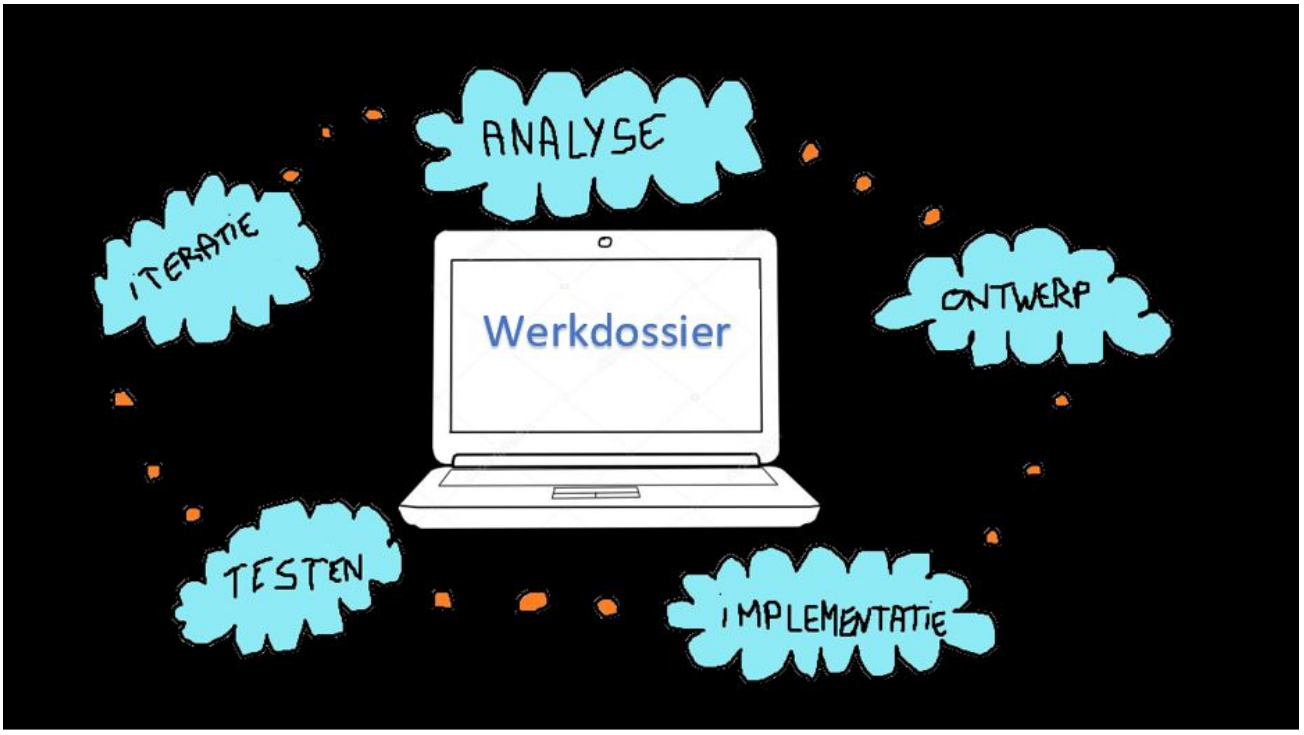
Project Toegepaste Informatica I



Werkdossier Mastermind

gerealiseerd door Lars Vandenberghe, Dries Schoolaert
Stef Bondroit & Nathan Cammerman
Toegepaste informatica I
schooljaar 2017-2018

HoGent

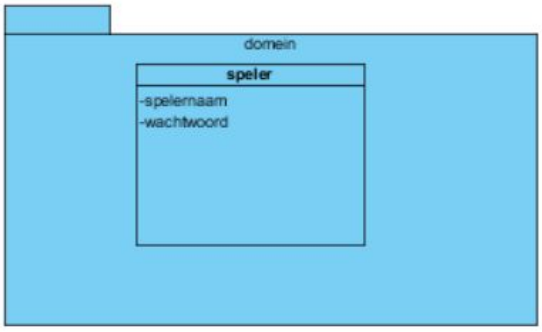
Inhoud

1. UC1 4
   1.1. Domeinmodel 4
   1.2. Activity Diagram 4
   1.3. SSD 5
   1.4. SSD Alternatief verloop + OC 5
   1.5. DCD 6
   1.6 SD 7
2. UC2 8
   2.1. Domeinmodel 8
   2.2. Activity Diagram 9
   2.3. SSD 10
   2.4. OC 10
   2.5. DCD 11
   2.6 SD 12
3. UC3 13
   3.1. Domeinmodel 13
   3.2. Activity Diagram 14
   3.3. SSD 15
   3.4. OC 15
   3.5. DCD 16
   3.6 SD 17
4. UC4 19
   4.1. Domeinmodel 19
   4.2. Activity Diagram 19
   4.3. SSD 20
   4.4. OC 20
   4.5. DCD 21
   4.6 SD 21

5. UC 22
5.1. Domeinmodel 22
5.2. Activity Diagram 23
5.3. SSD 24
5.4. OC 24
5.5. DCD 25
5.6 SD 27
6. UC6 28
6.1. Domeinmodel 28
6.2. Activity Diagram 29
6.3. SSD 30
6.4. OC 30
6.5. DCD 31
6.6 SD 33
7. UC7 34
7.1. Domeinmodel 34
7.2. Activity Diagram 34
7.3. SSD 35
7.4. DCD 36
7.5. SD 37

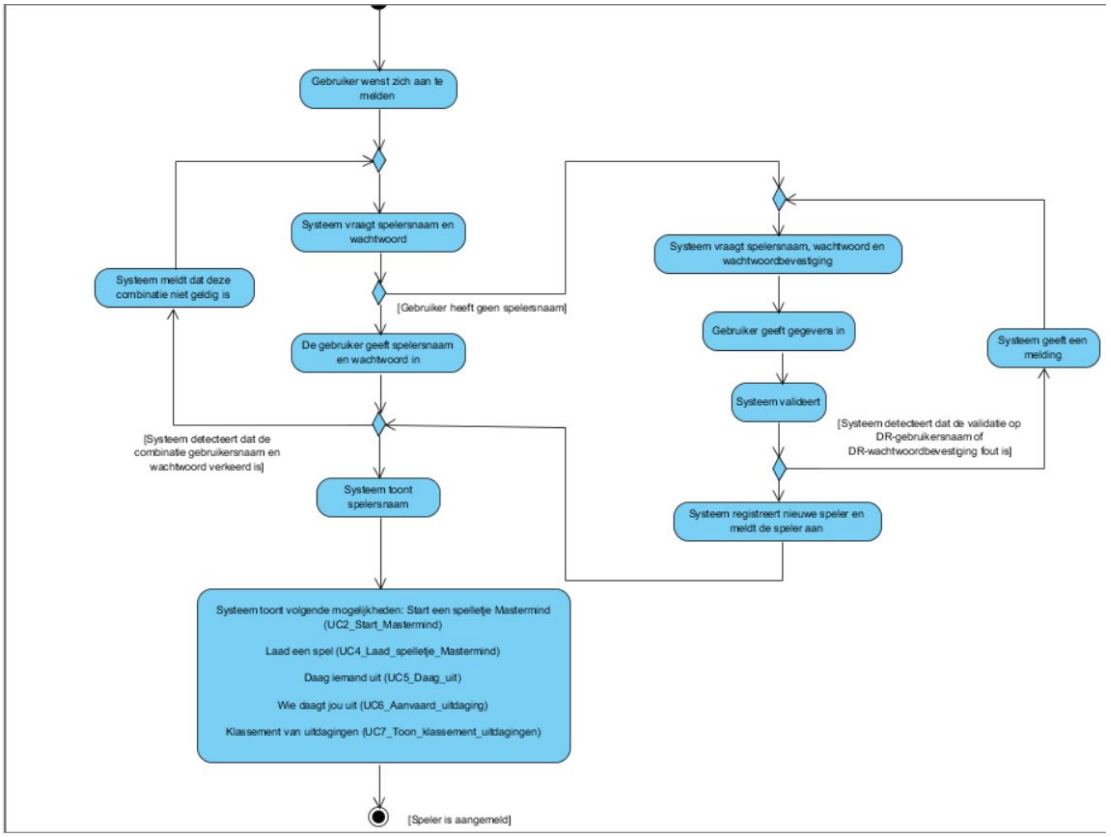
1 UC1

1.1 Domeinmodel



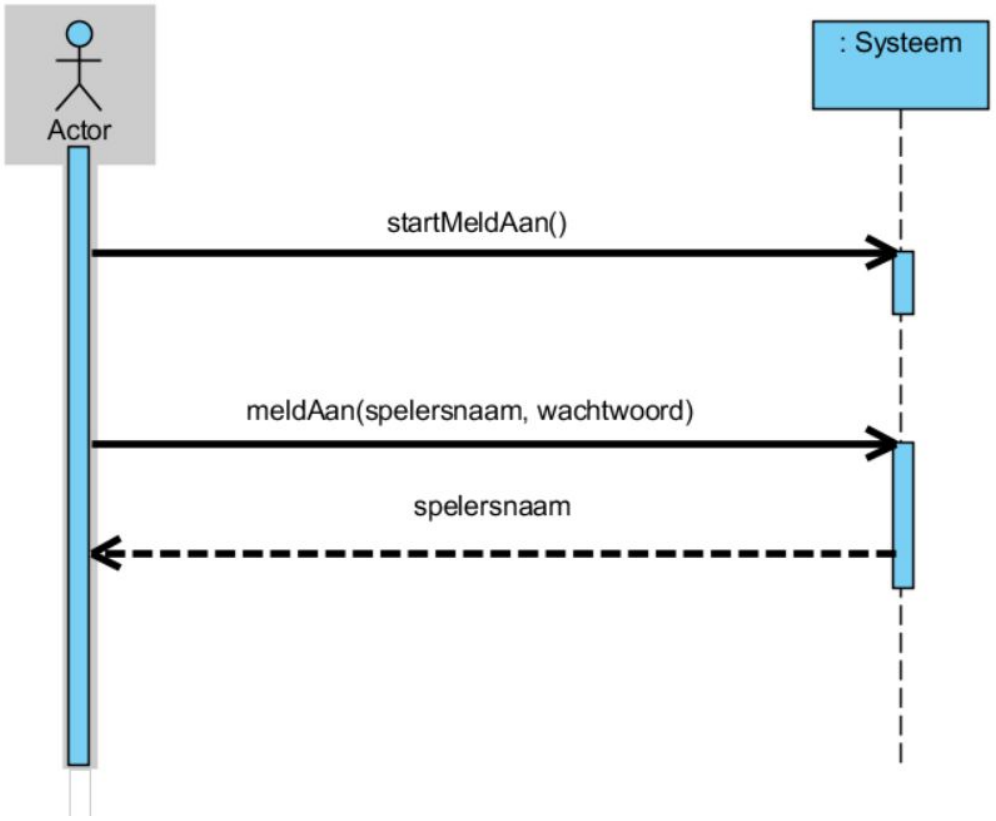
We hebben er voor gekozen om gebruiker, zoals aangegeven in de UC1 op chamilo, speler te noemen. Dit schept een beter beeld en is duidelijker in het lange termijn.

1.2 Activity diagram



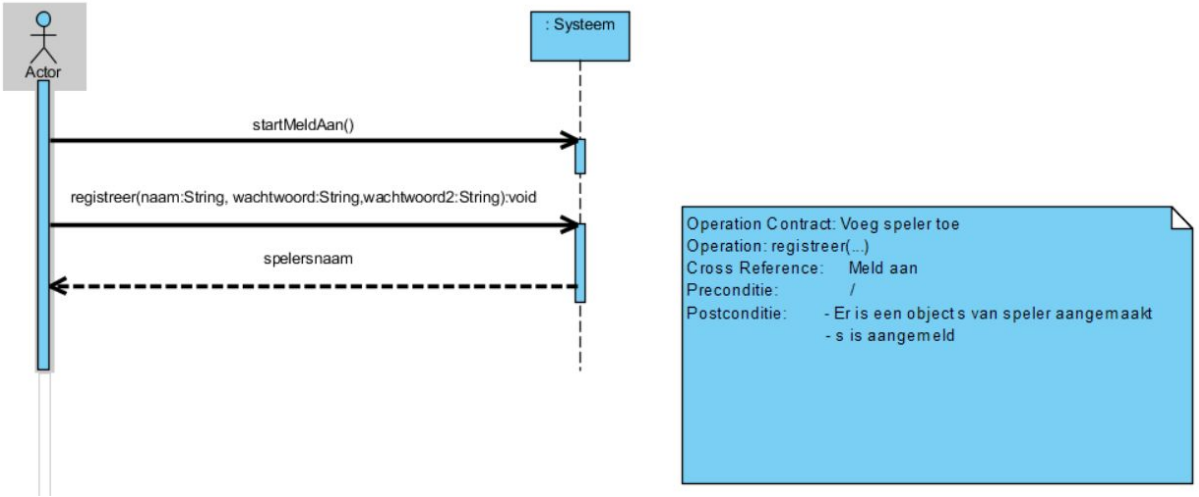
Geen extra commentaar. Dit is stapsgewijs opgebouwd.

1.3 SSD



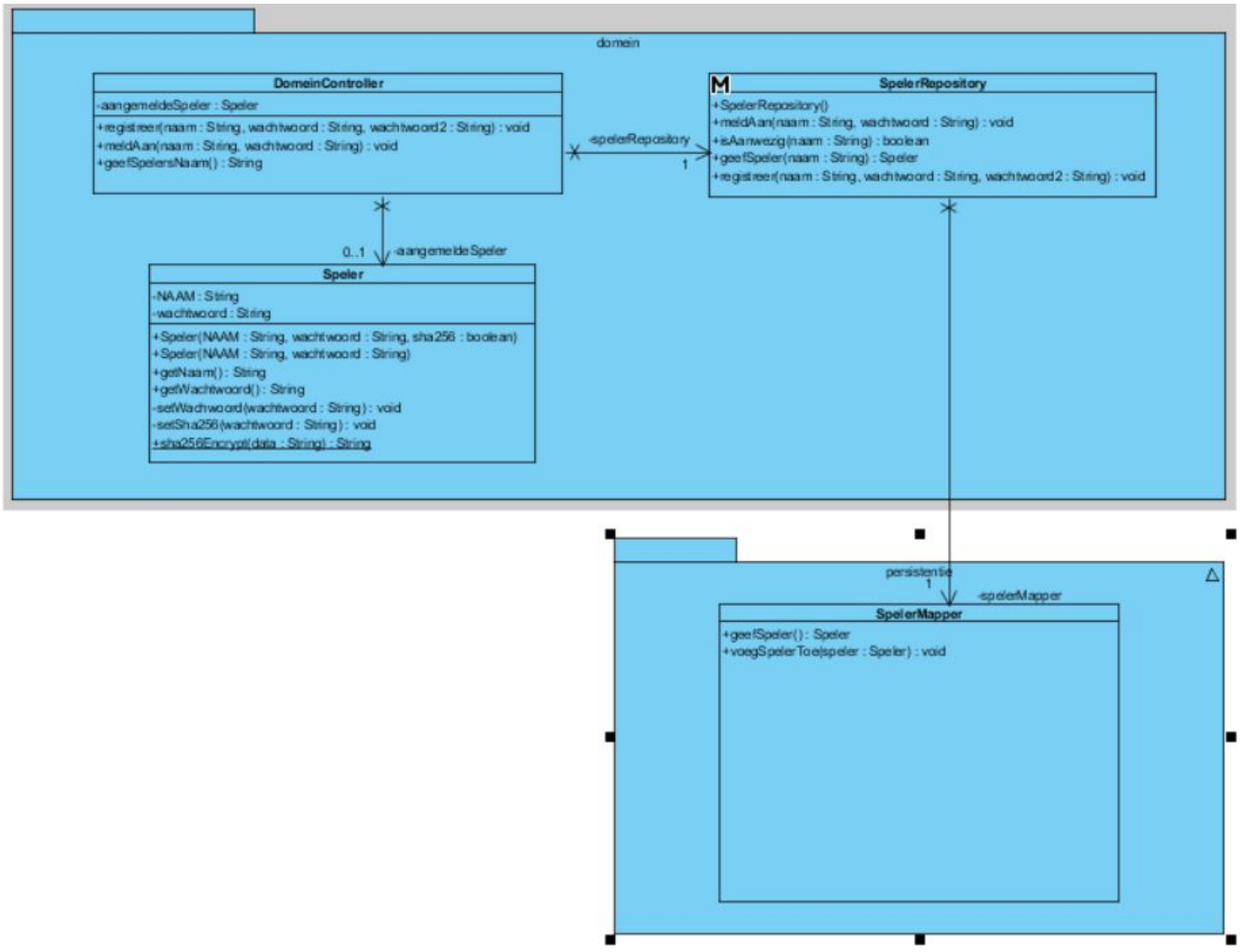
Geen extra commentaar. Dit is stapsgewijs opgebouwd.

