DAS PROJEKT ANTZEIT

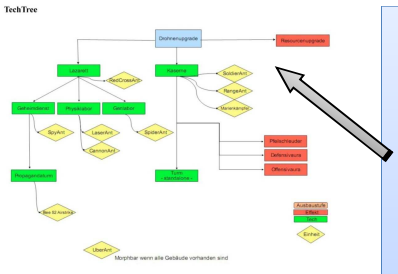
der Bundeswehr Universität München

ALLGEMEINE INFORMATIONEN & SPIELIDEE

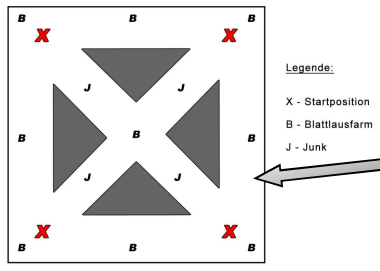
Im Rahmen des an der UniBw-München durchgeführten Praktikums „Game-Development“ ging es darum, den iterativen Prozess der Spielentwicklung zu erleben. Unser Team einigte sich auf die Erstellung eines Multiplayer-Echtzeitstrategiespiels im Universum der Ameisen. Das Ergebnis war ein Strategiespiel mit Grundfunktionalität, welches in Form einer Studienarbeit zu einer serverzentrierten Simulation weiterentwickelt wurde. Den Entwicklungsprozess des Praktikums haben wir auf Basis einer eigenen Website dokumentiert.

Genre	Echtzeit-Strategie
Modus	Multiplayer
Sicht	Iso-Perspektive
Sprachplattform	Java JDK 1.6
Framework	SLICK (OpenGL)

GAME DESIGN (DOCUMENT)



Map 1: „Crossroads“



Spielmechanik

In AntZeit steuert der Spieler ein Ameisenvolk. Die technologische Entwicklung ist baumartig strukturiert. Der Spieler sammelt Ressourcen um neue Technologien und Einheiten zu erforschen und um letztendlich die Dominanz über die Karte zu erlangen. Dabei ist eine gerechte Aufteilung der Spielwelt zu Beginn nötig.

GUI-Entwurf

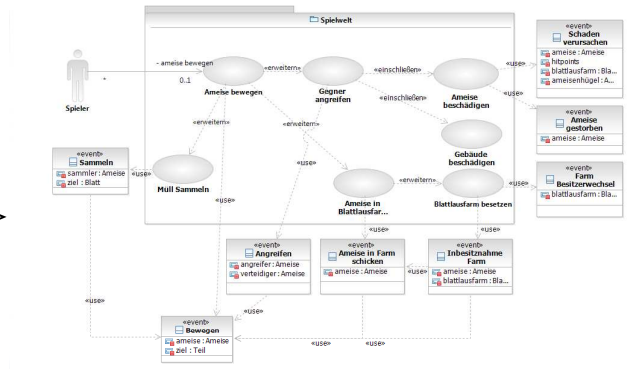
Eine erste Überlegung zum User Interface – wir haben uns an bekannten Spielen orientiert und Ideen kombiniert.

PLANUNG & ANALYSE

Aus dem Gamedesign extrahierten wir die technischen Anforderungen und teilten das Projekt in Meilensteine auf. Nach der Priorisierung und einer Aufwandsschätzung kamen wir zu dem Schluss, das Pathfinding, den Iso–Layer und das Kampfsystem in **Wegwerfprototypen** zu testen und so das Risiko zu minimieren.

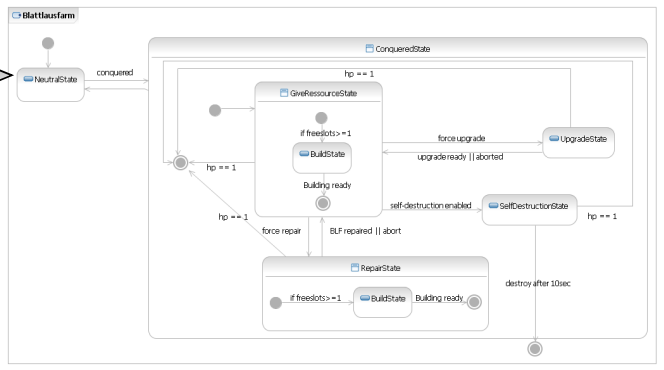
Use-Cases

Die technischen Anforderungen verfeinerten wir durch den Einsatz von Use-Case Diagrammen. Die Aktionen in der Spielwelt werden durch Ereignisse realisiert. Hier wird die Integration der möglichen Spieleraktionen in das Ereignissystem abgebildet.



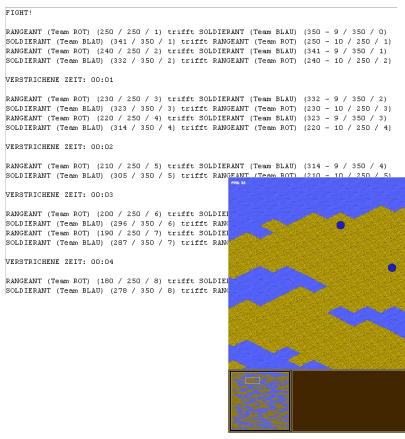
State-Charts

Gebäude und Einheiten sind Zustandsautomaten. Exemplarisch ist hier die Blattlausfarm dargestellt.

