Fontys Hogeschool ICT

*Handleiding voor het ontwerpen van use case diagrammen, klassendiagrammen, sequentiediagrammen, componentendiagrammen en deploymentdiagrammen.*

Visual Paradigm 11.2

Handleiding 2014

Inhoudsopgave

[Inleiding 2](#_Toc401912558)

[Use case diagram 3](#_Toc401912559)

[Stap 1 – Diagram maken 3](#_Toc401912560)

[Stap 2 – Diagram tekenen 4](#_Toc401912561)

[Stap 3 – Actors toevoegen 4](#_Toc401912562)

[Stap 4 – Use cases toevoegen 5](#_Toc401912563)

[Stap 5 – Use cases vullen 5](#_Toc401912564)

[Klassendiagram 8](#_Toc401912565)

[Stap 1 – Klasse toevoegen 8](#_Toc401912566)

[Stap 2 – Veld toevoegen 8](#_Toc401912567)

[Stap 3 – Methode toevoegen 8](#_Toc401912568)

[Stap 4 – Multipliciteit en compositiesymbolen 9](#_Toc401912569)

[Sequentiediagram 10](#_Toc401912570)

[Stap 1 - Lifeline 10](#_Toc401912571)

[Stap 2 – Loops en vergelijkingen 12](#_Toc401912572)

[Componentendiagram 13](#_Toc401912573)

[Stap 1 - Componenten 13](#_Toc401912574)

[Stap 2 – Koppeling tussen componenten 13](#_Toc401912575)

[Stap 3 – Overige klassen 14](#_Toc401912576)

[Deploymentdiagram 15](#_Toc401912577)

[Stap 1 – Devices toevoegen 15](#_Toc401912578)

[Stap 2 – Componenten toevoegen 15](#_Toc401912579)

[Stap 3 – Onderlinge verbinding 15](#_Toc401912580)

# Inleiding

In deze handleiding zal met behulp van screenshots met duidelijke stappen beschreven worden hoe Visual Paradigm gebruikt kan worden voor het maken van de UML diagrammen:

Use case diagram

Klassendiagram

Sequentiediagram

Componentendiagram

Deploymentdiagram

In deze handleiding wordt gebruik gemaakt van Visual Paradigm 11.2. Als voorbeelden worden UML diagrammen voor een Kolonisten van Catan project gebruikt. Kolonisten van Catan is een bekend bordspel ontworpen door Klaus Teuber (<http://nl.wikipedia.org/wiki/Kolonisten_van_Catan>).



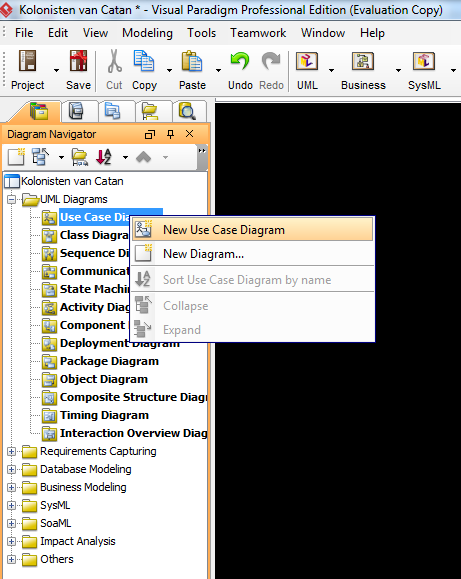
Figuur 1: Het bordspel Colonisten van Catan (Bron: http://www.bord-spel.com/spelregels/kolonisten-van-catan-spelregels/)

# Use case diagram

Use case diagrammen worden in Visual Paradigm onder UML diagrammen gevonden, net als alle andere UML diagrammen. In dit hoofdstuk wordt beschreven hoe een use case diagram getekend wordt en hoe een use case gevuld wordt in Visual Paradigm.

## Stap 1 – Diagram maken

Klik met de rechtermuisknop op ‘Use Case Diagram’ en dan op ‘New Use Case Diagram’ of klik met de rechtermuisknop op ‘UML diagrams’ en selecteer ‘Use Case Diagram’.