1.4 SSD Alternatief verloop + OC



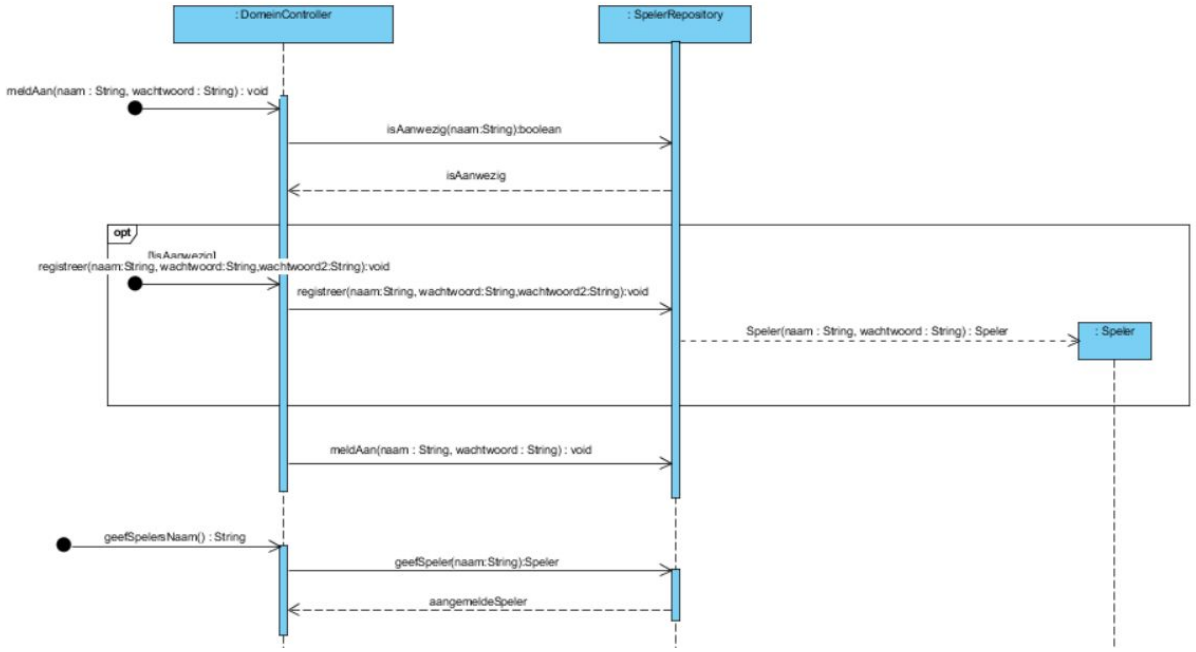
Er is een SSD gemaakt van het alternatieve verloop wegens dat het een grote impact heeft op onze applicatie. Namelijk dat we een speler moeten registreren. Hier moest ook een OC gemaakt worden die ernaast is beschreven.

1.5 DCD



We hebben direct ook gedacht aan de databank vandaar de SpelerMapper in de persistentie map.

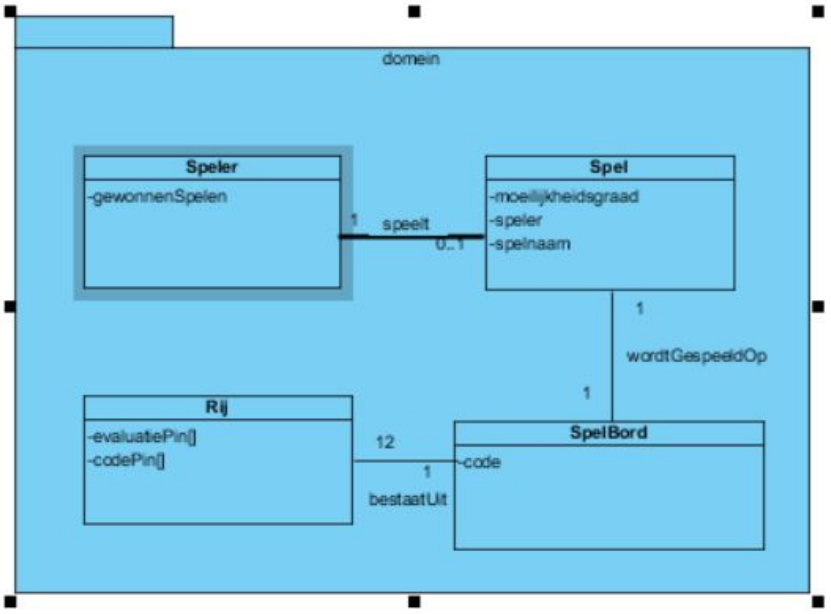
1.6 SD



Voor het alternatieve verloop van het SSD is er besloten om deze te vertalen in een optioneel vak in het SD.

2 UC2

2. 1 Domeinmodel

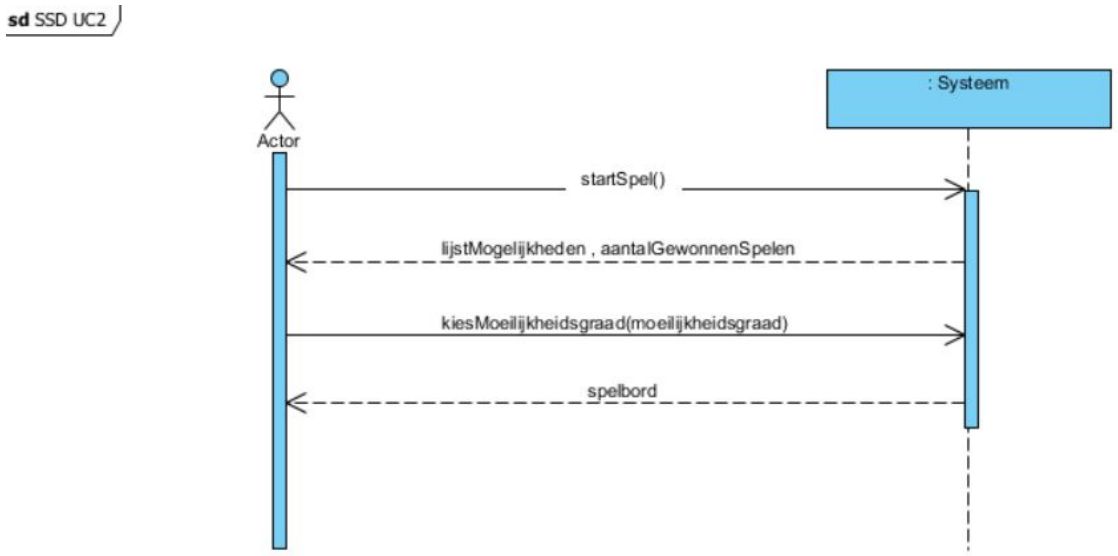


Dit domeinmodel is van UC1 en UC2 samengevoegd. Dit werd gedaan nadat we de opmerking gekregen om deze samen te voegen om zo de applicatie te zien “groeien” en er een totaaloverzicht te krijgen in de latere fases.

2.2 Activity diagram

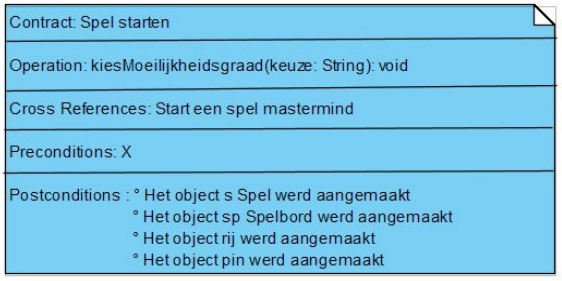
Er zijn geen alternatieve verlopen.

2.3 SSD



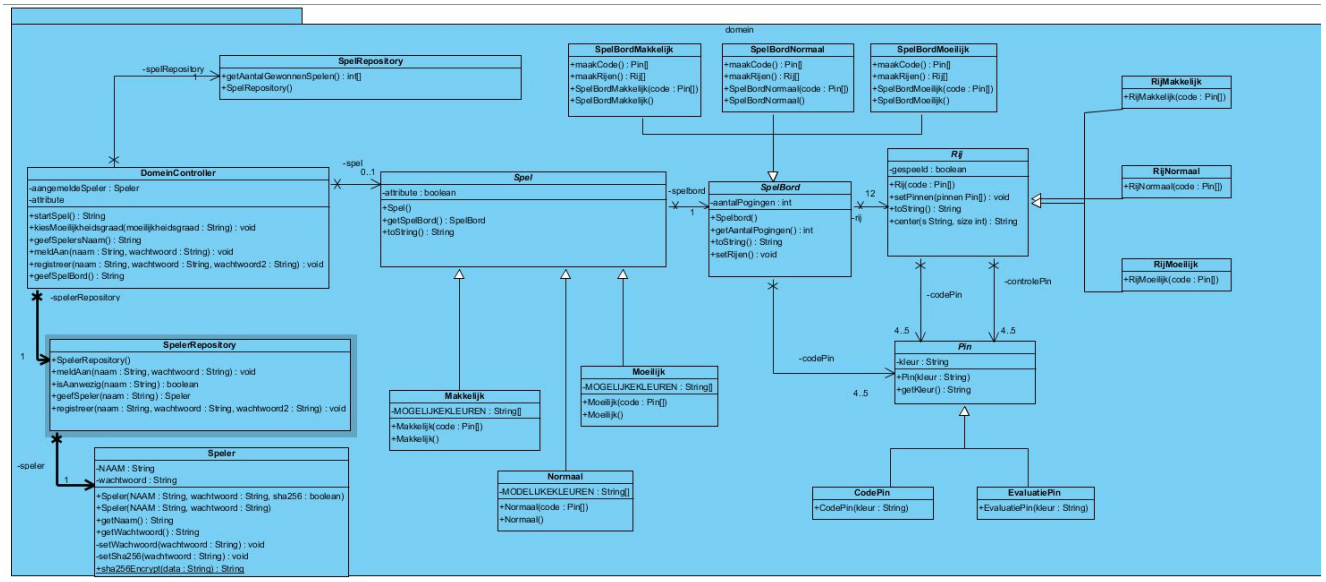
Geen extra commentaar. Dit is stapsgewijs opgebouwd.

2.4 OC



Een OC die gemaakt wordt voor de methode startSpel().

2.5 DCD

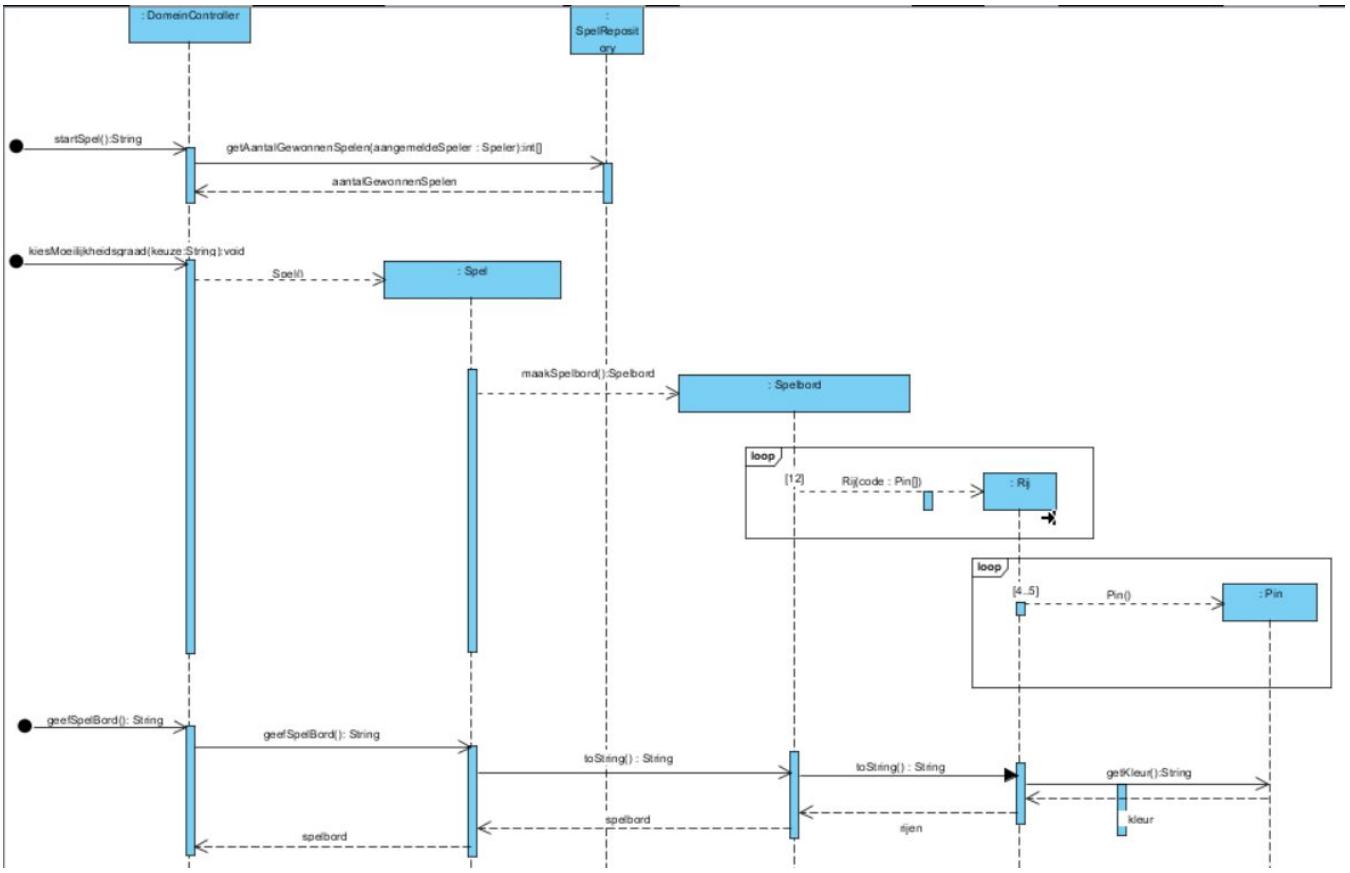


De klasse Spel, spelBord, Rij en Pin zijn abstract gemaakt. Deze hebben elk hun subklassen.

We hebben gekozen voor geen klasse moeilijkheidsgraad te implementeren wegens dat de moeilijkheidsgraden een instantie van een spel is. (vandaar dat Spel abstract is)

Bij de eerste presentatie was er een associatie tussen Speler en Spel. Dit was fout. We hebben nu een associatie tussen speler en spelerRepository.

