



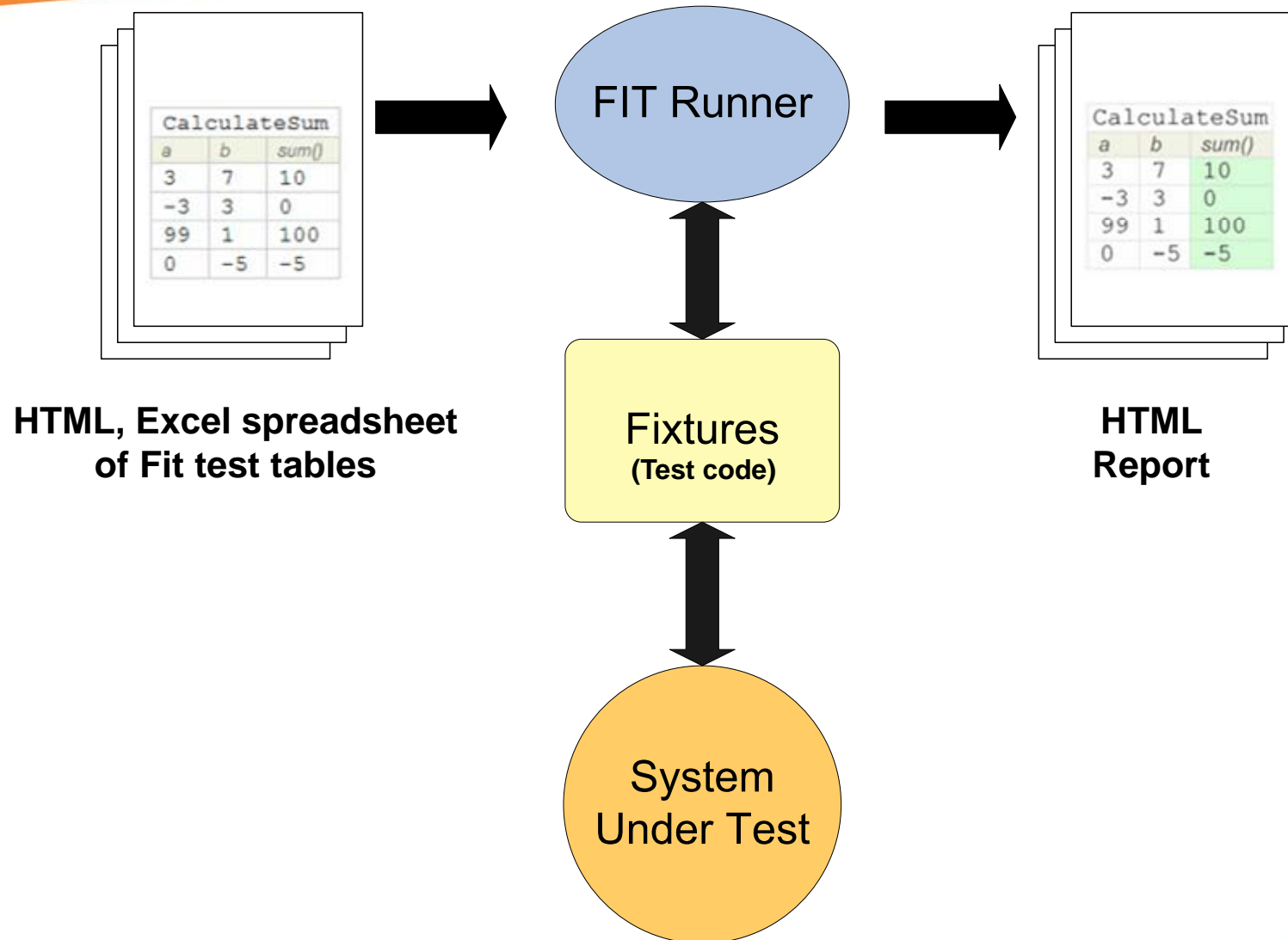
An introduction to Framework for Integrated Tests (FIT) and FITpro

What Is Fit?