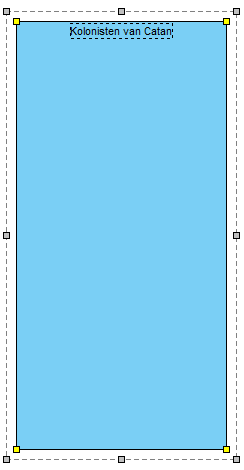
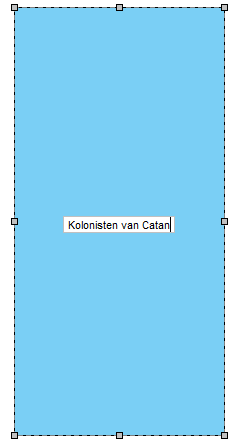
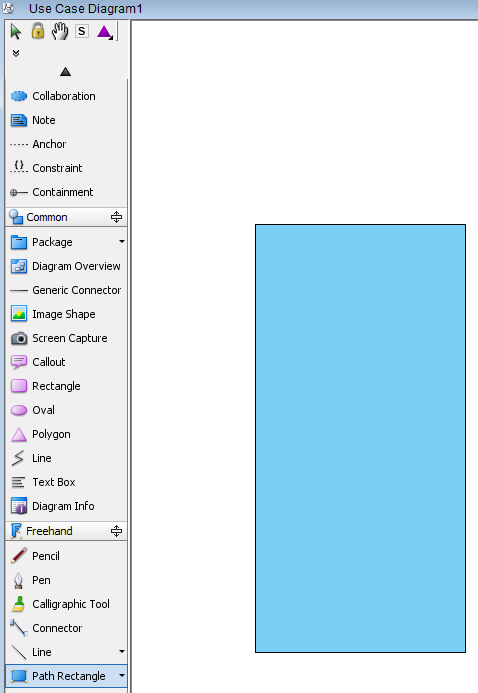
*Stap 1*

## Stap 2 – Diagram tekenen

Een lege use case diagram verschijnt op het scherm waar in getekend kan worden. Om aan te geven bij welk systeem use cases horen maken we eerst het vierkant met de naam van het systeem erin.

*Alle onderdelen die op een diagram geplaatst kunnen worden kan je slepen òf selecteren. Als een onderdeel dubbel geklikt wordt kun je dit blijven plaatsen met de linkermuisknop tot een ander onderdeel geselecteerd wordt.*

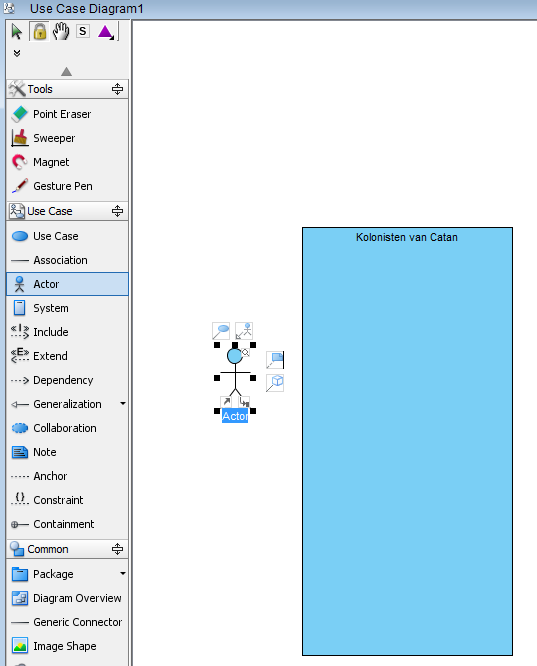
Scroll links in het diagram scherm naar beneden en plaats een ‘Path Rectangle’. Dubbelklik hier op en geeft het een naam. De tekst kan naar de bovenkant van het vierkant gesleept worden.



*Stap 2*

## Stap 3 – Actors toevoegen

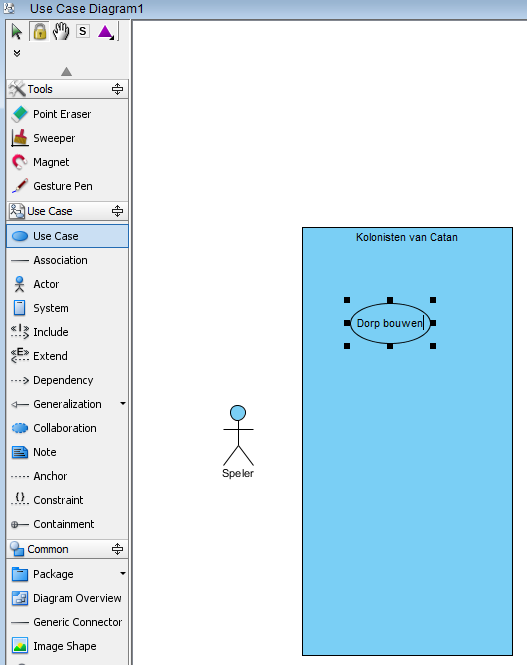
Plaats de actor(s), te vinden links in de lijst in de diagram. Het systeem vraagt direct een naam in te vullen.



*Stap 3*

## Stap 4 – Use cases toevoegen

Op dezelfde manier kunnen nu use cases toegevoegd worden.