2.6 SD



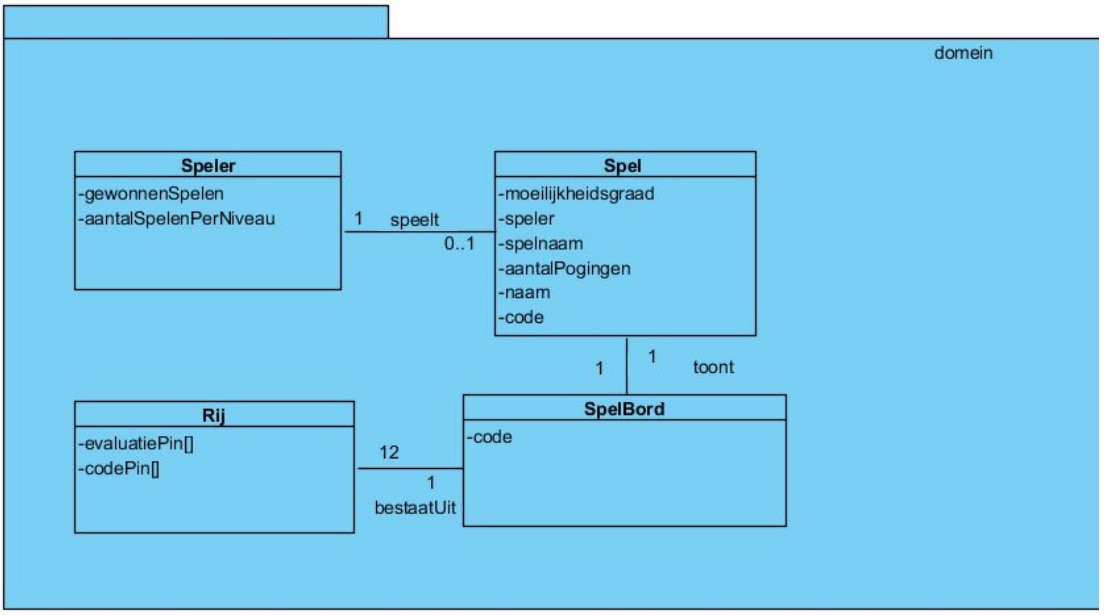
De methode getAantalGewonnenSpelen(aangemeldeSpeler : Speler):int[] kan opgevraagd worden in SpelRepository, het moet dus niet berekend worden in een andere klasse. We geven het attribuut aangemeldeSpeler als parameter mee, om het de aantal gewonnen spelen van die speler terug te vragen.

De methode kiesMoeilijkheidsgraad(keuze : String) : void , creëert een instantie van een spel (makkelijk, normaal of moeilijk) afhankelijk van de keuze. Spel creëert dan een instantie van spelBord (spelBordMakkelijk, spelBordNormaal of spelBordMoeilijk). Welk soort spelbord dat wordt gemaakt hangt af van de instantie van Spel. SpelBord maakt daarna 12 rijen aan. Uiteindelijk worden er 4 à 5 pinnen gecreëerd (afhankelijk van welke “moeilijkheidsgraad”).

De laatste methode in het SD is “geefSpelbord():String” We willen het volledige spelbord opvragen om te tonen aan de spelende speler. We gebruiken de methode toString():String om het spelbord weer te geven als een string.

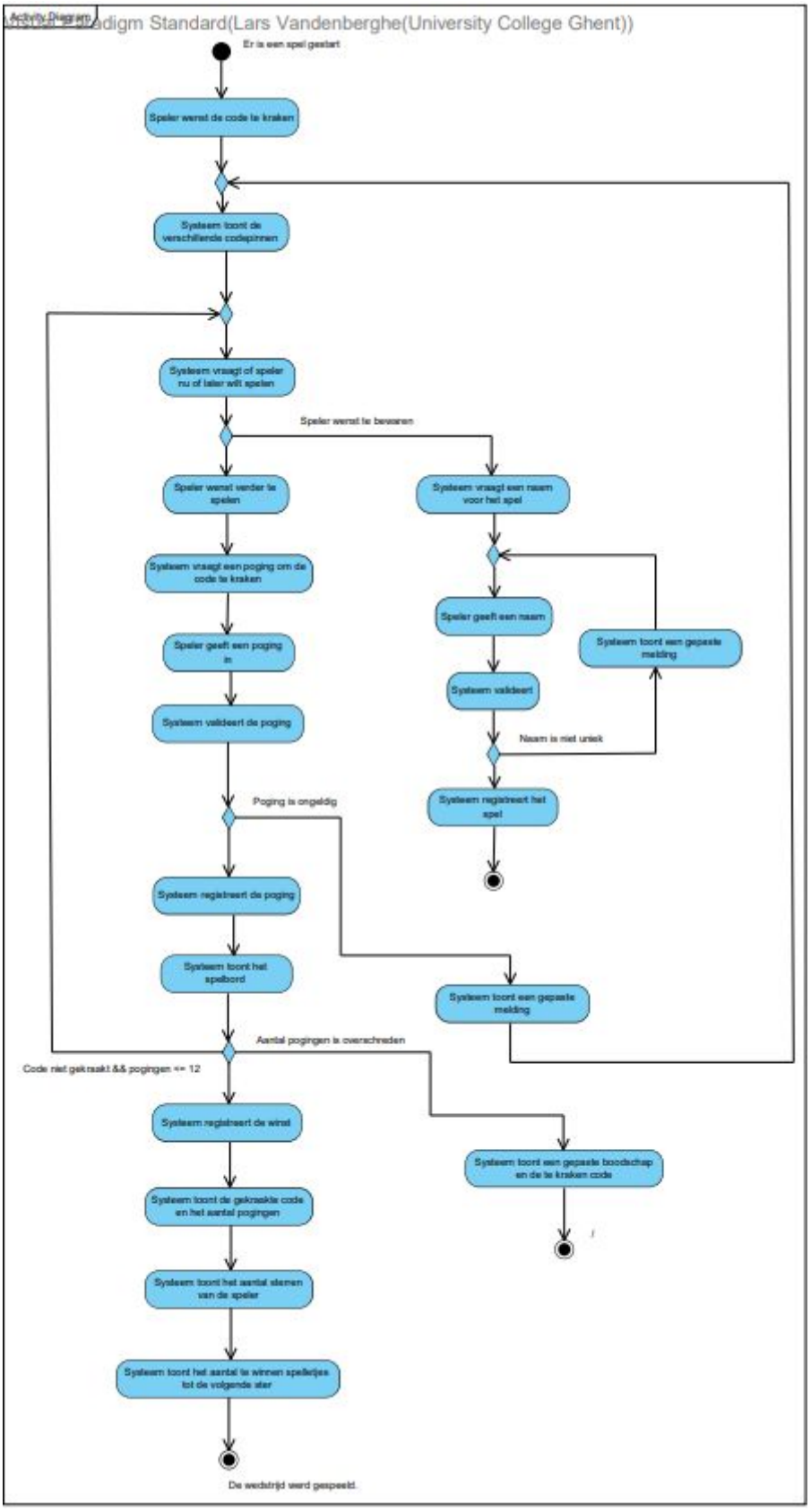
3 UC3

3.1 domein model



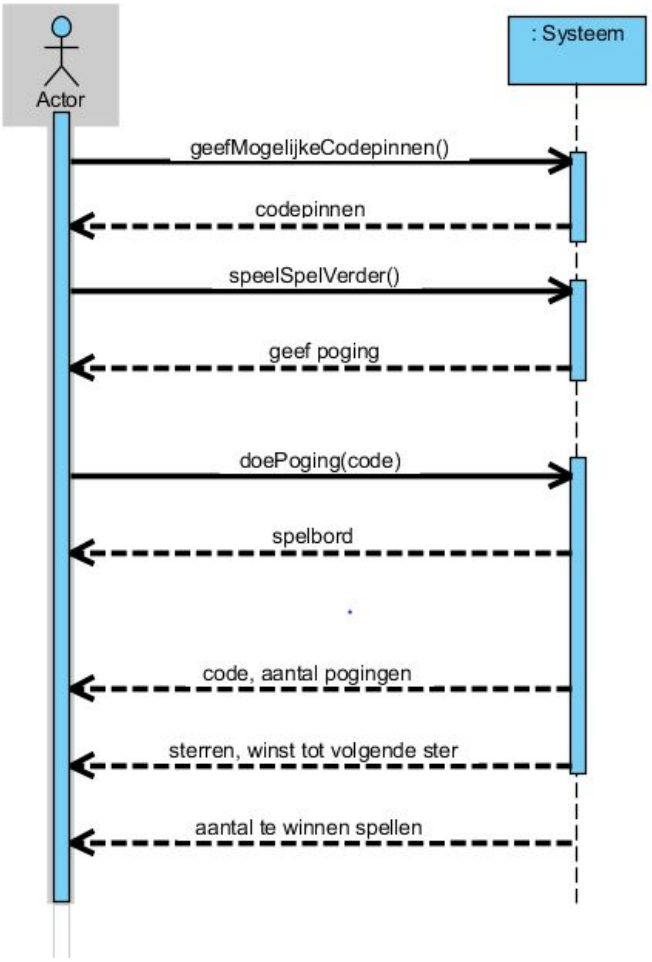
Domeinmodel van UC3 samen met UC2 en UC1.

3.2 Activity Diagram



Geen extra commentaar. Dit is stapsgewijs opgebouwd.

3.3 SSD



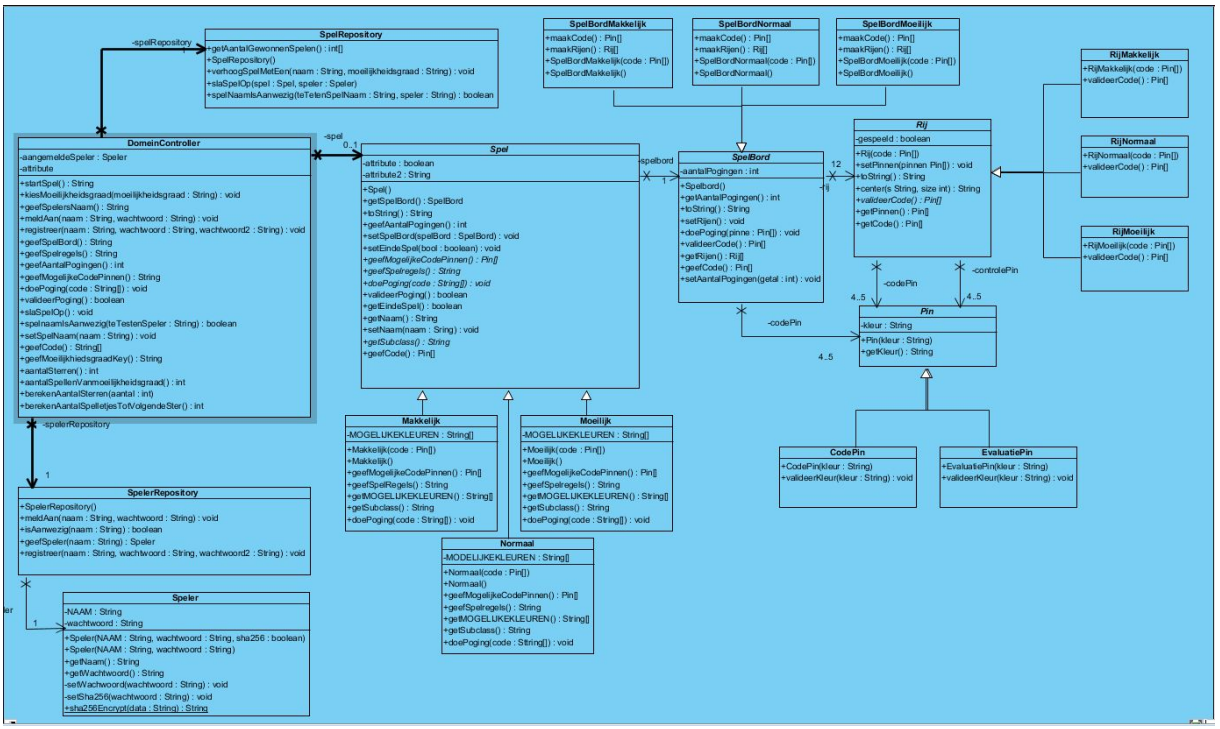
Geen extra commentaar. Dit is stapsgewijs opgebouwd.

3.4 OC

Contract: doe poging
Operation: doePoging(code)
Cross References: Speel mastermind
Preconditions: - Het spel is reeds opgestart
Postconditions; - Associatie tussen spelbord en evaluatiepin werd aangemaakt - Evaliatiepinnen en spelbord werden aangepast

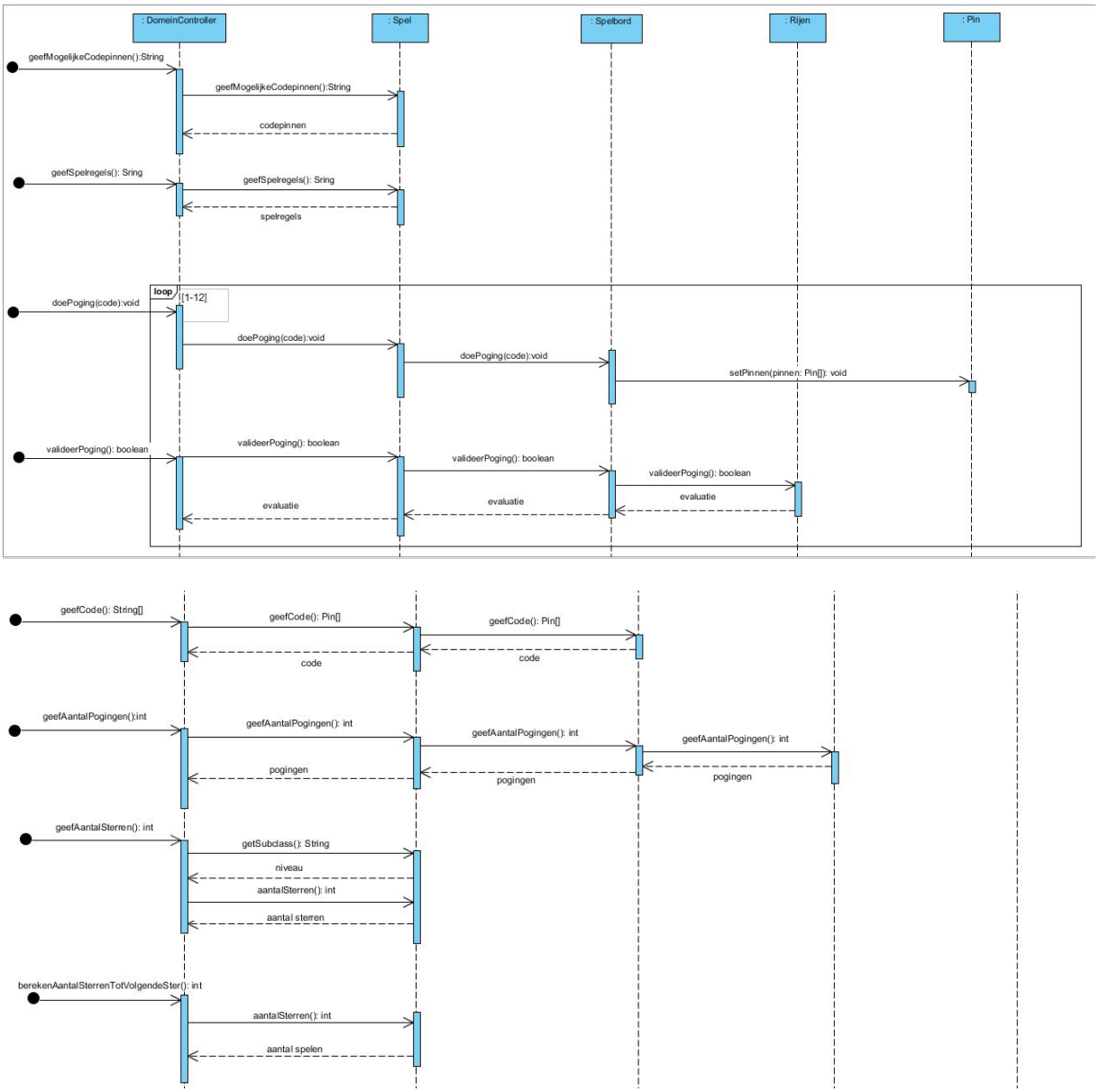
Een OC voor doePoging(code) want deze is een doing.

3.5 DCD



Er zijn geen extra klassen aangemaakt. Bij elke klasse zijn een aantal methodes toegevoegd.

