Pacman

Mikael Larsson, Kjell Gunnarsson, Jessica Stenberg, Marcus Kristoffersson

Version 0.1

31/5/2018

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Datum** | **Kommentar** | **Utförare** |
| 0.1 | 24/5 2018 | Uppdaterat inledningen av SAD-dokumentet | Marcus Billgren |
| 0.2 | 28/5-2018 | Uppdaterat Arkitekturell översikt. | Jessica Stenberg |
|  |  |  |  |
|  |  |  |  |

Table of Contents

Contents

[1. Inledning 3](#_Toc516068999)

[1.1 Syfte 3](#_Toc516069000)

[1.2 Målgrupp 3](#_Toc516069001)

[1.3 Dokumentets omfattning 3](#_Toc516069002)

[1.4 Definitioner, Akronymer och förkortningar 3](#_Toc516069003)

[1.5 Referenser 3](#_Toc516069004)

[2. Arkitektuell översikt 4](#_Toc516069005)

[2.1 Use-Case View 5](#_Toc516069006)

[2.2 Use Case Realization 6](#_Toc516069007)

[2.3 Logical View 14](#_Toc516069008)

[2.4 Interactions 18](#_Toc516069009)

[3 Icke-funktionella krav 19](#_Toc516069010)

[3.1 Svarstider 19](#_Toc516069011)

[3.2 Tillgänglighet 19](#_Toc516069012)

[3.3 Test 19](#_Toc516069013)

[5. Driftaspekter 20](#_Toc516069014)

[5.1 Programvara 20](#_Toc516069015)

[5.2 Versionshantering 20](#_Toc516069016)

[5.3 Fysisk produktionsmiljö 20](#_Toc516069017)

[5.4 Programvaror 20](#_Toc516069018)

[5.5 Programspråk 20](#_Toc516069019)

# 1. Inledning

## 1.1 Syfte

Det huvudsakliga syftet med projektet är att enligt vår beställares önskemål, skapa en nyversion av spelet Pacman som stöds av dagens hårdvara. Vår beställare är övertygad om att det gjordes bättre spel förr, och vill att dagens unga spelare ska ges möjligheten att få uppleva ”spel från förr - fast i ny version”.

Vår beställare hoppas att en nyversion av spelet Pacman blir en succé som kommer få spelutvecklare att börja utveckla spel utifrån att de ska vara roliga att spela och inte bara vara grafikdemonstrationer utan färdig spelbarhet.

## 1.2 Målgrupp

De huvudsakliga målgrupperna för detta dokument är: systemägaren(beställaren), systemarkitekterna, testarna och utvecklingsteamet

## 1.3 Dokumentets omfattning

Detta dokument har för avsikt att förklara systemarkitekturen för nyversionen av Pacman samt att förklara de delar av systemdesignen som bedöms som viktiga för att utvecklingsteamet ska kunna skapa den produkt som beställaren har önskat, samt för beställaren att kunna kontrollera om de önskemål och krav som finns på produkten är inkluderade i det färdiga systemet.

## 1.4 Definitioner, Akronymer och förkortningar

**libGDX** – ramverk för att utveckla spel till diverse plattformar med java som programmeringsspråk.

**API** – Application Programming Interface

**IDE** – Integrated Development Environment

**JRE** – Java Runtime Enviroment

**Direct X –** en samling API för spelprogrammering som är utvecklat av Microsoft för deras plattformar.

## 1.5 Referenser

Följande dokument är inspirerat av riktlinjerna för SAD-dokument från följande [webbadress](http://sce.uhcl.edu/helm/rationalunifiedprocess/process/modguide/md_sad.htm#References).

# 2. Arkitektuell översikt

Detta dokument specificerar arkitekturen av spelet Pacman genom följande översikter:

Användarfallsöversikt (Use case View):

Målgrupp: Samtliga delaktiga aktörer i projektet, slutanvändaren inkluderat.

Avsett område: Avsnittet beskriver de användarfall som utgör en betydande del för funktionen av det färdiga spelet. Detta avsnitt beskriver även de förväntningar som slutanvändaren kan tänkas ha på det färdiga spelet (use case bild från bouml, utökade use case beskrivningar som vi gjorde till lab 3).

Teknisk lösningsöversikt (Logical view, Data view):

Målgrupp: Systemdesigners och utvecklare.

Avsett område: Funktionella krav på spelet: Visar hur man har för avsikt att realisera de krav som specificerats på spelet genom användarfallen och av uppdragsgivaren. I detta avsnitt förklaras även hur spelets komponenter interagerar med varandra samt vilka konventioner som gäller vid designen av systemet. (UML, DFD, Sekvensdiagram, Avsnitt om felhantering, paketnamngivning etc.)