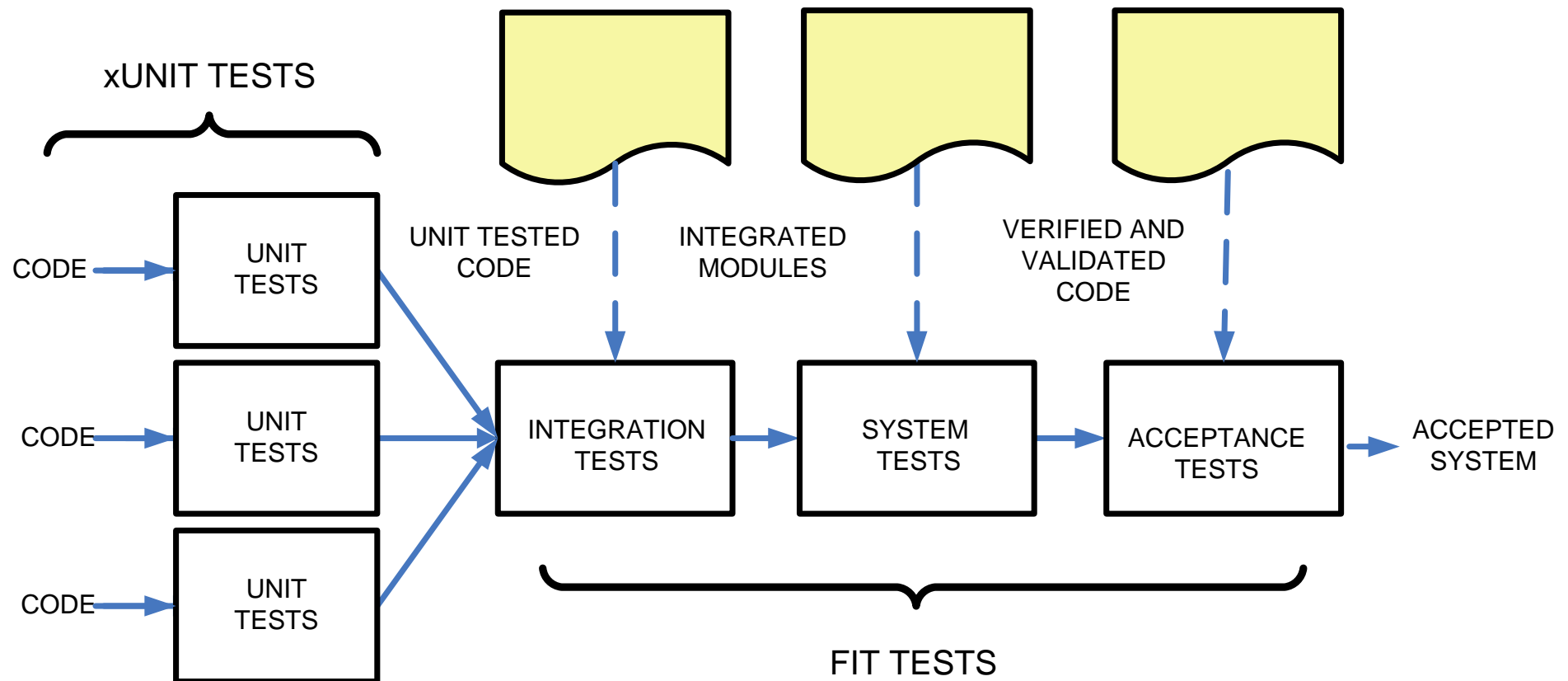


- Test harness for acceptance testing
- Defines tests/requirements in table form
- Collaboration tool
- Provides common language - tenet of domain driven design
- Available for various languages - Java, DotNet, C++, Python, Ruby, Smalltalk

How does Fit work?

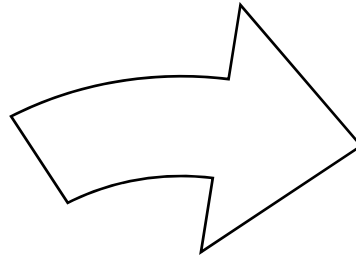


Acceptance Testing

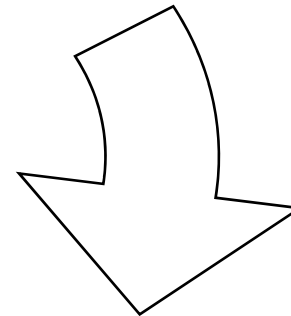


Test-Driven Development & Fit

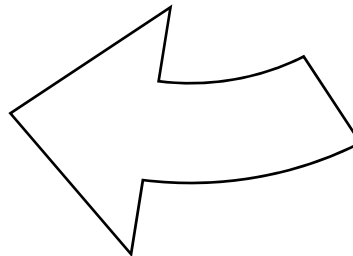
**1. Clarify
requirement
with Fit tests**



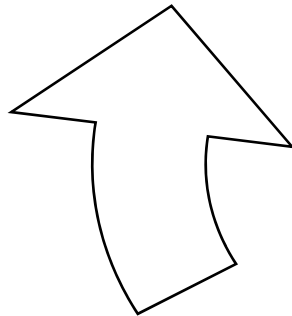
**2. Write high level
requirements**



3. Write code



**4. Test code
with Fit tests**



Writing Requirements with Fit

- Business Analyst will write high level requirements, and then supplement them with Fit tests that provide concrete examples.

High Level Requirements for an Online Bookstore

Business Rule 1. A customer gets free shipping if they spend \$50 or more, else shipping adds a 10% fee.

Business Rule 2: An employee may ask for a list of all books currently in the system. The list includes the book name, author and price

Business Rule 3. A customer selects several books. The running total is accumulated and shown to customers.



Fit Test File

Business Rule 1. A customer gets free shipping if they spend \$50 or more, else shipping adds a 10% fee.