3.6 SD



geefMogelijkeCodepinnen(): String geef codepinnen terug die gehaald worden vanuit de klasse Spel. Deze codepinnen worden aan de hand van een String teruggegeven.

geefSpelregels(): String zal de spelregels terug geven vlak voor het spel gespeeld wordt. Uiteraard doen we dit met een String.

doePoging(code): String zorgt ervoor dat de speler een poging kan doen bij het spelen van Mastermind. Aan de hand van de loop wordt ervoor gezorgd dat de speler maximaal 12 zetten kan doen. De code wordt meegegeven die dan berekent zal worden.

In de volgende methode valideerPoging(): boolean wordt berekend hoeveel pinnen juist gezet zijn. De aantal evaluatiepinnen worden teruggegeven om aan te tonen hoeveel pinnen juist zijn. Bij een juiste code zal dat een true opleveren.

geefCode(): String[] zal de code opvragen met een Stringarray. In de klasse Spel en Spelbord zal het returntype een array van pinnen zijn zodat alle pinnen worden opgevraagd en meegegeven.

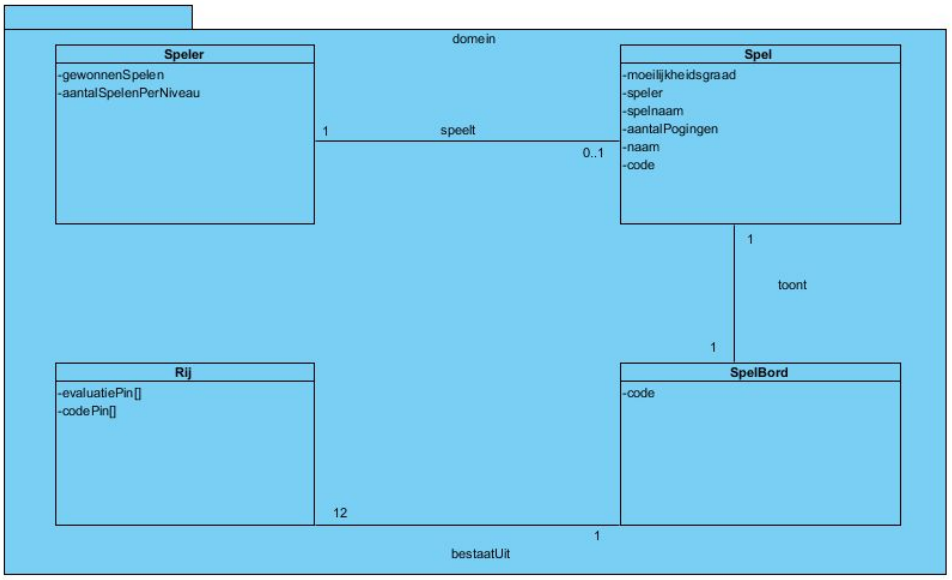
geefAantalPogingen(): int zal de aantal pogingen opvragen en weergeven. Deze methode wordt opgevraagd in de klasse Rij die onthoud hoeveel rijen er gebruikt zijn.

getAantalSterren(): int zal de aantal sterren bereken en teruggeven. Per niveau worden de gewonnen spellen berekent die dan omgezet worden naar sterren.

berekenSterrenTotVolgendeSter():int vraagt aan de klasse Spel hoeveel spelen er nog gespeeld moeten worden voor een volgende ster. Het aantal spelen zal teruggegeven worden met een int.

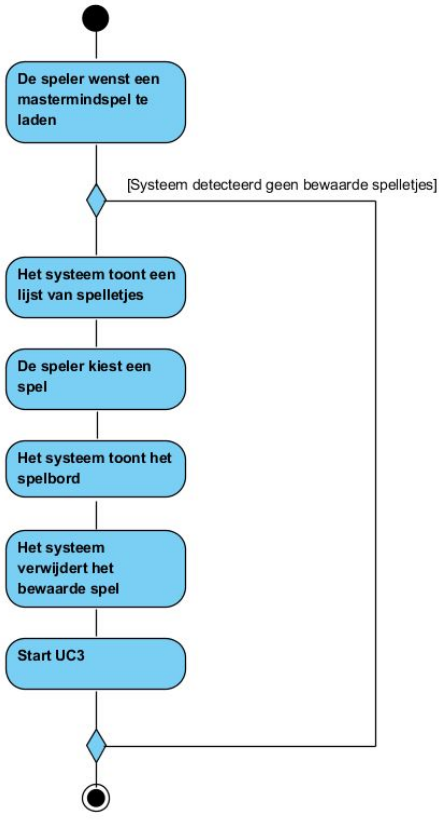
4 UC4

4.2 Domeinmodel



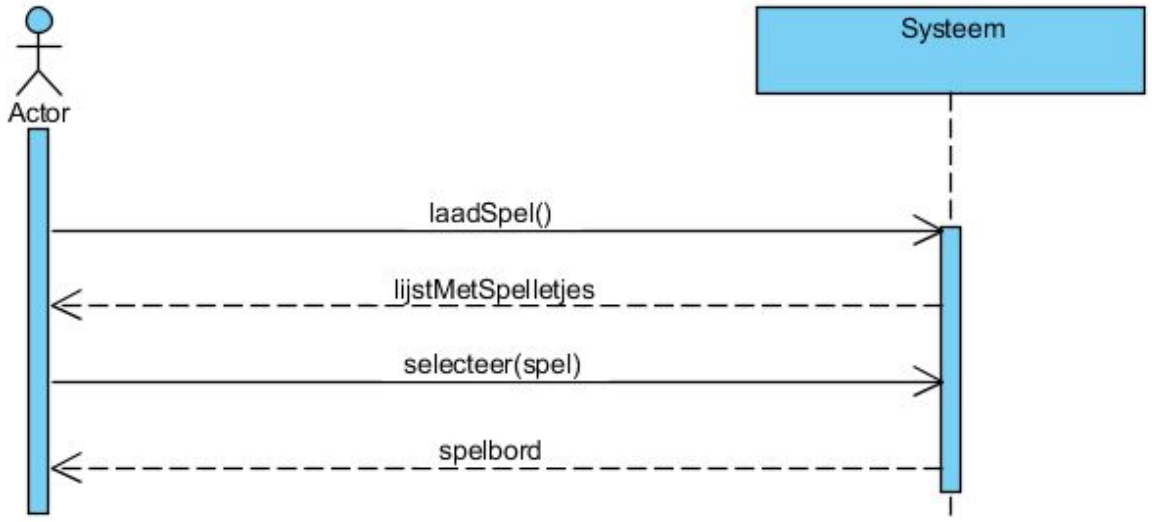
Dit is een domeinmodel van UC4 vermengd met de vorige UC'en.

4.1 Activity Diagram



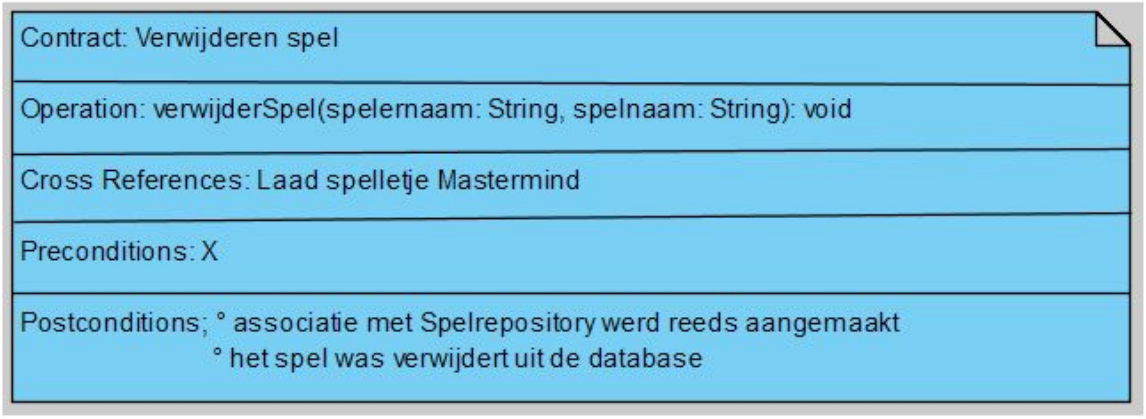
Geen extra commentaar. Dit is stapsgewijs opgebouwd.

4.3 SSD



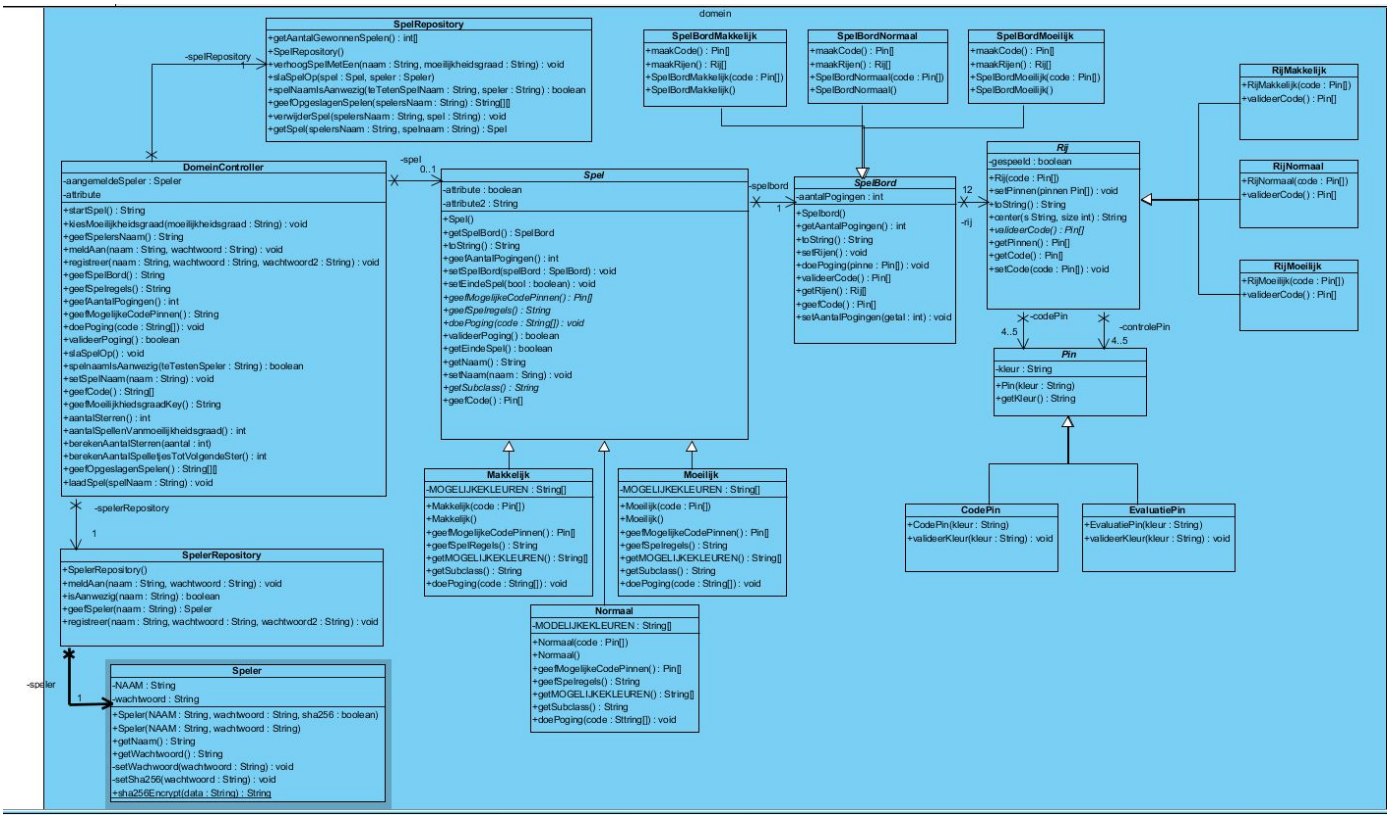
Geen extra commentaar. Dit is stapsgewijs opgebouwd.

4.4 OC



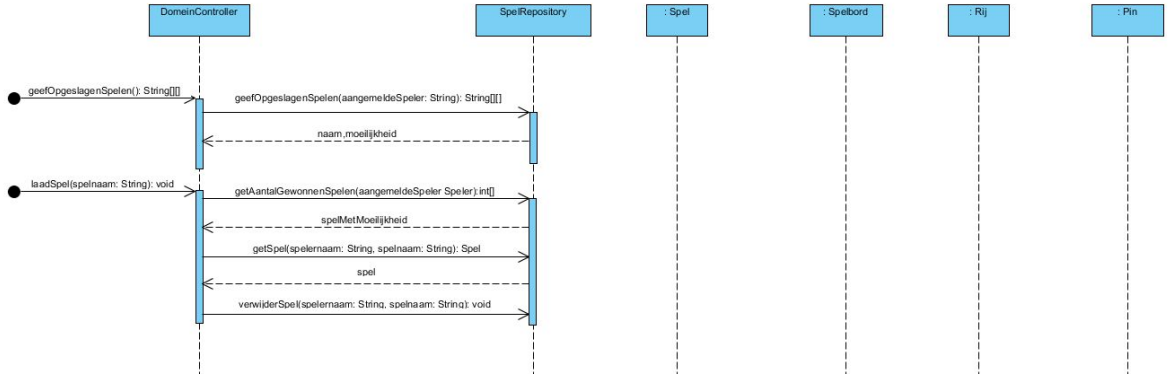
OC gemaakt voor het verwijderen van het spel. verwiderSpel(spelernaam: String, spelnaam: String): void is namelijk een doing.

4.5 DCD



Bij dit DCD is er opnieuw geen klasse bijgevoegd. Er zijn enkel een aantal methoden toegevoegd.

3.6 SD

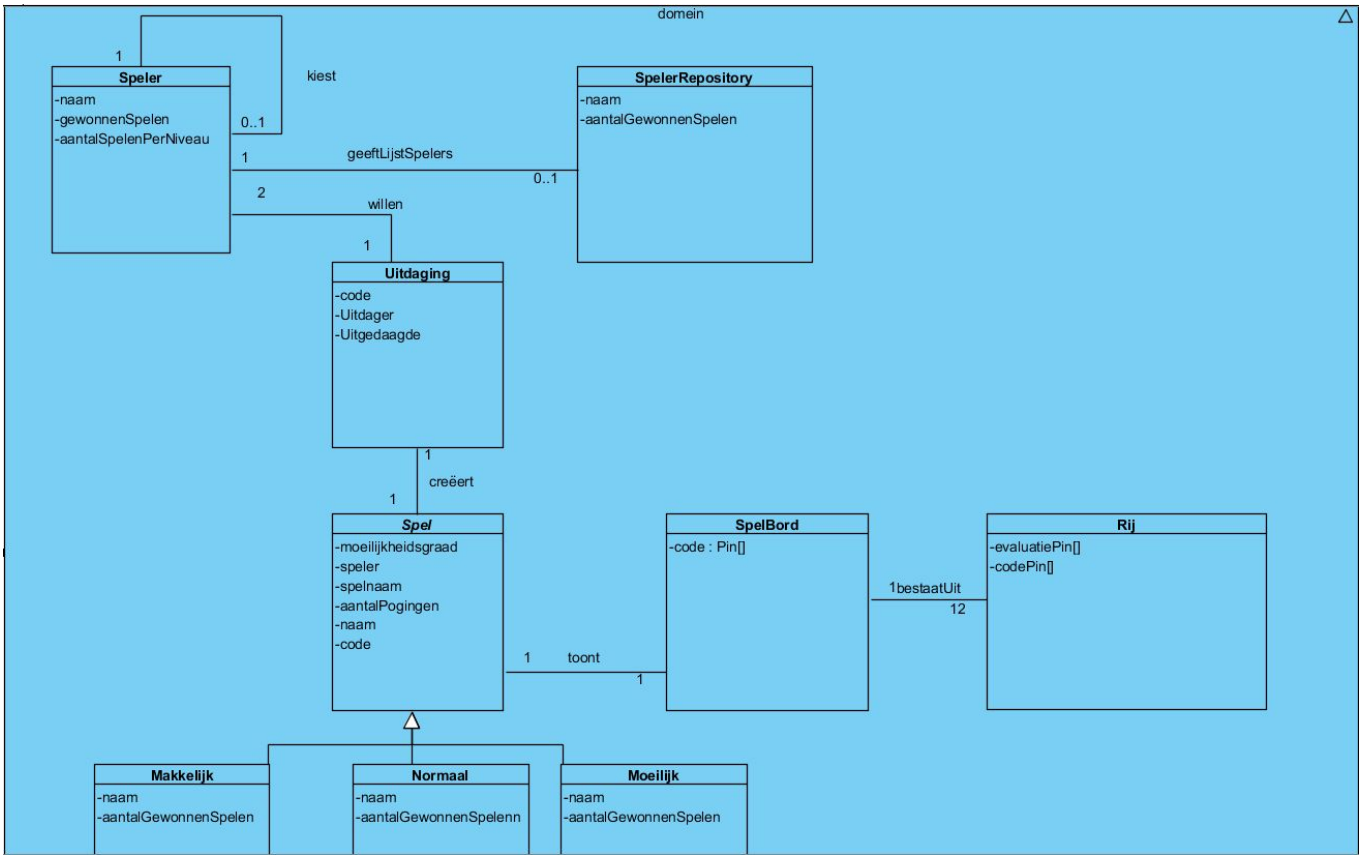


geefOpgeslagenSpelen():String[][] zal een lijst van opgeslagen spelen geven. Dit wordt met een dubbele array gedaan zodat elke speler zijn eigen spelen kan bijhouden.

laadSpel(spelnaam: String): void zorgt ervoor dat het gekozen spel verder kan gespeeld worden. de parameter spelnaam is de naam van het opgeslagen dat de speler wil afspelen. Eens de speler begonnen is met het spel wordt het opgeslagen spel onmiddellijk verwijdert.