Distributionsöversikt (Deploy):

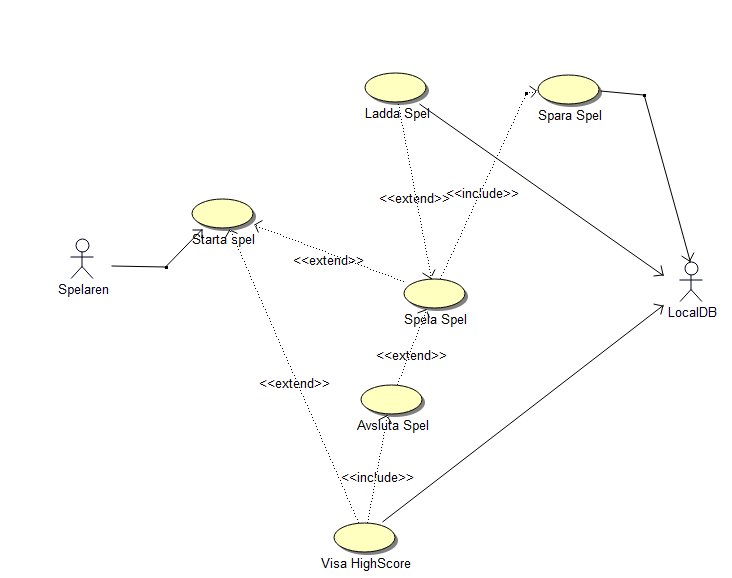
Målgrupp: Distributörer av den digitala produkten

Avsett område: Går igenom hur systemet är tänkt att distribueras och vilka krav som ställs på den aktuella miljön (driftaspekter, Deploy diagram).

## 2.1 Use-Case View

Bilden visar en överblick på de Use-Cases vi har i vår applikation där Spelaren är vår Actor och LocalDB är de externa aktörerna. Att kommunikationen mellan spelare och databas fungerar felfritt är absolut prio - annars kan inga spel skapas, laddas eller sparas. Om förbindelsen inte är aktiv är menyn det enda som kommer kunna visas och systemet kommer inte kunna utföra något av valen.

Man kan även se var och vilka Use-Cases som inkluderar varandra i de olika scenariers som finns, men även de som exkluderas (inte är beroende av ett tidigare case).



## 2.2 Use Case Realization

Här beskrivs några av de Use-Cases som finns tillsammans med tillhörande Story. Ett gemensamt krav som finns är bl.a. att det måste finnas tillräckligt med RAM för att spelet ska fungera korrekt.

|  |  |
| --- | --- |
| User story 1 | Som användare vill jag komma till en huvudmeny där jag kan göra val när spelet startar. |
| Acceptanskriterier | 1. Innan huvudmenyn startar ska företagets logotyp visas. 2. När huvudmenyn körs ska musik spelas i bakgrunden 3. Följande val måste finnas tillgängliga för användaren: Starta spel, Ladda spel, Titta på highscore och Avsluta spel |

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | 1 | | |
| Use Case Name: | Starta spelet | | |
| Created By: | Marcus | Last Updated By: | Kjell |
| Date Created: | 2018-05-11 | Date Last Updated: | 2018-05-12 |

|  |  |
| --- | --- |
| Actor: | Spelaren |
| Description: | Användaren startar applikationen och huvudmenyn visas med alternativ att starta spelet, ladda ett befintligt spel eller titta på highscore listan. |
| Preconditions: | * Tillräckligt med RAM ledigt för att spelet ska kunna fungera. |
| Postconditions: | När spelaren har gjort ett val kommer spelaren att slussas vidare till nästa del av programmet, eller avsluta detsamma. |
| Priority: | Hög |
| Frequency of Use: | Varje gång spelet ska startas samt när spelaren väljer att spara eller avsluta sitt spel från use caset ”spela spel”. |
| Normal Course of Events: | 1. Spelaren startar applikationen. 2. Titelskärmen visas med de alternativ som finns tillgängliga för spelaren. 3. Applikationen inväntar input från spelaren 4. Applikationen exekverar funktion utifrån användarens val. |
| Alternative Courses: | - |
| Exceptions: | 2.EX1. Spelaren försöker ladda ett spel men programmet upptäcker att sparfilen har blivit korrupt. Spelaren ska då få ett felmeddelande och återvända till huvudmenyn. |
| Includes: | Spela spel, Ladda spel, Avsluta spel, Titta på highscore |
| Special Requirements: |  |
| Assumptions: |  |
| Notes and Issues: |  |

|  |  |
| --- | --- |
| User story #2 | Som användare vill jag kunna ladda spel som jag tidigare har spelat. |
| Acceptanskriterier: | 1. Spelaren ska kunna komma åt menyn via tilldelad tangent för detta. 2. Där ska val som Spara spel, Pausa och Avsluta spel finnas. 3. Efter man valt ”ladda spel” dyker en textrad upp som meddelar när spelet är laddat så spelaren vet när processen är klar. 4. Spelaren får upp en ny vy där det laddade spelet syns och är pausat till dess att spelaren startar spelet. |

|  |  |
| --- | --- |
| Actor: | Spelaren |
| Description: | Användaren laddar ett sparat spel från en tidigare session från huvudmenyn. |
| Preconditions: | * Ett sparat spel från en tidigare session. * Tillräckligt med RAM ledigt för att spelet ska kunna fungera. |
| Postconditions: | När spelaren har gjort ett val kommer spelaren att slussas vidare till nästa del av programmet, eller avsluta detsamma. |
| Priority: | Medel |
| Frequency of Use: | Varje gång spelaren väljer att ladda ett spel. |
| Normal Course of Events: | 1. Spelaren startar applikationen. 2. Titelskärmen visas med de alternativ som finns tillgängliga för spelaren. 3. Spelaren väljer ”ladda spel” 4. Applikationen exekverar funktionen ”ladda spel”. |
| Alternative Courses: | - |
| Exceptions: | 2.EX1. Spelaren försöker ladda ett spel men programmet upptäcker att sparfilen har blivit korrupt. Spelaren ska då få ett felmeddelande och återvända till huvudmenyn. |
| Includes: | Ladda spel |
| Special Requirements: | * Ett redan existerande sparat spel. |
| Assumptions: |  |
| Notes and Issues: |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | 2 | | |
| Use Case Name: | Ladda spel | | |
| Created By: | Kjell | Last Updated By: |  |
| Date Created: | 2018-05-13 | Date Last Updated: |  |

