Specification by Example
A must for Continuous Integration

Chris C. Schotanus - CGI
Nederlandse Testdag
6 October 2015

Experience the commitment®

Agenda

Short introduction to Test Driven Development

Short introduction to Specification by Example

CI and Specification by Example

Some important aspects to bear in mind

Conclusion

Test Driven Development

Automated Testing at Unit Test Level

6 October 2015

CGI

Experience the commitment®

eXtreme Programming’s “Test First” principle

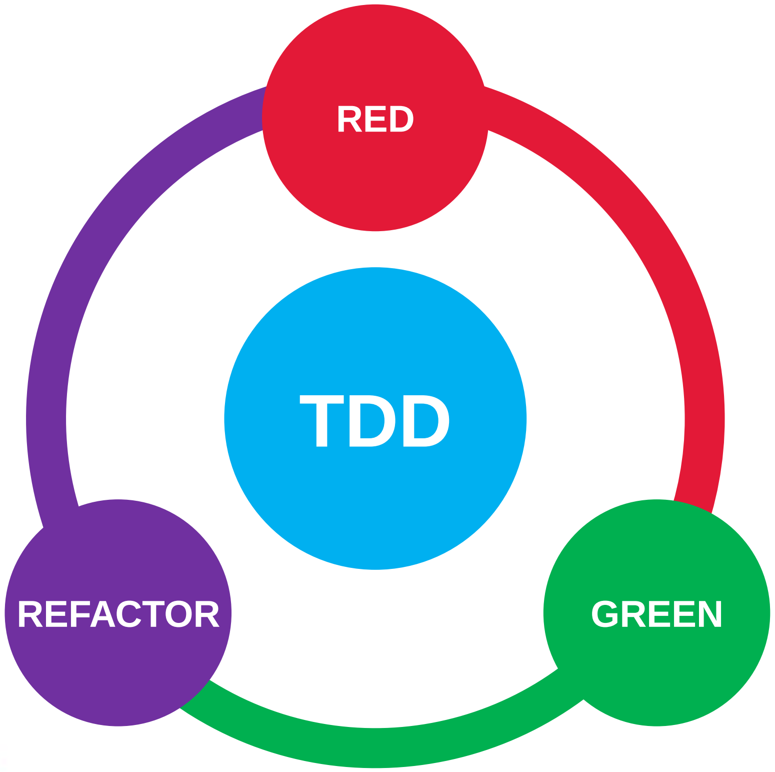
- Create test cases before writing code
  - Business representatives write acceptance tests to demonstrate that user stories are correctly implemented
  - Programmers continually write component tests which must run flawlessly for development to continue
- “We only write new code when we have a test that doesn’t work”

I'LL GO UP AND FIND OUT WHAT THEY NEED AND THE REST OF YOU START CODING!
SHELTON

Test Driven Development

Test Driven Development (TDD) is a software development technique!

Although the technique is even older, Kent Beck introduced the name TDD in 2003. The principle is that the developer starts writing tests which are used to test the code that the developer will write.



The biggest benefits of Test-Driven Development

Defect Reduction	Functionality that doesn't function provides no business value. TDD allows us to start with high quality, thus providing real value.
Faster Feature Time to Market	TDD allows developers to add innovative features as rapidly as can be, without damaging the original investment
Improved Focus	TDD helps developers think clearly about the task at hand, without having to continuously hold the whole complex system in their heads. They avoid over-extrapolation and over-design.
Parallel Efforts Without Conflict	Since, with disciplined TDD, each developer (or pair of developers) is expected to run all of the tests in the regression suite and confirm that 100% pass before committing new changes, there is much less opportunity to disrupt or destroy someone else's hard work.
Test Assets	During a TDD development process programmers create test code that is automatically executed and therefor valuable as regression test ware