CalculateShippingFees	
amount	shippingFees()
0	0
10	1
25	2.50
49	4.90
50	0
100	0

Business Rule 2: An employee may ask for a list of all books currently in the system. The list includes the book name, author and price.

BookList		
bookname	author	price
Rapid Development	Steve McConnell	100.00
UNIX in a Nutshell	Tim Robbins	22.00

Business Rule 3. A customer selects several books. The running total is accumulated and shown to customers.

Fit.ActionFixture		
start	BuyActions	
check	total	0.00
enter	bookname	Rapid Development
enter	price	100.00
Press	buy	
Check	total	100.00

Fit Test – Column Fixture

- Here's a Fit test table example using a column fixture
- Business Rule 1. A customer gets free shipping if they spend \$50 or more, else shipping adds a 10% fee.

CalculateShippingFees	
amount	shippingFees()
0	0
10	1
25	2.50
49	4.90
50	0
100	0

Fixture Code for Fit Test



- A developer will write test code called a “fixture” to hook the Fit tests into the system under test.
- Here’s example fixture code for our Column Fixture example:

```
public class CalculateShippingFees extends fit.ColumnFixture {  
    public double amount;  
    private ShippingFees application = new ShippingFee();  
  
    public double shippingFees(){  
        return application.getShippingFees(amount);  
    }  
}
```


Fit Report Results

CalculateShippingFees	
Amount	shippingFees()
0	0
10	1
25	2.50
49	4.90
50	0
100	0



CalculateShippingFees	
Amount	shippingFees()
0	0
10	1
25	2.50
49	4.90
50	0 <i>expected</i>
	<hr/>
	5 <i>actual</i>
100	0

Fit Test - Row Fixture

- Business Rule 2: An employee may ask for a list of all books currently in the system. The list includes the book name, author and price.

BookList		
bookname	author	price
Rapid Development	Steve McConnell	100.00
UNIX in a Nutshell	Tim Robbins	22.00

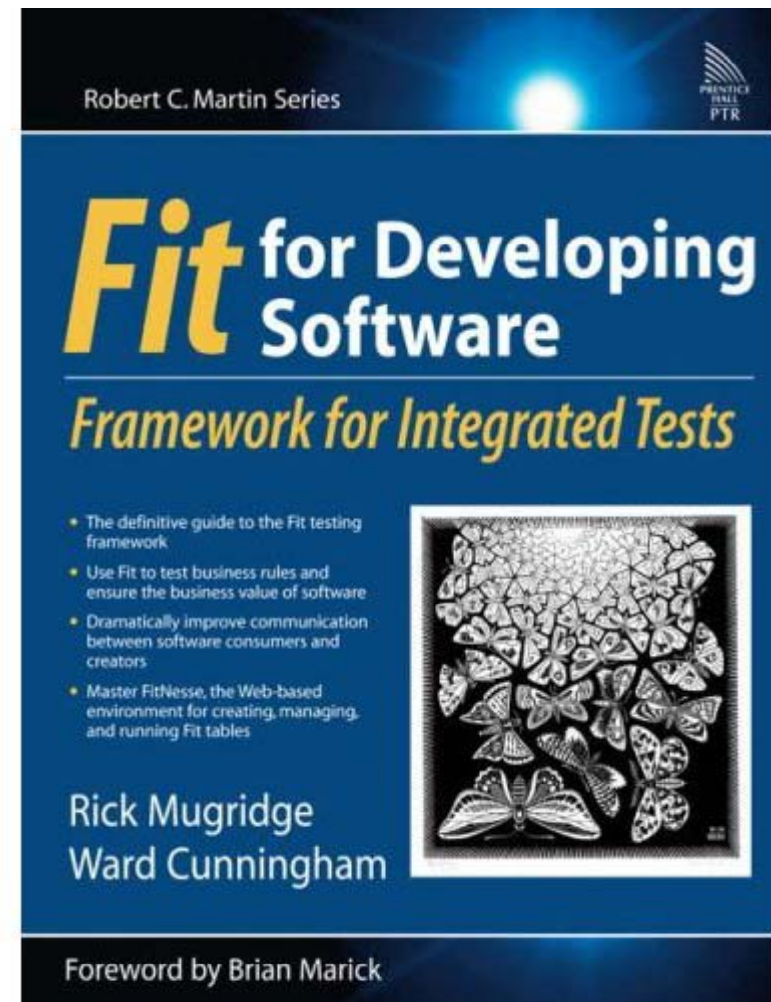
Fit Test - Action Fixture

- Business Rule 3. A customer selects several books. The running total is accumulated and shown to customers.