|  |  |
| --- | --- |
| User story 3 | Som användare vill jag kunna se high score så att jag kan jämföra mig själv med tidigare spelomgångar. |
| Acceptanskriterier | 1. Från huvudmenyn ska valet ”visa highscore” finnas. 2. Så fort ett spel har spelats ska poängen från sessionen sparas i en databas för poäng. 3. När man väljer ”visa high score” så visas poängen från databasen. |

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | 3 | | |
| Use Case Name: | Visa highscore | | |
| Created By: | Mikael | Last Updated By: |  |
| Date Created: | 2018-05-13 | Date Last Updated: |  |

|  |  |
| --- | --- |
| Actor: | Spelaren |
| Description: | Användaren hämtar hem data från databasen |
| Preconditions: | - Det finns data att hämta.  - Uppkopplingen mellan användaren och databasen lyckades. |
| Postconditions: | Användaren får en visuell översikt av det senaste 10 rekorden. |
| Priority: | Låg |
| Frequency of Use: | - |
| Normal Course of Events: | 1. Spelaren startar programmet.  2. Väljer menyvalet “Show highscore”  3. Applikationen visar en lista med rekord. |
| Alternative Courses: |  |
| Exceptions: | 1. En uppkoppling mellan databasen och användaren kunde inte etableras.  2. Användaren försöker att hämta data som inte finns. |
| Includes: | Titta på highscore |
| Special Requirements: | - |
| Assumptions: | - |
| Notes and Issues: | - |

|  |  |
| --- | --- |
| User story 4 | Som spelare vill jag kunna spara spelet för att ändringar/framsteg inte ska gå förlorade. |
| Acceptanskriterier | 1. Spelaren ska kunna komma åt menyn via tilldelad tangent för detta. 2. Där ska val som Spara spel, Pausa och Avsluta spel finnas. 3. Det ska dyka upp en textrad som meddelar när spelet är sparat så spelaren vet när processen är klar. |

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | 4 | | |
| Use Case Name: | Spara spel | | |
| Created By: | Jessica | Last Updated By: |  |
| Date Created: | 2018-05-14 | Date Last Updated: |  |

|  |  |
| --- | --- |
| Actor: | Spelaren |
| Description: | Användaren sparar alla ändringar som har skett efter den senaste sparade sessionen. |
| Preconditions: | - Det finns ändringar/framsteg att spara.  - Tillräckligt med RAM ledigt för att spelet ska kunna fungera. |
| Postconditions: | När användaren har sparat spelet så kan denna fortsätta spela eller välja att avsluta. |
| Priority: | Medel |
| Frequency of Use: | Varje gång spelaren vill spara sin session eller om Autosave är aktiverat. |
| Normal Course of Events: | 1. Spelaren trycker på den tangent som är avsedd för Meny.  2. En vy visas med tillgängliga val.  3. Spelaren väljer ”Spara spel”  4. Applikationen exekverar funktionen ”Spara spel”. |
| Alternative Courses: |  |
| Exceptions: | 1. Spelet kan inte skriva ändringarna till den befintliga filen pga. att den har kraschat/blivit korrupt – i så fall får spelaren ett felmeddelande. |
| Includes: | Spara spel |
| Special Requirements: | - |
| Assumptions: | - |
| Notes and Issues: | - |

|  |  |
| --- | --- |
| User story 5 | Som spelare vill jag kunna pausa spelet för att ta en paus utan att behöva stänga av spelet. |
| Acceptanskriterier | 1. Spelaren ska kunna komma åt menyn via tilldelad tangent för detta. 2. Där ska val som Spara spel, Pausa och Avsluta spel finnas. |

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | 5 | | |
| Use Case Name: | Pausa spel | | |
| Created By: | Jessica | Last Updated By: |  |
| Date Created: | 2018-05-14 | Date Last Updated: |  |

|  |  |
| --- | --- |
| Actor: | Spelaren |
| Description: | Användaren pausar spelet för att sedan kunna fortsätta vid samma läge. |
| Preconditions: | - Ett spel är startat och igång.  - Tillräckligt med RAM ledigt för att spelet ska kunna fungera. |
| Postconditions: | När användaren har pausat spelet kan denna välja att fortsätta spela igen. |
| Priority: | Medel - Hög |
| Frequency of Use: | Varje gång spelaren vill sluta spela utan att stänga av spelet men ändå inte ha sessionen aktivt igång. |
| Normal Course of Events: | 1. Spelaren trycker på den tangent som är avsedd för Meny.  2. En vy visas med tillgängliga val.  3. Spelaren väljer ”Pausa”  4. Applikationen exekverar funktionen ”Pausa”. |
| Alternative Courses: |  |
| Exceptions: | 1. Spelet stannar inte den aktuella sessionen utan fortsätter vara aktiv i bakgrunden så spelaren kanske dör eller missar något innehåll/event. |
| Includes: | Pausa spel |
| Special Requirements: | - |
| Assumptions: | - |
| Notes and Issues: | - |

|  |  |
| --- | --- |
| User story 6 | Som användare vill jag kunna spela pacman, få poäng och avancera till nästa nivå. |
| Acceptanskriterier | 1. Spelaren ska kunna styra pacman med piltangenterna. 2. Poäng ska ges till spelaren när denne går över ett mynt. 3. Om spökena och pacman hamnar på samma position ska ett liv dras, om alla liv är slut är spelet slut. |