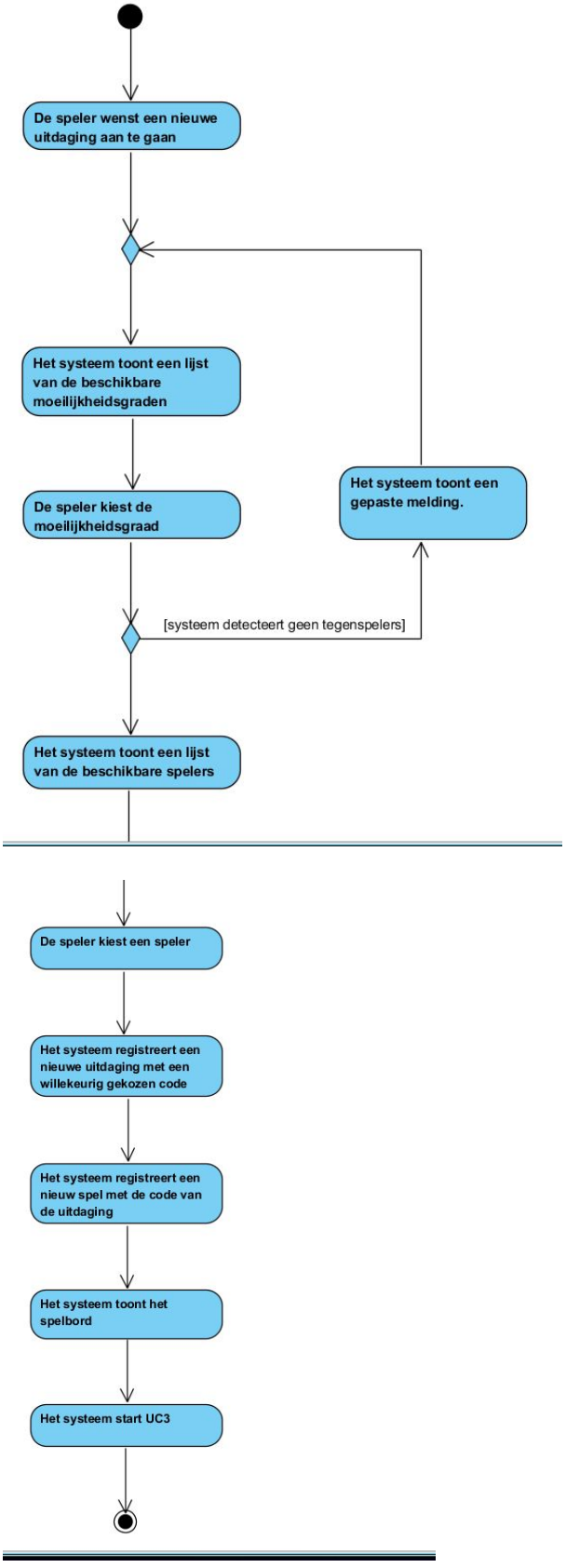
5 UC5

5.1 Domeinmodel



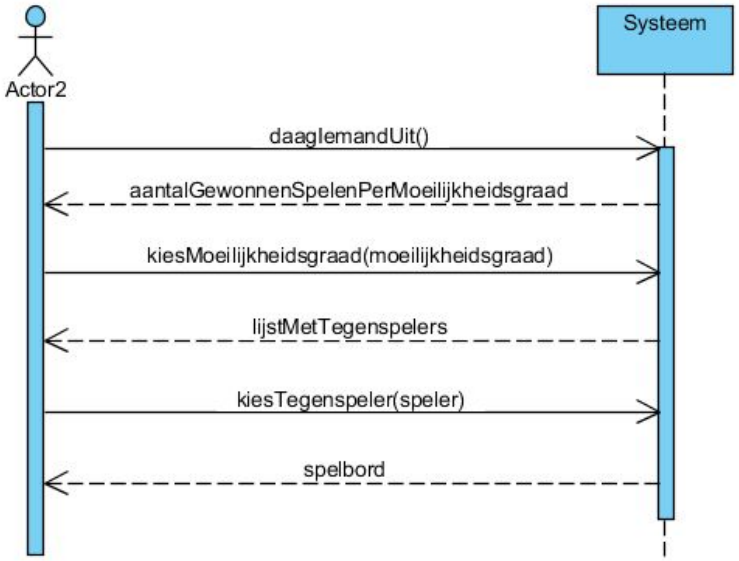
attributen van UC5 bij de vorige UC'en toegevoegd.

5.2 Activity Diagram



Geen extra commentaar. Dit is stapsgewijs opgebouwd.

5.3 SSD



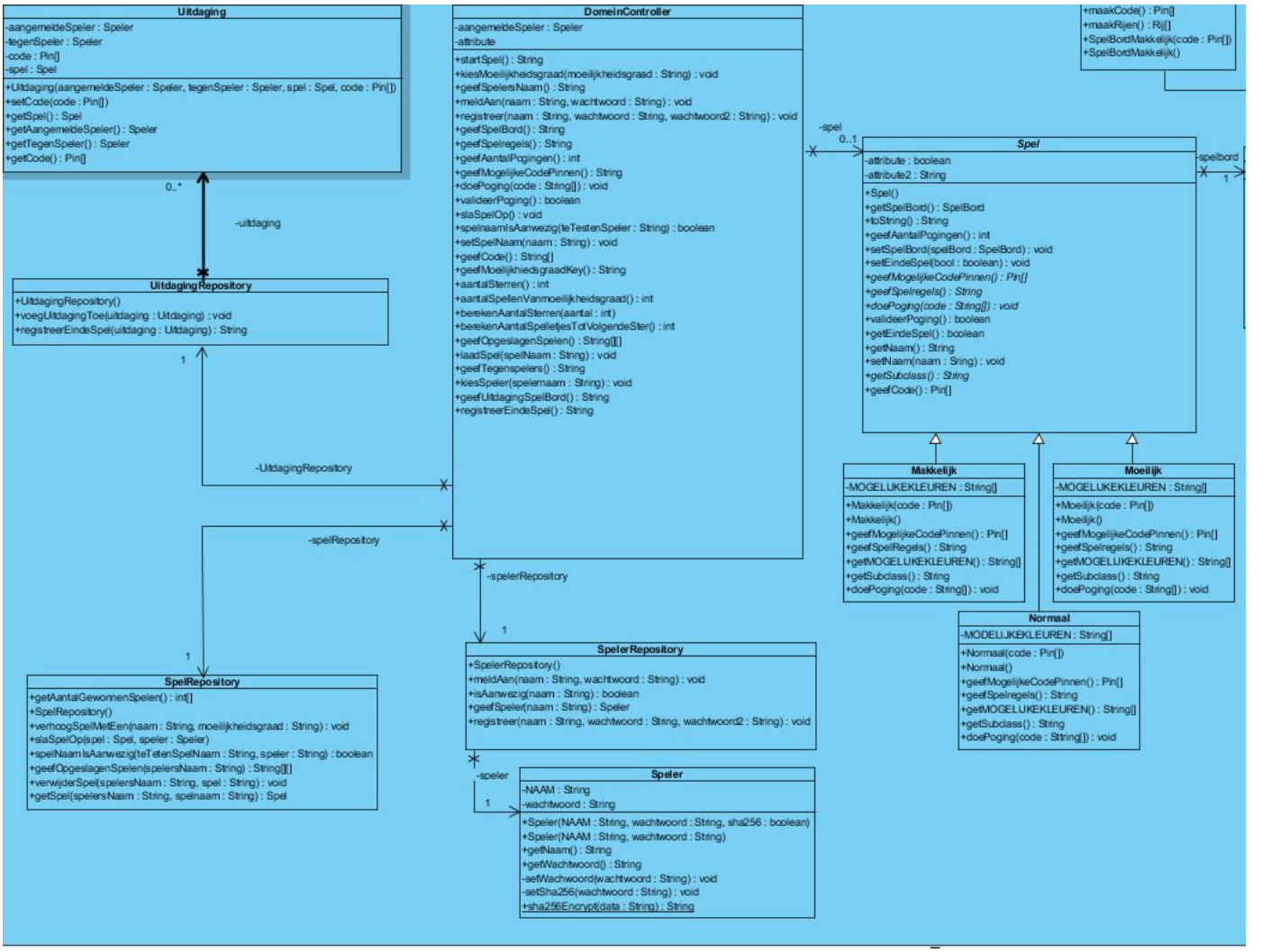
Geen extra commentaar. Dit is stapsgewijs opgebouwd.

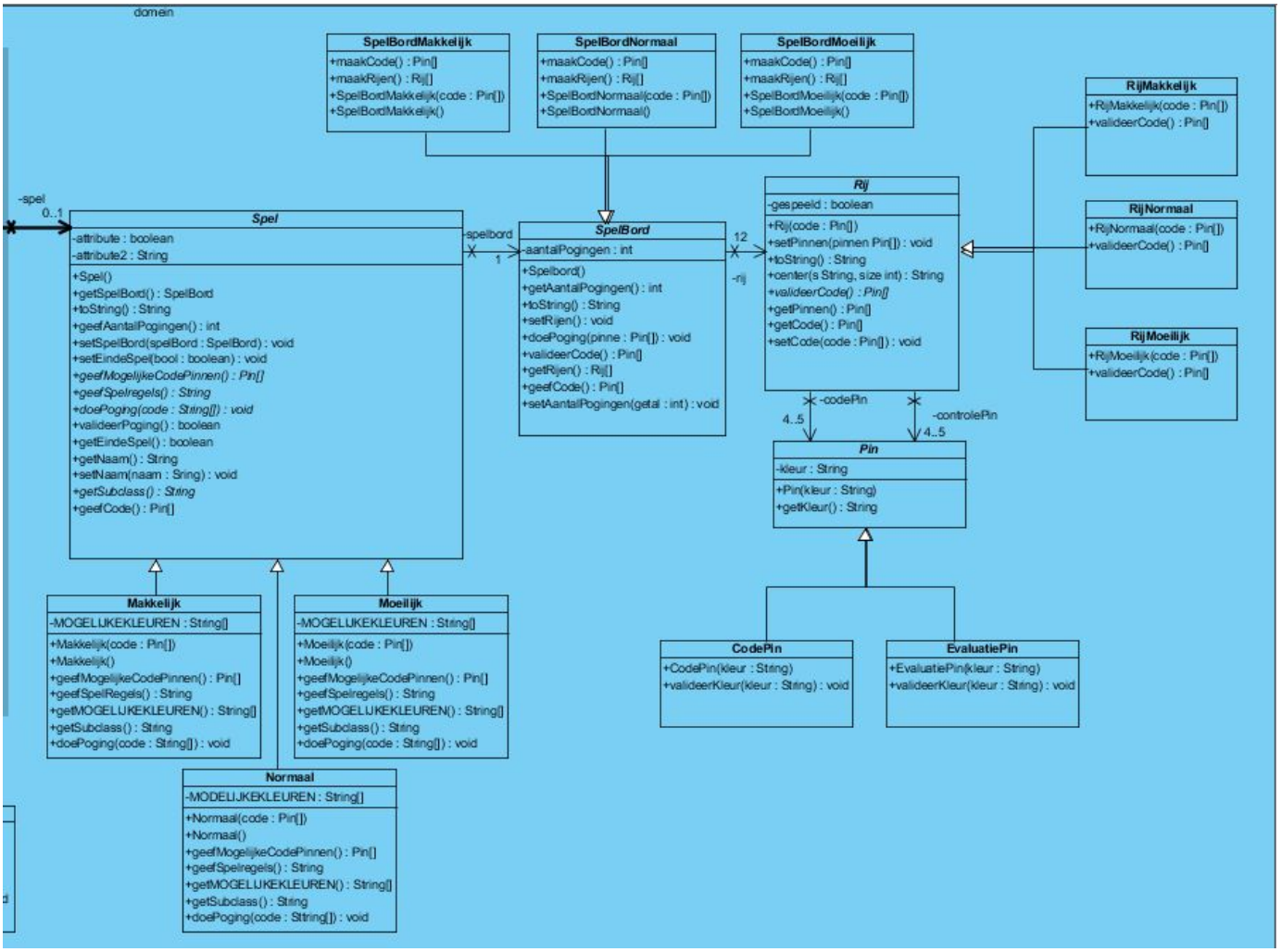
5.4 OC

Contract: Speler kiezen
Operation: kiesSpeler(spelernaam: String): void
Cross References: Daag uit
Preconditions: ° tegenspelers werden weergegeven
Postconditions: ° het object Uitdaging werd aangemaakt ° uitdaging werd reeds toegevoegd

OC voor de methode kiesSpeler(spelernaam: String): void want deze is een doing.

