FROM TDD TO BDD AND BACK
HANDS ON LAB

GETTING STARTED GUIDE CONCORDION

Mieke Kemme and Elke Steegmans
Nov 4, 2015

TABLE OF CONTENTS

INTRODUCTION

CONCORDION

Concordion is an open source tool for automating Specification by Example.

Official Website: http://concordion.org/

THIS LAB

This lab should give you a practical introduction to writing executable specifications with *Concordion*.

You receive:

- a small demo-application written in Java
- a few user stories
- scenarios for one of the stories

The goal of this lab is to write a few specifications for this application and make them executable, using the tool *Concordion.*

This lab is organized as follows:

1. *installation* of the demo-application and the software needed to write the tests,
2. follow the *step-by-step* guide to make the scenarios provided executable. This should give you the general idea of the tool,
3. play around and write the *scenarios* for another user story yourself. Look at the *whole picture*: can you make better user stories this way?

At the end we reserve half an hour to discuss with other participants and *compare* your tool with the tools they used.

SETUP

PREREQUISITES

- Eclipse
- Java
- Maven
- Maven-eclipse-plugin: http://download.eclipse.org/technology/m2e/releases

INSTALLATION ECLIPSE PLUGIN

Eclipse installer:

- In the menu choose *Help* | *Install New Software …*
- Click *Add…* to add a new software site
- Enter `Concordion` as *Name* and http://concordion-eclipse.googlecode.com/svn/trunk/org.concordion.ide.eclipse.update-site-helios as *Location*.
- Choose *OK*
- Select *Concordion Eclipse Plugin* and choose *Next*
- *Next*
- Accept the license agreement and choose *Finish*
- Ignore the warning and choose *OK*
- Choose *Yes* to restart Eclipse

CREATE ECLIPSE PROJECT

1. Create a new workspace: *File* | *Switch Workspace* | *Other …*
2. Via your file system, unzip the bmi-application you received on the USB-stick to this workspace
3. Import de bmi-application in *Eclipse*:
    - *File* | *Import…* | *Maven* | *Existing Maven Projects*
    - Click *Next*
    - For the field *Root Directory:* browse to the folder bmi-app and choose *Open*
    - The pom.xml should appear in the *Projects* list
    - Click *Finish*

    The project is created. In the root of your project, you will find the *pom.xml*. Click on the file and choose *Run As | Maven install* to check if you can build the application.

4. Edit the *pom.xml*. Add the plugins and dependencies needed for *Cucumber*. See Attachment 1: Additions to pom.xml.

DEVELOPMENT

INTRODUCTION: THE GENERAL IDEA

To write executable specifications with *Concordion*, you always have to write 2 types of file:

1. The **feature**: an html file containing the specifications in Given-When-Then style. The filename is in lowercase and extension .html.

   Example: `showPatientDetails.html`

2. A file containing the **fixture** and the **story steps**: a Java test class that is used to run as a JUnit test class and that contains the java code to make each Given-, When- or Then- step executable. The filename is the same name as the name of the story + Test, but in CamelCase and with extension .java

   Example: `ShowPatientDetailsTest.java`

STEP 1: WRITE SPECIFICATIONS

1. Create a new source folder `src/test/specs`.
2. Create a package `org.ucll.demo.service` in this new folder.
3. Create a new html file:
   - Choose *New* | *Other…* | *Web* | *HTML File*
   - Click *Next*
   - Enter the name of the feature, i.e. `showPatientDetails.html`
   - Click *Finish*
   - The HTML file is created.
4. Add the story and the scenario given in Attachment 2: Specification_1.txt
   - A css file is already made by Concordion for you, you don't need to make it yourself, you just add it with the necessary HTML tags.
5. Choose *Save*.

```
<h1>Show patient details</h1>
<h2>Narrative</h2>
<div>
<p>In order to check the physical condition of a patient</p>
<p>As a caretaker</p>
<p>I want to consult his/her personal details</p>
</div>
<h2>Acceptance Criteria</h2>
<h3>Scenario: the personal details of a registered patient are given</h3>
<div>
    <p>
        Given a patient with the social security number 93051822361, gender male and birthdate 1993-05-18
    </p>
    <p>
        And on 2000-04-10 the patient had a length of 180 cm and a weight of 75000 gr
    </p>
    <p>
        And the patient is registered
    </p>
    <p>
        When I ask for the details of the patient using his social security number
    </p>
    <p>
        Then the personal details social security number, gender and birthdate are given
    </p>
    <p>
        And the examination details length, weight and last examination date are given
    </p>
    <p>
        And the calculated bmi 23.15 is given
    </p>
   <hr></hr>
</div>
```

Figure 1. Example HTML show patient details

STEP 2: CREATE JAVA FIXTURE AND STEPS

1. Create in the folder `src/test/java` and in the package `org.ucll.demo.service` a Java class for the fixture, i.e. `ShowPatientDetailsTest.java`.

2. Add the following before the name of the class:

   ```
   @RunWith(ConcordionRunner.class)
   ```

   and add the import statements:

   ```
   import org.concordion.integration.junit4.ConcordionRunner;
   import org.junit.runner.RunWith;
   ```

EXECUTE SPECIFICATIONS

1. Select the Java file `ShowPatientDetailsTest.java`, right mouse click and choose *Run as JUnit Test.*

   In the **output console** you can see the result:

   - The path to the html file in which the results are shown
   - The number of successes and the number of failures

```
/var/folders/qb/kyvsxvcj3vd3bzwx9gbz8fp00000gn/T/concordion/org/ucll/demo/service/ShowPatientDetails.html
Successes: 0, Failures: 0
```

Figure 2. Example console output

2. Copy the link to the html file to your browser to open **HTML output**.
   Once the tests are implemented, colors will indicate the results.