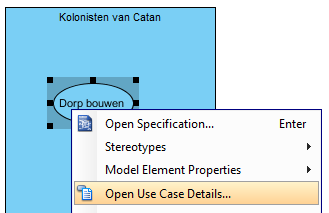


*Stap 4*

Maak tussen de actor en de use cases daarna ‘Associations’ (lijnen) aan.

## Stap 5 – Use cases vullen

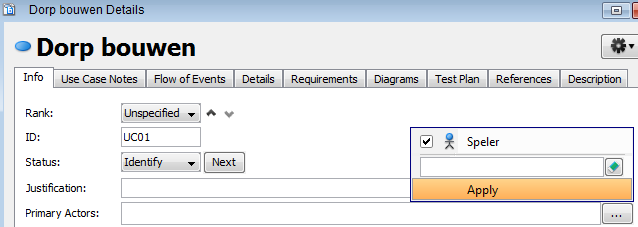
Het belangrijkste van de use case diagram is dat de use cases gevuld zijn met relevante informatie. Dit doen we door naar de ‘Use Case Details’ te gaan.



*Stap 5*

Details van de use case worden nu weergegeven op een scherm met verschillende tabs. De voor ons revelante tabs zijn ‘Info’, ‘Flow of events’ en ‘Details’.

Op het info tab vullen we de ‘use case ID’ en de actors in. Als associaties tussen de actor en de use case al aanwezig zijn zal de actor al bij de rol toegevoegd zijn. Door bij primary actors op de ‘...’ knop te drukken weergeeft Visual Paradigm een lijst van actors in het project ter selectie. Druk op apply als de actors gekozen zijn.



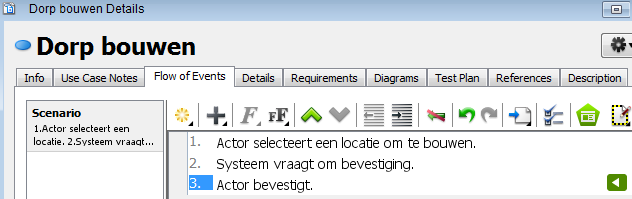
*Stap 5*

Op het ‘Details’ tab vullen we de assumptions, pre en post conditions in.



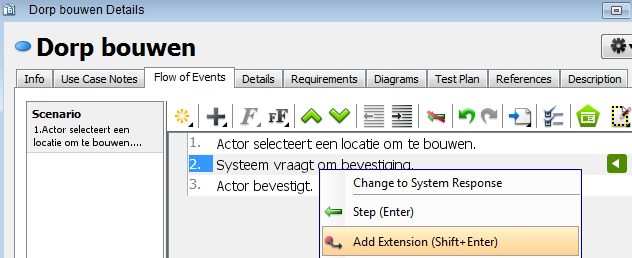
*Stap 5*

In het ‘Flow of events’ tab vullen we de stappen van de use case in.



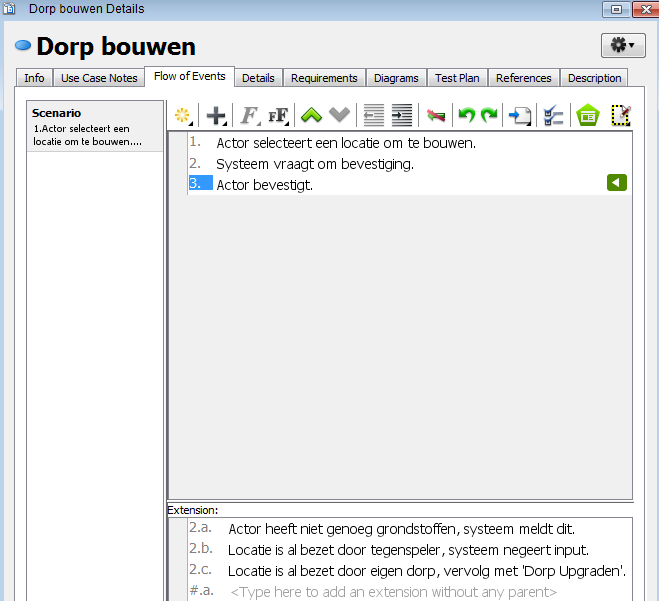
*Stap 5*

Voor uitzonderingen bij stappen gebruiken we extensions.