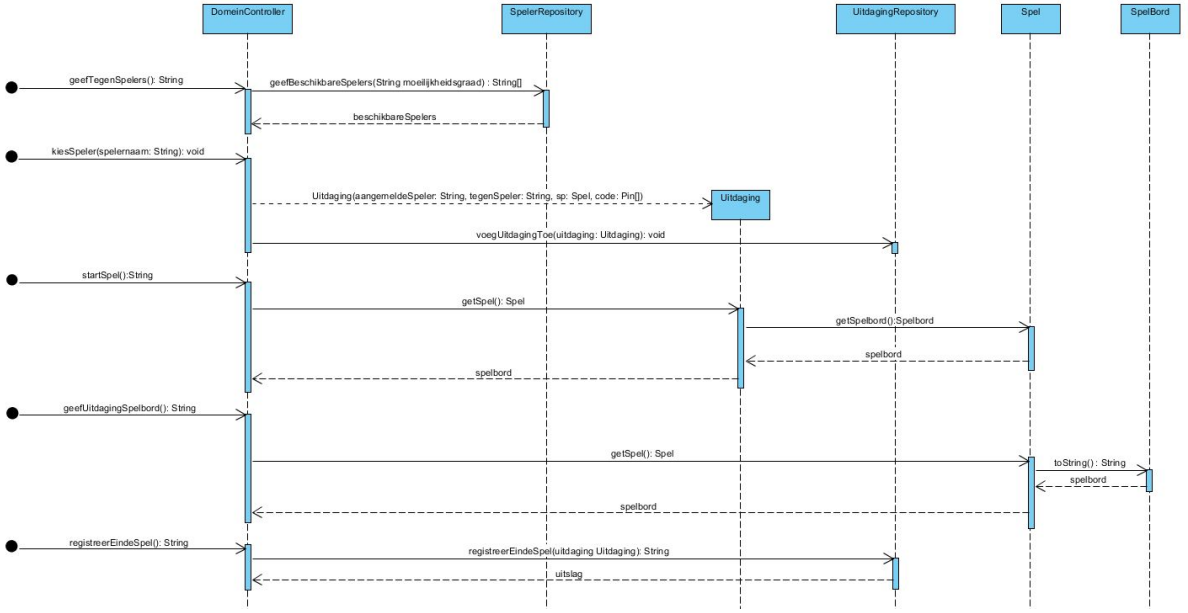
5.5 DCD


Uitdaging
-aangemeldeSpeler : Speler
-tegenSpeler : Speler
-code : Pin[]
-spel : Spel
+Uitdaging(aangemeldeSpeler : Speler, tegenSpeler : Speler, spel : Spel, code : Pin[])
+setCode(code : Pin[])
+getSpel() : Spel
+getAangemeldeSpeler() : Speler
+getTegenSpeler() : Speler
+getCode() : Pin[]
0..*
-uitdaging
UitdagingRepository
+UitdagingRepository()
+voegUitdagingToe(uitdaging : Uitdaging) : void
+registreerEindeSpel(uitdaging : Uitdaging) : String
1
-UitdagingRepository
-spelRepository
1
SpelRepository
+getAantalGewonnenSpelen() : int[]
+SpelRepository()
+verhoogSpelMetEen(naam : String, moeilijkheidsgraad : String) : void
+slaSpelOp(spel : Spel, speler : Speler)
+spelNaamIsAanwezig(teTetenSpelNaam : String, speler : String) : boolean
+geefOpgeslagenSpelen(spelersNaam : String) : String[][]
+verwijderSpel(spelersNaam : String, spel : String) : void
+getSpel(spelersNaam : String, spelnaam : String) : Spel
DomeinController
-aangemeldeSpeler : Speler
-attribute
+startSpel() : String
+kiesMoeilijkheidsgraad(moeilijkheidsgraad : String) : void
+geefSpelersNaam() : String
+meldAan(naam : String, wachtwoord : String) : void
+registreer(naam : String, wachtwoord : String, wachtwoord2 : String) : void
+geefSpelBord() : String
+geefSpelregels() : String
+geefAantalPogingen() : int
+geefMogelijkeCodePinnen() : String
+doePoging(code : String[]) : void
+valideerPoging() : boolean
+slaSpelOp() : void
+spelnaamIsAanwezig(teTestenSpeler : String) : boolean
+setSpelNaam(naam : String) : void
+geefCode() : String[]
+geefMoeilijkheidsgraadKey() : String
+aantalSterren() : int
+aantalSpellenVanmoeilijkheidsgraad() : int
+berekenAantalSterren(aantal : int)
+berekenAantalSpellenTotVolgendeSter() : int
+geefOpgeslagenSpelen() : String[][]
+laadSpel(spelNaam : String) : void
+geefTegenspelers() : String
+kiesSpeler(spelernaam : String) : void
+geefUitdagingSpelBord() : String
+registreerEindeSpel() : String
-spelerRepository
1
SpelerRepository
+SpelerRepository()
+meldAan(naam : String, wachtwoord : String) : void
+isAanwezig(naam : String) : boolean
+geefSpeler(naam : String) : Speler
+registreer(naam : String, wachtwoord : String, wachtwoord2 : String) : void
-speler
1
Speler
-NAAM : String
-wachtwoord : String
+Speler(NAAM : String, wachtwoord : String, sha256 : boolean)
+Speler(NAAM : String, wachtwoord : String)
+getNaam() : String
+getWachtwoord() : String
-setWachwoord(wachtwoord : String) : void
-setSha256(wachtwoord : String) : void
+sha256Encrypt(data : String) : String
+maakCode() : Pin[]
+maakRijen() : Rij[]
+SpelBordMakkelijk(code : Pin[])
+SpelBordMakkelijk()
-spel
0..1
Spel
-attribute : boolean
-attribute2 : String
+Spel()
+getSpelBord() : SpelBord
+toString() : String
+geefAantalPogingen() : int
+setSpelBord(spelBord : SpelBord) : void
+setEindeSpel(bool : boolean) : void
+geefMogelijkeCodePinnen() : Pin[]
+geefSpelregels() : String
+doePoging(code : String[]) : void
+valideerPoging() : boolean
+getEindeSpel() : boolean
+getNaam() : String
+setNaam(naam : Sring) : void
+getSubclass() : String
+geefCode() : Pin[]
-spelbord
1
Makkelijk
-MOGELIJKEKLEUREN : String[]
+Makkelijk(code : Pin[])
+Makkelijk()
+geefMogelijkeCodePinnen() : Pin[]
+geefSpelRegels() : String
+getMOGELIJKEKLEUREN() : String[]
+getSubclass() : String
+doePoging(code : String[]) : void
Moeilijk
-MOGELIJKEKLEUREN : String[]
+Moeilijk(code : Pin[])
+Moeilijk()
+geefMogelijkeCodePinnen() : Pin[]
+geefSpelregels() : String
+getMOGELIJKEKLEUREN() : String[]
+getSubclass() : String
+doePoging(code : String[]) : void
Normaal
-MODELIJKEKLEUREN : String[]
+Normaal(code : Pin[])
+Normaal()
+geefMogelijkeCodePinnen() : Pin[]
+geefSpelregels() : String
+getMOGELIJKEKLEUREN() : String[]
+getSubclass() : String
+doePoging(code : String[]) : void



We hebben een klasse Uitdaging en UitdagingRepository aangemaakt. Opnieuw zijn er een aantal nieuwe methodes aan toegevoegd.

5.6 SD



geefTegenSpelers(): String zal een String geven van alle spelers die beschikbaar zijn.

kiesSpeler(spelernaam: String) void zal een nieuwe klasse Uitdaging aanmaken. De uitdaging zal aangemaakt worden en het spel kan beginnen.

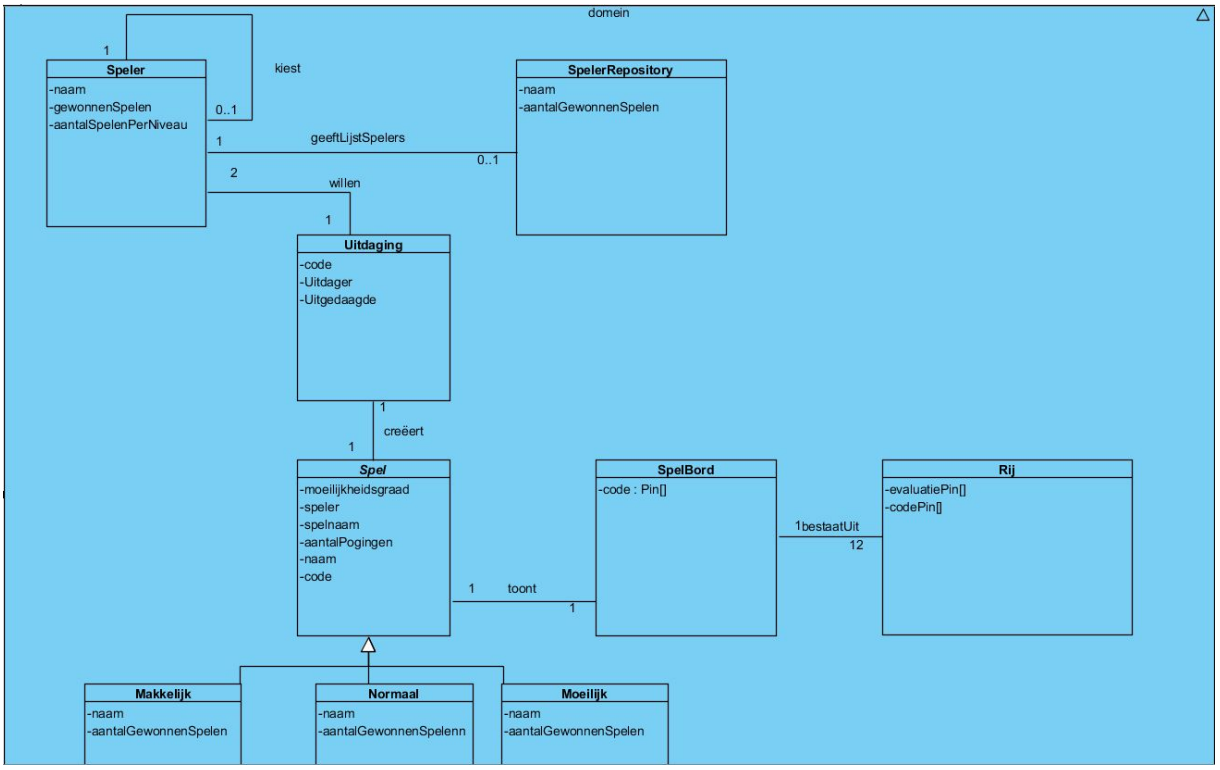
starSpel():String zal een spel starten maar wordt opgeroepen in de klasse UitdagingRepository.

registreerEindeSpel(): String zal de zien welke speler gewonnen is en wordt daarna in de database opgeslagen voor het klassement. De uitslag wordt weergegeven met een String.

De methodes die gebruikt worden in UC3 worden opnieuw uitgevoerd om een spel te spelen.

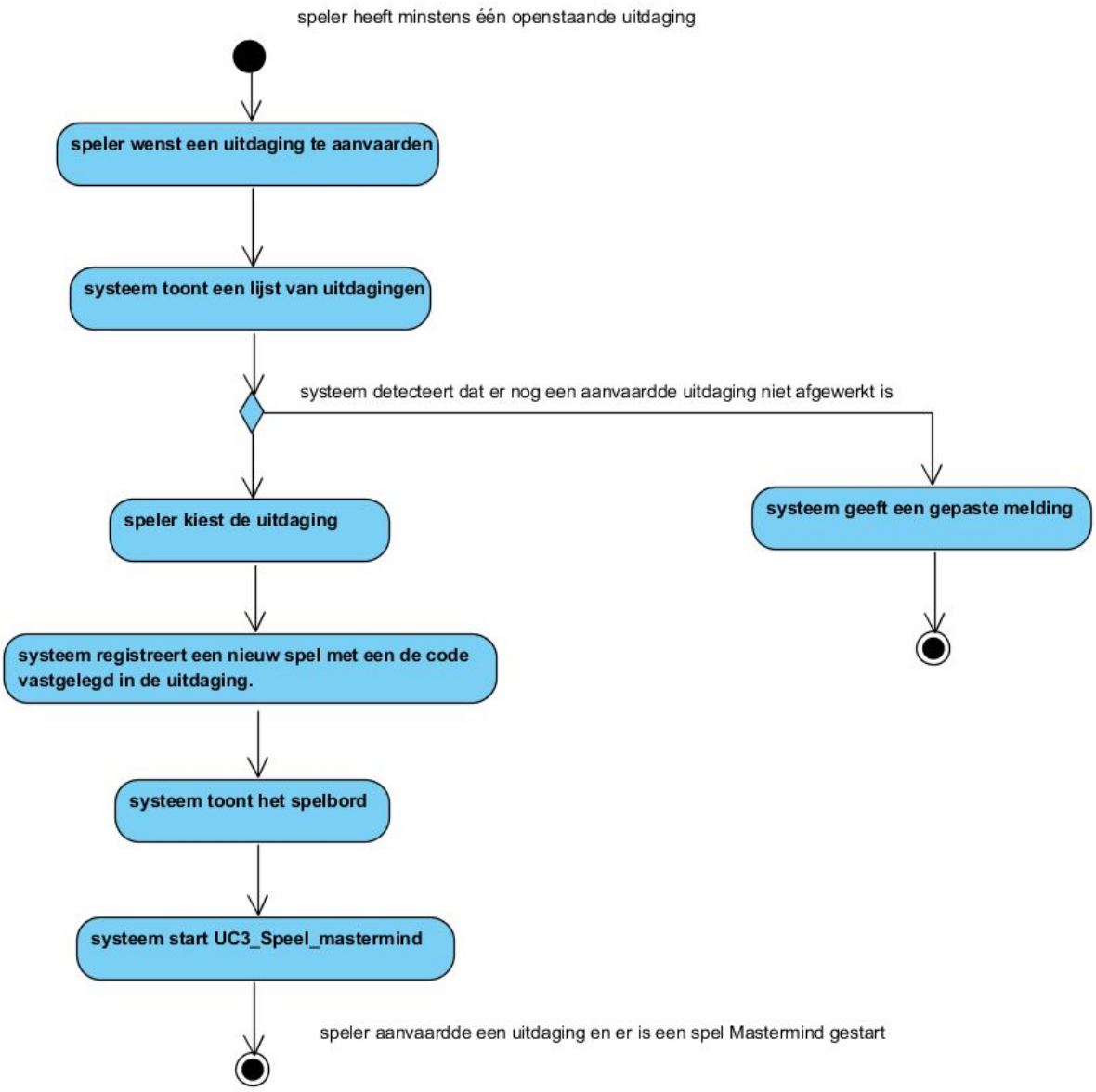
6 UC6

6.2 Domeinmodel



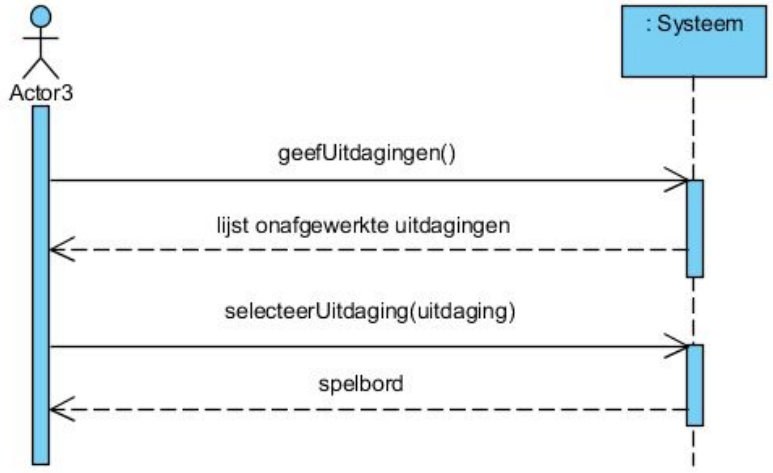
Hetzelfde domeinmodel als UC5.

6.2 Activity Diagram



Geen extra commentaar. Dit is stapsgewijs opgebouwd.