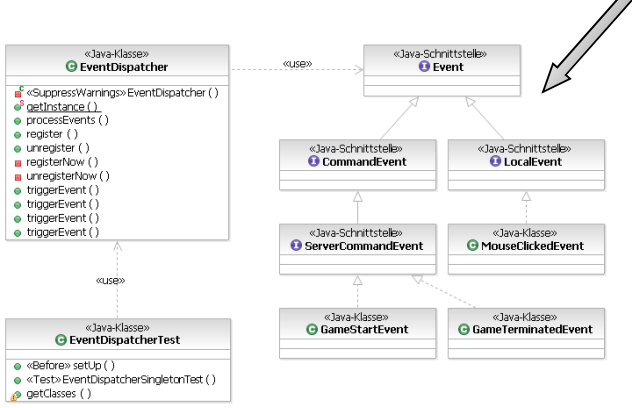


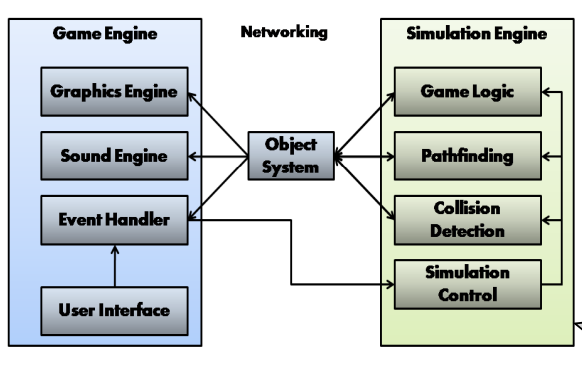
Klassendiagramme

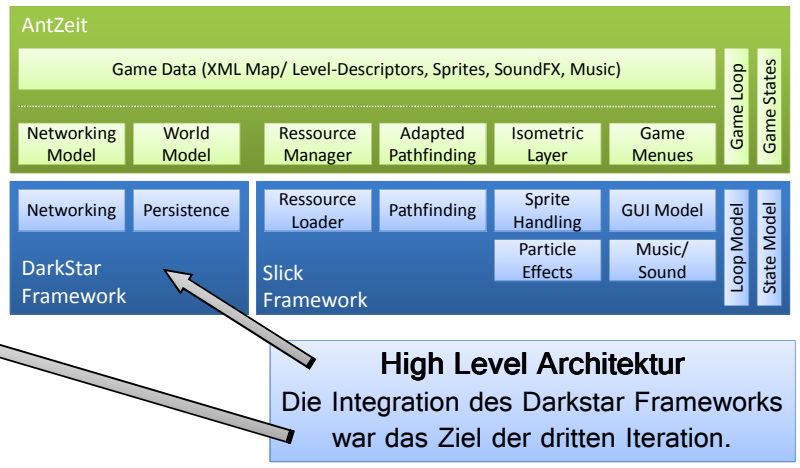
Die statischen Strukturen des Spiels werden mit Klassendiagrammen modelliert. Hier ist exemplarisch die Ereignisstruktur und ein zugehöriger JUnit-Test modelliert.



DESIGN

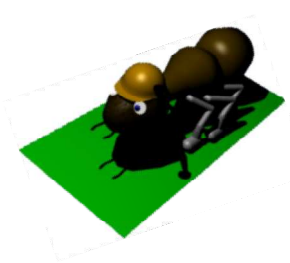






High Level Architektur

Die Integration des Darkstar Frameworks war das Ziel der dritten Iteration.



IMPLEMENTIERUNG

Iso-Layer

Die grafische Ausgabe wird über eine gekachelte isometrische Ansicht realisiert. Die Spieltwelt wird von Slick mittels OpenGL gerendert.

Pathfinding

Die AntZeit-Einheiten nutzen den A∗-Algorithmus zur Wegfindung. Dieser ähnelt dem Dijkstra-Algorithmus und gilt als einer der effizientesten „Best-First“- Algorithmen.

GUI-Elemente

Das GUI besteht aus drei Einzelelementen, die situationssensitiv Informationen darstellen.

Minimap

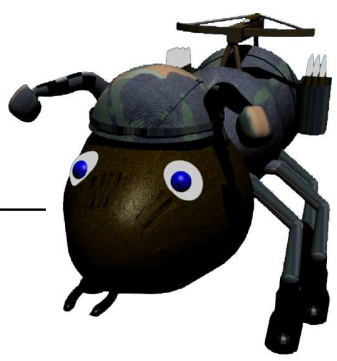
Die Minimap ist eine Miniaturansicht der Karte, um die Übersichtlichkeit zu verbessern.

Testen

Der **vertikale Prototyp** wurde fortlaufend auf Basis der Use-Cases getestet.

METRIKEN

Teamgröße	6 Personen	**Netto-Arbeitszeit**	1500 Stunden
Projektdauer	3 Monate Praktikum (6) + 3 Monate Studienarbeit (1)	**Wochenstunden**	21 h / Person
		LOC	~22.000
		Anzahl Klassen	~200



Team KAROSHI [DEV]

{rene.froehlich|erik.haenel|markus.knopp|daniel.nemschok|alexander.schramm}@unibw.de

Institut für Softwaretechnologie, Universität der Bundeswehr München, {daniel.volk}@unibw.de