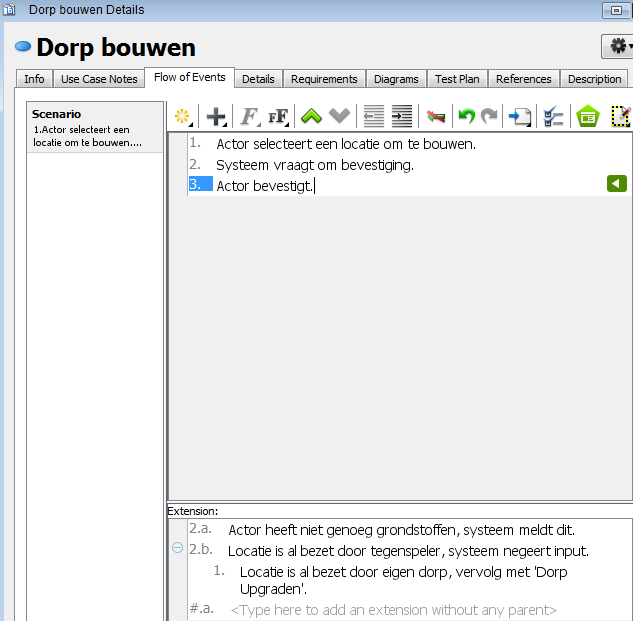


*Stap 5*

Klik voor elke uitzondering opnieuw op ‘Add Extension’. Op enter drukken in het extension scherm zal een extra regel toevoegen, maar dit zal een stap zijn die onderdeel is van een uitzondering terwijl we hier niet dieper op in willen gaan.

 **Goed**

*Stap 5*

** **Fout**

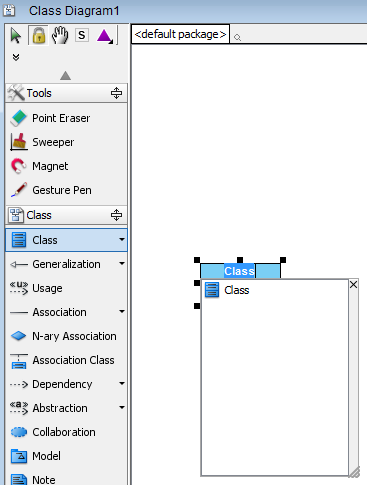
*Stap 5*

# Klassendiagram

In dit hoofdstuk wordt beschreven hoe in Visual Paradigm een klassendiagram gemaakt kan worden.

## Stap 1 – Klasse toevoegen

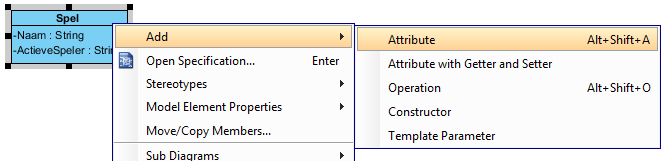
Na een nieuw ‘Class Diagram’ gemaakt te hebben kunnen er klassen op het scherm gesleept worden. Visual Paradigm vraagt meteen om een naam voor deze klasse.



*Stap 1*

## Stap 2 – Veld toevoegen

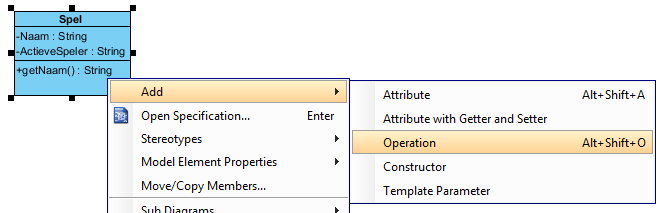
Door met de rechtermuisknop op de klasse te klikken kunnen er velden toegevoegd door op ‘Add’ te gaan staan en vervolgens op ‘Attribute’ te klikken. Dit kan eventueel met automatisch gegenereerde getter/setter.



*Stap 2*

## Stap 3 – Methode toevoegen

Op dezelfde manier kan een methode toegevoegd worden door op ‘Operation’ te klikken.

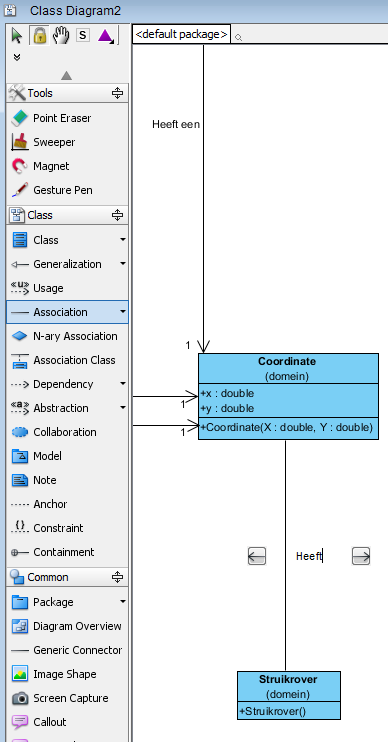


*Stap 3*

Het is ook mogelijk een constructor toe te voegen. Het voordeel van de constructor knop ten opzichte van de operation knop is dat de naam van de klasse automatisch ingevuld wordt.