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | 6 | | |
| Use Case Name: | Spela spelet | | |
| Created By: | Marcus | Last Updated By: |  |
| Date Created: | 2018-05-15 | Date Last Updated: |  |

|  |  |
| --- | --- |
| Actor: | Spelaren |
| Description: | Har för avsikt att beskriva vad som händer under tiden som spelet körs. |
| Preconditions: | Spelaren måste ha valt ”spela spel” från huvudmenyn, alternativt valt att ”ladda spel” från samma meny |
| Postconditions: | Spelet kommer köras fram till att spelaren väljer att avsluta, spara eller pausa. |
| Priority: | Hög |
| Frequency of Use: | Varje gång spelet användaren startar, eller laddar ett spel. Kan temporärt pausas av användaren under pågående spelsession. |
| Normal Course of Events: | 1. Spelplanen ritas upp. 2. Spelaren och spökena instansieras till sina respektive startpositioner. 3. Spelomgången startar och spelet lyssnar till input från spelaren 4. Spelet ser till att förflytta spökena och håller koll på att spelaren är vid liv samt hur mycket poäng som har samlats. |
| Alternative Courses: | 6.AC1. Spelaren blir tagen av spökena.  6.AC1.1 Har spelaren fler liv startar omgången om, annars skickas spelaren vidare till huvudmenyn.  6.AC2. Spelaren pausar spelet. En meny med alternativen spara, ladda, återgå och avsluta visas. Beroende på vilket val spelaren gör, hänvisas läsaren till respektive use case specifikation för mer information.  6.AC3 Användaren försöker flytta pacman till en plats på spelskärmen där det finns en vägg.  6.AC3.1 Pacman ska inte förflytta sig och spelet fortsätter att lyssna efter korrekt användarinput.  6.AC4 Spelaren flyttar pacman över till en position där det finns ett gult mynt.  6.AC4.1 Pacman flyttar till den önskade positionen  6.AC4.2 Myntet försvinner från spelplanen och användarens poäng höjs med 20 poäng. |
| Exceptions: | 6.EX1. Input från användaren kastar ett felmeddelande  6.EX1.1 Fånga felmeddelande och fortsätt lyssna efter korrekt input från användaren. |
| Includes: | Spela spel, Ladda spel, Avsluta spel. |
| Special Requirements: |  |
| Assumptions: |  |
| Notes and Issues: |  |

|  |  |
| --- | --- |
| User story 7 | Som användare vill jag kunna avsluta spelet ifall jag har behov av det. |
| Acceptanskriterier | 1. Spelaren ska kunna komma åt menyn via tilldelad tangent för detta. 2. Där ska val som Spara spel, Pausa och Avsluta spel finnas. 3. Det ska dyka upp en textrad som meddelar när spelet är avslutat. 4. Spelet avslutas. |

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | 7 | | |
| Use Case Name: | Avsluta spel | | |
| Created By: | Kjell | Last Updated By: |  |
| Date Created: | 2018-05-17 | Date Last Updated: |  |

|  |  |
| --- | --- |
| Actor: | Spelaren |
| Description: | Användaren avslutar spelet. |
| Preconditions: | - Ett spel är startat och igång.  - Tillräckligt med RAM ledigt för att spelet ska kunna fungera.  - En knapp för att avsluta spelet. |
| Postconditions: | När användaren har pausat spelet kan denna välja att fortsätta spela igen. |
| Priority: | Medel - Hög |
| Frequency of Use: | Varje gång spelaren vill avsluta spelet. |
| Normal Course of Events: | 1. Spelaren trycker på den tangent som är avsedd för Meny.  2. En vy visas med tillgängliga val.  3. Spelaren väljer ”avsluta spel”  4. Applikationen exekverar funktionen ”avsluta spel”. |
| Alternative Courses: |  |
| Exceptions: | 1. Spelet stannar inte den aktuella sessionen utan fortsätter vara aktiv i bakgrunden så spelaren kanske dör eller missar något innehåll/event. |
| Includes: | Avsluta spel |
| Special Requirements: | - |
| Assumptions: | - |
| Notes and Issues: | - |
|  |  |

## 2.3 Logical View

Standarder för projektet.

