Modul 426

Agile Softwareentwicklung

Motivation

https://www.youtube.com/watch?v=wdzHgN7_Hs8

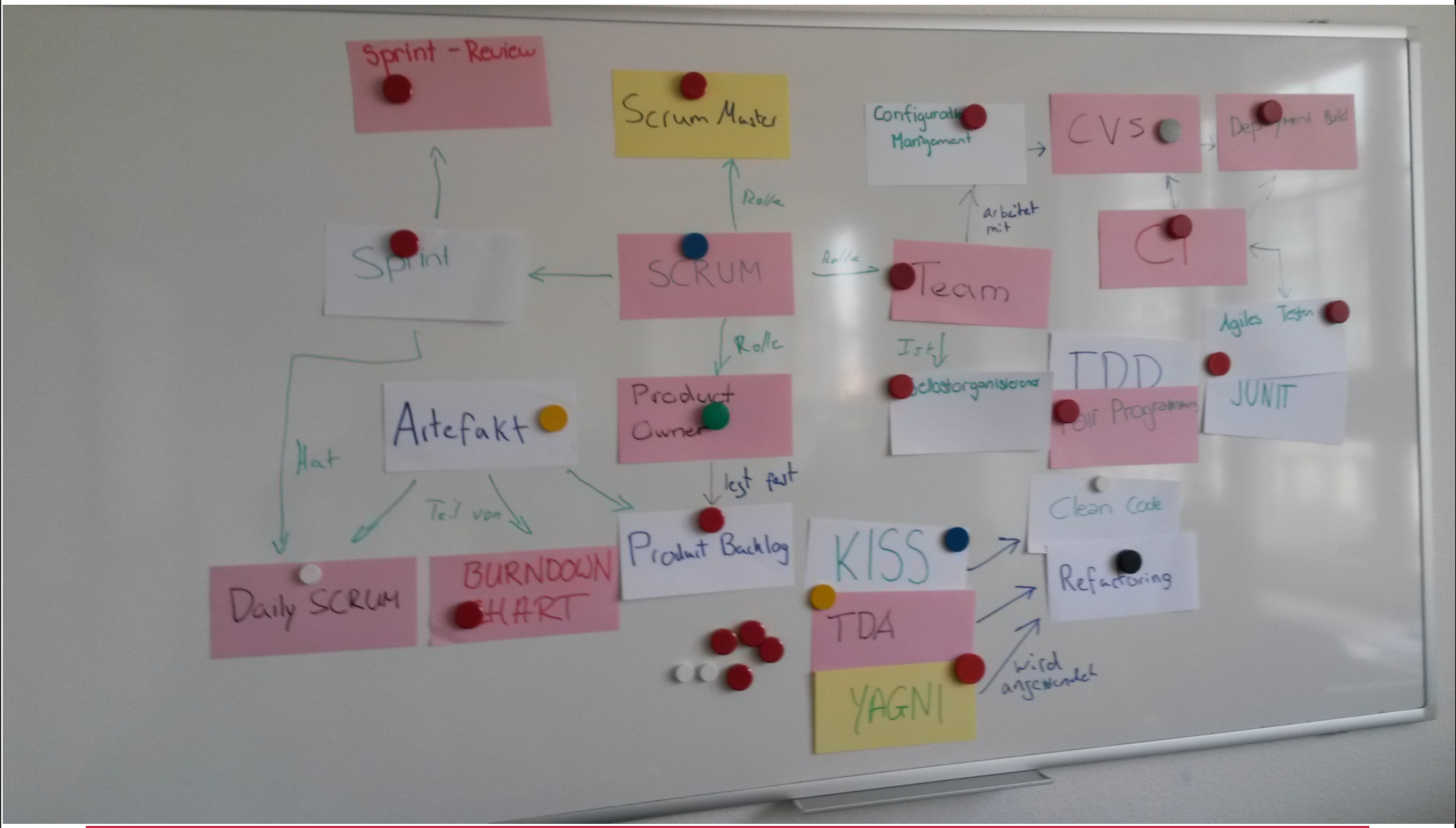
Tagesplan

- A little repetition
- Daily Scrum
- Putting things together
- Lunch
- Scrum Team
- Sprint Review & Retrospektive

Warmup: Netzwerk erstellen

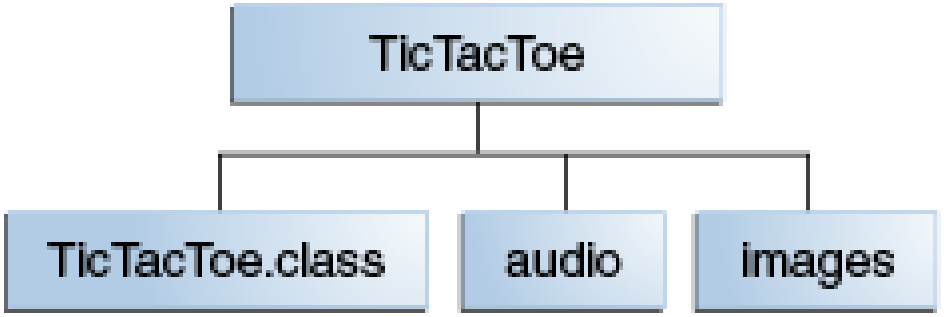
- KISS
- Configuration Management
- Agiles Testen
- Selbstorganisierend
- JUNIT
- SCRUM
- Pair Programming
- Deployment Build
- Sprint
- Clean Code
- Artefakt
- Scrum Master
- Refactoring
- TDD
- Product Backlog
- Daily Scrum
- Product Owner
- CVS
- Team
- TDA