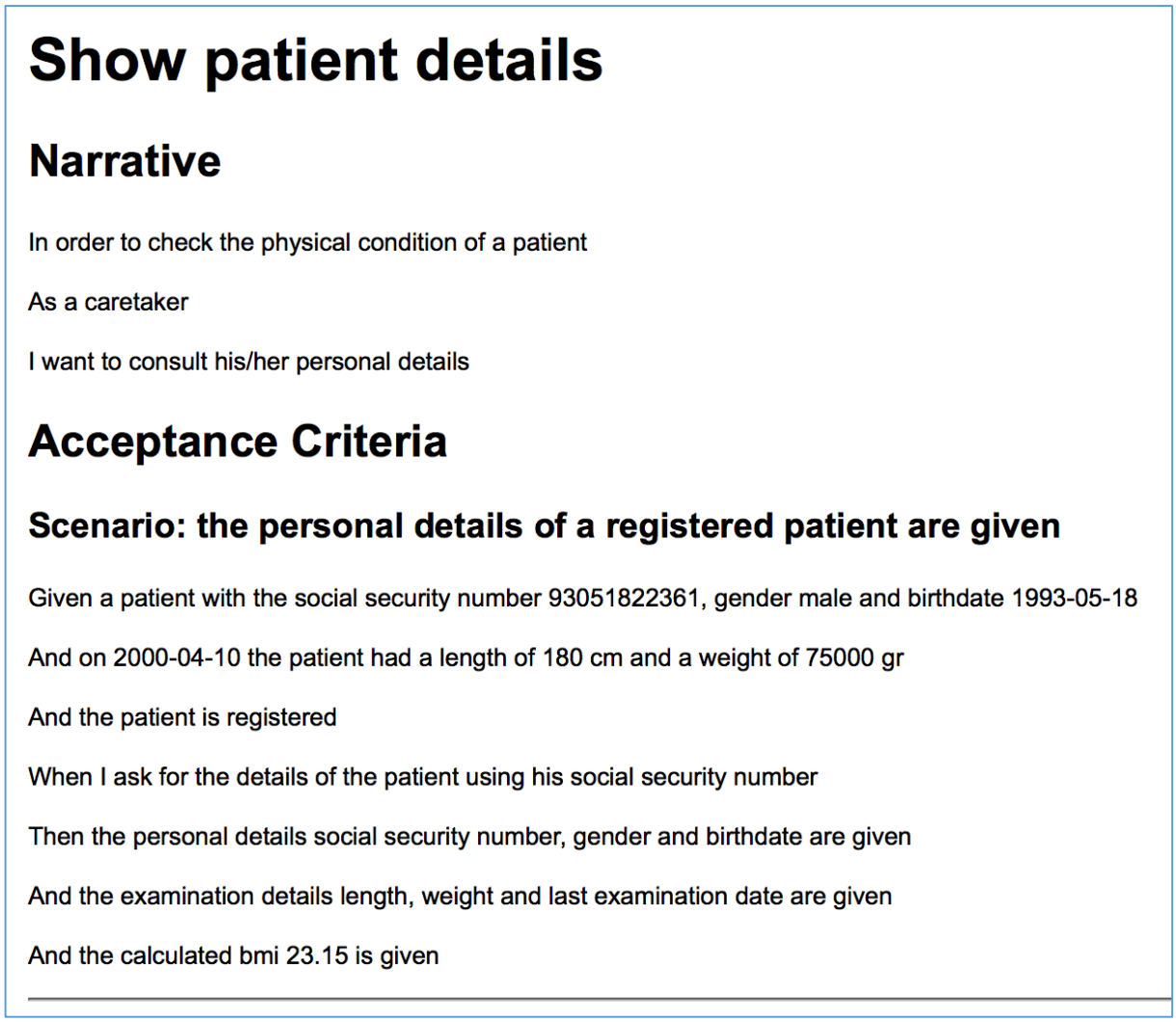

Figure 3. Example HTML output

STEP 4: MAKE THE STEPS EXECUTABLE

INTRODUCTION

In the html file you need to add specific *Concordion* tags to the existing html tags.

There exist different types of Concordion specific tags, they all start with "`concordion:*`" where the asterisk needs to be replaced by a command. You can add the Concordion specific tags to a lot of existing HTML tags, such as for example <label>, <p>, …

An example of each Concordion specific tag:

- `<label concordion:set="#number>93051822361</label>`
  This created a temporary variable **number** with the value **93051822361**

- `<p concordion:execute="setSocialSecurityNumber(#number)"></p>`
  This calls and executes the method **setSocialSecurityNumber** with the temporary variable created before as a parameter.
  In this case, the "setSocialSecurityNumber" is a *Java* method on **story steps**, a .java class that represents this *.html* , file such as

  ```
  public void setSocialSecurityNumber(String number){
  …
  }
  ```

  You can also call a method which returns something and put the return value in a variable. For example:

  ```
  <p concordion:execute="#validNumber =
  checkSocialSecurityNumber(#number)"></p>
  ```

- `<label concordion:assertTrue="validSocialSecurityNumber(#number)">true</label>`
  This calls the **checkSocialSecurityNumber** method with the temporary variable *#number*. It then checks/asserts if the result is **true**
  In this case, the "checkSocialSecurityNumber" is a *Java* method on **story steps**, a .java class that represents this *.html* , file such as

  ```
  public boolean checkSocialSecurityNumber(String number){
  …
  }
  ```

  The label with the assert will display a green or red background depending on the result of the test. Also `assertFalse`, `assertEquals`, .. can be used here. For example:
  `<label concordion:assertEquals="#number">93051822361</label>`

Remark: in eclipse you can type `<label con` and then do *ctrl-space* to get an overview of all the existing Concordion specific tags.

CODE

To make the specifications executable:

1. Add Concordion specific tags on the right places.

```
<h1>Show patient details</h1>
<h2>Narrative</h2>
<div>
<p>In order to check the physical condition of a patient</p>
<p>As a caretaker</p>
<p>I want to consult his/her personal details</p>
</div>
<h2>Acceptance Criteria</h2>
<h3>Scenario: the personal details of a registered patient are given</h3>
<div>
    <p concordion:execute="initializePersonDetails(#socialSecurityNumber, #gender, #birthDate)">
        Given a patient with the social security number <label concordion:set="#socialSecurityNumber">93051822361</label>,
        gender <label concordion:set="#gender">male</label> and birthdate <label concordion:set="#birthDate">1993-05-18</label>
    </p>
    <p>
        And on 2000-04-10 the patient had a length of 180 cm and a weight of 75000 gr
    </p>
    <p>
        And the patient is registered
    </p>
    <p>
        When I ask for the details of the patient using his social security number
    </p>
    <p>
        Then the personal details social security number, gender and birthdate are given
    </p>
    <p>
        And the examination details length, weight and last examination date are given
    </p>
    <p>
        And the calculated bmi 23.15 is given
    </p>
   <hr></hr>
</div>
```

Figure 4. Example adding Concordion specfic tags

2. Implement each method called in the Concordion specific tags in the .java file.
    - call the necessary code to connect to your actual domain classes…

```
<h1>Show patient details</h1>
<h2>Narrative</h2>
<div>
<p>In order to check the physical condition of a patient</p>
<p>As a caretaker</p>
<p>I want to consult his/her personal details</p>
</div>
<h2>Acceptance Criteria</h2>
<h3>Scenario: the personal details of a registered patient are given</h3>
<div>
    <p concordion:execute="initializePersonDetails(#socialSecurityNumber, #gender, #birthDate)">
        Given a patient with the social security number <label concordion:set="#socialSecurityNumber">93051822361</label>,
        gender <label concordion:set="#gender">male</label> and birthdate <label concordion:set="#birthDate">1993-05-18</label>
    </p>
    <p>
        And on 2000-04-10 the patient had a length of 180 cm and a weight of 75000 gr
    </p>
    <p>
        And the patient is registered
    </p>
    <p>
        When I ask for the details of the patient using his social security number
    </p>
    <p>
        Then the personal details social security number, gender and birthdate are given
    </p>
    <p>
        And the examination details length, weight and last examination date are given
    </p>
    <p>
        And the calculated bmi 23.15 is given
    </p>
   <hr></hr>
</div>
```

Figure 5. Example step implementation

3. Implement all the steps until you have a successful test.

See *Attachment 3: Example scenario - HTML* and *Attachment 4: Example scenario - implementation* for a first implementation of all the steps

EXTRA

REUSE STEPS

Identical steps:

1. Add the scenario's given in Attachment 5: Specification_2.txt
2. Run the test.
3. In *Concordion* you need to add *Concordion* specific tags to each step again, and you can reuse the same method in these different steps, you don't need to make different methods. In the example in Figure 6 And the patient is registered step is implemented by the same method.

```
    </p>
    <p concordion:execute="registerPatient()">
        And the patient is registered
    </p>
    <p concordion:execute="askPatientDetails()">
        When I ask for the details of the patient using his social security number
    </p>
    <p concordion:assertTrue="verifyPatientDetails()">
        Then the personal details social security number, gender and birthdate are given
    </p>
    <p concordion:assertTrue="verifyExaminationDetails()">
        And the examination details length, weight and last examination date are given
    </p>
    <p>
        And the calculated bmi <label concordion:assertEquals="verifyBmi()">23.15</label> is given
    </p>
   <hr></hr>
</div>
<h3>Scenario: the physical data of the most recent examination are given</h3>
<div>
    <p concordion:execute="initializePersonDetails(#socialSecurityNumber)">
        Given a patient with the social security number <label concordion:set="#socialSecurityNumber">93051822361</label>
    </p>
    <p concordion:execute="initializeExaminationDetails(#examinationDate, #length, #weight)">
        And on <label concordion:set="#examinationDate">2000-04-10</label> the patient had
        a length of <label concordion:set="#length">180</label> cm and
        a weight of <label concordion:set="#weight">75000</label> gr
    </p>
    <p concordion:execute="registerPatient()">
        And the patient is registered
    </p>
    <p concordion:execute="addOlderExaminationDetails(#examinationDate, #length, #weight)">
        And on <label concordion:set="#examinationDate">2000-04-17</label> the patient had
        a length of <label concordion:set="#length">180</label> cm and
```

Figure 6. Calling the same method in different scenarios

SCENARIO'S WITH EXPECTED EXCEPTIONS

1. Add the scenario's given in Attachment 6: Specification_3.txt.
2. Declare an instance variable: `private boolean errorThrown;`
3. In the methods for the *when* steps, where an exception might be thrown*:*
    - put a try-catch block around the code.
    - In the catch block you can set the instance variable `errorThrown` to `true`;
4. In the *then* steps where you want to check if an exception is thrown, you can check the value of the instance variable…

5. In the then step in the html page you can use than in the right html tags of the *then* steps *concordion assertTrue="exceptionThrown()"*
6. The result of the execution of this step is colored now: red if it has failed and green if it has succeeded.

