

# Software Engineering Projekt (SEP) mit ROBOCODE

Klaus Knopper <[robocode@knopper.net](mailto:robocode@knopper.net)>

16.07.2007

<http://robocode.sourceforge.net/>



## Kurzbeschreibung

Es wird mit den Methoden des Software Engineering in Teamarbeit ein virtuelles Roboter-  
team von Planung (Spezifikation, Design, Gewinnstrategie, Ablaufplan, ...) bis zur Imple-  
mentation aufgebaut, und in Form einer gemeinsamen Abschlusspräsentation der Teams  
sowie eines technischen Handbuchs dokumentiert.

Lernziele:

Praktische Anwendung des Software Engineering, Fähigkeit zum Arbeiten im Team, Fähig-  
keit zur Dokumentation und Präsentation der Arbeitsergebnisse.

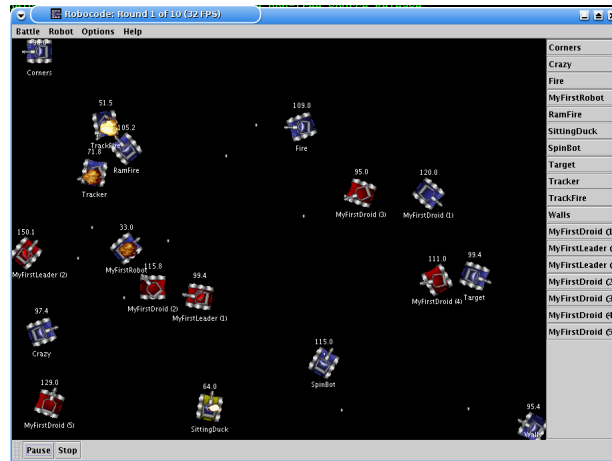
Voraussetzung für die Teilnehmer:

- Grundkenntnisse in JAVA
- Grundlagen Software-Engineering

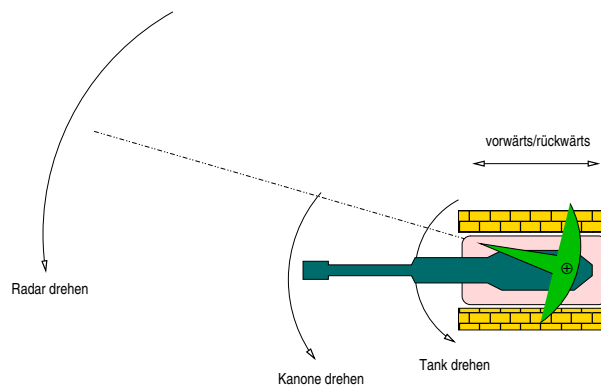
Das Projekt ist betriebssystemunabhängig, und kann sowohl auf den Übungsrechnern im  
PC-Pool, als auch auf mitgebrachten, eigenen Notebooks implementiert werden.

# 1 Einführung: Robocode

Robocode ist ein OpenSource-Projekt von IBM, das zum Erlernen bzw. Vertiefen der objektorientierten Programmiersprache JAVA eingesetzt wird. Da in Robocode eine umfangreiche graphische Laufzeit- und Entwicklungsumgebung vorhanden ist, mit einer ausgezeichneten Beschreibung der API für den Programmierer, sind hiermit auch die im Studium erlernten Methoden der zielgerichteten Softwareentwicklung praktisch einsetzbar.



Robocode ist eine Simulation von Robotern (Tanks), die sich in einer virtuellen Arena einstellbarer Größe bewegen, ihre Umgebung mit Hilfe eines Radars scannen und sich mit Hilfe von Projektilen gegenseitig bekämpfen können. Weiterhin sind Methoden zum Nachrichtenaustausch zwischen „befreundeten“ Robotern, Hilfen zur optischen Kontrolle der Strategieverfolgung (Farbwechsel, Sichtbarmachen der gescannten Umgebung) und Kollisionserkennung vorhanden, die mitverwendet werden können.



## Vereinfachte Spielregeln

Jeder Roboter hat zu Beginn der Simulation eine bestimmte Energiemenge zur Verfügung, die durch Bewegung, Kollisionen mit anderen Robotern oder Treffer fremder Projektilen, Abfeuern eigener Projektilen oder Eigenverbrauch reduziert wird. Ist die verbliebene Energie 0, explodiert der Roboter, und das Spiel ist für ihn damit beendet. Gewonnen hat am Ende der übriggebliebene Roboter, bzw. dessen Team.

## 2 Aufgabenstellung

### 2.1 Ziel

Ziel des Projektes ist es, ein Team von genau 2 Robotern zu **planen**, zu **implementieren** und zu **dokumentieren**. Das Roboterteam soll mit einer gewinnenden Strategie sowohl gegen die von den anderen Kursteilnehmern programmierten Roboterteams, als auch gegen die in der Robocode-Umgebung vorhandenen Beispielroboter und -teams erfolgreich antreten können.

### 2.2 Projektbearbeitung in Teams

Es werden zu Beginn des Kurses Teams von 3 (empfohlene Teamgröße), maximal 4 Personen gebildet. Innerhalb des Teams sollen einerseits alle teilnehmenden Personen mit fest zugewiesenen, speziellen Aufgaben betraut werden, andererseits auch frei an Teilprojekten (z.B. Implementation, Dokumentation) mitarbeiten. Eine Person aus dem Team wird zum Projektleiter/zur Projektleiterin gewählt, der oder die für das Ressourcen-, Zeit- und Qualitätsmanagement verantwortlich ist. Analyse, Planung, Implementation, Tests und Präsentation können von jeweils einer oder mehreren Personen geleitet werden, speziell an der Abschlusspräsentation sollten alle Teammitglieder beteiligt sein und einen Teil vortragen. In der Dokumentation (s.S. 4) soll ersichtlich werden, welche Person während des Projektes für welche Aufgabe(n) verantwortlich war, um neben der Gesamtwertung für die Gruppe auch eine personenbezogene Einzelbewertung durchführen zu können.