Fit.ActionFixture		
start	BuyActions	
check	total	0.00
enter	bookname	Rapid Development
Press	buy	
Check	total	100.00

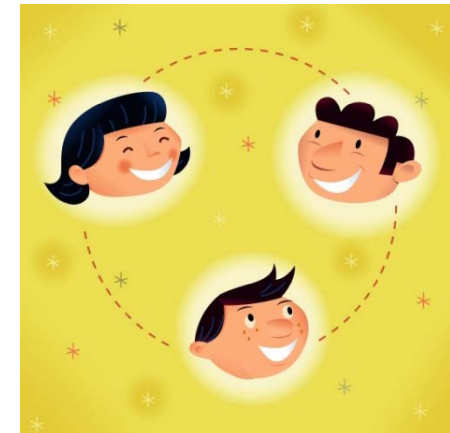
Further Examples

- More examples of use of Fit in testing complex systems in “Fit for Developing Software” by Mugridge and Cunningham



Fit Benefits for Business People

- Clearer communication
- Transparency – business people can clearly see what is tested
- Ensures compliance with your business rules
- Allows for traceability between requirements, tests and code



Fit Benefits for Developers

- Takes guess-work out of interpreting requirements
- Gives confidence that code changes have not broken business logic
- Indicates features meet expectations



Fit Benefits for Testers

- Fit tests are quicker to develop than other automated tests
- Easy to develop tests before code is complete
- Can bypass changing GUI to test business logic
- Finds recurring bugs

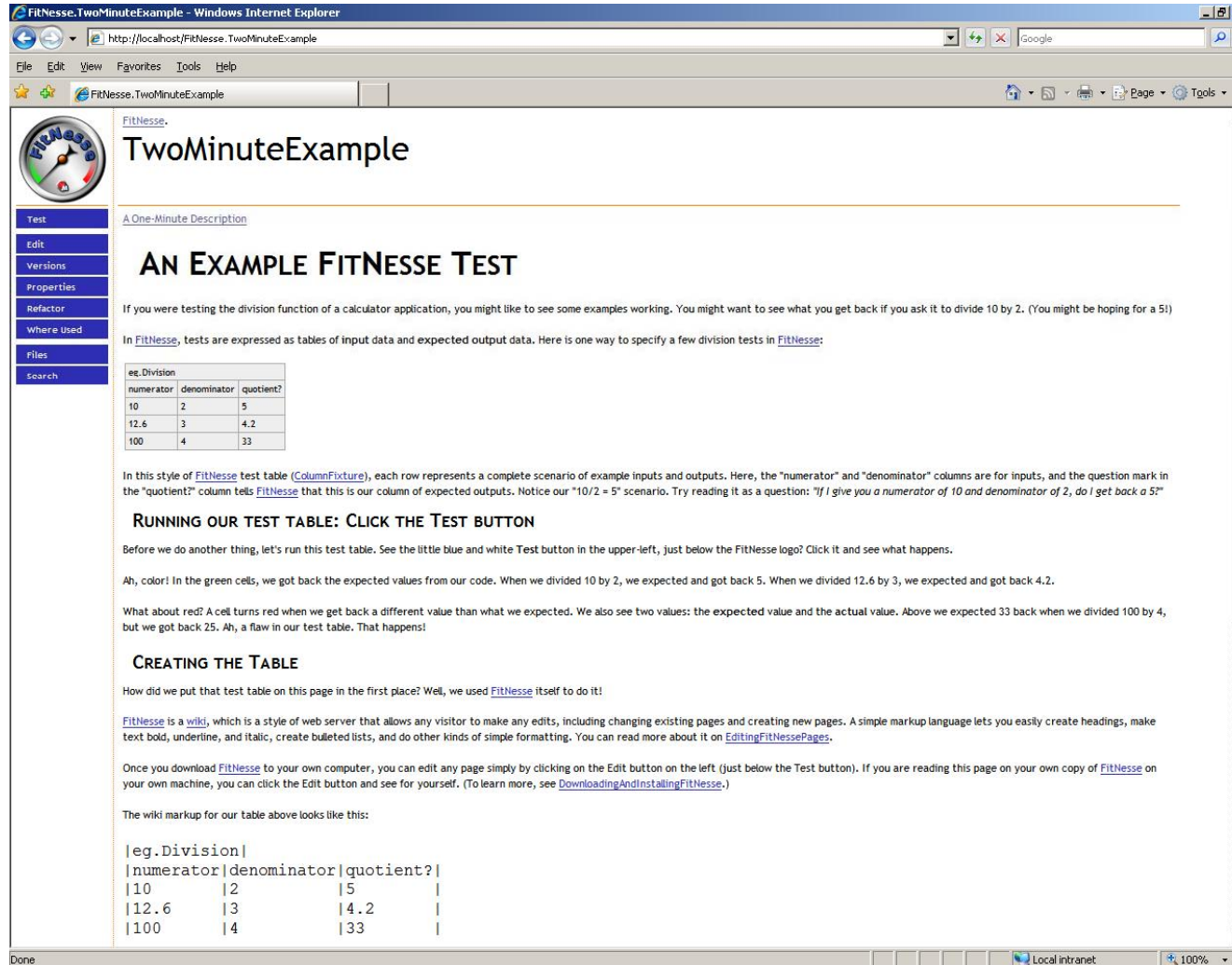


Tools that supplement Fit

- FitLibrary provides new fixture, Do Fixture
- Allows to execute folders of tests
- Various GUI-based Fit Runners that execute Fit tests
- Selenium + Fit can do UI testing within a browser