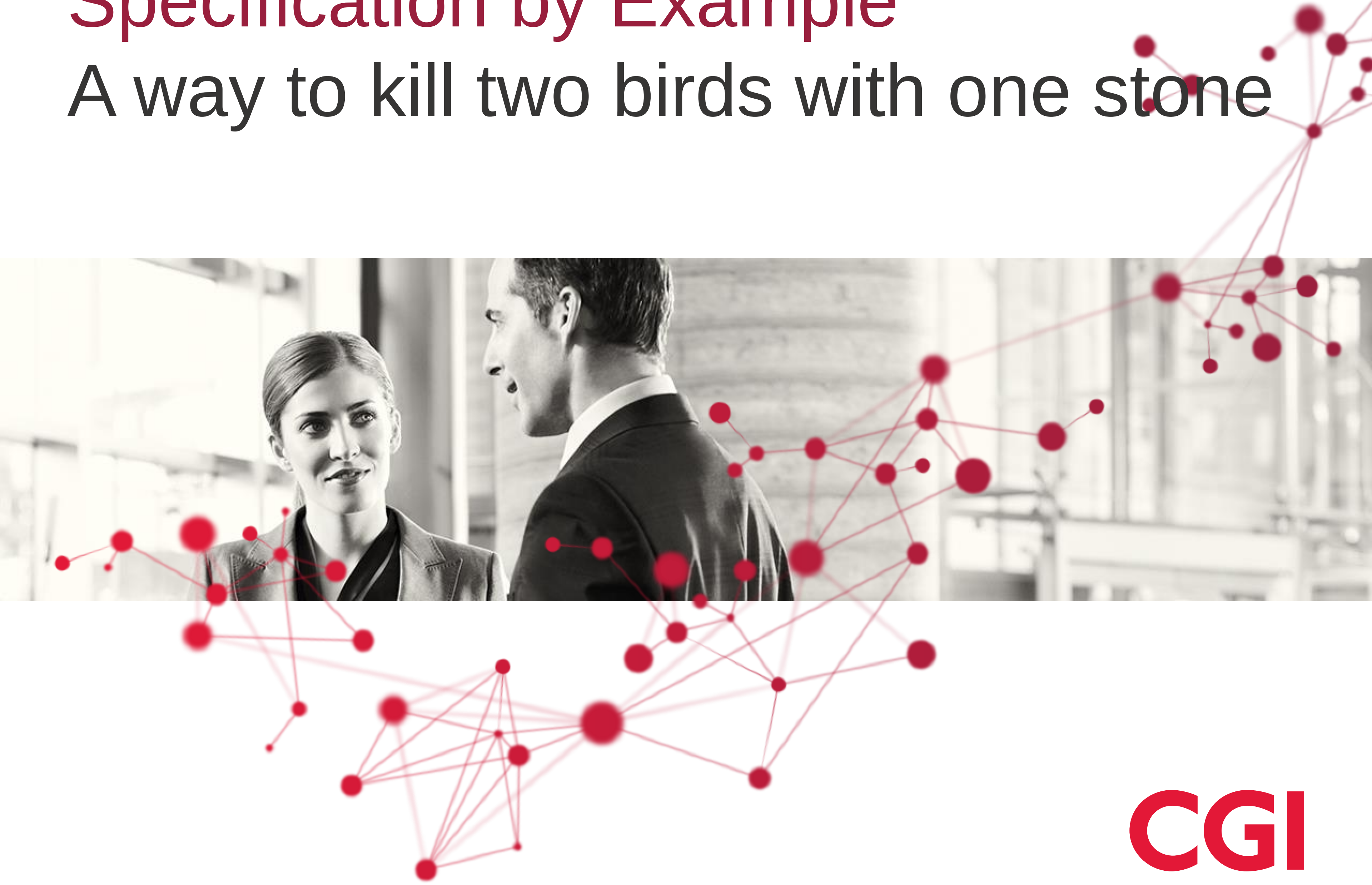
Source: Agileinstitute.com

However, there are limitations to TDD

TDD leads to high quality software, not necessarily to the correct software

Specification by Example

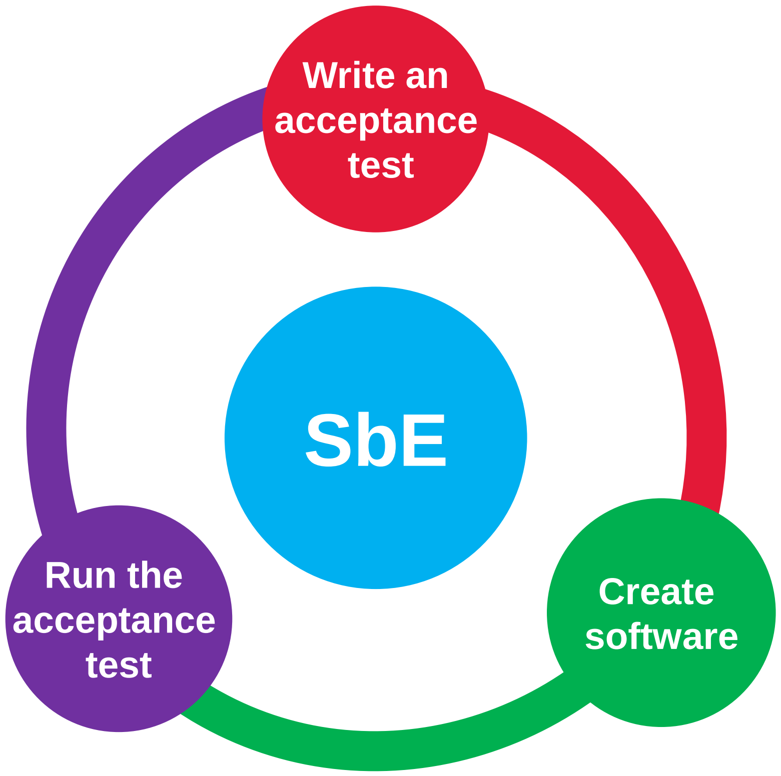
A way to kill two birds with one stone



Experience the commitment®

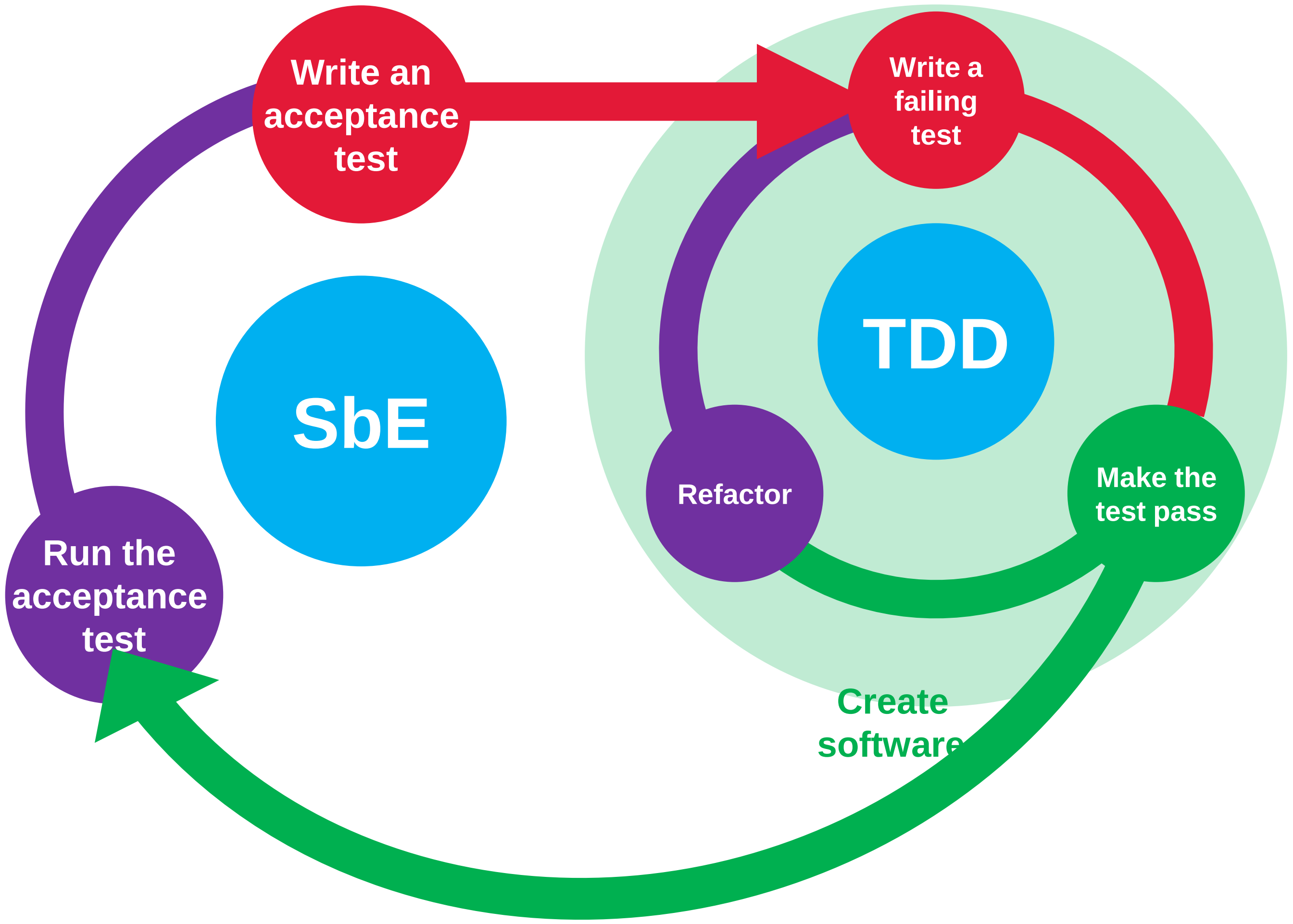
Specification by Example

TDD combined with acceptance testing

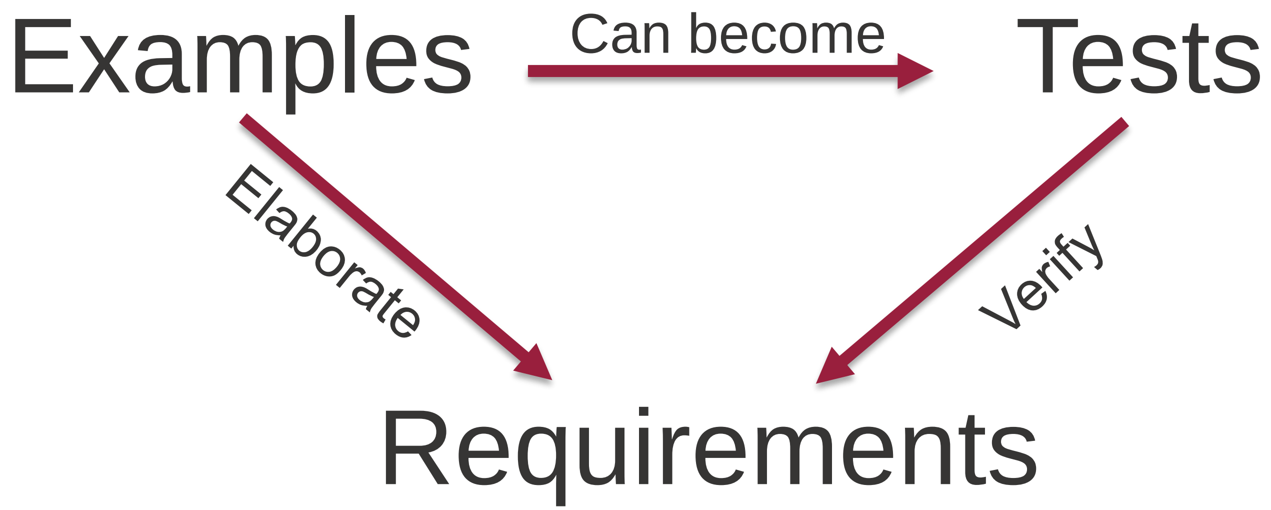


Specification by Example

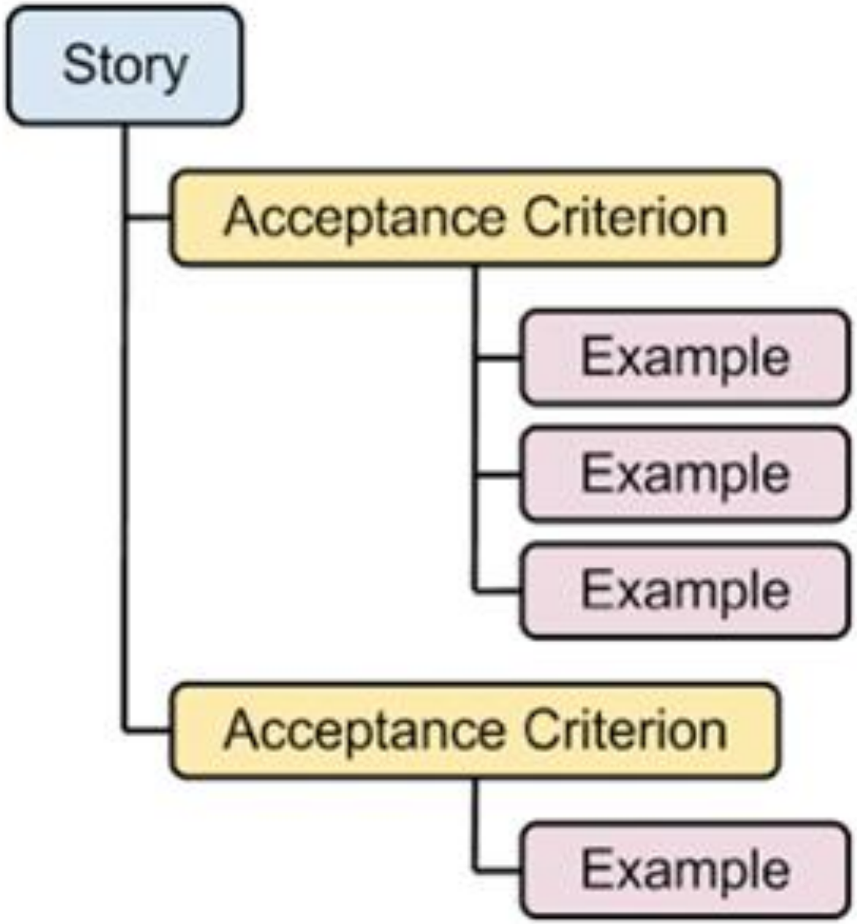
TDD combined with acceptance testing



The basics



Source: Gojko Adzic: Specification by Example


Story
Acceptance Criterion
Example
Example
Example
Acceptance Criterion