Paketnamn ska inledas med com.ffs.\*

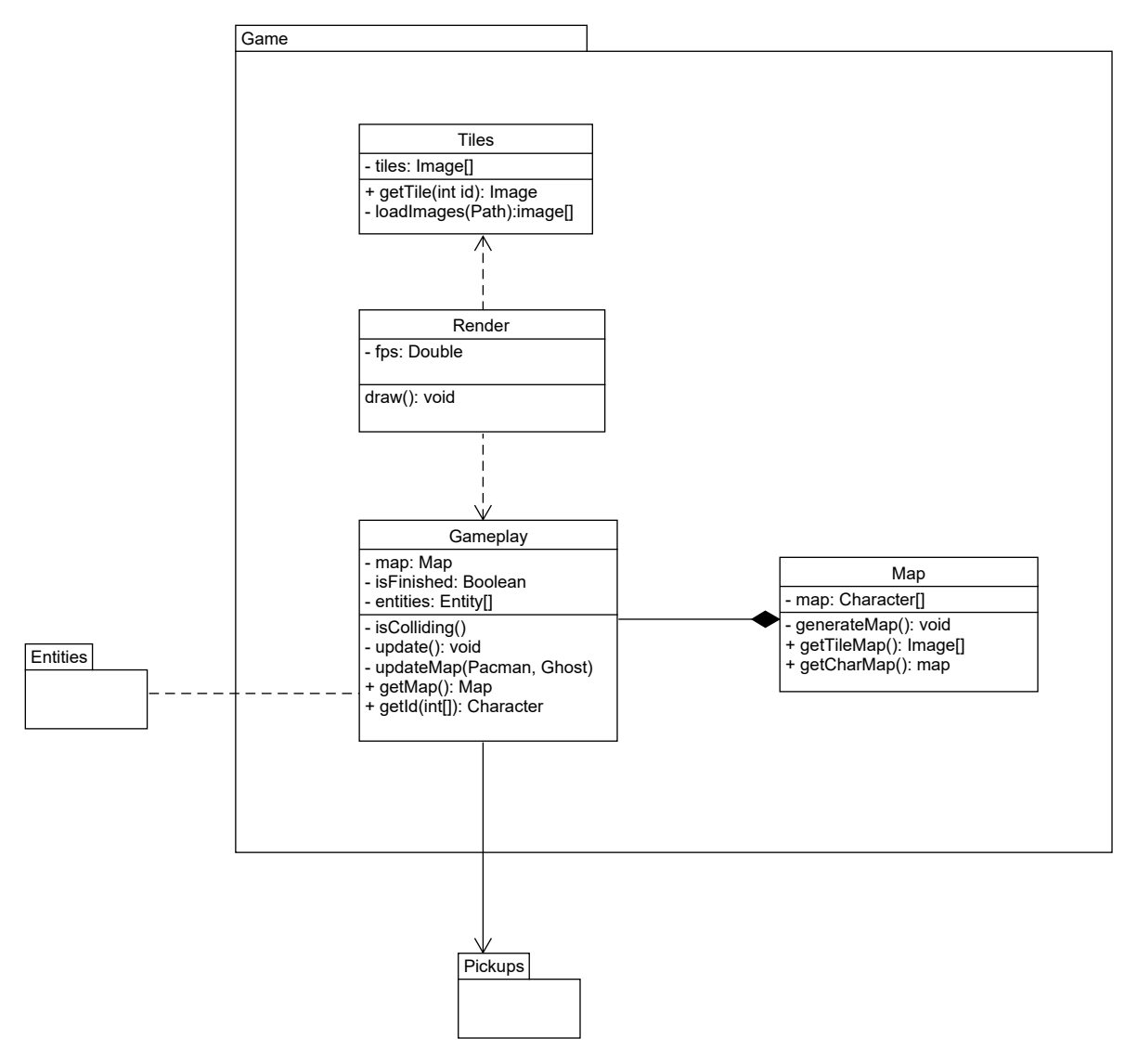
Kodformatering i Eclipse ska ske enligt följande [mall](http://git.eclipse.org/c/jetty/org.eclipse.jetty.admin.git/tree/jetty-eclipse-java-format.xml).

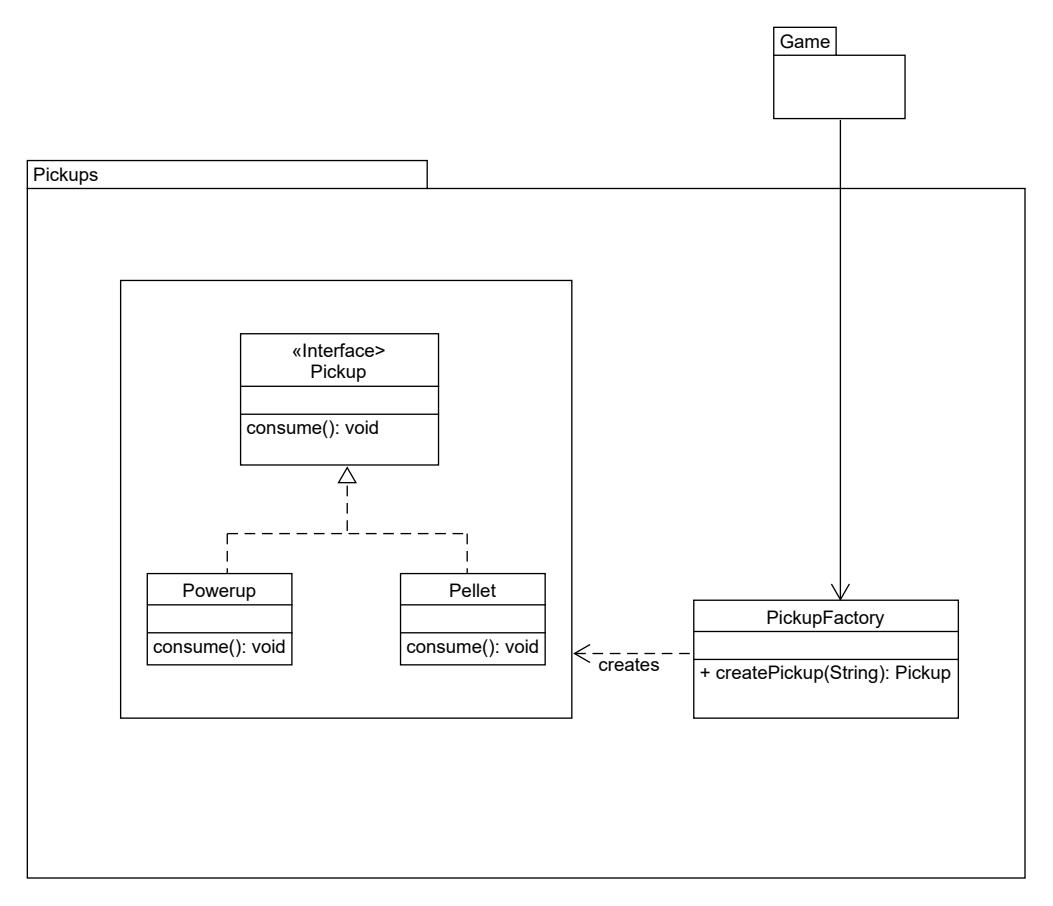
För att genomföra de grundläggande enhetstester som krävs skapas ett separat Java projekt som refererar till JUnit5 och de klasser från spelet som ska testas.   
Testklasserna ska ha samma namn som de klasserna det avser att testa, med tillägget ”Test” på slutet. Testklasserna ska även paketeras på samma vis som de är paketerade i det skarpa projektet, med tillägget unittest i slutet.