FitNesse Wiki Server



FitNesse.TwoMinuteExample - Windows Internet Explorer

http://localhost/fitNesse.TwoMinuteExample

File Edit View Favorites Tools Help

FitNesse.TwoMinuteExample

FitNesse

TwoMinuteExample

[Test](#)
[Edit](#)
[Versions](#)
[Properties](#)
[Refactor](#)
[Where Used](#)
[Files](#)
[Search](#)

[A One-Minute Description](#)

AN EXAMPLE FITNESSE TEST

If you were testing the division function of a calculator application, you might like to see some examples working. You might want to see what you get back if you ask it to divide 10 by 2. (You might be hoping for a 5!)

In [FitNesse](#), tests are expressed as tables of input data and expected output data. Here is one way to specify a few division tests in [FitNesse](#):

eg. Division		
numerator	denominator	quotient?
10	2	5
12.6	3	4.2
100	4	33

In this style of [FitNesse](#) test table ([ColumnFixture](#)), each row represents a complete scenario of example inputs and outputs. Here, the "numerator" and "denominator" columns are for inputs, and the question mark in the "quotient?" column tells [FitNesse](#) that this is our column of expected outputs. Notice our "10/2 = 5" scenario. Try reading it as a question: "If I give you a numerator of 10 and denominator of 2, do I get back a 5?"

RUNNING OUR TEST TABLE: CLICK THE TEST BUTTON

Before we do another thing, let's run this test table. See the little blue and white Test button in the upper-left, just below the FitNesse logo? Click it and see what happens.

Ah, color! In the green cells, we got back the expected values from our code. When we divided 10 by 2, we expected and got back 5. When we divided 12.6 by 3, we expected and got back 4.2.

What about red? A cell turns red when we get back a different value than what we expected. We also see two values: the expected value and the actual value. Above we expected 33 back when we divided 100 by 4, but we got back 25. Ah, a flaw in our test table. That happens!

CREATING THE TABLE

How did we put that test table on this page in the first place? Well, we used [FitNesse](#) itself to do it!

[FitNesse](#) is a [wiki](#), which is a style of web server that allows any visitor to make any edits, including changing existing pages and creating new pages. A simple markup language lets you easily create headings, make text bold, underline, and italic, create bulleted lists, and do other kinds of simple formatting. You can read more about it on [EditingFitNessePages](#).

Once you download [FitNesse](#) to your own computer, you can edit any page simply by clicking on the Edit button on the left (just below the Test button). If you are reading this page on your own copy of [FitNesse](#) on your own machine, you can click the Edit button and see for yourself. (To learn more, see [DownloadingAndInstallingFitNesse](#).)

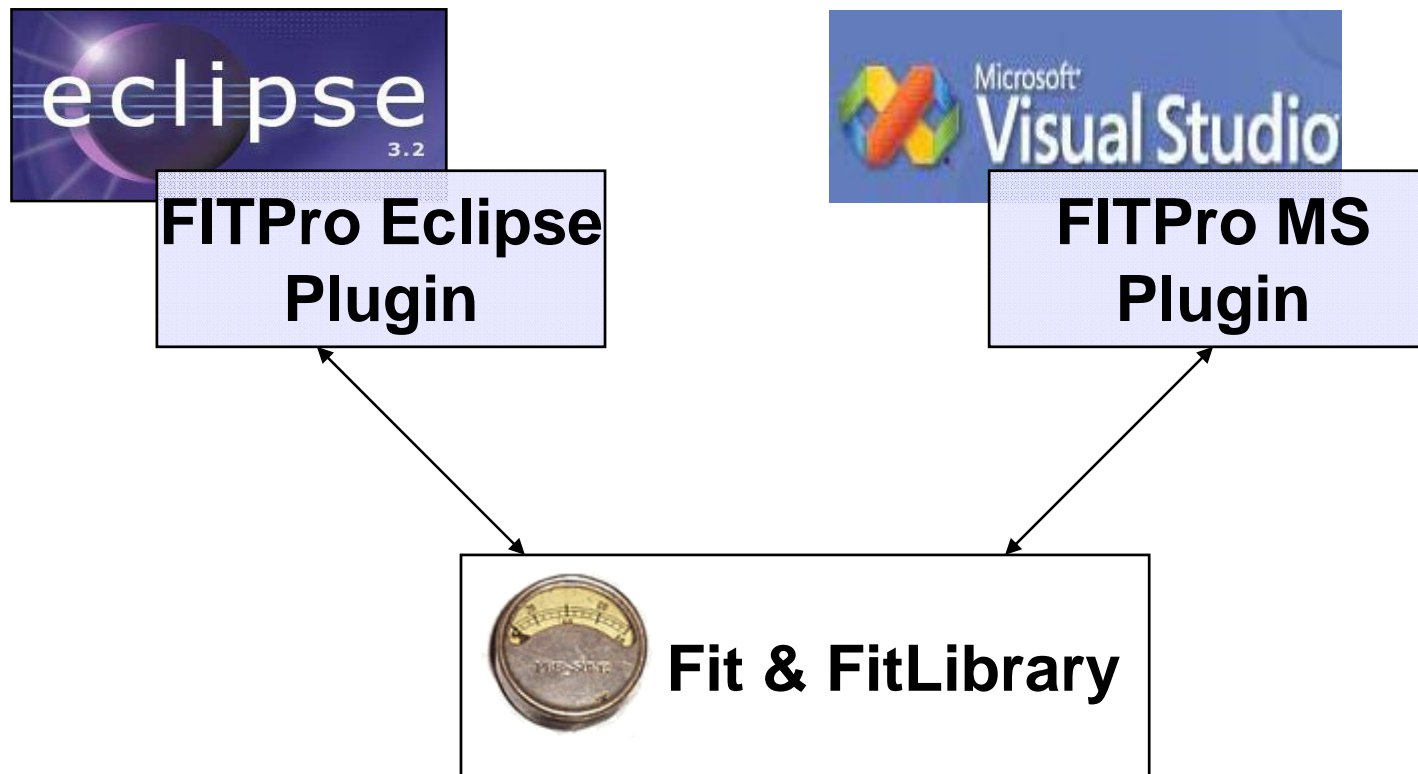
The wiki markup for our table above looks like this:

```
|eg.Division|
|numerator|denominator|quotient?|
|10        |2          |5          |
|12.6      |3          |4.2        |
|100       |4          |33         |
```