Example

Implementing Acceptance Criteria

Validate that a story has been developed with the functionality the Product Owner had in mind.

- Test cases fill in the details of the story.
- Write tests before programming – Test first, build next.
- Execute test at end of sprint as demo
- Test cases are delivered as system documentation.

Title:	***Book a flight***	
ID:	***US12a***	
Scenario:	***As a member of the F&H club I can use Fresh points to pay for a flight so that I can travel for free***	
Estimation:	***13 points***	Priority: ***High***

Examples in Gherkin

Acceptance Criterion:

The return date must be later than or equal to the arrival date

Given user "123.456.789.0" with password "abcd" is logged in
And departure city is "AMS"
And destination city is "CDG"
And return trip = “Y”
When departure date is 17 days after today
And return date is 15 days after today
And query is submitted
Then the alert "Date of return flight is incorrect!" is shown

Automating the Test Execution

Keyword driven testing:

- Automation scripts per keyword
- Parameters contain test data

Given user "123.456.789.0" with password "abcd" is logged in
And departure city is "AMS"
And destination city is "CDG"
And return trip = “Y”
When departure date is 17 days after today
And return date is 15 days after today
And query is submitted
Then the alert "Date of return flight is incorrect!" is shown

Automating the Test Execution

Keyword driven testing:

- Automation scripts per keyword
- Parameters contain test data

Given user "123.456.789.0" with password "abcd" is logged in
And departure city is "AMS"
And destination city is "CDG"
And return trip = "Y"
When departure date is 17 days after today
And return date is 15 days after today
And query is submitted
Then the alert "Date of return flight is incorrect!" is shown

Reuse using tables

Data driven testing

user	password	Dept city	Dest city	return	Dept date (now+#days)	Ret Date (now+#days)	Alert
123.456.789.0	abcd	AMS	CDG	N	13	-	Bookingdate must be 14 days in the future
123.456.789.0	abcd	AMS	CDG	Y	17	16	Date of return flight is incorrect!"

