|  |
| --- |
| Advanced Programming, Advanced software engineering |
| Music matching voor Dare2Date |
| Software architectuur document |

|  |
| --- |
| Hanna Hermsen 474296 Jeroen Jansen 483264  8-5-2014  Versie 0.1 |

|  |  |  |  |
| --- | --- | --- | --- |
| Versie | Naam | Datum | Wijziging |
| V0.1 | Jeroen & Hanna | 8-5-2014 | Begin document |
|  |  | 8-5-2014 |  |

Inhoudsopgave

[1. Inleiding 4](#_Toc387315631)

[2. Architecturale eisen 5](#_Toc387315632)

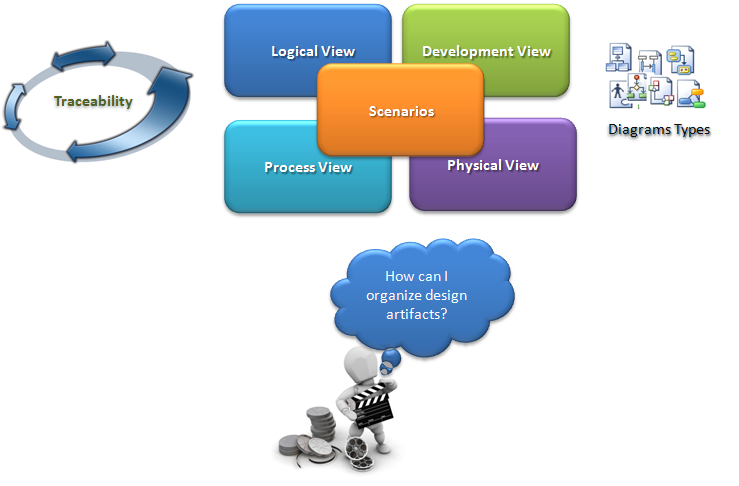
[2.1. Niet functionele eisen 5](#_Toc387315633)

[2.2. Use case view (functionele requirements) 5](#_Toc387315634)

# Inleiding

Voor u ligt het Software Architectuur Document (SAD) van Jeroen Jansen en Hanna Hermsen.  
Het bevat een uitgebreide architecturale kijk op de music matching functionaliteit die wij bouwen voor de dating website Dare2Date.   
In principe is het document bedoelt voor Rody Middelkoop, docent van het vak Advanced Software Engineering van de minor Advanced Programming die wij nu volgen. Desalniettemin is dit document ook geschikt voor andere technische stakeholders van Dare2Date.

Dare2Date is een (nu nog fictieve) website waar 30+'ers hun ideale partner kunnen ontmoeten. Als toevoeging op het matching gedeelte zullen wij een music matching functionaliteit ontwerpen en implementeren die gebruik maakt van last.fm. Dit is een website die bijhoudt wat voor muziek iemand luistert en bevat ook sociale aspecten zoals een berichtenservice en groepen. Ook kunnen concerten/festivals worden toegevoegd.



Figuur : 4 + 1 Model

Bij het ontwerpen maken we gebruik van het 4 + 1 model, waarmee verschillende aspecten van het systeem worden toegelicht. Dit model stelt de verschillende belanghebbenden in staat vanuit hun eigen perspectief de invloed van de gekozen architectuur te bepalen. De Process View (communicatie van processen) is niet als los hoofdstuk uitgewerkt maar ondergebracht bij de hoofdstukken 3.3 en 5.