6.3 SSD



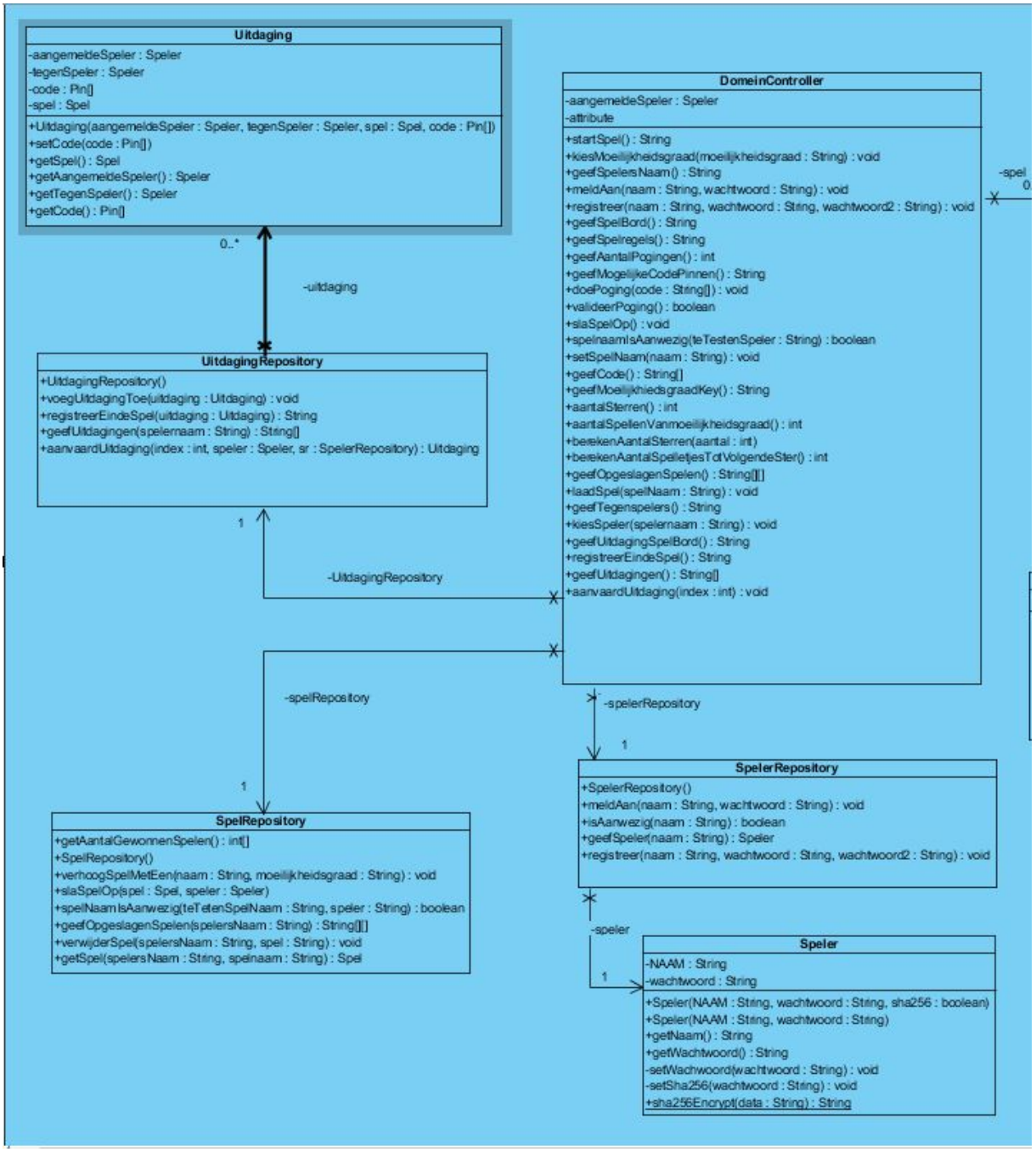
Geen extra commentaar. Dit is stapsgewijs opgebouwd.

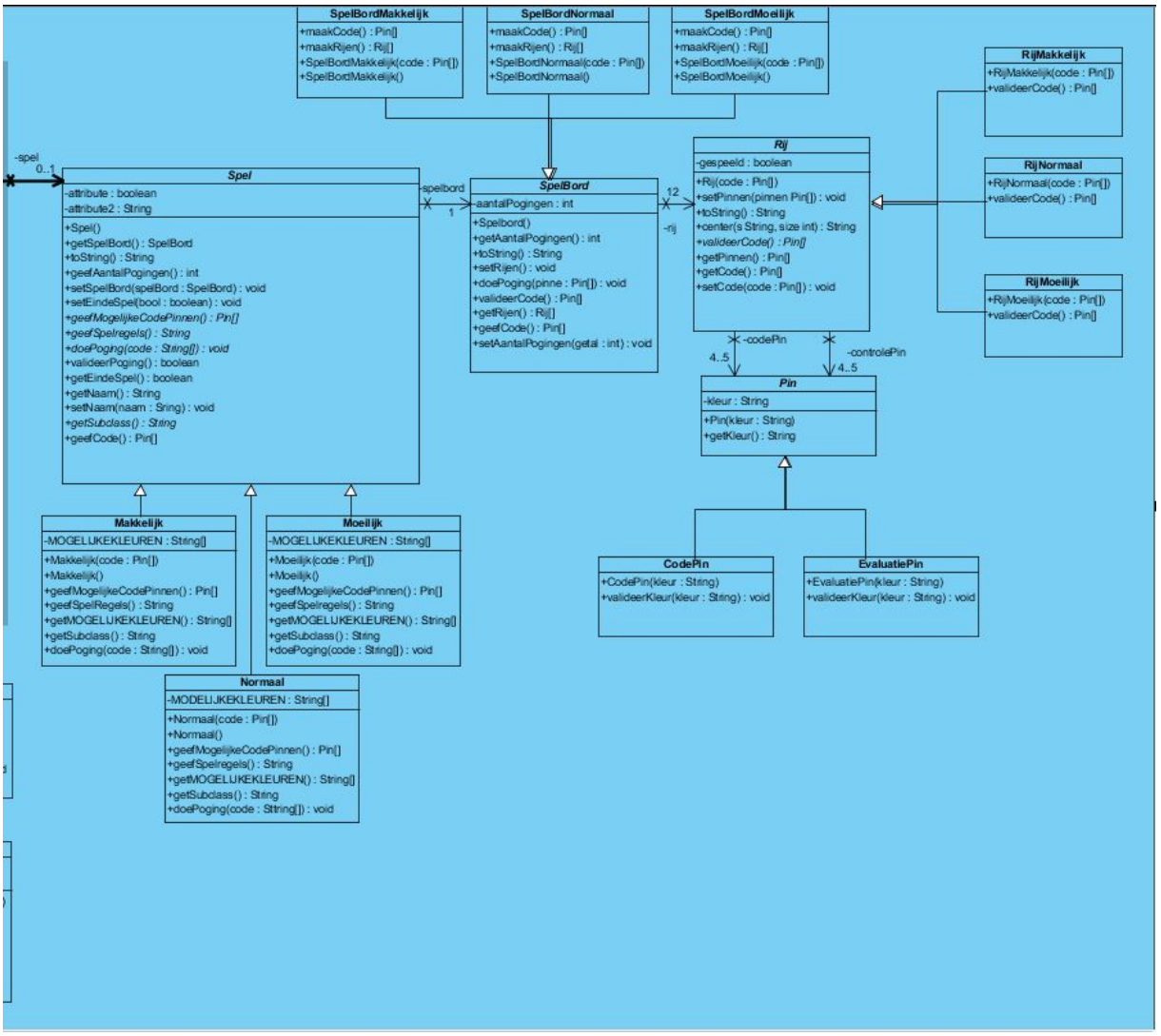
6.4 OC

Contract: Uitdaging aanvaarden
Operation: aanvaardUitdaging(index: int): void
Cross References: Aanvaard uitdaging
Preconditions: ° uitdagingen werden weergegeven
Postconditions: ° associatie met de UitdagingRepository ° uitdaging was aanvaard

OC van de methode aanvaardUitdaging(index: int): void want deze is een doing.

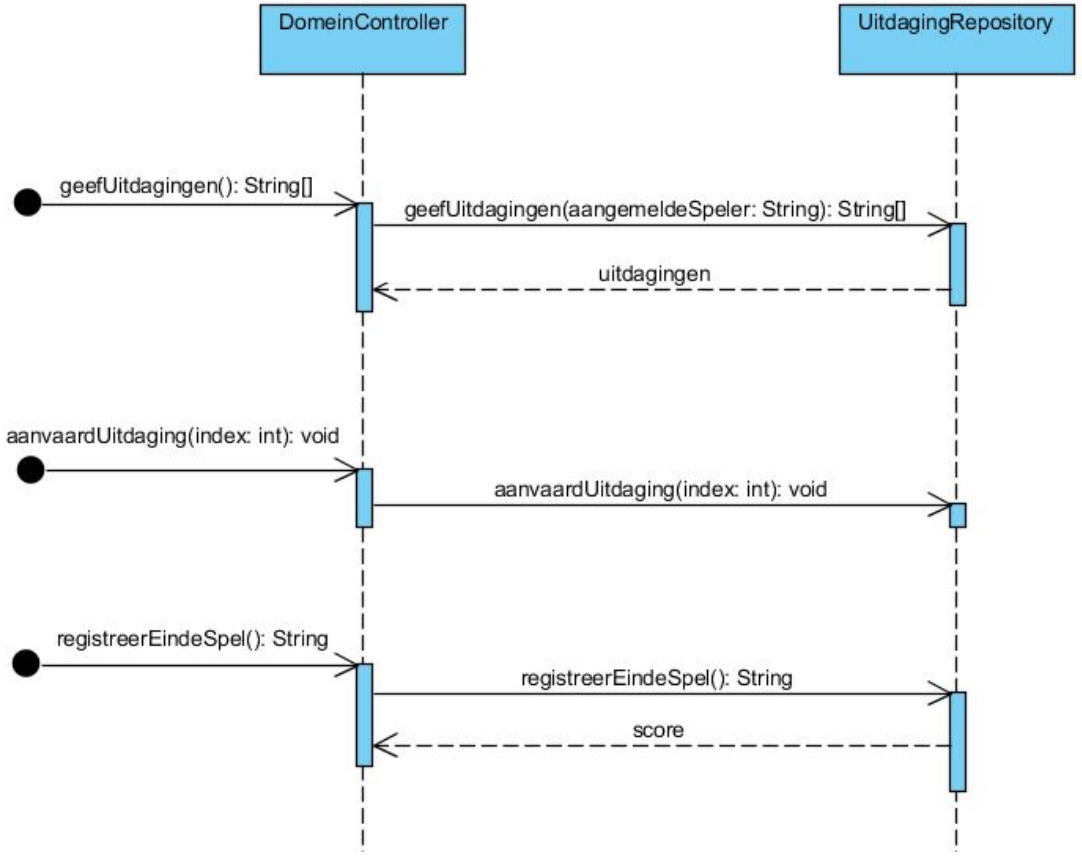
6.5 DCD


Uitdaging
-aangemeldeSpeler : Speler
-tegenSpeler : Speler
-code : Pin[]
-spel : Spel
+Uitdaging(aangemeldeSpeler : Speler, tegenSpeler : Speler, spel : Spel, code : Pin[])
+setCode(code : Pin[])
+getSpel() : Spel
+getAangemeldeSpeler() : Speler
+getTegenSpeler() : Speler
+getCode() : Pin[]
0..*
-uitdaging
UitdagingRepository
+UitdagingRepository()
+voegUitdagingToe(uitdaging : Uitdaging) : void
+registreerEindeSpel(uitdaging : Uitdaging) : String
+geefUitdagingen(spelernaam : String) : String[]
+aanvaardUitdaging(index : int, speler : Speler, sr : SpelerRepository) : Uitdaging
1
-UitdagingRepository
DomeinController
-aangemeldeSpeler : Speler
-attribute
+startSpel() : String
+kiesMoeilijkheidsgraad(moeilijkheidsgraad : String) : void
+geefSpelersNaam() : String
+meldAan(naam : String, wachtwoord : String) : void
+registreer(naam : String, wachtwoord : String, wachtwoord2 : String) : void
+geefSpelBord() : String
+geefSpelregels() : String
+geefAantalPogingen() : int
+geefMogelijkeCodePinnen() : String
+doePoging(code : String[]) : void
+valideerPoging() : boolean
+slaSpelOp() : void
+spelnaamIsAanwezig(teTestenSpeler : String) : boolean
+setSpelNaam(naam : String) : void
+geefCode() : String[]
+geefMoeilijkheidsgraadKey() : String
+aantalSterren() : int
+aantalSpellenVanmoeilijkheidsgraad() : int
+berekenAantalSterren(aantal : int)
+berekenAantalSpelletjesTotVolgendeSter() : int
+geefOpgeslagenSpelen() : String[][]
+laadSpel(spelNaam : String) : void
+geefTegenspelers() : String
+kiesSpeler(spelernaam : String) : void
+geefUitdagingSpelBord() : String
+registreerEindeSpel() : String
+geefUitdagingen() : String[]
+aanvaardUitdaging(index : int) : void
-spel
-spelRepository
1
-spelerRepository
1
SpelRepository
+getAantalGewonnenSpelen() : int[]
+SpelRepository()
+verhoogSpelMetEen(naam : String, moeilijkheidsgraad : String) : void
+slaSpelOp(spel : Spel, speler : Speler)
+spelNaamIsAanwezig(teTetenSpelNaam : String, speler : String) : boolean
+geefOpgeslagenSpelen(spelersNaam : String) : String[][]
+verwijderSpel(spelersNaam : String, spel : String) : void
+getSpel(spelersNaam : String, spelnaam : String) : Spel
SpelerRepository
+SpelerRepository()
+meldAan(naam : String, wachtwoord : String) : void
+isAanwezig(naam : String) : boolean
+geefSpeler(naam : String) : Speler
+registreer(naam : String, wachtwoord : String, wachtwoord2 : String) : void
-speler
1
Speler
-NAAM : String
-wachtwoord : String
+Speler(NAAM : String, wachtwoord : String, sha256 : boolean)
+Speler(NAAM : String, wachtwoord : String)
+getNaam() : String
+getWachtwoord() : String
-setWachwoord(wachtwoord : String) : void
-setSha256(wachtwoord : String) : void
+sha256Encrypt(data : String) : String



Dit DCD is een uitgebreidere versie van het DCD van UC5. er zijn extra methodes aan toegevoegd.

6.6 SD



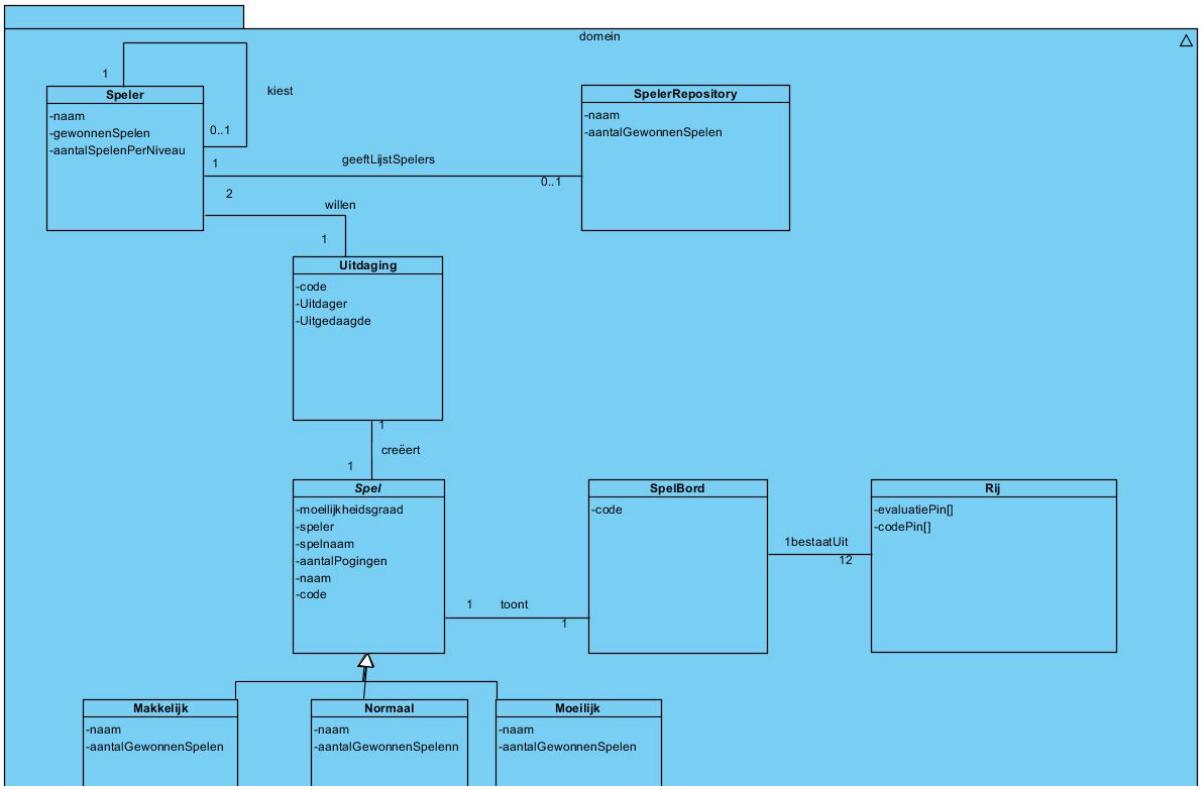
geefUitdagingen(): String[] zal een Stringarray teruggeven van alle uitdagingen die bestaan. Dit zal gevraagd worden aan de UitdagingRepository.

aanvaarUitdaging(index: int): void zal de uitdaging aanvaarden die de speler wilt spelen. Na de aanvaarding zal UC3 starten waarbij het spelbord gespeeld kan worden.

registreerEindeSpel(): String zal de winnaar en verliezer registreren en zal een String teruggeven voor de speler die de uitdaging heeft aanvaard.