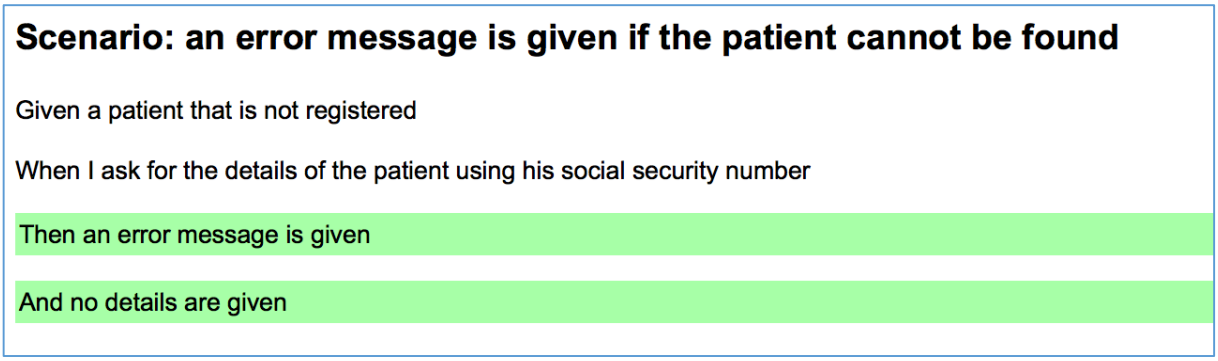

Figure 7. HTML output of a successful assertTrue

DATA TABLES

1. Add the scenario given in Attachment 7: Specification_4.txt. This scenario is different from the previous ones:
    - Instead of concrete values, you can see a table with parameters and concrete values

```
<h3>Scenario Outline: the bmi is rounded to 2 digits</h3>
<div>
    <p>Given a patient that is registered with a length in cm and weight in gr</p>
    <p>When I ask for details of the patient</p>
    <p>Then the bmi given is rounded to 2 digits</p>
    <div class="example">
        <h3>Examples</h3>
        <table concordion:execute="registerPatient(#length, #weight)">
            <tr>
                <th concordion:set="#length">length</th>
                <th concordion:set="#weight">weight</th>
                <th concordion:assertEquals="askPatientDetailsAndVerifyBmi()">bmi</th>
            </tr>
            <tr>
                <td>160</td>
                <td>65000</td>
                <td>25.39</td>
            </tr>
            <tr>
                <td>160</td>
                <td>65001</td>
                <td>25.39</td>
            </tr>
            <tr>
                <td>160</td>
                <td>65009</td>
                <td>25.39</td>
            </tr>
            <tr>
                <td>180</td>
                <td>75000</td>
                <td>23.15</td>
            </tr>
            <tr>
                <td>180</td>
                <td>75000</td>
                <td>23.15</td>
```

Figure 8.Table with examples

2. Implement all steps and all methods used and run as JUnit test.

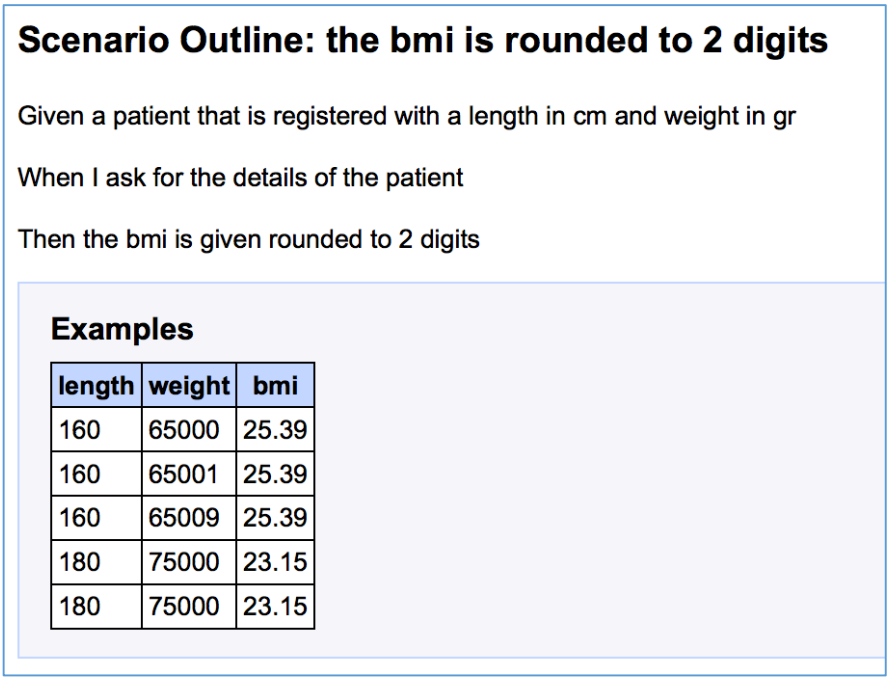

Figure 9. HTML output for an example table

See *Attachment 9: Example steps after implementing the 4 scenarios* for an example implementation of all scenarios

AND THEN …

Write specifications for the user story *Add physical examination data*.

1. How easy is it to focus on the content, without thinking about technical aspects?
2. Investigate other possibilities of *Concordion*. Look at features not described in this manual.
    - Can you avoid duplicate methods using a setup or teardown method?
    - How can you include images in your HTML pages?
    - …

In the attachments you can find an example of *Concordion* used in combination with *Mockito* and *Selenium*.

ATTACHMENTS

ATTACHMENT 1: ADDITIONS TO POM.XML

```
<project ...
    <properties>
        <concordion.version>1.4.4</concordion.version>
        ...
    </properties>

    <dependencies>
        <dependency>
                <groupId>org.concordion</groupId>
                <artifactId>concordion</artifactId>
                <version>${concordion.version}</version>
        </dependency>
        ...
    </dependencies>
</project>
```

ATTACHMENT 2: SPECIFICATION_1.TXT

```
<html xmlns:concordion="http://www.concordion.org/2007/concordion">
<head>
<link href="concordion.css" rel="stylesheet" type="text/css" />
<meta charset="utf-8"></meta>
<script src="js/jquery.js"></script>
<script src="js/jquery-ui.js"></script>
<script>
$(function() {
        $("#accordion").accordion();
});
</script>
</head>
<body>
    <h1>Show patient details</h1>
        <h2>Narrative</h2>
        <div>
        <p>In order to check the physical condition of a patient</p>
        <p>As a caretaker</p>
        <p>I want to consult his/her personal details</p>
        </div>
        <h2>Acceptance Criteria</h2>
    <h3>Scenario: the personal details of a registered patient are given</h3>
    <div>
                <p>
                        Given a patient with the social security number 93051822361, gender male and birthdate 1993-05-18
                </p>
                <p>
                        And on 2000-04-10 the patient had a length of 180 cm and a weight of 75000 gr
                </p>
                <p>
                        And the patient is registered
                </p>
                <p>
                        When I ask for the details of the patient using his social security number
                </p>
                <p>
                        Then the personal details social security number, gender and birthdate are given
                </p>
```

```
                <p>
                        And the examination details length, weight and last examination date are given
                </p>
                <p>
                        And the calculated bmi 23.15 is given
                </p>
        <hr></hr>
        </div>
 </body>
</html>
```