### 2.3 Methoden

Es sollen die Methoden des Software Engineering angewandt werden, also

1. Planung,
2. Analyse,
3. Entwurf,
4. Programmierung,
5. Tests,
6. ... begleitende Prozesse, insbesondere Dokumentation,

mit Validierung und Verifizierung in geeigneten Phasen des Projektverlaufs.

## 3 Vorgehen

### 3.1 Projektplan/Zeitplan

Vom Projektleiter ist in einem frühen Projektstadium in Absprache mit den übrigen Teammitgliedern ein Projektplan zu erstellen, die Aufgaben sind festzulegen und zu verteilen.

Beispielsweise könnte der Projektplan für Tag 1 eine „Orientierungsphase“ vorsehen, um die Robocode Laufzeit- und Entwicklungsumgebung kennenzulernen, und einen ersten Blick in die Beispielprogramme und die Robocode-API zu werfen, Tag 2 für Analyse der Möglichkeiten, die die Robocode-API bietet, Design (Entwurf einer Gewinnstrategie) und Planung (wer kümmert sich um welchen Teil der Implementation, welche Module sind wann fertigzustellen und zu testen?), Tag 3 für die Implementierung und Tests.

### 3.2 Analyse und Design

Auch wenn es den einen oder anderen Teilnehmer vielleicht reizt, gleich „draufloszuprogrammieren“, soll die Arbeit an der Software zielgerichtet ablaufen. Es wird zunächst gemeinsam die Strategie, die jeder der Teamroboter verfolgen kann, festgelegt, und in der Implementationsphase soll diese Strategie nach Möglichkeit nicht mehr geändert werden. Da hierzu die Kenntnis der Möglichkeiten Voraussetzung ist, müssen sich die Teilnehmer vorher eingehend mit der Robocode-API, insbesondere der Klasse **TeamRobot** und den darin vorhandenen Basisklassen beschäftigt haben.

### 3.3 Implementation

In der Implementationsphase sollen (Vorgabe), ausgehend von Objekten der Klasse **TeamRobot**, zwei Roboter programmiert werden. Diese sollen sich gegenseitig bei der Erreichung des Ziels (s.S. 2) unterstützen, was durch geeignete Schnittstellen erreicht werden kann. Das Team kann in der virtuellen Arena von Robocode getestet werden.

Um fair gegenüber den anderen Projektteams zu bleiben, dürfen nur die in Robocode vorhandenen Beispiele als Basis dienen, und keine 1:1-Kopien aus dem Internet heruntergeladener „Killermaschinen“, die oft auch kaum verständlich programmiert und unzureichend dokumentiert sind. Sie dürfen aber durchaus Strategien wiederverwenden, die Sie bei anderen Robotern gesehen haben, sofern Sie diese durchschauen und gut dokumentieren können.

### 3.4 Abschlusspräsentation/Dokumentation

Am letzten Tag des Projektes muss jedes Team

1. Eine Dokumentation
2. Eine Präsentation
3. Einen funktionsfähigen Prototyp bzw. ein fertiges Roboter-Team (**jar**-File)

abliefern.

### 3.4.1 Dokumentation

Die Dokumentation soll, pro Team eine, in elektronischer Form sowie in mindestens einem gedruckten Exemplar beim Kursleiter abgegeben werden und enthält:

- Den Namen des Projektes,
- Die Namen der in diesem Team beteiligten Personen und ihre Funktionen für das Projekt,
- Eine **kurze** Einführung („Worum geht es überhaupt?“),
- das Pflichtenheft, insbesondere
  - Eine Beschreibung des Vorgehens und der verwendeten Methoden bei der Realisierung des Projektes,
  - Eine Erläuterung der Strategie für jeden programmierten Roboter,
  - Die Umsetzung dieser Strategie, z.B. mit Beispielcode aus der Implementation,
- Ein Fazit: Wurde das Ziel erreicht? Welche Problempunkte sind noch ungelöst? Was könnte man anders/besser machen, wie hoch ist der Aufwand dafür?

### 3.4.2 Präsentation

Das Team stellt sein abgeschlossenes Projekt in Form einer Kurzpräsentation (max. 20 Minuten, plus 10 Minuten Diskussion) vor. Hierbei kann der Beamer oder die Tafel mitverwendet werden. Falls Live-Demonstrationen am Rechner gezeigt werden, sind diese im Vorfeld so vorzubereiten, dass keine zusätzliche Zeit für die Technik aufgewandt werden muss.

## 3.5 Demonstration

Nach Abschluss aller Präsentationen erfolgt (als Demonstration der Ergebnisse) eine Gegenüberstellung aller Teams in einer gemeinsamen, großen Robocode-Arena in 10 Runden, um das Team mit der „besten“ Strategie zu ermitteln. Hierzu ist dem Kursleiter ein Snapshot (**jar**-Export) des programmierten Roboter-Teams zur Verfügung zu stellen.

Während oder nach der Demonstration erfolgt eine Abschlussdiskussion mit allen Teilnehmern.