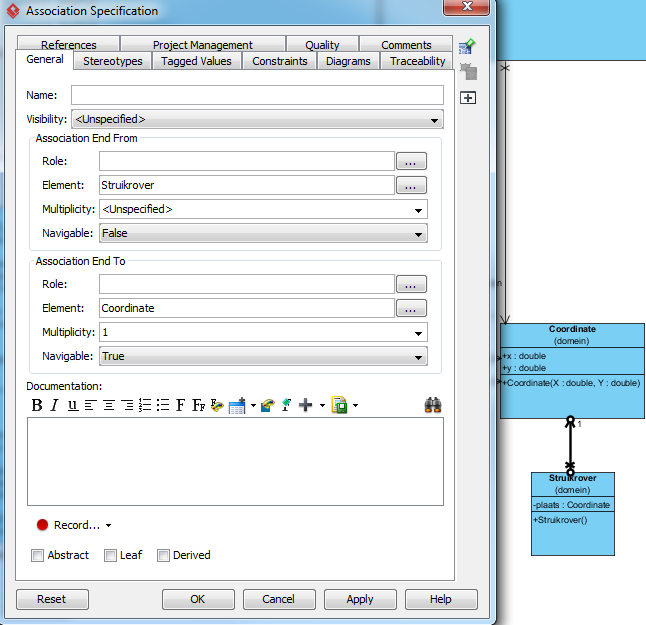
## Stap 4 – Multipliciteit en compositiesymbolen

Het aangeven van multipliciteit kan gedaan worden door een associatie tussen twee klassen aan te maken en deze aan te passen. Bij een associatie tussen twee klassen wordt een beschrijving verwacht, Visual Paradigm geeft meteen de mogelijkheid dit in te vullen.



*Stap 4*

Klik nu met de rechtermuisknop op de ‘Assocation’ en ga naar ‘Open Specification’. Er verschijnt nu een scherm waar alle relevante informatie ingevuld kan worden. Navigatie kan worden aangegeven door aan beide uiteinden van de associatie ‘Navigable’ op ‘True’ of ‘False’ te zetten. Als ‘Navigable’ aan een uiteinde op ‘True’ staat en aan het andere uiteinde op ‘False’, dan is er een pijl te zien aan de True-zijde en een kruis aan de False-zijde. Als ‘Navigable’ aan beide uiteinden op ‘True’ staat zijn er geen pijlen te zien. Multipliciteit kan aan beide uiteinden ingesteld worden onder ‘Multiplicity’.



Een compositiesymbool kan toegevoegd worden door met de rechtermuisknop op een associatie te klikken, naar rol A of B te gaan (klasse wordt aangegeven), daarna naar ‘Aggregation kind’ te gaan en hier ‘Composited’ aan te klikken.

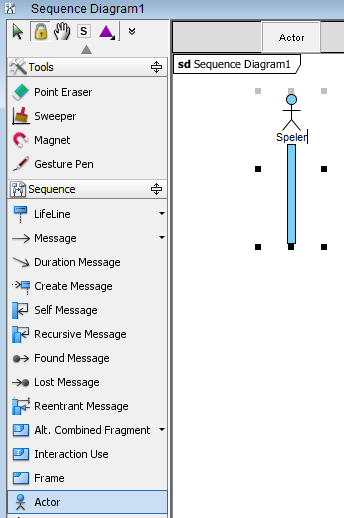
# Sequentiediagram

In dit hoofdstuk wordt beschreven hoe in Visual Paradigm een sequentiediagram gemaakt kan worden.

## Stap 1 - Lifeline

Maak eerst een nieuwe ‘Sequence Diagram’ aan.

Het eerste wat we toevoegen op de sequentiediagram is de actor die de actie uitvoert. Visual Paradigm zal meteen de naam voor deze actor vragen en de eerste lifeline aanmaken.

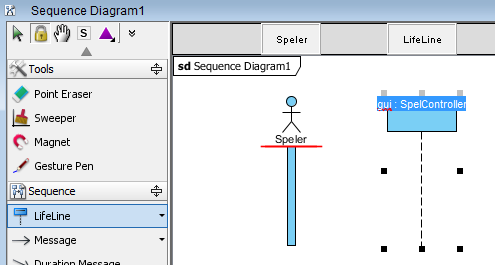
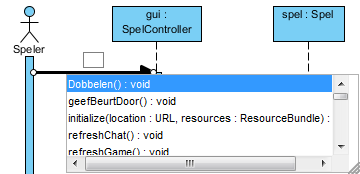