ATTACHMENT 3: EXAMPLE SCENARIO - HTML

```
<html xmlns:concordion="http://www.concordion.org/2007/concordion">
<head>
<link href="concordion.css" rel="stylesheet" type="text/css" />
<meta charset="utf-8"></meta>
<script src="js/jquery.js"></script>
<script src="js/jquery-ui.js"></script>
<script>
$(function() {
        $("#accordion").accordion();
});
</script>
</head>
<body>
    <h1>Show patient details</h1>
        <div>
        <h2>Narrative</h2>
        <p>In order to check the physical condition of a patient</p>
        <p>As a caretaker</p>
        <p>I want to consult his/her personal details</p>
        </div>
        <h2>Mock-up</h2>
        <img src="mockup_showpatientdetails.jpg" alt="Mockup show patient details" width="600" height="400"></img>
    <h2>Acceptance criteria</h2>
    <h3>Scenario: the personal details of a registered patient are given</h3>
    <div>
                <p concordion:execute="initializePersonDetails(#socialSecurityNumber, #gender, #birthDate)">
                        Given a patient with the social security number <label concordion:set="#socialSecurityNumber">93051822361</label>,
                        gender <label concordion:set="#gender">male</label> and birthdate <label concordion:set="#birthDate">1993-05-18</label>
                </p>
                <p concordion:execute="initializeExaminationDetails(#examinationDate, #length, #weight)">
                        And on <label concordion:set="#examinationDate">2000-04-10</label> the patient had
                        a length of <label concordion:set="#length">180</label> cm and
                        a weight of <label concordion:set="#weight">75000</label> gr
                </p>
                <p concordion:execute="registerPatient()">
                        And the patient is registered
                </p>
                <p concordion:execute="askPatientDetails()">
```

```
                When I ask for the details of the patient using his social security number
            </p>
            <p concordion:assertTrue="verifyPatientDetails()">
                Then the personal details social security number, gender and birthdate are given
            </p>
            <p concordion:assertTrue="verifyExaminationDetails()">
                And the examination details length, weight and last examination date are given
            </p>
            <p>
                And the calculated bmi <label concordion:assertEquals="verifyBmi()">23.15</label> is given
            </p>
        <hr></hr>
    </div>
</body>
</html>
```

ATTACHMENT 4: EXAMPLE SCENARIO - IMPLEMENTATION

```
package org.ucll.demo.service;

import java.text.SimpleDateFormat;
import java.util.Date;

import org.concordion.integration.junit4.ConcordionRunner;
import org.junit.runner.RunWith;
import org.ucll.demo.domain.Gender;
import org.ucll.demo.service.api.java.PersonServiceJavaApi;
import org.ucll.demo.service.api.java.to.ExaminationDetail;
import org.ucll.demo.service.api.java.to.PersonDetail;

@RunWith(ConcordionRunner.class)
public class ShowPatientDetailsTest {

	private PersonServiceJavaApi service = new PersonServiceJavaApi();

	private String socialSecurityNumber;
	private Gender gender = Gender.MALE;
	private Date birthDate;
	private int length;
	private int weight;
	private Date examinationDate;

	private PersonDetail detailsRetrieved;

	private static final SimpleDateFormat DATE_FORMATTER = new SimpleDateFormat("yyyy-MM-dd");
	private static final String REGISTERED_SOCIAL_SECURITY_NUMBER = "93051822361";
	private static final String UNKNOWN_SOCIAL_SECURITY_NUMBER = "93051822363";

	private boolean errorThrown = false;

	public void initializePersonDetails (String socialSecurityNumber, String gender, String birthDate) throws Exception {
		this.socialSecurityNumber = socialSecurityNumber;
		this.gender = Gender.valueOf(gender.toUpperCase());
		this.birthDate = DATE_FORMATTER.parse(birthDate);
		detailsRetrieved = new PersonDetail(this.socialSecurityNumber, this.gender, this.birthDate);
	}
```

```
        public void initializePersonDetails () throws Exception {
                this.socialSecurityNumber = UNKNOWN_SOCIAL_SECURITY_NUMBER;
                this.gender = Gender.MALE;
                this.birthDate = DATE_FORMATTER.parse("1993-05-18");
                detailsRetrieved = new PersonDetail(this.socialSecurityNumber, this.gender, this.birthDate);
        }

        public void initializePersonDetails (String socialSecurityNumber) throws Exception {
                this.socialSecurityNumber = socialSecurityNumber;
                this.gender = Gender.MALE;
                this.birthDate = DATE_FORMATTER.parse("1993-04-18");
                detailsRetrieved = new PersonDetail(this.socialSecurityNumber, this.gender, this.birthDate);
        }

        public void initializeExaminationDetails (String examinationDate, int length, int weight) throws Exception {
                this.examinationDate = DATE_FORMATTER.parse(examinationDate);
                this.length = length;
                this.weight = weight;
                detailsRetrieved.setExaminationDetail(new ExaminationDetail(length, weight, this.examinationDate));
        }

        public void addOlderExaminationDetails (String examinationDate, int length, int weight) throws Exception {
                service.addExamination(new ExaminationDetail(length, weight, DATE_FORMATTER.parse(examinationDate)), this.socialSecurityNumber);
        }

        public void registerPatient () {
                clearTestData();

                service.addPerson(detailsRetrieved);
        }

        public void registerPatient (int length, int weight) throws Exception {
                clearTestData();

                PersonDetail patient = new PersonDetail(REGISTERED_SOCIAL_SECURITY_NUMBER, Gender.MALE, DATE_FORMATTER.parse("1993-05-18"), new
ExaminationDetail(length, weight, DATE_FORMATTER.parse("2000-04-10")));
                service.addPerson(patient);
        }

        public void askPatientDetails () {
                detailsRetrieved = null;
```

```
		try {
			detailsRetrieved = service.getPerson(this.socialSecurityNumber);
		}
		catch (Exception exc) {
			errorThrown = true;
		}
	}

	public boolean verifyPatientDetails () {
		return (this.socialSecurityNumber.equals(detailsRetrieved.getSocialSecurityNumber()) &&
				this.gender.equals(detailsRetrieved.getGender()) &&
				differNoMoreThanFewSeconds(this.birthDate, detailsRetrieved.getBirthdate()));
	}

	public boolean verifyExaminationDetails () {
		ExaminationDetail examinationDetailsRetrieved = service.getPerson(this.socialSecurityNumber).getExaminationDetail();
		return (this.length == examinationDetailsRetrieved.getLength() &&
				this.weight == examinationDetailsRetrieved.getWeight() &&
				differNoMoreThanFewSeconds(this.examinationDate, examinationDetailsRetrieved.getExaminationDate()));
	}

	public boolean verifyExaminationDetails (int length, int weight, String examinationDate) throws Exception {
		ExaminationDetail examinationDetailsRetrieved = service.getPerson(this.socialSecurityNumber).getExaminationDetail();
		return (length == examinationDetailsRetrieved.getLength() &&
				weight == examinationDetailsRetrieved.getWeight() &&
				differNoMoreThanFewSeconds(DATE_FORMATTER.parse(examinationDate),
examinationDetailsRetrieved.getExaminationDate()));
	}

	public double verifyBmi() {
		return detailsRetrieved.getBmi();
	}

	public double askPatientDetailsAndVerifyBmi() {
		return service.getPerson(REGISTERED_SOCIAL_SECURITY_NUMBER).getBmi();
	}

	public boolean verifyException () {
		return errorThrown;
	}

	public boolean verifyNoDetailsRetrieved () {
```

```
                return detailsRetrieved == null;
        }

        private boolean differNoMoreThanFewSeconds(Date date, Date otherDate) {
                return date.compareTo(otherDate) <= 0 && date.compareTo(otherDate) >= -5;
        }

        private void clearTestData() {
                //TODO replace with DBUnit, ...
                service.deletePerson(REGISTERED_SOCIAL_SECURITY_NUMBER);
        }

}
```