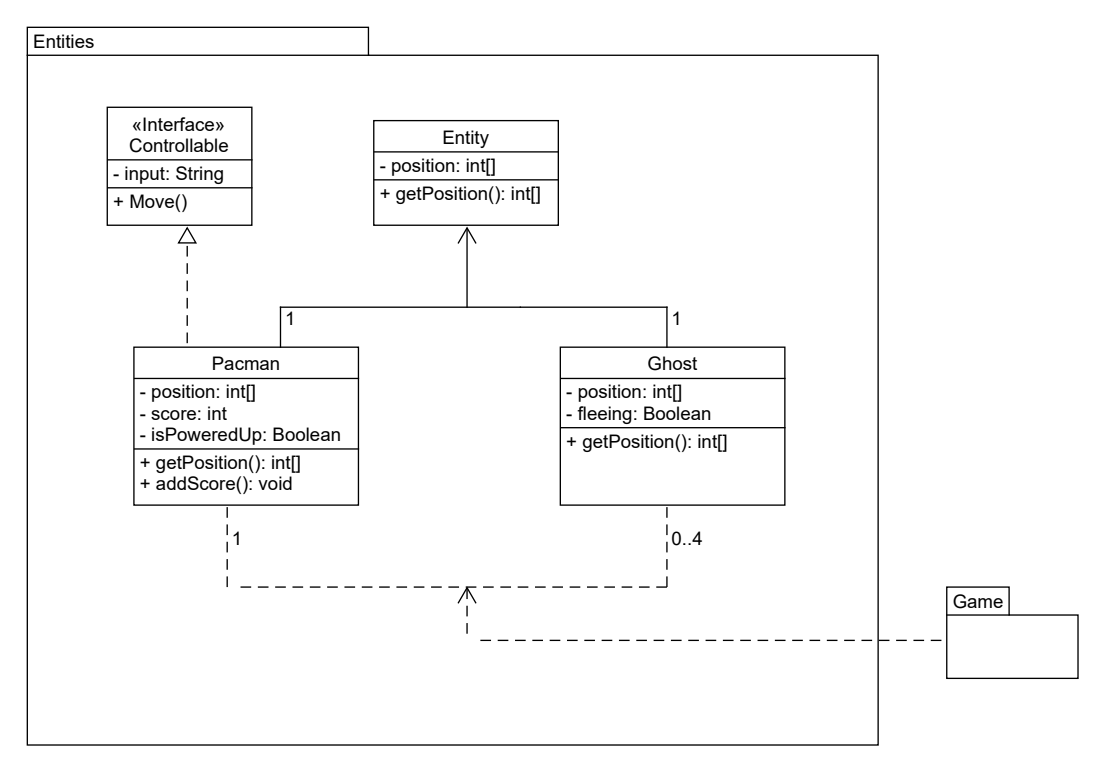
Om egna klasser för felhantering behöver skapas ska dessa läggas i paketet util.exceptions.x.y där ”x” står för den klass och som använder sig av det aktuella falmeddelandet och ”y” vilken metod som avses.