*Stap 1*

De voorbeeld sequentiediagram zal het gooien van de dobbelsteen en uitdelen van grondstoffen beschrijven en grotendeels over de methode DeelGrondstoffenUit in de Spel klasse gaan. (Zie figuur 2)

*De eerste klasse die de actor aanspreekt is de GUI klasse. Omdat het diagram hieronder een domeinmodel is staan sommige klassen, waaronder de ‘SpelController’ (GUI klasse) er niet in. Deze klasse komt wel in het sequentiediagram voor omdat de actie hier begint.*

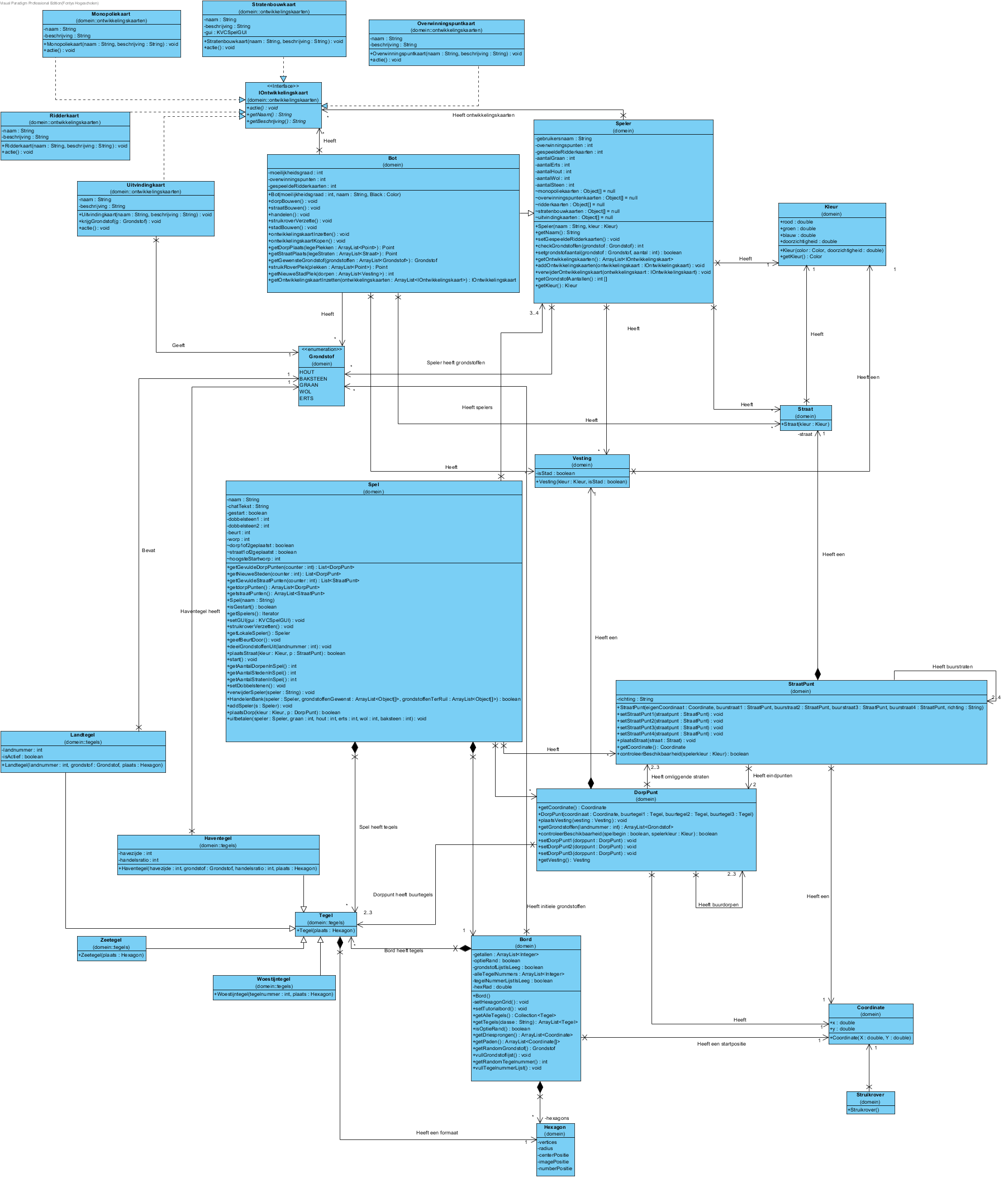
Plaats nu een nieuwe lifeline voor de eerste klasse die aangesproken wordt bij de actie die je beschrijft.