ATTACHMENT 5: SPECIFICATION_2.TXT

```
<h3>Scenario: the physical data of the most recent examination are given</h3>
<div>
          <p>
                Given a patient with the social security number 93051822361
          </p>
          <p>
                And on 2000-04-17 the patient had a length of 180 cm and a weight of 80000 gr
          </p>
          <p>
                And the patient is registered
          </p>
          <p>
                And on 10-04-2000 the patient had a length of 180 cm and a weight of 75000 gr but these data were added later
          </p>
          <p>
                When I ask for the details of the patient using his social security number
          </p>
          <p>
                Then the length 180, weight 80000, and date of the examination 2000-04-17 are given
          </p>
          <p>
                And the calculated bmi 24.69 is based on these data
          </p>
  <hr></hr>
 </div>
```

ATTACHMENT 6: SPECIFICATION_3.TXT

```
<h3>Scenario: an error message is given if the patient cannot be found</h3>
        <div>
              <p>
                    Given a patient that is not registered
              </p>
              <p>
                    When I ask for the details of the patient using his social security number
              </p>
              <p>
                    Then an error message is given
              </p>
              <p>
                    And no details are given
              </p>
              <hr></hr>
        </div>
```

ATTACHMENT 7: SPECIFICATION_4.TXT

```
<h3>Scenario Outline: the bmi is rounded to 2 digits</h3>
	<div>
		<p>Given a patient that is registered with a length in cm and weight in gr</p>
		<p>When I ask for the details of the patient</p>
		<p>Then the bmi is given rounded to 2 digits</p>
		<div class="example">
			<h3>Examples</h3>
			<table>
				<tr>
					<th>length</th>
					<th>weight</th>
					<th>bmi</th>
				</tr>
				<tr>
					<td>160</td>
					<td>65000</td>
					<td>25.39</td>
				</tr>
				<tr>
					<td>160</td>
					<td>65001</td>
					<td>25.39</td>
				</tr>
				<tr>
					<td>160</td>
					<td>65009</td>
					<td>25.39</td>
				</tr>
				<tr>
					<td>180</td>
					<td>75000</td>
					<td>23.15</td>
				</tr>
				<tr>
					<td>180</td>
					<td>75000</td>
					<td>23.15</td>
				</tr>
```

```
                </table>
        </div>
<hr></hr>
 </div>
```

ATTACHMENT 8: EXAMPLE HTML AFTER THE 4 SCENARIOS

```
<html xmlns:concordion="http://www.concordion.org/2007/concordion">
<head>
<link href="concordion.css" rel="stylesheet" type="text/css" />
<meta charset="utf-8"></meta>
<script src="js/jquery.js"></script>
<script src="js/jquery-ui.js"></script>
<script>
$(function() {
        $("#accordion").accordion();
});
</script>
</head>
<body>
    <h1>Show patient details</h1>
        <div>
        <h2>Narrative</h2>
        <p>In order to check the physical condition of a patient</p>
        <p>As a caretaker</p>
        <p>I want to consult his/her personal details</p>
        </div>
    <h2>Acceptance criteria</h2>
    <h3>Scenario: the personal details of a registered patient are given</h3>
    <div>
                <p concordion:execute="initializePersonDetails(#socialSecurityNumber, #gender, #birthDate)">
                        Given a patient with the social security number <label concordion:set="#socialSecurityNumber">93051822361</label>,
                        gender <label concordion:set="#gender">male</label> and birthdate <label concordion:set="#birthDate">1993-05-18</label>
                </p>
                <p concordion:execute="initializeExaminationDetails(#examinationDate, #length, #weight)">
                        And on <label concordion:set="#examinationDate">2000-04-10</label> the patient had
                        a length of <label concordion:set="#length">180</label> cm and
                        a weight of <label concordion:set="#weight">75000</label> gr
                </p>
                <p concordion:execute="registerPatient()">
                        And the patient is registered
                </p>
                <p concordion:execute="askPatientDetails()">
                        When I ask for the details of the patient using his social security number
                </p>
```

```
            <p concordion:assertTrue="verifyPatientDetails()">
                    Then the personal details social security number, gender and birthdate are given
            </p>
            <p concordion:assertTrue="verifyExaminationDetails()">
                    And the examination details length, weight and last examination date are given
            </p>
            <p>
                    And the calculated bmi <label concordion:assertEquals="verifyBmi()">23.15</label> is given
            </p>
    <hr></hr>
</div>
<h3>Scenario: the physical data of the most recent examination are given</h3>
<div>
            <p concordion:execute="initializePersonDetails(#socialSecurityNumber)">
                    Given a patient with the social security number <label concordion:set="#socialSecurityNumber">93051822361</label>
            </p>
            <p concordion:execute="initializeExaminationDetails(#examinationDate, #length, #weight)">
                    And on <label concordion:set="#examinationDate">2000-04-10</label> the patient had
                    a length of <label concordion:set="#length">180</label> cm and
                    a weight of <label concordion:set="#weight">75000</label> gr
            </p>
            <p concordion:execute="registerPatient()">
                    And the patient is registered
            </p>
            <p concordion:execute="addOlderExaminationDetails(#examinationDate, #length, #weight)">
                    And on <label concordion:set="#examinationDate">2000-04-17</label> the patient had
                    a length of <label concordion:set="#length">180</label> cm and
                    a weight of <label concordion:set="#weight">80000</label> gr
                    but these data were added later
            </p>
            <p concordion:execute="askPatientDetails()">
                    When I ask for the details of the patient using his social security number
            </p>
            <p concordion:assertTrue="verifyExaminationDetails(#length, #weight, #examinationDate)">
                    Then the length <label concordion:set="#length">180</label>, weight <label concordion:set="#weight">80000</label>,
                    and date of the examination date <label concordion:set="#examinationDate">2000-04-17</label> are given
            </p>
            <p>
                    And the calculated bmi <label concordion:assertEquals="verifyBmi()">24.69</label> is based on these data
            </p>
    <hr></hr>
     </div>
```

```
<h3>Scenario: an error message is given if the patient cannot be found</h3>
<div>
	<p concordion:execute="initializePersonDetails()">
		Given a patient that is not registered
	</p>
	<p concordion:execute="askPatientDetails()">
		When I ask for the details of the patient using his social security number
	</p>
	<p concordion:assertTrue="verifyException()">
		Then an error message is given
	</p>
	<p concordion:assertTrue="verifyNoDetailsRetrieved()">
		And no details are given
	</p>
</div>
<h3>Scenario Outline: the bmi is rounded to 2 digits</h3>
<div>
	<p>Given a patient that is registered with a length in cm and weight in gr</p>
	<p>When I ask for details of the patient</p>
	<p>Then the bmi given is rounded to 2 digits</p>
	<div class="example">
		<h3>Examples</h3>
		<table concordion:execute="registerPatient(#length, #weight)">
			<tr>
				<th concordion:set="#length">length</th>
				<th concordion:set="#weight">weight</th>
				<th concordion:assertEquals="askPatientDetailsAndVerifyBmi()">bmi</th>
			</tr>
			<tr>
				<td>160</td>
				<td>65000</td>
				<td>25.39</td>
			</tr>
			<tr>
				<td>160</td>
				<td>65001</td>
				<td>25.39</td>
			</tr>
			<tr>
				<td>160</td>
				<td>65009</td>
				<td>25.39</td>
```

```
                                </tr>
                                <tr>
                                        <td>180</td>
                                        <td>75000</td>
                                        <td>23.15</td>
                                </tr>
                                <tr>
                                        <td>180</td>
                                        <td>75000</td>
                                        <td>23.15</td>
                                </tr>
                        </table>
                </div>
        <hr></hr>
        </div>
</body>
</html>
```