Resultat


Sprint - Review
Scrum Master
Configuration Management
CVS
Deployment Build
Rolle
arbeitet mit
Sprint
SCRUM
Rolle
Team
CI
Rolle
Ist
Agiles Testen
TDD
Pair Programming
JUNIT
Product Owner
selbstorganisierend
Hat
Artefakt
Teil von
Ist fest
Clean Code
Product Backlog
KISS
Refactoring
Daily SCRUM
BURNDOWN CHART
TDA
wird angewendet
YAGNI

Vom Quellcode zum Programm

- `javac *.java`
- `junit *.class`
- `jar -cf TicTacToe.jar TictacToe.class audio images`



Mit ANT zu Continous Integration

ANT (Another Neat Tool): Automatisieren komplexer Compiler, Test und Pack-Prozessen, build.xml

```
<?xml version="1.0"?>
<project name="Ant-Test" default="main" basedir=".">
<!-- Variables used for JUnit testin -->
    <property name="test.dir" location="src" />
    <property name="test.report.dir" location="testreport" />

    <!-- Deletes the existing build, docs and dist directory-->
    <target name="clean">
     <echo>Cleaning up...</echo>
            <delete dir="${build.dir}" />
            <delete dir="${docs.dir}" />
            <delete dir="${dist.dir}" />
            <delete dir="${test.report.dir}" />
    </target>

    <!-- Creates the  build, docs and dist directory-->
    <target name="makedir">
     <echo>recreate directory structure</echo>
            <mkdir dir="${build.dir}" />
            <mkdir dir="${docs.dir}" />
            <mkdir dir="${dist.dir}" />
            <mkdir dir="${test.report.dir}" />
    </target>

<!-- Run the JUnit Tests -->
    <!-- Output is XML, could also be plain-->
<target name="junit" depends="compile" description="execute JUnit tests" >
     <junit printsummary="on" fork="true" haltonfailure="yes">
         <classpath refid="junit.class.path"/>
         <formatter type="xml" />
         <batchtest todir="${test.report.dir}">
            <fileset dir="${src.dir}">
               <include name="**/*Test*.java" />
            </fileset>
         </batchtest>
    </junit>
</target>
```

```
    <!-- Compiles the java code (including the usage of library for processing -->
<target name="compile_model" description="compiles the model and networking portion" >
<echo>Compiling the model and networking packages</echo>
    <javac srcdir="${src.dir}" destdir="${build.dir}" includeantruntime="false">
     <include name="ch/m226a/gameobjects/**,ch.m226a.net"/>
     <exclude name="test/m226a/**" />
          <classpath refid="junit.class.path" />
        </javac>
    </target>

    <target name="compile" depends="clean, makedir, compile_model" description="compiles the entire
package">
     <echo>Compiling everything else</echo>
        <javac srcdir="${src.dir}" destdir="${build.dir}" includeantruntime="false">
          <classpath refid="junit.class.path" />
        </javac>
    </target>

    <!-- Creates Javadoc -->
    <target name="docs" depends="compile" description="generate the documentation">
        <javadoc packagenames="src" sourcepath="${src.dir}" destdir="${docs.dir}">
         <!-- Define which files / directory should get included, we include all -->
          <fileset dir="${src.dir}"><include name="**" /> </fileset>
       </javadoc>
    </target>

    <!--Creates the deployable jar file  -->
    <target name="jar" depends="compile" description="generate the jar file for the app">
     <tstamp/>
        <jar destfile="${dist.dir}/robigame_${DSTAMP}.jar" basedir="${build.dir}">
          <!-- add the processing library -->
          <zipfileset includes="**/*.class" src="/home/sven/apps/processing-3.3/core/library/core.jar"/>
            <manifest>
                    <attribute name="Main-Class"
                         value="ch.m226a.gamecontrol.GameStart"/>
               <!-- unneeded since processing library is included now -->
               <!--attribute name="Class-Path"
               value="${junit.class.path}" / doesn't work that easy... -->
            </manifest>
        </jar>
    </target>

    <target name="main" depends="compile, jar">
          <description>Main target</description>
    </target>
</project>
```

ANT Erweiterungen für CI

- Tasks: javac, junit, jar, mkdir, rmdir, javadoc, copy, ...
- Abhängigkeiten

Weitere Build-Tools für CI

PHPUnit

- Intro:

  http://slides.com/johnnorton/be-nice-to-future-you-write-unit-tests-with-phpunit#/0/13

- Installation (im Entwicklungssystem):

  https://phpunit.de/manual/current/en/installation.html