

# Arcade Game

<b>Documentsoort:</b>	Behoeftespecificatie
<b>Versie:</b>	1.0
<b>Datum:</b>	17 Feb 2015
<b>Auteur:</b>	Quinten Soetens
<b>Status:</b>	Opgeleverd

## Samenvatting

Dit document bevat de specificaties voor de basis van een arcade game. Het is geschreven in het kader van het vak "Inleiding software Engineering" (1ste bachelor informatica - Universiteit Antwerpen).

## Legende

Een typische use-case bevat de volgende onderdelen.

*Referentennummer & titel:*

Wordt gebruikt om naar een bepaalde use-case te verwijzen.

*Prioriteit:*

De specificatie van een systeem vraagt meer dan wat binnen de voorziene tijd op te leveren is. Vandaar dat we per use-case aangeven in hoeverre die functionaliteit belangrijk is. In volgorde van belangrijkheid kan hier staan: VERPLICHT (deze use-case moet opgeleverd worden), BELANGRIJK (niet essentieel maar bij voorkeur toch opleveren), NUTTIG (interessant maar kan weggelaten worden).

*Doel:*

Summiere beschrijving van het waarom van de use-case, t.t.z. wat de use-case bijdraagt tot de gehele functionaliteit.

*Preconditie:*

Summiere beschrijving van de uitgangspunten bij aanvang van de use-case.

*Succesvol einde:*

Summiere beschrijving van wat opgeleverd zal worden als er niks fout is gegaan.

*Stappen:*

Een sequentiële beschrijving van hoe de use-case precies zal verlopen als alles goed gaat (het zogenaamde "happy day scenario"). De stappen zijn genummerd en kunnen controle instructies (WHILE, IF, ...) bevatten.

*Uitzonderingen:*

Een lijst van mogelijke probleemgevallen en hoe die behandeld zullen worden. Een probleem geval (a) verwijst naar het nummer van de stap waar het probleem kan

optreden, (b) bevat een conditie die aangeeft wanneer het probleemgeval optreedt, (c) omschrijft heel kort (een lijn) hoe het probleem behandeld zal worden.

*Voorbeeld:*

Een voorbeeld van wat in- of uitgevoerd kan worden.

Soms is een use-case een uitbreiding van een andere use-case, en dan zijn volgende onderdelen relevant.

*Uitbreiding:*

Een referte naar de use-case waarvan deze een uitbreiding is.

*Stappen:*

Een lijst van extra en/of aangepaste stappen t.o.v de use-case waarvan deze een uitbreiding is.

Een uitbreiding (a) verwijst naar het nummer van de stap die uitgebreid wordt, (b) zegt of de uitbreiding voor, na of tijdens de normale stap zal gebeuren, (c) omschrijft wat precies in de uitbreiding zal gebeuren.

# Behoeftes

Hieronder volgt een opsomming van alle use-cases inclusief hun prioriteit.

Use-case	Prioriteit
<i>1: Invoer</i>	
1.1. Speelveld inlezen	VERPLICHT
1.2. Bewegingen inlezen	VERPLICHT
<i>2: Uitvoer</i>	
2.1. Huidig speelveld wegschrijven	VERPLICHT
2.2. Resterende bewegingen wegschrijven	VERPLICHT
<i>3: Simulatie</i>	
3.1. Bewegen over het speelveld	VERPLICHT
3.2. Verplaatsen van beweegbare obstakels	VERPLICHT
3.3. Automatisch uitvoeren van bewegingen	VERPLICHT