ATTACHMENT 9: EXAMPLE STEPS AFTER IMPLEMENTING THE 4 SCENARIOS

```
package org.ucll.demo.service;

import java.text.SimpleDateFormat;
import java.util.Date;

import org.concordion.integration.junit4.ConcordionRunner;
import org.junit.runner.RunWith;
import org.ucll.demo.domain.Gender;
import org.ucll.demo.service.api.java.PersonServiceJavaApi;
import org.ucll.demo.service.api.java.to.ExaminationDetail;
import org.ucll.demo.service.api.java.to.PersonDetail;

@RunWith(ConcordionRunner.class)
public class ShowPatientDetailsTest {

	private PersonServiceJavaApi service = new PersonServiceJavaApi();

	private String socialSecurityNumber;
	private Gender gender = Gender.MALE;
	private Date birthDate;
	private int length;
	private int weight;
	private Date examinationDate;

	private PersonDetail detailsRetrieved;

	private static final SimpleDateFormat DATE_FORMATTER = new SimpleDateFormat("yyyy-MM-dd");
	private static final String REGISTERED_SOCIAL_SECURITY_NUMBER = "93051822361";
	private static final String UNKNOWN_SOCIAL_SECURITY_NUMBER = "93051822363";

	private boolean errorThrown = false;

	public void initializePersonDetails (String socialSecurityNumber, String gender, String birthDate) throws Exception {
		this.socialSecurityNumber = socialSecurityNumber;
		this.gender = Gender.valueOf(gender.toUpperCase());
		this.birthDate = DATE_FORMATTER.parse(birthDate);
		detailsRetrieved = new PersonDetail(this.socialSecurityNumber, this.gender, this.birthDate);
	}
```

```
        public void initializePersonDetails () throws Exception {
                this.socialSecurityNumber = UNKNOWN_SOCIAL_SECURITY_NUMBER;
                this.gender = Gender.MALE;
                this.birthDate = DATE_FORMATTER.parse("1993-05-18");
                detailsRetrieved = new PersonDetail(this.socialSecurityNumber, this.gender, this.birthDate);
        }

        public void initializePersonDetails (String socialSecurityNumber) throws Exception {
                this.socialSecurityNumber = socialSecurityNumber;
                this.gender = Gender.MALE;
                this.birthDate = DATE_FORMATTER.parse("1993-04-18");
                detailsRetrieved = new PersonDetail(this.socialSecurityNumber, this.gender, this.birthDate);
        }

        public void initializeExaminationDetails (String examinationDate, int length, int weight) throws Exception {
                this.examinationDate = DATE_FORMATTER.parse(examinationDate);
                this.length = length;
                this.weight = weight;
                detailsRetrieved.setExaminationDetail(new ExaminationDetail(length, weight, this.examinationDate));
        }

        public void addOlderExaminationDetails (String examinationDate, int length, int weight) throws Exception {
                service.addExamination(new ExaminationDetail(length, weight, DATE_FORMATTER.parse(examinationDate)), this.socialSecurityNumber);
        }

        public void registerPatient () {
                clearTestData();

                service.addPerson(detailsRetrieved);
        }

        public void registerPatient (int length, int weight) throws Exception {
                clearTestData();

                PersonDetail patient = new PersonDetail(REGISTERED_SOCIAL_SECURITY_NUMBER, Gender.MALE, DATE_FORMATTER.parse("1993-05-18"), new
ExaminationDetail(length, weight, DATE_FORMATTER.parse("2000-04-10")));
                service.addPerson(patient);
        }

        public void askPatientDetails () {
                detailsRetrieved = null;
```

```
		try {
			detailsRetrieved = service.getPerson(this.socialSecurityNumber);
		}
		catch (Exception exc) {
			errorThrown = true;
		}
	}

	public boolean verifyPatientDetails () {
		return (this.socialSecurityNumber.equals(detailsRetrieved.getSocialSecurityNumber()) &&
				this.gender.equals(detailsRetrieved.getGender()) &&
				differNoMoreThanFewSeconds(this.birthDate, detailsRetrieved.getBirthdate()));
	}

	public boolean verifyExaminationDetails () {
		ExaminationDetail examinationDetailsRetrieved = service.getPerson(this.socialSecurityNumber).getExaminationDetail();
		return (this.length == examinationDetailsRetrieved.getLength() &&
				this.weight == examinationDetailsRetrieved.getWeight() &&
				differNoMoreThanFewSeconds(this.examinationDate, examinationDetailsRetrieved.getExaminationDate()));
	}

	public boolean verifyExaminationDetails (int length, int weight, String examinationDate) throws Exception {
		ExaminationDetail examinationDetailsRetrieved = service.getPerson(this.socialSecurityNumber).getExaminationDetail();
		return (length == examinationDetailsRetrieved.getLength() &&
				weight == examinationDetailsRetrieved.getWeight() &&
				differNoMoreThanFewSeconds(DATE_FORMATTER.parse(examinationDate),
examinationDetailsRetrieved.getExaminationDate()));
	}

	public double verifyBmi() {
		return detailsRetrieved.getBmi();
	}

	public double askPatientDetailsAndVerifyBmi() {
		return service.getPerson(REGISTERED_SOCIAL_SECURITY_NUMBER).getBmi();
	}

	public boolean verifyException () {
		return errorThrown;
	}

	public boolean verifyNoDetailsRetrieved () {
```

```
                return detailsRetrieved == null;
        }

        private boolean differNoMoreThanFewSeconds(Date date, Date otherDate) {
                return date.compareTo(otherDate) <= 0 && date.compareTo(otherDate) >= -5;
        }

        private void clearTestData() {
                //TODO replace with DBUnit, ...
                service.deletePerson(REGISTERED_SOCIAL_SECURITY_NUMBER);
        }

}
```

ATTACHMENT 10: EXAMPLE USING MOCKITO

```
package org.ucll.demo.service.api.java;

import static org.mockito.BDDMockito.*;
import static org.mockito.MockitoAnnotations.initMocks;

import org.mockito.Mock;
import org.concordion.integration.junit4.ConcordionRunner;
import org.junit.runner.RunWith;
import org.ucll.demo.domain.Person;
import org.ucll.demo.service.PersonService;
import org.ucll.demo.service.api.java.PersonServiceJavaApi;
import org.ucll.demo.service.api.java.assembler.ExaminationToAssembler;
import org.ucll.demo.service.api.java.assembler.PersonToAssembler;
import org.ucll.demo.service.api.java.to.PersonDetail;

@RunWith(ConcordionRunner.class)
public class PersonServiceJavaApiTest {
	private static final String SOCIAL_SECURITY_NUMBER = "93051822361";

	@Mock
	private PersonService mockPersonService;
	@Mock
	private ExaminationToAssembler mockExaminationToAssembler;
	@Mock
	private PersonToAssembler mockPersonToAssembler;
	@Mock
	private Person mockPerson;
	@Mock
	private PersonDetail mockPersonDetail;
	private PersonServiceJavaApi service;

	private PersonDetail patientRetrieved;
	private boolean exceptionThrown = false;


	private void setUp() {
		initMocks(this);
		service = new PersonServiceJavaApi(mockPersonService, mockExaminationToAssembler, mockPersonToAssembler);
```

```
}

public void registerSocialSecurityNumber() {
        setUp();

        given(mockPersonService.getPerson(SOCIAL_SECURITY_NUMBER)).willReturn(mockPerson);
        given(mockPersonToAssembler.createPersonDetailTo(mockPerson)).willReturn(mockPersonDetail);
}

public void getPatientUsingSocialSecurityNumber() {
        try{
                patientRetrieved = null;
                patientRetrieved = service.getPerson(SOCIAL_SECURITY_NUMBER);
        } catch(Exception e){
                exceptionThrown = true;
        }
}

public boolean aPersonDetailObjectIsReturned() {
        then(mockPersonService).should().getPerson(SOCIAL_SECURITY_NUMBER);
        then(mockPersonToAssembler).should().createPersonDetailTo(any(Person.class));
        return patientRetrieved != null;
}

public void doNotRegisterSocialSecurityNumber() {
        setUp();

        given(mockPersonService.getPerson(SOCIAL_SECURITY_NUMBER)).willReturn(null);
}

public void getPatientUsingNoSocialSecurityNumber() {
        try{
                patientRetrieved = null;
                patientRetrieved = service.getPerson(null);
        } catch(Exception e){
                exceptionThrown = true;
        }
}

public boolean exceptionThrown() {
        then(mockPersonService).should().getPerson(any(String.class));
        then(mockPersonToAssembler).should(never()).createPersonDetailTo(any(Person.class));
```

```
                return exceptionThrown;
        }

        public void noSocialSecurityNumberGiven () {
                setUp();

         willThrow(new RuntimeException()).given(mockPersonService).getPerson(null);
        }

}
```