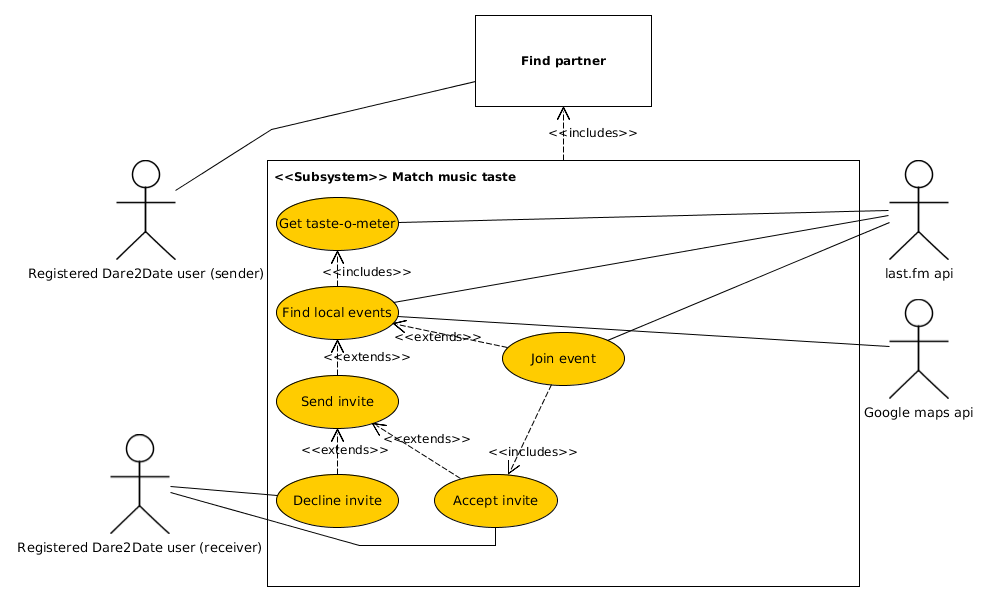
# Architecturale eisen

Deze sectie beschrijft de software eisen die voor het ontwikkelen van de software architectuur van belang zijn.

## Niet functionele eisen

|  |  |  |
| --- | --- | --- |
| Naam | Architecturele relevantie | Paragraaf |
| Systeem onafhankelijkheid | Er zal ontwikkeld worden in java. |  |
| Ontwikkeltools | Voor de ontwikkeling van de software zal er gebruik worden gemaakt van de Netbeans IDE versie 7.0.1. |  |
| Versiebeheer | Voor versiebeheer van documenten en code zal er gebruik worden gemaakt van Git reposetories. |  |

## Use case view (functionele requirements)

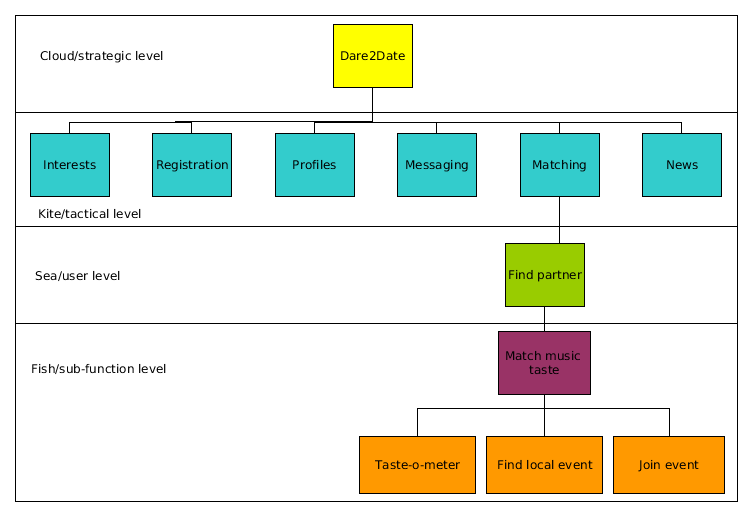


Figuur : use case matching

# Logical view

Hier wordt de architectureel significante logische opbouw van het systeem beschreven. Denk hierbij aan de decompositie in lagen en deelsystemen. Ook de manier waarop Use Cases, rekening houdend met deze logische decompositie, technisch worden vertaald naar Use Case Realizations worden hier beschreven.