Vision for FITPro Solution

Fit Solution that provides:

- Improved productivity & usability of Fit



FITPro Plug-in For Eclipse

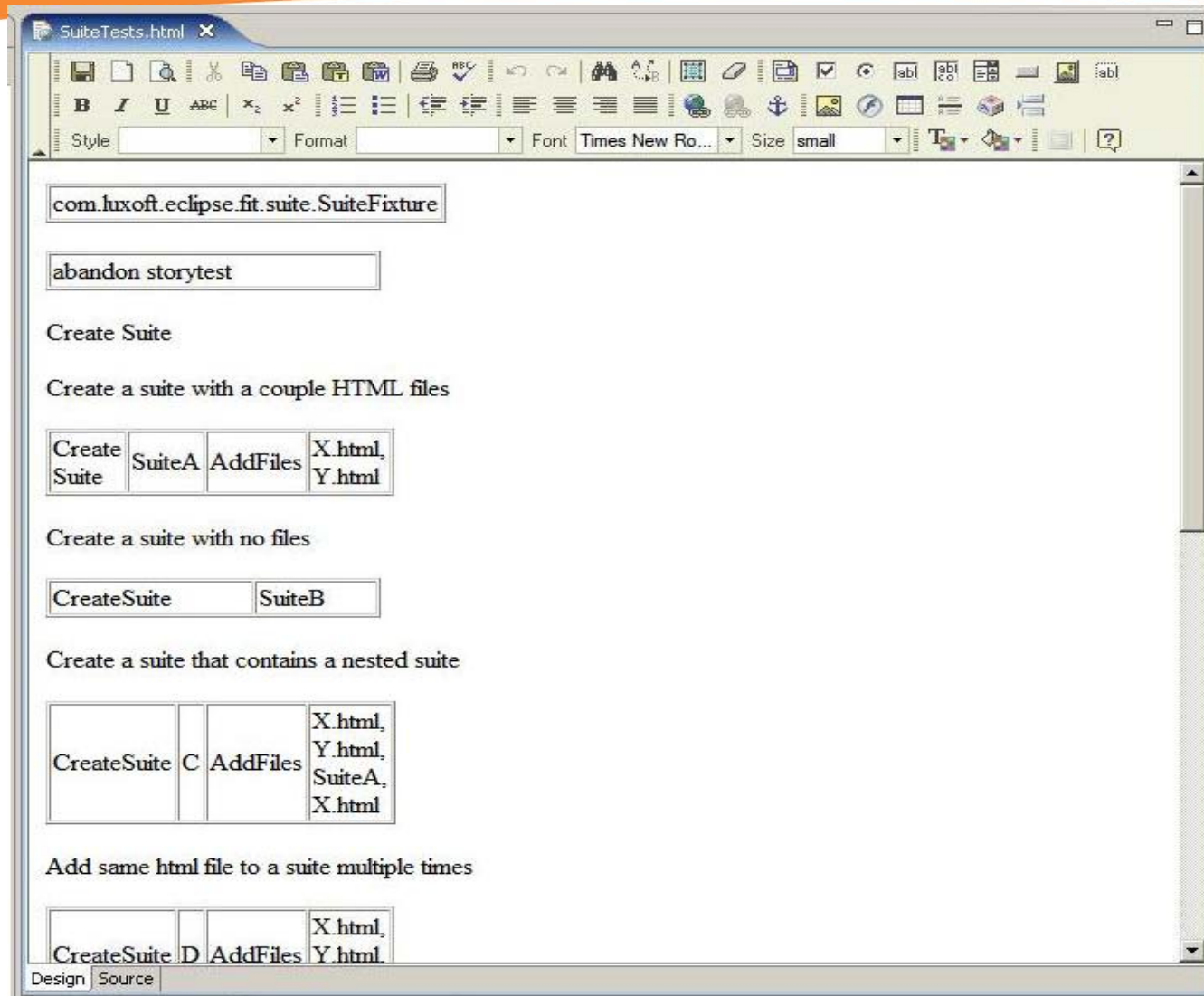


FITPro Eclipse Plug-in:

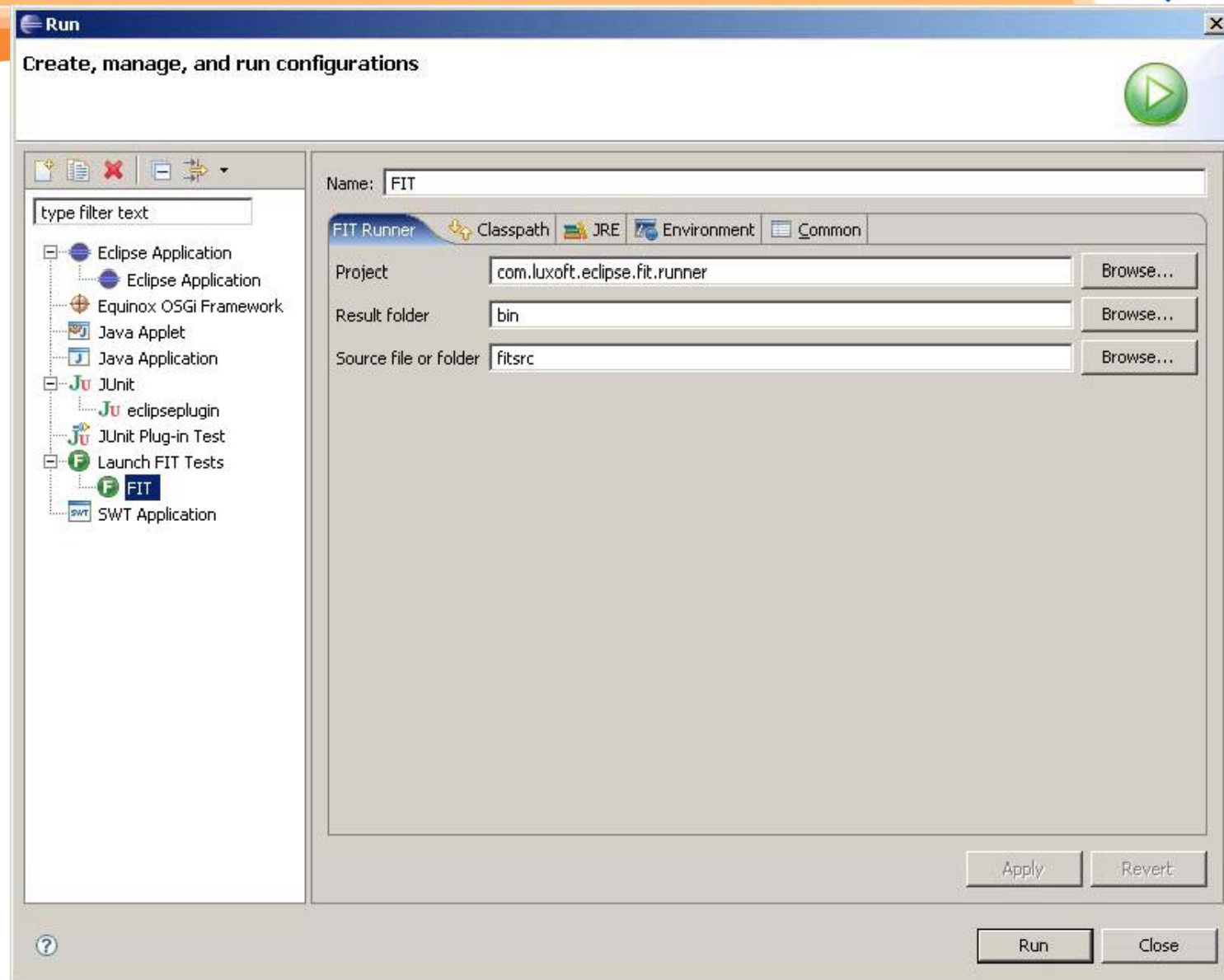
- Simplifies requirement/test creation
- Eases burden in managing tests
- Provides flexible test execution
- Allows for transparency in reporting
- Provides linking between requirements/tests and code