ATTACHMENT 11: EXAMPLE USING SELENIUM

```
package org.ucll.demo.ui;

import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.NoSuchElementException;
import org.concordion.integration.junit4.ConcordionRunner;
import org.junit.runner.RunWith;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.ucll.demo.domain.Gender;
import org.ucll.demo.ui.pages.ExaminationDetailPage;
import org.ucll.demo.ui.pages.PersonDetailPage;
import org.ucll.demo.ui.pages.PersonOverviewPage;

@RunWith(ConcordionRunner.class)
public class ShowPatientDetailsITTest {

        private String registeredSocialSecurityNumber;
        private Date registeredBirthdate;
        private Gender registeredGender;
        private int registeredLength, registeredWeight;
        private Date registeredExaminationDate;
        private PersonDetailPage personDetailPage;
        private ExaminationDetailPage examinationDetailPage;
        private PersonOverviewPage personOverviewPage;
        private WebDriver driver;
        private final static String HOST_AND_PORT_URL = "http://localhost:8080";
        private final static String CONTEXT_URL = HOST_AND_PORT_URL + "/bmi";
        private static final SimpleDateFormat DATE_FORMATTER = new SimpleDateFormat("yyyy-MM-dd");

        public void setUp(){
                driver = new FirefoxDriver();
                driver.get(CONTEXT_URL);
        }

        public void clean(){
                driver.get(CONTEXT_URL);
                clearTestData();
```

```
                driver.quit();
        }

        public void initializePersonWithAllDetails(String socialSecurityNumber, String gender, String birthDate) throws Throwable {
                //TODO replace with DBUnit, ...
                registeredSocialSecurityNumber = socialSecurityNumber;
                registeredGender = Gender.valueOf(gender.toUpperCase());
                registeredBirthdate = DATE_FORMATTER.parse(birthDate);
        }

        public void initializeExaminationDetails (String examinationDate, int length, int weight) throws Throwable {
                //TODO replace with DBUnit, ...
                registeredLength = length;
                registeredWeight = weight;
                registeredExaminationDate = DATE_FORMATTER.parse(examinationDate);
        }

        public void registerPatient () throws Throwable {
                personOverviewPage = new PersonOverviewPage(driver);
                personDetailPage = personOverviewPage.addPerson();
                personOverviewPage = personDetailPage.addPerson(registeredSocialSecurityNumber, registeredBirthdate, registeredGender,
registeredLength, registeredWeight, registeredExaminationDate);
        }

        public void askPatientDetails () throws Throwable {
                personDetailPage = personOverviewPage.showPerson(registeredSocialSecurityNumber);
        }

        public boolean verifyPatientDetails () throws Throwable {
                return ((registeredSocialSecurityNumber.equals(personDetailPage.getSocialSecurityNumber())) &&
                                (registeredGender.equals(personDetailPage.getGender())) &&
                                (differNoMoreThanFewSeconds(registeredBirthdate, personDetailPage.getBirthdate())));
        }

        public boolean verifyExaminationDetails () throws Throwable {
                return ((registeredLength == personDetailPage.getLength()) &&
                                (registeredWeight == personDetailPage.getWeight()) &&
                                (differNoMoreThanFewSeconds(registeredExaminationDate, personDetailPage.getExaminationDate())));
        }

        public double verifyBmi () throws Throwable {
```

```
		try {
			return personDetailPage.getBmi();
		}
		finally {
			clean();
		}
	}

	public void initializePersonDetails(String socialSecurityNumber) throws Throwable {
		//TODO replace with DBUnit, ...
		registeredSocialSecurityNumber = socialSecurityNumber;
		registeredGender = Gender.MALE;
		registeredBirthdate = DATE_FORMATTER.parse("1993-05-18");
	}

	public void addOlderExaminationDetails(String examinationDate, int length, int weight) throws Throwable {
		personDetailPage = personOverviewPage.showPerson(registeredSocialSecurityNumber);
		examinationDetailPage = personDetailPage.addExamination();
		personDetailPage = examinationDetailPage.addExamination(length, weight, DATE_FORMATTER.parse(examinationDate));
		personOverviewPage = personDetailPage.cancel();
	}

	public boolean verifyGivenExaminationDetails(int length, int weight, String examinationDate) throws Throwable {
		return ((length == personDetailPage.getLength()) &&
				(weight == personDetailPage.getWeight()) &&
				(differNoMoreThanFewSeconds(DATE_FORMATTER.parse(examinationDate), personDetailPage.getExaminationDate())));
	}

	private void clearTestData() {
		//TODO replace with DBUnit, ...
		personOverviewPage = new PersonOverviewPage(driver);
		try {
			personOverviewPage.deletePerson();
		} catch (NoSuchElementException mightBeExpectedIfPersonWasNotRegistered) {
		}
	}

	private boolean differNoMoreThanFewSeconds(Date date, Date otherDate) {
		return date.compareTo(otherDate) <= 0 && date.compareTo(otherDate) >= -5;
	}

}
```

ATTACHMENT 12: EXAMPLE USING SELENIUM WITH DATA TABLES

```
package org.ucll.demo.ui;

import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.NoSuchElementException;
import org.concordion.integration.junit4.ConcordionRunner;
import org.junit.runner.RunWith;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.ucll.demo.domain.Gender;
import org.ucll.demo.ui.pages.PersonDetailPage;
import org.ucll.demo.ui.pages.PersonOverviewPage;

@RunWith(ConcordionRunner.class)
public class ShowPatientDetailsRoundedBmiITTest {

        private String registeredSocialSecurityNumber;
        private Date registeredBirthdate;
        private Gender registeredGender;
        private int registeredLength, registeredWeight;
        private Date registeredExaminationDate;

        private PersonDetailPage personDetailPage;
        private PersonOverviewPage personOverviewPage;

        private WebDriver driver;
        private final static String HOST_AND_PORT_URL = "http://localhost:8080";
        private final static String CONTEXT_URL = HOST_AND_PORT_URL + "/bmi";
        private static final SimpleDateFormat DATE_FORMATTER = new SimpleDateFormat("yyyy-MM-dd");

        private static final String SOCIAL_SECURITY_NUMBER = "93051822361";

        public void registerPatient(int length, int weight) throws Throwable {
                stopDriverAndStartDriverAgain();
                //TODO replace with DBUnit, ...
                registeredSocialSecurityNumber = SOCIAL_SECURITY_NUMBER;
                registeredGender = Gender.MALE;
                registeredBirthdate = DATE_FORMATTER.parse("1993-05-18");
```

```
                registeredLength = length;
                registeredWeight = weight;
                registeredExaminationDate = new Date();
                personOverviewPage = new PersonOverviewPage(driver);
                personDetailPage = personOverviewPage.addPerson();
                personOverviewPage = personDetailPage.addPerson(registeredSocialSecurityNumber, registeredBirthdate, registeredGender,
registeredLength, registeredWeight, registeredExaminationDate);
                personDetailPage = personOverviewPage.showPerson(registeredSocialSecurityNumber);
        }

        public void setUp(){
                driver = new FirefoxDriver();
                driver.get(CONTEXT_URL);
        }

        public void clean(){
                driver.get(CONTEXT_URL);
                clearTestData();
                driver.quit();
        }

        public double verifyBmis () throws Throwable {
                return personDetailPage.getBmi();
        }

        private void clearTestData() {
                //TODO replace with DBUnit, ...
                personOverviewPage = new PersonOverviewPage(driver);
                try {
                        personOverviewPage.deletePerson();
                } catch (NoSuchElementException mightBeExpectedIfPersonWasNotRegistered) {
                }
        }

        private void stopDriverAndStartDriverAgain() {
                if (personOverviewPage != null) {
                        clean();
                        setUp();
                }
        }

}
```