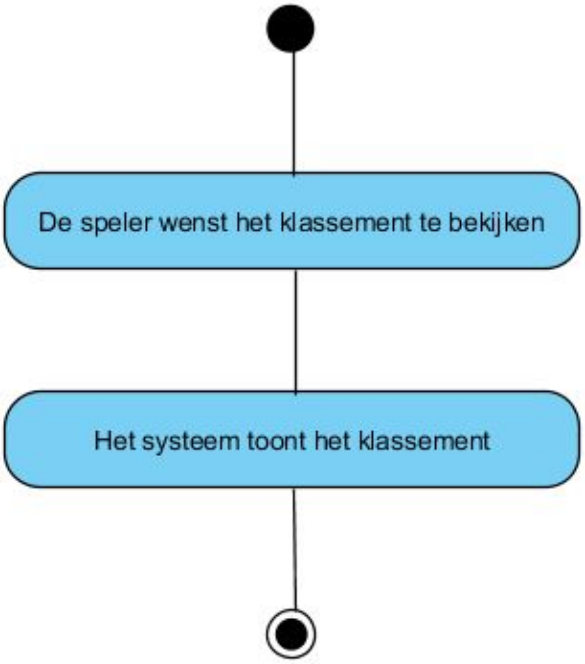
7 UC7

7.1 Domeinmodel

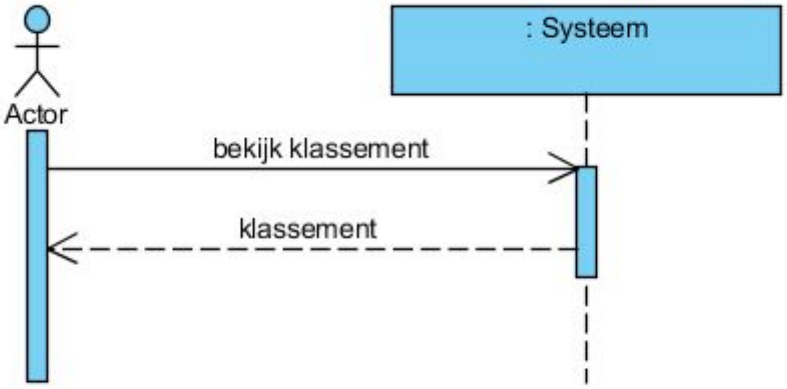


Hetzelfde domeinmodel als UC6. klassement zal niet als attribuut meegegeven worden.

7.2 Activity Diagram

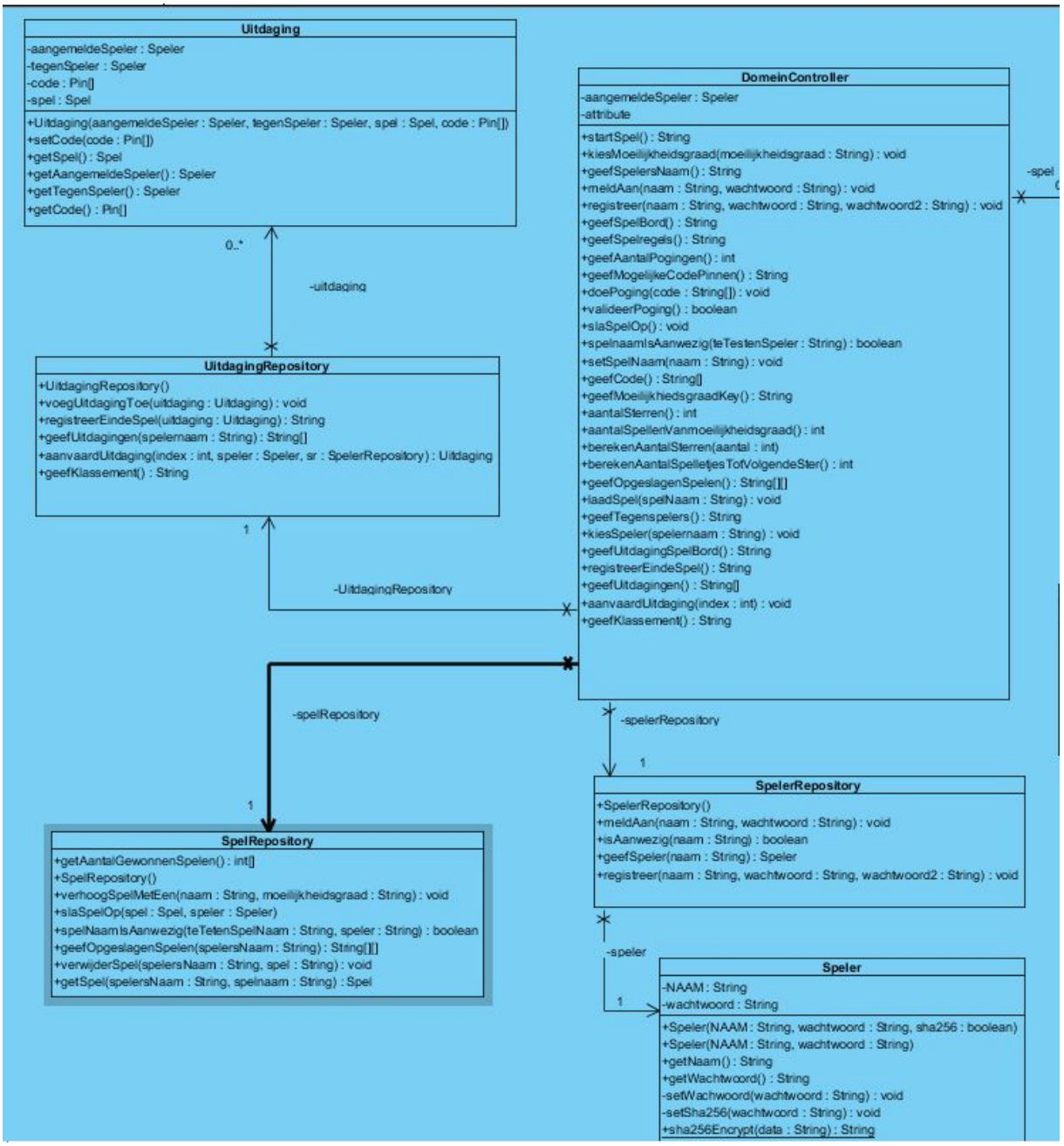


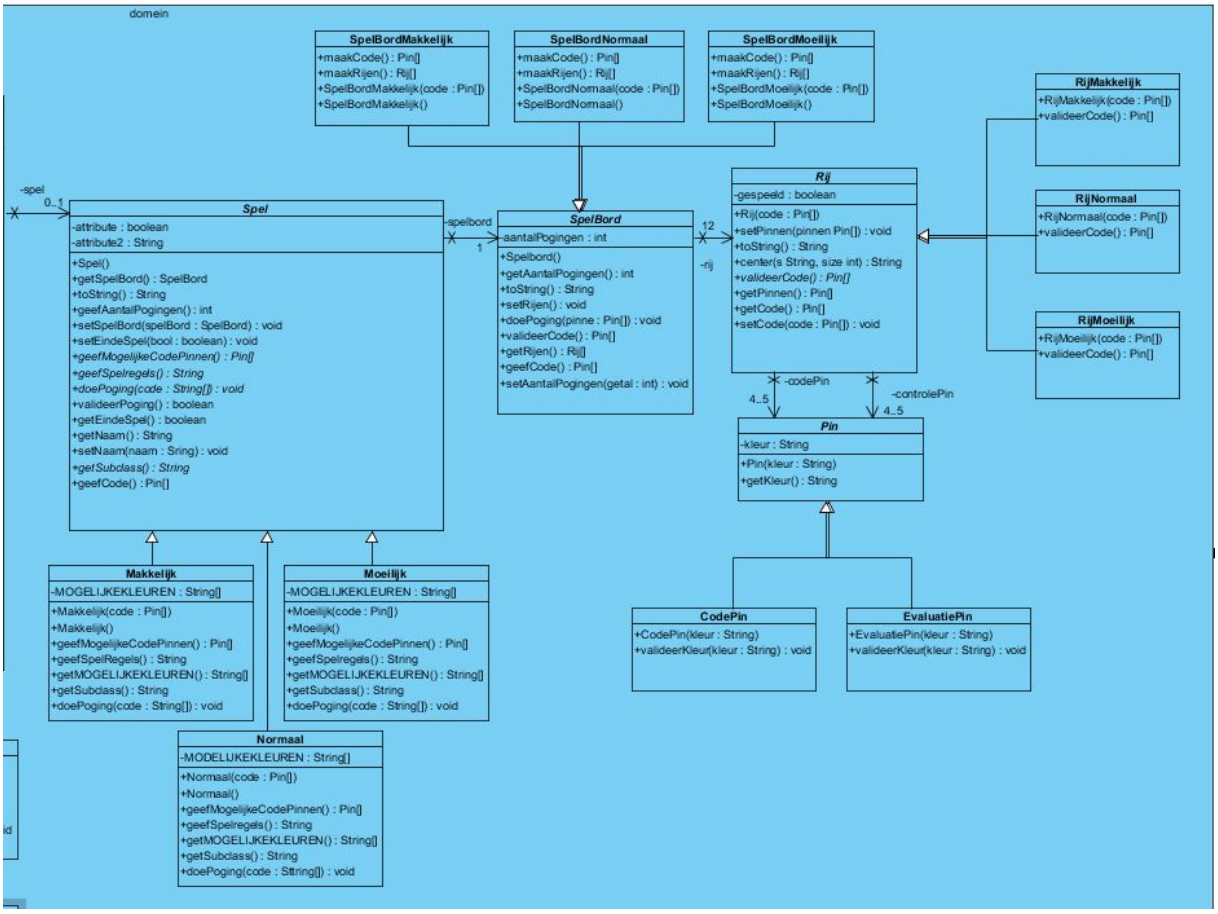
7.3 SSD



Geen extra commentaar. Dit is stapsgewijs opgebouwd.

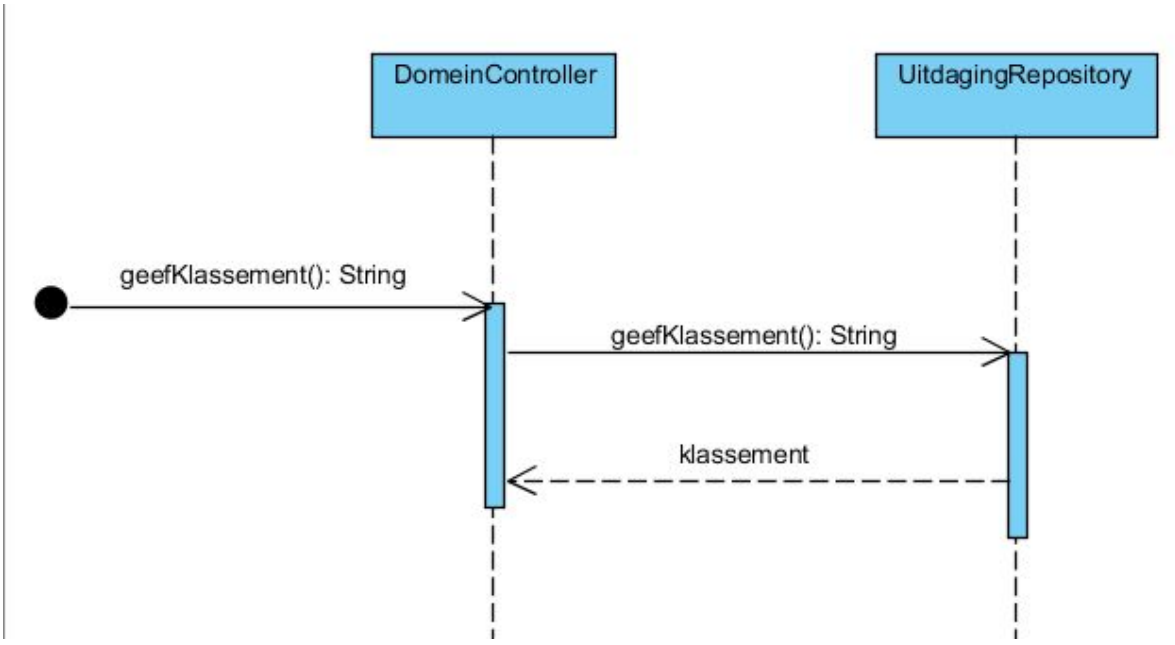
7.4 DCD


Uitdaging
-aangemeldeSpeler : Speler
-tegenSpeler : Speler
-code : Pin[]
-spel : Spel
+Uitdaging(aangemeldeSpeler : Speler, tegenSpeler : Speler, spel : Spel, code : Pin[])
+setCode(code : Pin[])
+getSpel() : Spel
+getAangemeldeSpeler() : Speler
+getTegenSpeler() : Speler
+getCode() : Pin[]
0..*
-uitdaging
UitdagingRepository
+UitdagingRepository()
+voegUitdagingToe(uitdaging : Uitdaging) : void
+registreerEindeSpel(uitdaging : Uitdaging) : String
+geefUitdagingen(spelernaam : String) : String[]
+aanvaardUitdaging(index : int, speler : Speler, sr : SpelerRepository) : Uitdaging
+geefKlassement() : String
1
-UitdagingRepository
DomeinController
-aangemeldeSpeler : Speler
-attribute
+startSpel() : String
+kiesMoeilijkheidsgraad(moeilijkheidsgraad : String) : void
+geefSpelersNaam() : String
+meldAan(naam : String, wachtwoord : String) : void
+registreer(naam : String, wachtwoord : String, wachtwoord2 : String) : void
+geefSpelBord() : String
+geefSpelregels() : String
+geefAantalPogingen() : int
+geefMogelijkeCodePinnen() : String
+doePoging(code : String[]) : void
+valideerPoging() : boolean
+slaSpelOp() : void
+spelnaamIsAanwezig(teTestenSpeler : String) : boolean
+setSpelNaam(naam : String) : void
+geefCode() : String[]
+geefMoeilijkheidsgraadKey() : String
+aantalSterren() : int
+aantalSpellenVanmoeilijkheidsgraad() : int
+berekenAantalSterren(aantal : int)
+berekenAantalSpelletjesTotVolgendeSter() : int
+geefOpgeslagenSpelen() : String[][]
+laadSpel(spelNaam : String) : void
+geefTegenspelers() : String
+kiesSpeler(spelernaam : String) : void
+geefUitdagingSpelBord() : String
+registreerEindeSpel() : String
+geefUitdagingen() : String[]
+aanvaardUitdaging(index : int) : void
+geefKlassement() : String
-spel
-spelRepository
1
-spelerRepository
1
SpelRepository
+getAantalGewonnenSpelen() : int[]
+SpelRepository()
+verhoogSpelMetEen(naam : String, moeilijkheidsgraad : String) : void
+slaSpelOp(spel : Spel, speler : Speler)
+spelNaamIsAanwezig(teTetenSpelNaam : String, speler : String) : boolean
+geefOpgeslagenSpelen(spelersNaam : String) : String[][]
+verwijderSpel(spelersNaam : String, spel : String) : void
+getSpel(spelersNaam : String, spelnaam : String) : Spel
SpelerRepository
+SpelerRepository()
+meldAan(naam : String, wachtwoord : String) : void
+isAanwezig(naam : String) : boolean
+geefSpeler(naam : String) : Speler
+registreer(naam : String, wachtwoord : String, wachtwoord2 : String) : void
-speler
1
Speler
-NAAM : String
-wachtwoord : String
+Speler(NAAM : String, wachtwoord : String, sha256 : boolean)
+Speler(NAAM : String, wachtwoord : String)
+getNaam() : String
+getWachtwoord() : String
-setWachwoord(wachtwoord : String) : void
-setSha256(wachtwoord : String) : void
+sha256Encrypt(data : String) : String



DCD van de vorige UC'en waaraan de methodes van UC7 zijn toegevoegd.

7.5 SD



geefKlassement(): String zal een alle gewonnen en verloren uitdagingen berekenen en dit in een klassement stoppen. Het klassement wordt teruggegeven aan de hand van een String.