*Stap 1*

In deze lifeline staat ‘gui : SpelController’. De naam van het object wat aangesproken wordt is gui, de klasse is SpelController. Als de klasse bekend is in Visual Paradigm bijvoorbeeld in een klassediagram, dan geeft Visual Paradigm automatisch suggesties voor methodes bij het maken van een ‘Message’. Een message is een aanroep van een methode op een (ander) object, in een sequentiediagram vertegenwoordigd als lifeline. Maak een message door links ‘Message’ te selecteren of door met de muis over een lifeline te gaan en de pijl naar de volgende lifeline te trekken. Let op, return messages kunnen alleen gevonden worden als dropdown onder ‘Message’.

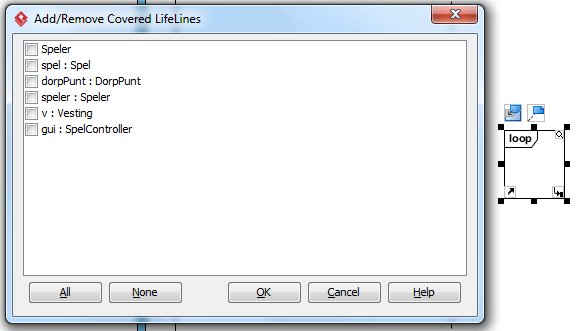
*Als je de instellingen van de nummering van messages wil veranderen kan dit door met de rechtermuisknop op de achtergrond van de diagram te klikken (let op dat er niets geselecteerd is), ga naar ‘Sequence Number’ en kies ‘Single Level’.*



Figuur 2: Klassendiagram Kolonisten van Catan project

## Stap 2 – Loops en vergelijkingen

Door links in het scherm op het dropdown tekentje te klikken bij ‘Alt combined fragment’ kan een loop toegevoegd worden. Visual Paradigm zal vragen welke lifelines betrokken zijn in de loop. De loop wordt automatisch over de lifelines heen getrokken, maar zal bijgesteld moeten worden naarmate de diagram groter wordt.

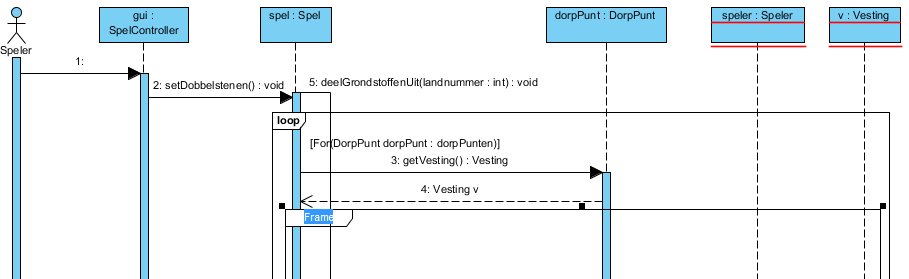


*Stap 2*

Wanneer je daarna dubbel klikt op ‘Loop’ binnen dit vak kan beschreven worden waar doorheen geloopt word.

In de voorbeeldsituatie gaat het spel na het bepalen van het nieuwe getal van de dobbelstenen grondstoffen uitdelen aan de hand van nummering van tegels op het bord. De ‘deelGrondstoffenUit’ methode in de spel klasse wordt aangeroepen met het nieuw gedobbelde getal en tijdens de methode wordt door alle punten geloopt om eventueel grondstoffen aan dorpen rond de tegel uit te delen.

Een if statement kunnen we maken met een ‘Frame’. Nadat een frame toegevoegd wordt kan, wanneer het als if statement dient, de vergelijking hier in gezet worden. Er staat daarna bijvoorbeeld **‘sd**v!=null’. Klik met de rechtermuisknop op het frame en ga naar ‘Open Specification’. Verander hier het type naar ‘Assert’.



*Stap 2*

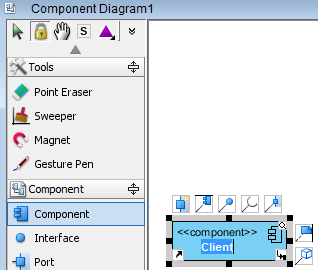
# Componentendiagram

In dit hoofdstuk wordt beschreven hoe in Visual Paradigm een componentendiagram gemaakt kan worden.

Een componentendiagram biedt overzicht over de klassen waaruit componenten binnen een applicatie bestaan. In het Kolonisten van Catan project zijn twee componenten namelijk de client en de server.

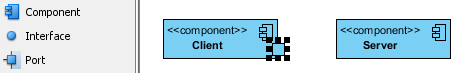
## Stap 1 - Componenten

Maak een ‘Component Diagram’ aan. Plaats nu een component op het diagram en geef het een naam.



*Stap 1*

Voeg afhankelijk van het aantal gebruikte of aangeboden interfaces ports aan de componenten toe.