## 3.1 Lagen & deelsystemen

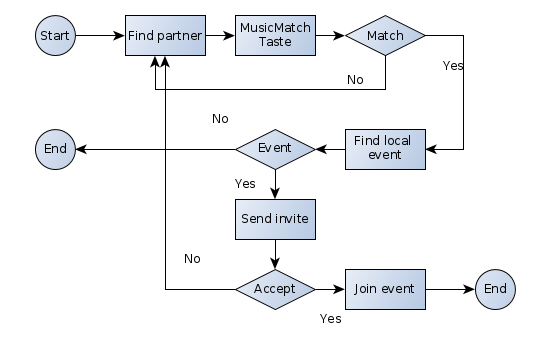


## 

## 3.2 Use Case Realizations

Hier wordt geillusteerd op welke wijze waarop de eerder genoemde Use Case Specifications technisch vertaald worden naar Use Case Realizations.

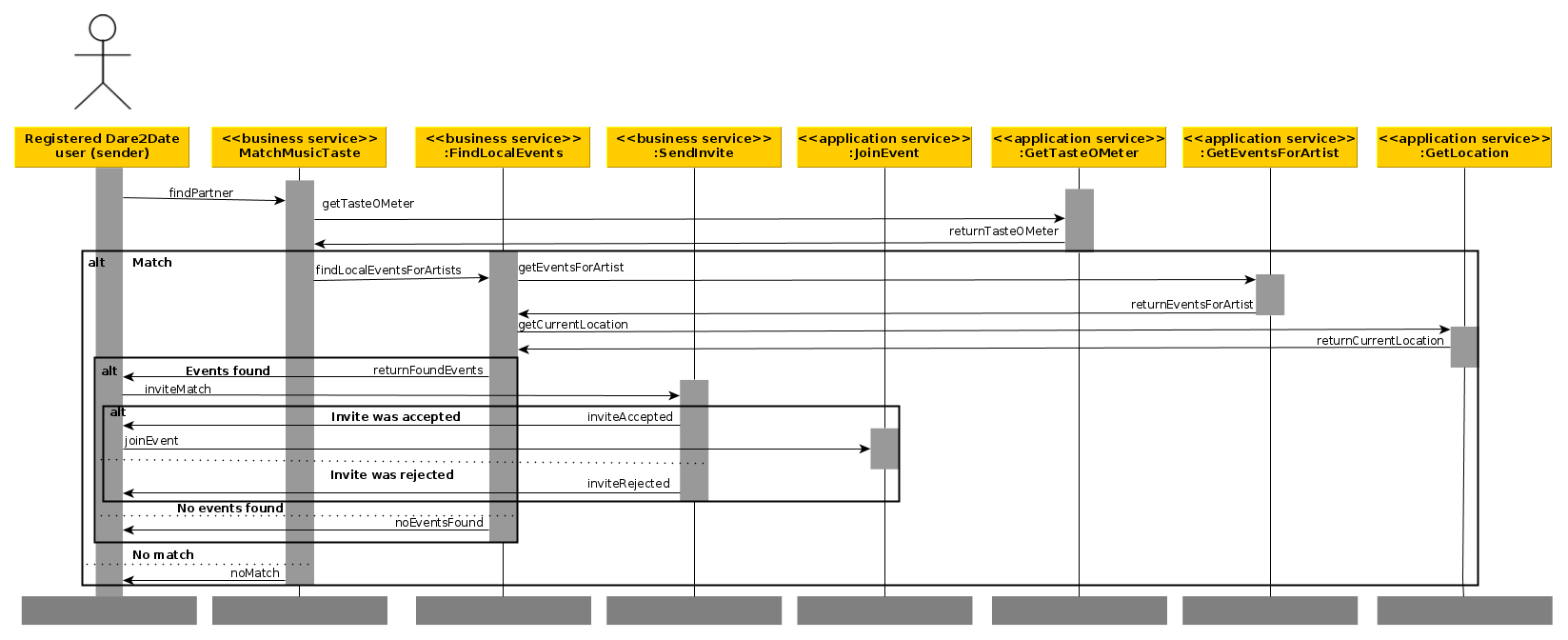
Flow diagram matching service



## 

High level sequence diagram matching service

# High level diagram.png

Service seguence diagram

# Implementation view