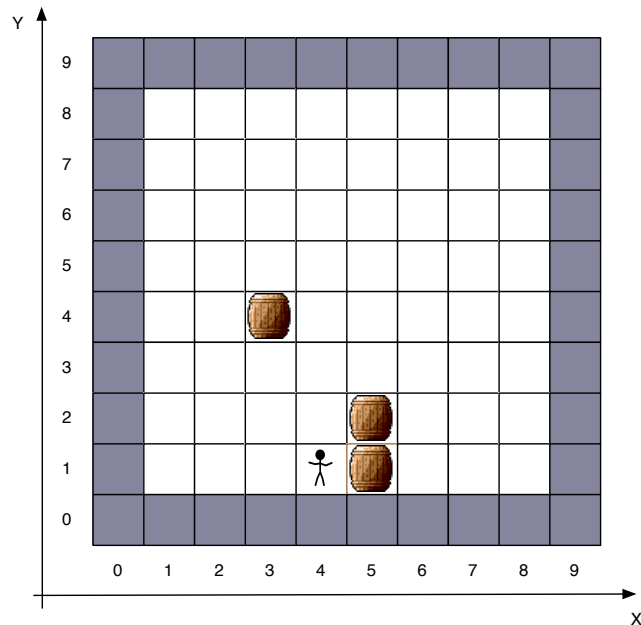
# 1. Invoer

1.1. Speelveld inlezen	
<b>Prioriteit</b>	VERPLICHT
<b>Doel</b>	De begin-situatie van het systeem bestaat uit een speelveld waarop één speler en meerdere obstakels gepositioneerd zijn. En waarin Informatie over de vorm van het speelveld en de posities van de speler en obstakels is beschikbaar in een XML-gestructureerde file. Doel van deze usecase is het inlezen van de XML-file en controleren of de beginsituatie fysisch haalbaar is.
<b>Preconditie</b>	Een XML bestand met daarop een beschrijving van het speelveld. (Zie Appendix 1 voor meer informatie over het XML formaat)
<b>Succesvol einde</b>	Het systeem bevat de beginsituatie van het speelveld.
<b>Stappen</b>	<ol style="list-style-type: none"><li>1. Open het invoerbestand</li><li>2. Parse het bestand met TinyXML)</li><li>3. WHILE Er zijn nog elementen in het geparse document<ol style="list-style-type: none"><li>3.1. Herken het soort element (één van VELD, SPELER of OBSTAKEL)</li><li>3.2. Lees de verder informatie voor dit element</li><li>3.3. IF Verifieer de geldigheid van het element<ol style="list-style-type: none"><li>3.3.1. THEN voeg element toe aan het speelveld</li></ol></li></ol></li><li>4. Verifieer de consistentie van het speelveld</li></ol>
<b>Uitzonderingen</b>	<ol style="list-style-type: none"><li>3.1. [Onherkenbaar element] <b>Foutboodschap</b> + positioneer op volgende element in het bestand =&gt; verdergaan vanaf stap 3</li><li>3.3. [Ongeldige informatie] <b>Foutboodschap</b> + positioneer op volgende element in het bestand =&gt; verdergaan vanaf stap 3</li><li>4. [Inconsistent speelveld] <b>Foutboodschap</b></li></ol>

## 1.1. Speelveld inlezen

### Voorbeeld

Een speelveld (zoals op onderstaande prent) staat beschreven in het bestand Speelveld1.0.xml.



## 1.2. Bewegingen inlezen

### Prioriteit

VERPLICHT

### Doel

Lees de verschillende bewegingen in voor de speler.

### Preconditie

Een XML bestand met de beschrijvingen van de bewegingen.

### Succesvol einde

Het systeem bevat de bewegingen die sequentieel in de volgorde van voorkomen in het bestand uitgevoerd kunnen worden.

### Stappen

1. Open het invoerbestand met de bewegingen
2. Parse het bestand (met TinyXML)
3. WHILE Er zijn nog elementen in het geparse document
  - 3.1. Herken het soort element (BEWEGING)
  - 3.2. Lees de verder informatie voor dit element
  - 3.3. IF Verifieer de geldigheid van het element
    - 3.3.1. THEN voeg beweging toe aan de overige bewegingen

## 1.2. Bewegingen inlezen

<b>Uitzonderingen</b>	3.1. [Onherkenbaar element] Foutboodschap + positioneer op volgende element in het bestand => verdergaan vanaf stap 3 3.3. [Ongeldige informatie] Foutboodschap + positioneer op volgende element in het bestand => verdergaan vanaf stap 3
<b>Voorbeeld</b>	Zie het bestand Bewegingen1.0.xml

