editor

```

define TEST_SYSTEM {slim}
In this scenario we create a requirement document, and create a trace to a test case afterwards. Then, we check for the suspect flag, which should be unset. Afterwards we perform a significant update, and recheck the suspect flag again. Now it should be flagged as suspect. Finally we use a trigger to clear the suspect flag again.

1. Initialization - Find Items already in Integrity
Search for the project item with Summary "Project for Release 1" and return the ID, lookup also the Test Case previously created.

/ Integrity / $(HOST) |
Type | Summary | Text | State | find Issue? | Value? |
Project for Release 1 | | Defined | $IDP= | >0 |
Test Case | | 2020 TC2 | | $IDTC2= | >0 |


2. Create Requirement Document and Requirement
Create a simple requirement document.

Integrity / $(HOST) |
Module | Command | Type | Parent ID | Document Short Title | Project | Text | Category | Result? | Message? | Value? |
im | createsegment | Requirement Document | | The Requirement Title | /Projects/Release1 | | | 0 | | $IDRD= |
im | createcontent | Requirement | $IDRD | | | Any Requirement Text | Functional Requirement | 0 | | $IDR1= |


```

10

Execute Test Cases
PTC®



FrontPage > IntegrityTesting > TestSuite
RequirementDocument



Assertions: 21 right, 0 wrong, 12 ignored, 0 exceptions (1,924 seconds)

1. Initialization - Find Items already in Integrity

Search for the project item with Summary "Project for Release 1" and return the ID, lookup also the Test Case

Integrity	localhost:7001
Type	Summary
ALM_Project	Project for Release 1
Test Case	2020 TC2

2. Create Requirement Document and Requirement

Create a simple requirement document.

Module	Command	Type	Parent ID	Document Short Title	Project	Text	Category	Result?	Message?	Value?
im	createsegment	Requirement Document		The Requirement Title	/ALM_Projects/Release1			0	Item with ID '3247' has been created	\$IDRD-<[3247]
im	createcontent	Requirement	\$IDRD->[3247]			Any Requirement Text	Functional Requirement	0	Item with ID '3248' has been created	\$IDR1-<[3248]

createcontent	Requirement	\$IDRD->[3254]			Requirement Text	Functional Requirement	[-1] expected [0]	MKS124066: Type "Requirement" does not exist.	\$IDR1-<[]
---------------	-------------	----------------	--	--	------------------	------------------------	-------------------	---	-------------

Passed

Failed

11

PTC®

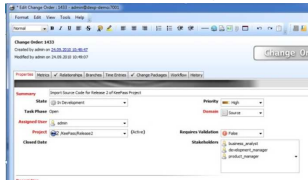
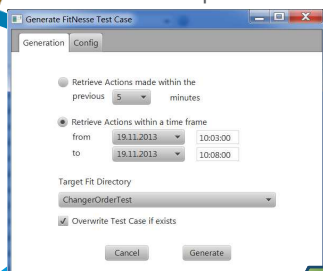
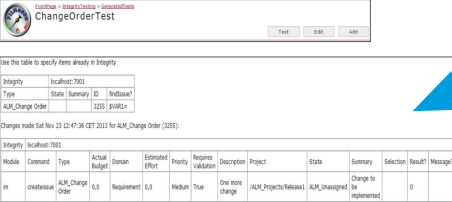
Retrieve Actions from Integrity

to generate Test Cases automatically

12

Retrieve Actions from Integrity

PTC®

- 1 Perform an Action in Integrity
 
- 2 Retrieve the Actions performed
 
- 3 Review and Execute in FitNesse
 

STeF

13

Conclusion

PTC®

14

PTC®
Conclusion

Setup Testing Framework for Integrity




- **Allows you to**
 - Enter Test Cases in FitNesse
 - Perform them against any Integrity Environment for Setup validation
- **Provides**
 - Easy to use interface
 - High Flexibility with Variables
 - Custom Templates
 - Excel Import / Export
- **Based on**
 - Java
 - FitNesse Library




15

PTC®
Conclusion

Setup Testing Framework for Integrity



- **Supports**
 - Step-by-step and/or fully automated execution
 - Automated Test Case generation
- **Can be used by**
 - System and Setup Responsibles
 - Testmanagers and Testers
 - Key Users
 - Developers
- **Main Business Advantages**
 - Improves Testing Quality
 - Reduces Test Durations
 - Provides a complete testing Protocol for Compliance Requirements
 - Reduced Costs



16

Thank you !

Your Questions

Volker Eckardt
Senior Consultant, PTC GSO
veckardt@ptc.com



17

PTC® PRODUCT & SERVICE
ADVANTAGE®