*Stap 1*

Door op de ports te klikken kunnen eigenschappen zoals aangeboden en gebruikte interfaces aangepast worden. De client gebruikt in dit geval de Ilobby interface aangeboden door de server voor onderlinge communicatie. Klik op ‘Realization’ voor het aanbieden van een interface en op ‘Usage’ voor het gebruik van een interface.



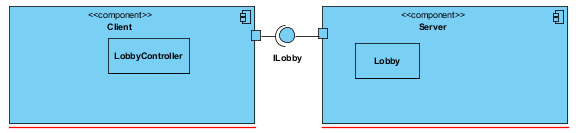


*Stap 1*

## Stap 2 – Koppeling tussen componenten

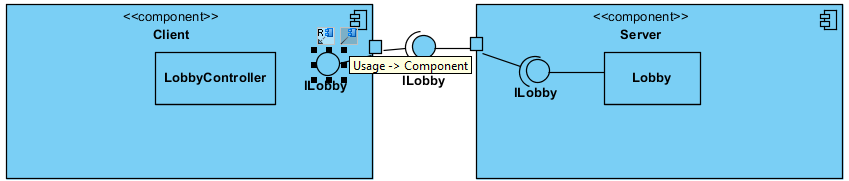
Bepaal per component welke klassen hierbij horen (tip: doe een component tegelijk om tijd te besparen). Kopieer deze klassen (CTRL+C) in het klassendiagram en plak ze in het componentendiagram. De klassen zijn nu allemaal vrij groot, daarom gaan we de velden en methodes verbergen. Selecteer alle klassen, klik er met de rechtermuisknop op, ga naar presentation options, daarna naar attributes en klik hier op hide all. Doe hetzelfde voor methodes en eventueel enumerators.

Plaats nu eerst de klassen die de componenten gebruiken voor onderlinge communicatie in de bijbehorende componenten.



*Stap 2*

Verbind deze klassen nu met de ports door op een port te klikken en maak duidelijk of de interface van buitenaf aangeboden wordt of geaccepteerd wordt in deze situatie. Wanneer een interface geaccepteerd wordt, zoals Ilobby bij de client, kies je bij de poort ‘Realization’. Er ontstaat binnen de component nog een ‘Realization’ symbooltje binnen de component. Selecteer dit symbool, klik hier nogmaals op ‘Usage’ en koppel het met de klasse.



*Stap 2*

Geef ook deze ‘Realizations’ namen.

## Stap 3 – Overige klassen

Plaats nu de overige klassen in de componenten afhankelijk van het gebruik binnen de code.

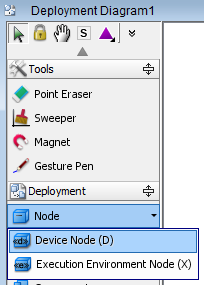
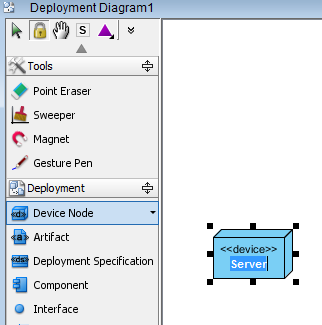
# Deploymentdiagram

In dit hoofdstuk wordt beschreven hoe in Visual Paradigm een deploymentdiagram gemaakt kan worden.

Een deploymentdiagram weergeeft hoe de software componenten gebruikt (kunnen) gaan worden in combinatie met hardware en hoe de componenten gaan communiceren.

## Stap 1 – Devices toevoegen

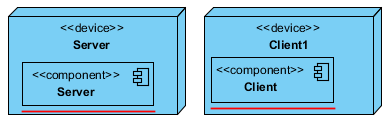
Maak in Visual Paradigm een ‘deployment diagram’. Gebruik nu het dropdown menu onder ‘Node’ en selecteer ‘Device Node’. Plaats voor alle benodigde of gewenste hardware een device op de diagram en voeg herkenbare namen toe.

*Stap 1*

## Stap 2 – Componenten toevoegen

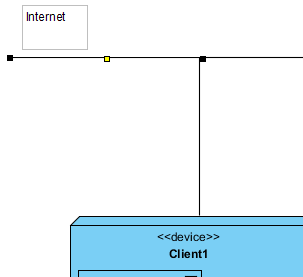
Kopieer alle componenten uit de componentendiagram naar de deploymentdiagram. Verwijder de klassen in de componenten en plaats ze op de bijbehorende hardware.



*Stap 2*

## Stap 3 – Onderlinge verbinding

Plaats voor het weergeven van een netwerk of internetverbinding ‘Lines’ gevonden onder ‘Common’. Gebruik ook de ‘Text Box’ om het type verbinding te beschrijven.



*Stap 3*