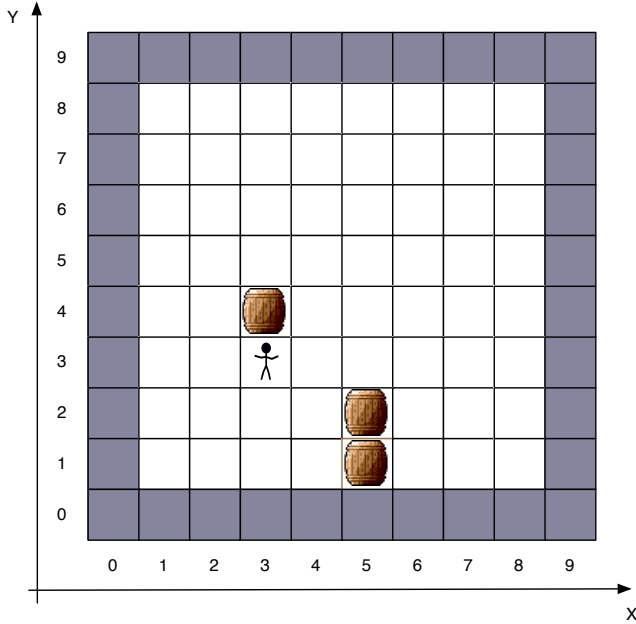
## 2. Uitvoer

2.1. Huidig speelveld wegschrijven	
<b>Prioriteit</b>	VERPLICHT
<b>Doel</b>	Volledige speelveld zoals hij op moment van uitvoeren is, wegschrijven naar een bestand. Dit bestand beschrijft dus a) hoe het speelveld er uit ziet, b) waar de speler en beweegbare obstakels zich bevinden.
<b>Preconditie</b>	Het systeem is correct geïntialiseerd Een speelveld werd ingelezen
<b>Succesvol einde</b>	Het systeem heeft een tekstbestand (ASCII) uitgevoerd, waarin de informatie over het speelveld netjes is uitgeschreven.
<b>Stappen</b>	1. Creëer uitvoerbestand 2. Open uitvoerbestand 3. Schrijf gegevens uit voor het speelveld. 4. FORALL Entiteiten 4.1. Schrijf gegevens uit voor de entiteit 5. Sluit uitvoerbestand
<b>Uitzonderingen</b>	1. [Creatie mislukt] Foutboodschap + schrijf naar console ipv bestand
<b>Voorbeeld</b>	<p>gegeven de input uit het voorbeeld van 1.1, waarbij we een speelveld hebben ingelezen met één speler en drie beweegbare obstakels: Bestand: HuidigSpeelveld.txt</p> <p>Het huidige speelveld is Level 1: Eigenschappen van dit veld: -Breedte 10 -Lengte 10</p> <p>Speler Chip bevindt zich in dit speelveld op positie (4,1).</p> <p>Er bevindt zich een ton op positie (5,1).</p> <p>Er bevindt zich een ton op positie (5,2).</p> <p>Er bevindt zich een ton op positie (3,4).</p>

2.2. Resterende bewegingen wegschrijven	
<b>Prioriteit</b>	VERPLICHT
<b>Doel</b>	Alle nog niet gemaakte bewegingen wegschrijven naar een bestand.
<b>Preconditie</b>	Het systeem is correct geïnitieerd Een speelveld werd ingelezen
<b>Succesvol einde</b>	Het systeem heeft een tekstbestand (ASCII) uitgevoerd, waarin de informatie over de resterende bewegingen netjes is uitgeschreven.
<b>Stappen</b>	<ol style="list-style-type: none"> <li>1. Creëer uitvoerbestand</li> <li>2. Open uitvoerbestand</li> <li>3. WHILE nog te maken bewegingen aanwezig <ol style="list-style-type: none"> <li>3.1. Schrijf gegevens uit voor de beweging</li> </ol> </li> <li>4. Sluit uitvoerbestand</li> </ol>
<b>Uitzonderingen</b>	<ol style="list-style-type: none"> <li>1. [Creatie mislukt] Foutboodschap + schrijf naar console ipv bestand</li> <li>3. [Geen te maken bewegingen aanwezig] Als er geen te maken bewegingen zijn, is de output "geen bewegingen"</li> </ol>
<b>Voorbeeld</b>	<p>gegeven de input uit het voorbeeld van 1.2,</p> <p>Bestand: ResterendeBewegingen.txt</p> <p>Speler Chip zal volgende beweging nog uitvoeren: Omhoog</p> <p>Speler Chip zal volgende beweging nog uitvoeren: Omhoog</p> <p>Speler Chip zal volgende beweging nog uitvoeren: Links</p> <p>Speler Chip zal volgende beweging nog uitvoeren: Omhoog</p> <p>Speler Chip zal volgende beweging nog uitvoeren: Links</p> <p>Speler Chip zal volgende beweging nog uitvoeren: Omhoog</p> <p>Speler Chip zal volgende beweging nog uitvoeren: Rechts</p>