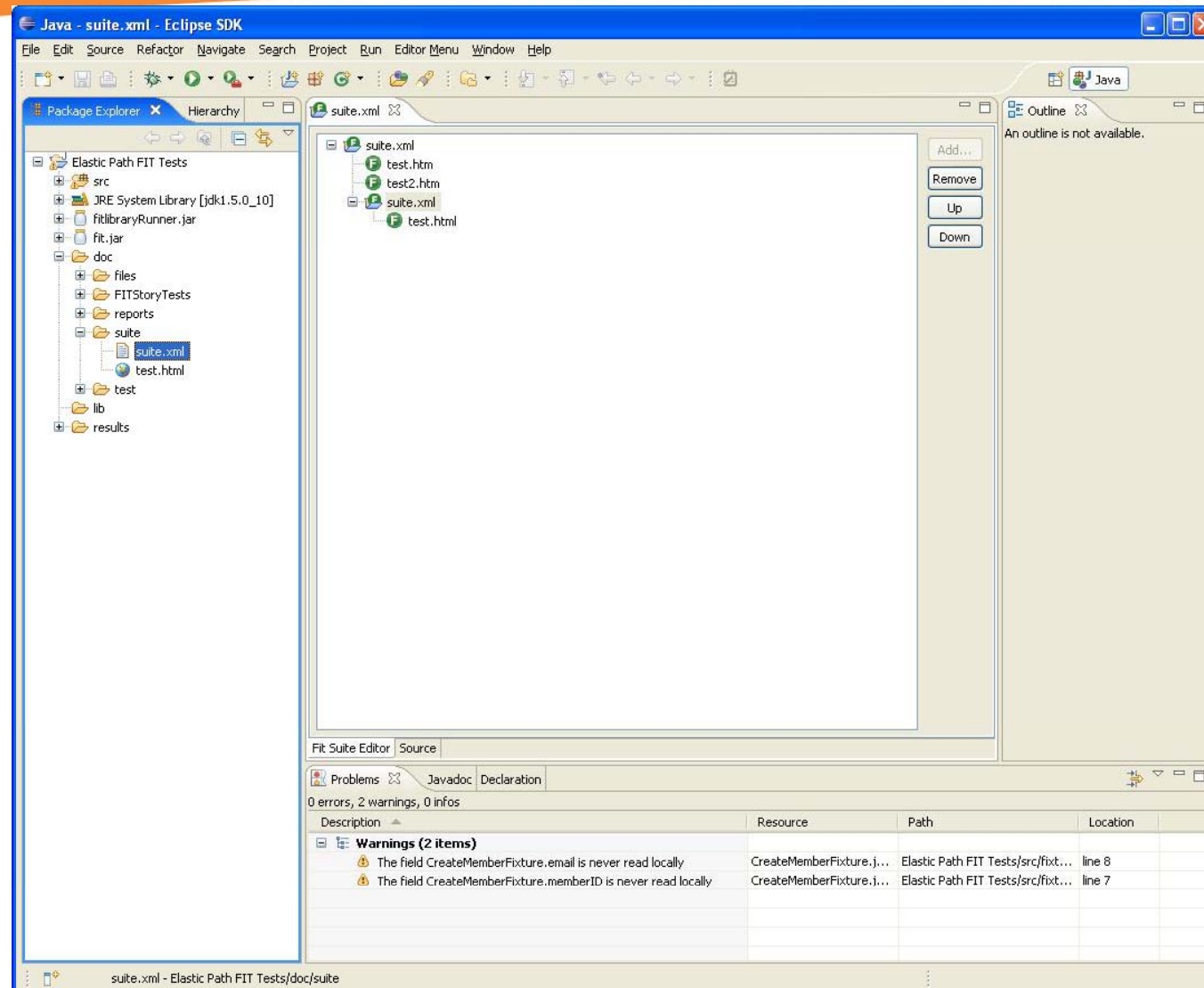
Capture requirements/tests in WYSWIG HTML



Fit Test Execution within Eclipse



Fit Test Suite Management



Link from Tests to Code

```
SuiteTests.fit SuiteFixture.java x
1 package com.luxoft.fit.suite;
2
3 import java.util.ArrayList;
4
11
12 /**
13  * This method hooks into the FIT tests for Suites.
14  */
15 public class SuiteFixture
16     extends DoFixture {
17
18     private Map<String, Suite> suites = new Hashtable<String, Suite>();
19
20     /**
21      * This method creates a FIT suite.
22      *
23      * @param suiteName name of suite you want created
24      * @return boolean indicating if successful in creating suite
25      */
26     public void createSuiteWithFilenameWithSuiteNameWithDescription(final String fileName,
27         final String description) {
28         Suite suite = new Suite(fileName);
29         suite.setLogicalName(logicalName);
30         suite.setDescription(description);
31         suites.put(fileName, suite);
32     }
33
34     public void suiteNameFilename(String fileName) {
35
36     }
37
38     public CalculateFixture checkSuiteProperties() {
39         return new SuitePropertiesFixture(suites);
40     }
41
42     /**
43      * This method creates a FIT suite and add files to it.
44      */
```