ATTACHMENT 13: SELENIUMPAGE CLASS

```
package org.ucll.demo.ui.pages;

import java.util.Calendar;
import java.util.Date;

import org.openqa.selenium.WebDriver;

public abstract class SeleniumPage {
	private final WebDriver driver;

	public SeleniumPage(WebDriver driver) {
		this.driver = driver;
	}

	public WebDriver getDriver() {
		return driver;
	}

	protected Date convertToDate(String dateAsString) {
		String[] dateParts = dateAsString.split("-");
		int year = Integer.parseInt(dateParts[0]);
		int month = Integer.parseInt(dateParts[1]);
		int day = Integer.parseInt(dateParts[2]);
		Calendar calendar = Calendar.getInstance();
		calendar.set(year, month - 1, day);
		return calendar.getTime();
	}

}
```

ATTACHMENT 14: PERSONOVERVIEWPAGE CLASS

```
package org.ucll.demo.ui.pages;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;

public class PersonOverviewPage extends SeleniumPage {

	public PersonOverviewPage(WebDriver driver) {
		super(driver);
	}

	public PersonDetailPage showPerson(String id) {
		WebElement userLink = getDriver().findElement(By.partialLinkText(id));
		userLink.click();
		return new PersonDetailPage(getDriver());
	}

	public boolean isCurrentPage() {
		WebElement heading = getDriver().findElement(By.cssSelector("h2"));
		return heading.getText().equals("Persons");
	}

	public PersonDetailPage addPerson() {
		WebElement addButton = getDriver().findElement(By.id("personOverviewForm:addPersonButton"));
		addButton.click();
		return new PersonDetailPage(getDriver());
	}

	public PersonDetailPage deletePerson() { // TODO do better
		WebElement deleteLink = getDriver().findElement(By.id("personOverviewForm:personTable:0:delete"));
		deleteLink.click();
		return new PersonDetailPage(getDriver());
	}

}
```

ATTACHMENT 15: PERSONDETAILPAGE CLASS

```
package org.ucll.demo.ui.pages;

import java.text.SimpleDateFormat;
import java.util.Date;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.ui.Select;
import org.ucll.demo.domain.Gender;

public class PersonDetailPage extends SeleniumPage {
	private ExaminationFieldsPage examinationfieldsPage;

	public PersonDetailPage(WebDriver driver) {
		super(driver);
		examinationfieldsPage = new ExaminationFieldsPage(getDriver(), "personForm");
	}

	public PersonOverviewPage addPerson(String socialSecurityNumber, Date birthdate,
			Gender gender, int length, int weight, Date examinationDate) {
		WebElement socialSecurityNumberField = getDriver().findElement(By.id("personForm:number"));
		socialSecurityNumberField.clear();
		socialSecurityNumberField.sendKeys(socialSecurityNumber);

	WebElement birthdateField = getDriver().findElement(By.id("personForm:birthDate"));
	birthdateField.clear();
	birthdateField.sendKeys(new SimpleDateFormat("yyyy-MM-dd").format(birthdate));

		Select genderField = new Select(getDriver().findElement(By.id("personForm:gender")));
		genderField.selectByVisibleText("MALE");

		examinationfieldsPage.addExaminationData(length, weight, examinationDate);

	WebElement saveButton = getDriver().findElement(By.id("personForm:actionSave"));
	saveButton.click();

	return new PersonOverviewPage(getDriver());
```

```
}

public String getSocialSecurityNumber() {
        WebElement socialSecurityNumberField = getDriver().findElement(By.id("personForm:number"));
        return socialSecurityNumberField.getAttribute("value");
}

public Gender getGender() {
        Select genderField = new Select(getDriver().findElement(By.id("personForm:gender")));
        String genderAsString = genderField.getFirstSelectedOption().getText();

        return Gender.valueOf(genderAsString.toUpperCase());
}

public Date getBirthdate() {
        WebElement birhtdateField = getDriver().findElement(By.id("personForm:birthDate"));
        String dateAsString = birhtdateField.getAttribute("value");
        return convertToDate(dateAsString);
}

public int getLength() {
        return examinationfieldsPage.getLength();
}

public int getWeight() {
        return examinationfieldsPage.getWeight();
}

public Date getExaminationDate() {
        return examinationfieldsPage.getExaminationDate();
}

public double getBmi() {
        WebElement bmiField = getDriver().findElement(By.id("personForm:bmi"));
        String bmiAsString = bmiField.getAttribute("value");
        return Double.parseDouble(bmiAsString);
}

public ExaminationDetailPage addExamination() {
        WebElement buttonExamination = getDriver().findElement(By.id("personToExaminationForm:addExamination"));
        buttonExamination.click();
```

```
                return new ExaminationDetailPage(getDriver());
        }

        public PersonOverviewPage cancel() {
                WebElement buttonCancel = getDriver().findElement(By.id("personToExaminationForm:cancel"));
                buttonCancel.click();
                return new PersonOverviewPage(getDriver());
        }

}
```

ATTACHMENT 16: EXAMINATIONDETAILPAGE CLASS

```
package org.ucll.demo.ui.pages;

import java.util.Date;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;

public class ExaminationDetailPage extends SeleniumPage {
        private ExaminationFieldsPage examinationfieldsPage;

        public ExaminationDetailPage(WebDriver driver) {
                super(driver);
                examinationfieldsPage = new ExaminationFieldsPage(getDriver(), "examinationForm");
        }

        public PersonDetailPage addExamination(int length, int weight, Date examinationDate) {
                examinationfieldsPage.addExaminationData(length, weight, examinationDate);

         WebElement saveButton = getDriver().findElement(By.id("examinationForm:saveExamination"));
         saveButton.click();

         return new PersonDetailPage(getDriver());
        }

}
```

ATTACHMENT 17: EXAMINATIONFIELDSPAGE CLASS

```
package org.ucll.demo.ui.pages;

import java.text.SimpleDateFormat;
import java.util.Date;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;

public class ExaminationFieldsPage extends SeleniumPage {
	private String formPrefix;

	public ExaminationFieldsPage(WebDriver driver, String formPrefix) {
		super(driver);
		this.formPrefix = formPrefix;
	}

	public void addExaminationData(int length, int weight, Date examinationDate) {
		WebElement lenghtField = getDriver().findElement(By.id(formPrefix + ":length"));
		lenghtField.clear();
		lenghtField.sendKeys(Integer.toString(length));

		WebElement weightField = getDriver().findElement(By.id(formPrefix + ":weight"));
		weightField.clear();
		weightField.sendKeys(Integer.toString(weight));

	 WebElement dateField = getDriver().findElement(By.id(formPrefix + ":examinationDate"));
	 dateField.clear();
	 dateField.sendKeys(new SimpleDateFormat("yyyy-MM-dd").format(examinationDate));
	}

	public int getLength() {
		WebElement lengthField = getDriver().findElement(By.id(formPrefix + ":length"));
		String lengthAsString = lengthField.getAttribute("value");
		return Integer.parseInt(lengthAsString);
	}

	public int getWeight() {
```

```
                WebElement weightField = getDriver().findElement(By.id(formPrefix + ":weight"));
                String weightAsString = weightField.getAttribute("value");
                return Integer.parseInt(weightAsString);
        }

        public Date getExaminationDate() {
                WebElement examinationDateField = getDriver().findElement(By.id(formPrefix + ":examinationDate"));
                String dateAsString = examinationDateField.getAttribute("value");
                return convertToDate(dateAsString);
        }

}
```