### 3. Simulatie

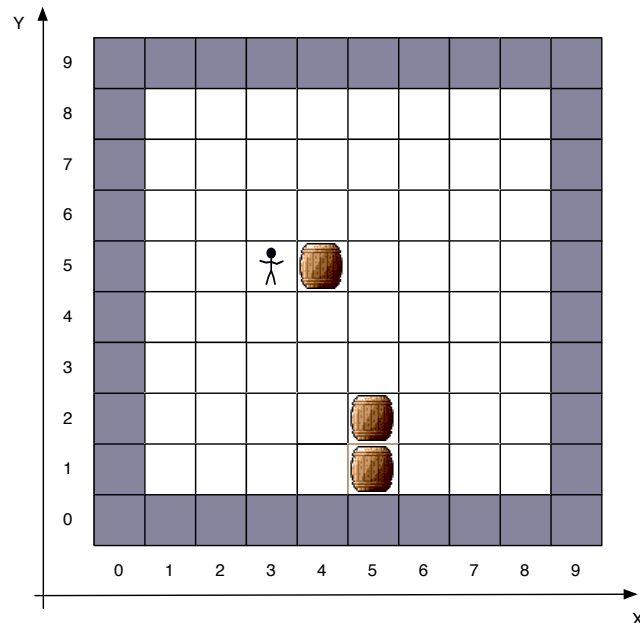
3.1: Bewegen over het speelveld	
<b>Prioriteit</b>	VERPLICHT
<b>Doel</b>	Het uitvoeren van een beweging door de speler.
<b>Preconditie</b>	Het systeem is correct geïnitieerd. Een speelveld werd ingelezen. Een beweging werd ingelezen.
<b>Succesvol einde</b>	De speler waarvoor de beweging werd gedefinieerd bevindt zich op de juiste eindpositie.
<b>Stappen</b>	1.1 IF richting beweging is “omhoog” 1.1.1 verplaats de speler 1 positie hoger op de y coördinaat. 1.2. ELSE IF richting beweging is “rechts” 1.2.1 verplaats de speler 1 positie hoger op de x coördinaat. 1.3. ELSE IF richting beweging is “omlaag” 1.3.1 verplaats de speler 1 positie lager op de y coördinaat. 1.4. ELSE IF richting beweging is “links” 1.4.1 verplaats de speler 1 positie lager op de x coördinaat.
<b>Uitzonderingen</b>	1.1.1,1.2.1 [Speler raakt onbeweegbaar obstakel] stop beweging & geef foutmelding wanneer nog niet over de volledige afstand verschoven.
<b>Voorbeeld</b>	<p>Na het uitvoeren van de eerste drie bewegingen uit voorbeeld 1.2 toegepast op de situatie uit voorbeeld 1.1 krijg je volgende situatie.</p> 