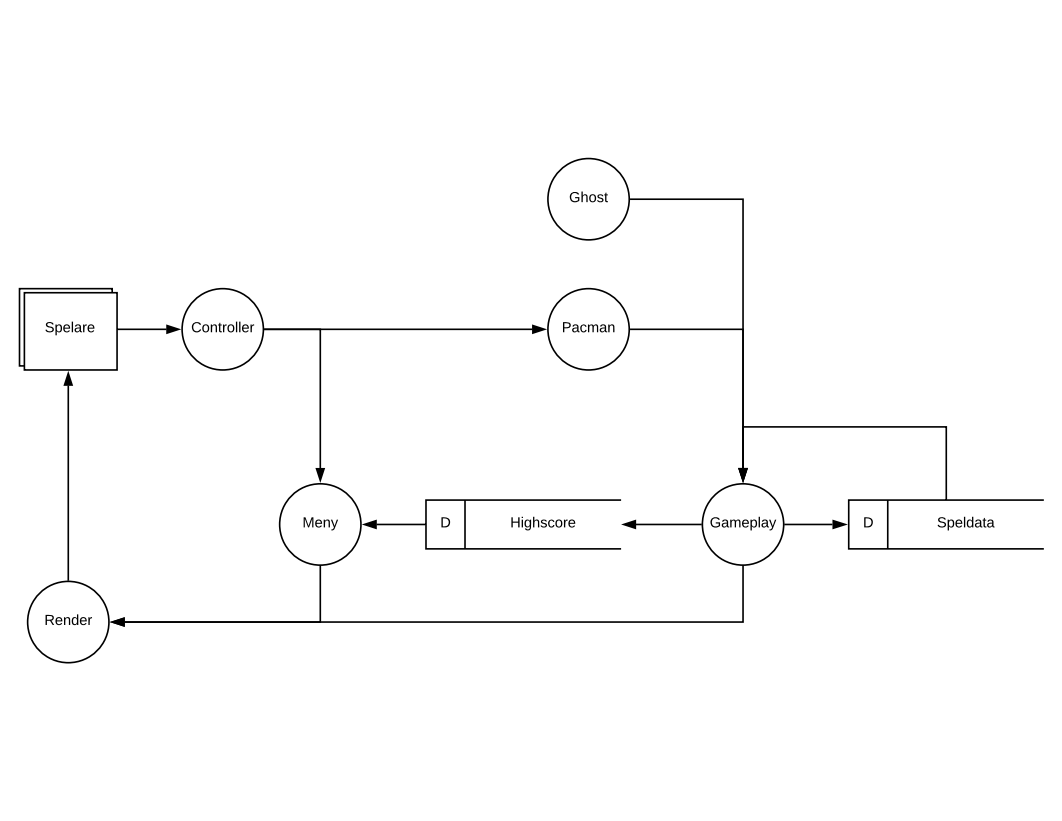
Programöversikt.

I bilden nedanför illustreras några av de funktioner och klasser som bygger systemet, även relationer dessa emellan. Några klasser är beroende av andra för att de ska kunna existera, medans andra instansieras oavsett.