Given user “@User" with password “@Password" is logged in
And departure city is “@Deptcity"
And destination city is “@Destcity"
And return trip “@return”
When departure date is “@Deptdate” days after today
And return date is “@Retdate” days after today
And query is submitted
Then the alert “@lert” is shown

The greatest benefits of SbE

from a project focus: First-Time-Right

It is not about "testing"	SbE is a design approach, another tool in the box to facilitate goal oriented communication and get everybody on the same page.
No more outdated documents	Another benefit is that there are less documents in the true sense of the word. Test cases and specs are the same and always in sync
Easy automatable test execution	SbE Specs (Examples) are written in a format that is directly automatable, right from the first execution
Automated regression test instantly available	Every sprint will deliver test cases and test scripts. It is just a matter of selection of Regression Test cases (depending on the product risks that were defined)
No more throwing artifacts over the wall	Since everyone is involved during refinement of the user stories, there will be a common understanding of what to achieve

from a business focus: we get what we need

The system just works	There are almost no findings (either because of errors or changes) after the software is released, so First-time-Right
Focus is on the problem not the solution	SbE specs must be written in a way that really describes the business problem to be solved and not how the user interface of the application will be used.
Faster time to market	The application of SbE leads to an overall reduction of project duration due to less misunderstanding

CI and Specification by Example

Continuous Acceptance Testing

Practices of Continuous Integration

- Maintain a Single Source Repository.
- Automate the Build
- Make Your Build Self-Testing
- Everyone Commits To the Mainline Every Day
- Every Commit Should Build the Mainline on an Integration Machine
- Fix Broken Builds Immediately
- Keep the Build Fast
- Test in a Clone of the Production Environment
- Make it Easy for Anyone to Get the Latest Executable
- Everyone can see what's happening
- Automate Deployment