3.2 Verplaatsen van beweegbare obstakels	
<b>Prioriteit</b>	VERPLICHT
<b>Doel</b>	De speler is in staat om beweegbare obstakels te verplaatsen.
<b>Preconditie</b>	Het systeem is correct geïnitieerd. Een speelveld werd ingelezen. Bewegingen werden ingelezen.
<b>Succesvol einde</b>	Een speler die een beweging maakt in de richting van een positie waar zich een beweegbaar obstakel bevindt, zal zich in die richting verplaatsen en tevens het beweegbaar obstakel in dezelfde richting verplaatsen.
<b>Stappen</b>	1.1 IF richting beweging is “omhoog” AND boven de speler bevindt zich een beweegbaar obstakel 1.1.1 verplaats het obstakel 1 positie hoger op de y coördinaat. 1.1.2 verplaats de speler 1 positie hoger op de y coördinaat. 1.2. ELSE IF richting beweging is “rechts” AND rechts van de speler bevindt zich een beweegbaar obstakel 1.2.1 verplaats het obstakel 1 positie hoger op de x coördinaat. 1.2.2 verplaats de speler 1 positie hoger op de x coördinaat. 1.3. ELSE IF richting beweging is “omlaag” AND onder de speler bevindt zich een beweegbaar obstakel 1.3.1 verplaats het obstakel 1 positie lager op de y coördinaat. 1.3.2 verplaats de speler 1 positie lager op de y coördinaat. 1.4. ELSE IF richting beweging is “links” AND links van de speler bevindt zich een beweegbaar obstakel 1.4.1 verplaats het obstakel 1 positie lager op de x coördinaat. 1.4.2 verplaats de speler 1 positie lager op de x coördinaat.
<b>Uitzonderingen</b>	1.1.1, 1.2.1, 1.3.1, 1.4.1 [Er staat nog een ander obstakel (beweegbaar of onbeweegbaar) achter het beweegbaar obstakel] stop beweging & geef <b>foutmelding</b> !

### 3.2 Verplaatsen van beweegbare obstakels

#### Voorbeeld

Na het uitvoeren van de laatste vier bewegingen uit voorbeeld 1.2 toegepast op de situatie uit voorbeeld 3.1 krijg je volgende situatie.



### 3.3 Automatisch uitvoeren van bewegingen

<b>Prioriteit</b>	VERPLICHT
<b>Doel</b>	Een volledig scenario van meerdere bewegingen wordt uitgevoerd.
<b>Preconditie</b>	Het systeem is correct geïnitieerd. Een speelveld werd ingelezen. Bewegingen werden ingelezen.
<b>Succesvol einde</b>	Er zijn geen onafgewerkte bewegingen. Alle spelers en obstakels bevinden zich in de juiste eindpositie.
<b>Stappen</b>	1. WHILE nog niet afgewerkte bewegingen 1.1. beweeg speler volgens beweging (zie use-case 3.1 & 3.2)
<b>Uitzonderingen</b>	Geen
<b>Voorbeeld</b>	

## Appendix 1

### Geldige informatie

We moeten nu nog de tags en hun waarde vastleggen die gelden voor ons probleem-domein. Dit bestaat uit het vastleggen van de mogelijke tags, Attributen en de verwachte inhoud (content).

De mogelijke tag-identifiers zijn:

veld, naam, lengte, breedte, speler, obstakel, type, bewegingen, beweging, spelernaam, richting

Coördinaten in het speelveld worden weergegeven met attribuut-identifiers X en Y.

Tag-identifiers en attribuut-identifiers zijn niet hoofdletter gevoelig, inhouden zijn dit wel.

Tags	Subtags	Attributen	Content
Veld	Naam		String
	Lengte		Integer
	Breedte		Integer
	Speler		Zie tag definitie hieronder
	Obstakel		Zie tag definitie hieronder
Speler		X	Integer
		Y	Integer
	Naam		String
Obstakel		X	Integer
		Y	Integer
		Beweegbaar	boolean
	Type		String
Beweging	Spelernaam		String
	Richting		String

## (In)consistent speelveld

Het bestand met het in te lezen speelveld wordt met de hand geschreven. Om het ingelezen speelveld te kunnen simuleren moet de informatie consistent zijn.

Een speelveld is consistent als:

- Een entiteit (speler/obstakel) binnen de grenzen van het veld ligt.
- Een speelveld geen negatieve dimensies (lengte, breedte) heeft.
- Een entiteit (speler/obstakel) geen negatieve coördinaten (x,y) heeft.
- Een speler een naam heeft.
- Er maar maximaal 1 entiteit (speler/obstakel) per coördinatenpaar op het speelveld staat.

Een beweging is consistent als:

- De richting “omhoog”, “links”, “omlaag” of “rechts” is.