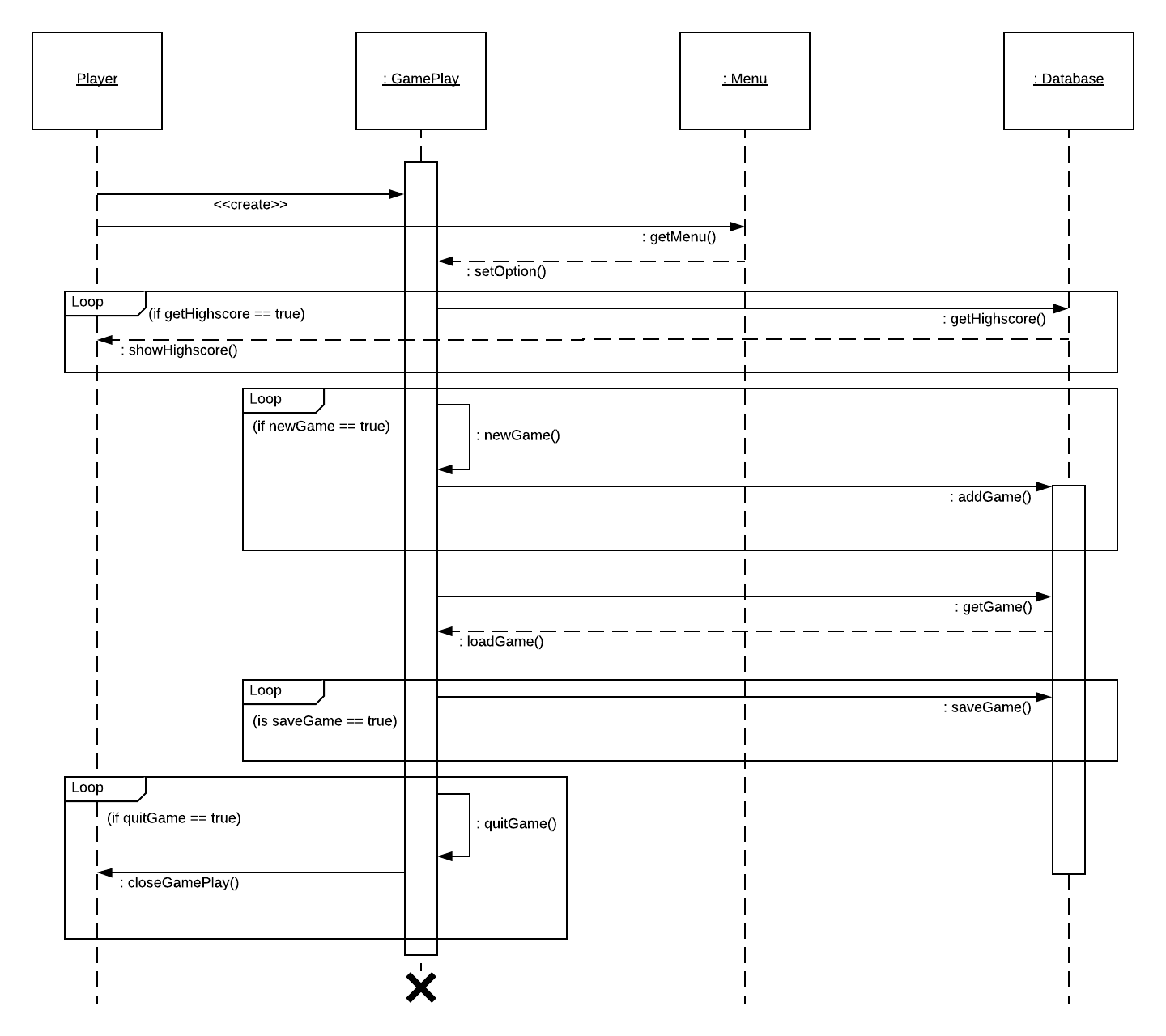


**Översikt av spelets huvudsakliga processer**

|  |  |
| --- | --- |
| **Process** | **Förklaring** |
| Controller | Den process som ansvarar för att ta emot input från användaren. |
| Pacman | Uppdaterar spelarens position på spelplanen samt kontrollerar om pacman kan äta spöken eller inte. |
| Ghost | Uppdaterar spökets position på spelplanen. |
| Menu | Den process som låter spelaren göra val att spara, ladda eller avsluta när spelet är pausat eller när användaren är i huvudmenyn. |
| Gameplay | Den process som ansvarar för att räkna poäng, upptäcka kollisioner och ha koll på om spelaren är i livet eller inte. Även den process som kommunicerar med databasen för sparade spel och hämtar dem om användaren begär det. |
| Render | Renderar spelets grafiska komponenter. |

## 2.4 Interactions

Följande diagram beskriver programmet från det att spelaren har skapat ett konto eller loggat in med befintligt konto. Spelaren kan sedan göra sitt val i menyn och diagrammet visar de funktioner som körs baserat på valet.



* Efter att GamePlay har skapats så kallas funktionen getMenu() som presenterar de olika valen som finns.
* När spelaren har gjort sitt val så skickas detta in i GamePlay där valet valideras och tilldelas med setOption().
* Sedan följer olika loopar/funktioner som körs beroende av spelarens val:  
  - getHighscore() hämtar data från databasen som sedan visas för spelaren.  
  - newGame() skapar ett nytt spel som sedan läggs till i databasen.  
  - loadGame() körs när ett sparat spel har hittats och hämtats från databasen.  
  - saveGame() sparar den aktuella sessionen, skriver över tidigare version i databasen.  
  - quitGame() avslutar GamePlay och förstör även det objektet.

# 3 Icke-funktionella krav

Känslan av ett spel kommer i stor del ifrån hur snabbt spelet kan reagera på användarens knapptryck. Ett dåligt spel brukar oftast beskrivas som ”laggigt”. Dessa krav ställs för att upprätthålla bl.a. Att spelaren känner sig i kontroll av ”Pacman” och att tillgängligheten stöds på så många plattformar som möjligt

## 3.1 Svarstider

Kartan ska kunna uppdateras och ritas på skärmen inom 20ms.

## 3.2 Tillgänglighet

## 3.3 Test

Under utvecklingens gång ska spelet omfatta sig utav alpha- och betatestare för att upptäcka buggar eller testa nya funktioner.

# Driftaspekter

Databashanteringen kommer att ske på användarens dator med hjälp av enkla SQLlite-databaser och därmed behövs inte en separat, fysisk server att användas. Dock så kommer det finnas krav på användarens egen hårdvara.

## 5.1 Programvara

Följande programvara måste en användare ha installerad för att kunna köra spelet:

* Java Runtime 7.0 eller en senare version.
* DirectX 10.0 eller senare.

## 5.2 Versionshantering

Projektet kommer versionshanteras via Git som är en DVCS. I varje utvecklingsmiljö finns möjligheten till att versionshantera internt. Det finns också möjlighet för versionshantering via terminalen på varje enskild enhet.

## 5.3 Fysisk produktionsmiljö

För att kunna köra programvarorna krävs det en dator (minimum windows 8/macOS 10.12 ”Sierra) med följande hårdvara:

* Dualcore 2.2 GHz processor.
* Ljudkort (stereo).
* Grafikkort (internt/externt) som kan köra directdraw.
* Monitor

## 5.4 Programvaror

I vår produktionsmiljö krävs följande program:

* Libgdx (open source under Apache 2.0 licens) version 1.9.7
* Java Development Kit SE 10
* Direct X 10.0
* Android Studio (valfritt)
* Eclipse/IntelliJ som IDE för javaprogrammering.

## 5.5 Programspråk

Följande programmeringsspråk kommer ingå i processen för att skapa spelet. Förklaringar följer varje språk.

* Java 9: Används i libgdx och är det huvudsakliga språker som kommer att användas för applikationen.
* XML: Används i Android studio för designrelaterade funktioner och är en del av spelutvecklingsmiljön.
* Kotlin: Används parallellt med java i spelutvecklingsmiljön, dock i en mindre skala och för specifika frågor.
* SQL: Används för databashantering där t.ex. användarens poäng och sparade spel placeras. MySQL används för översikt över all data.