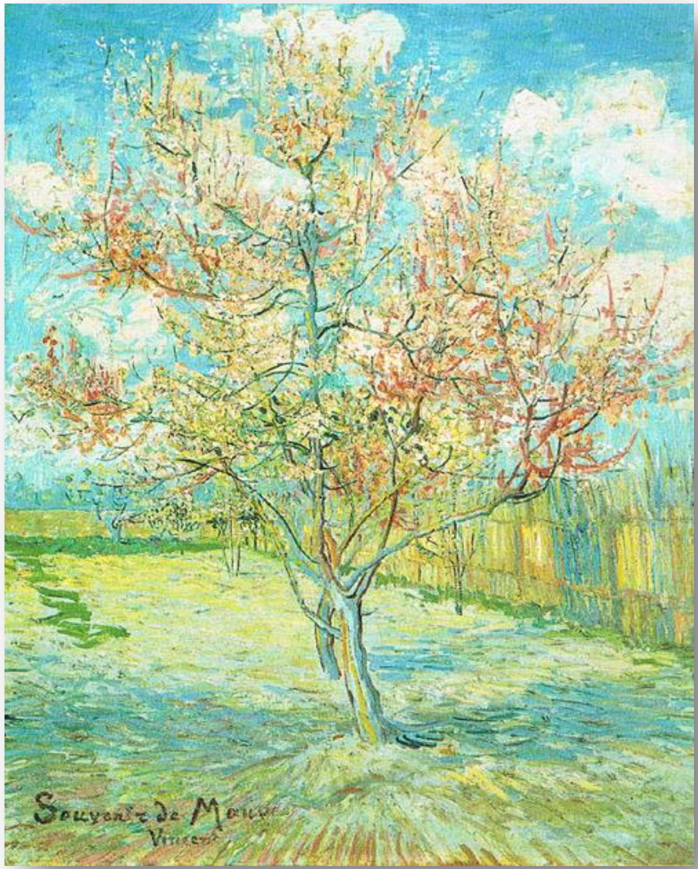
Automate the Build

Make Your Build Self-Testing

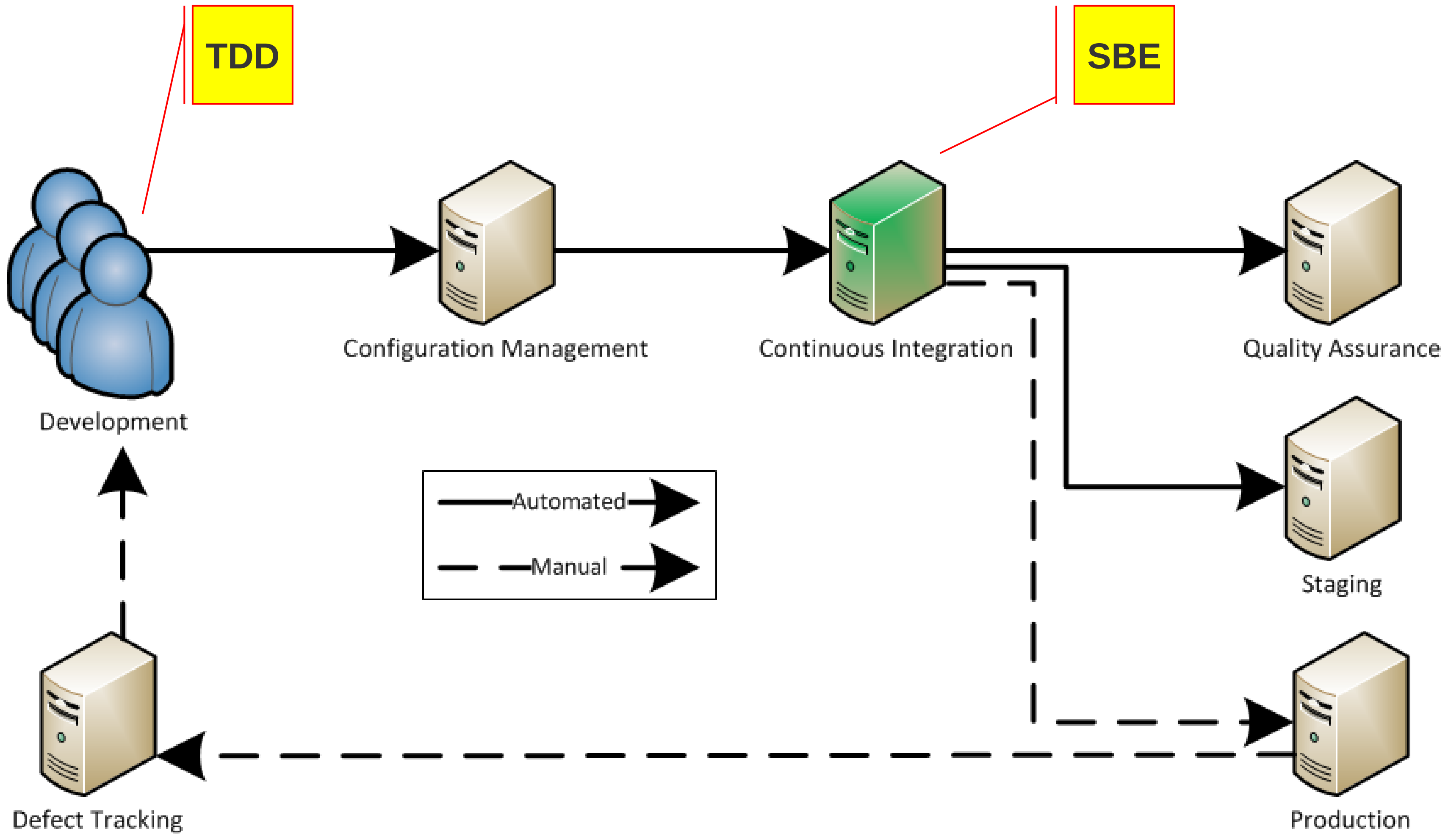
So, CI means that:

- The programmer will unit test the branch
- The branch will be integrated into the trunk
- On the build server the integrated software is built and tested

And this many times a day!



The CI process (simplified)



Conclusion

The advantages of Specification by Example

Specification by Example has numerous advantages:

- Improved, living documentation
- Common understanding by anyone involved (business & IT)
- Less production failures, first time right delivery
- Overall shorter development and QA cycles
- No “that’s not what I meant” complaints anymore, so no rework

- Easy automatable and therefor unmissable in Continuous Integration

And remember

Transformation into a solution of even the SMART-est requirements may fail due to misunderstanding and assumptions. This often leads to unhappy users or even high problem solving cost.

When business representative, product owner, developer and tester talk to each other and create acceptance tests, misunderstanding and assumptions tend to be discovered.

But beware: if you apply Specification by Example just as a test approach, you'll loose these advantages

chris.schotanus@cgi.com

Let’s end with two quotes:

“More than the act of testing, the act of designing tests is one of the best defect preventers known … The thought process that must take place to create useful tests can discover and eliminate problems at every stage of development” Boris Beizer

“One of the most effective ways of specifying something is to describe (in detail) how you would accept (test) it if someone gave it to you.” Bill Hetzel