

EMS30 Introductie week 5 t/m 8

Leerdoelen EMS30.

Je leert in week **1 t/m 4**:

- hoe je een betere **C**-programmeur wordt.

Je leert in week **5 t/m 8**:

- de basisbeginselen van objectgeoriënteerd programmeren in **C++**;
- de basisbeginselen van objectgeoriënteerd ontwerpen met **UML**.

Leerdoelen EMS30 week 5 t/m 8

#	Niveau	Weging	De student is in staat om ...
4	C	30%	... gebruik te maken van objectgeoriënteerde programmeertechnieken in C++ zoals encapsulation, templates, overerving en polymorfisme zodanig dat de student programmatuur kan ontwikkelen die uitbreidbaar, aanpasbaar en herbruikbaar is.
5	C	10%	... op grond van zijn/haar kennis van de big-O-notatie effectief gebruik te maken van de standaard in C++ aanwezige datastructuren en algoritmen.
6	C	10%	... een objectgeoriënteerd ontwerp te maken van een programma in UML door middel van use cases, klasse-, sequentie, toestands- en activiteitendiagrammen.

Objectgeoriënteerd Programmeren in C++.

- responsibility driven design (ontwerpen uitgangspunt van verantwoordelijkheden).
- information hiding (het afschermen van informatie door middel van het scheiden van interface en implementatie).
- abstractie (het afschermen van complexiteit door middel van het scheiden van interface en implementatie).
- inheritance (het mogelijk maken van hergebruik van code, ... is een ... in plaats van ... heeft een ...)
- polymorfisme (hergebruik mogelijk gemaakt door dynamic binding)

Objectgeoriënteerd Ontwerpen met UML.

- klasse- en objectdiagram
- use-case diagram.
- sequence- en communicatiediagram.
- toestands- en activiteitendiagram.

Herbruikbaarheid
Aanpasbaarheid
Uitbreidbaarheid

Toetsing en studielast EMS30

Toets	Leerdoelen	Weging	Deadline
Opdracht 1	1, 2 en 3	50 %	Lesweek 4, zondag 23.59 uur
Opdracht 2	4, 5 en 6	50 %	Lesweek T4, zondag 23.59 uur

Je werkt in **tweetallen** samen m.b.v. git (*je krijgt van ons een repository*) en levert je werk ook in op dat repository.

EMS30 levert je **6** studiepunten op dat betekent dat je $6/15 = 2/5 =$ **2 dagen / week** aan EMS30 dient te besteden.

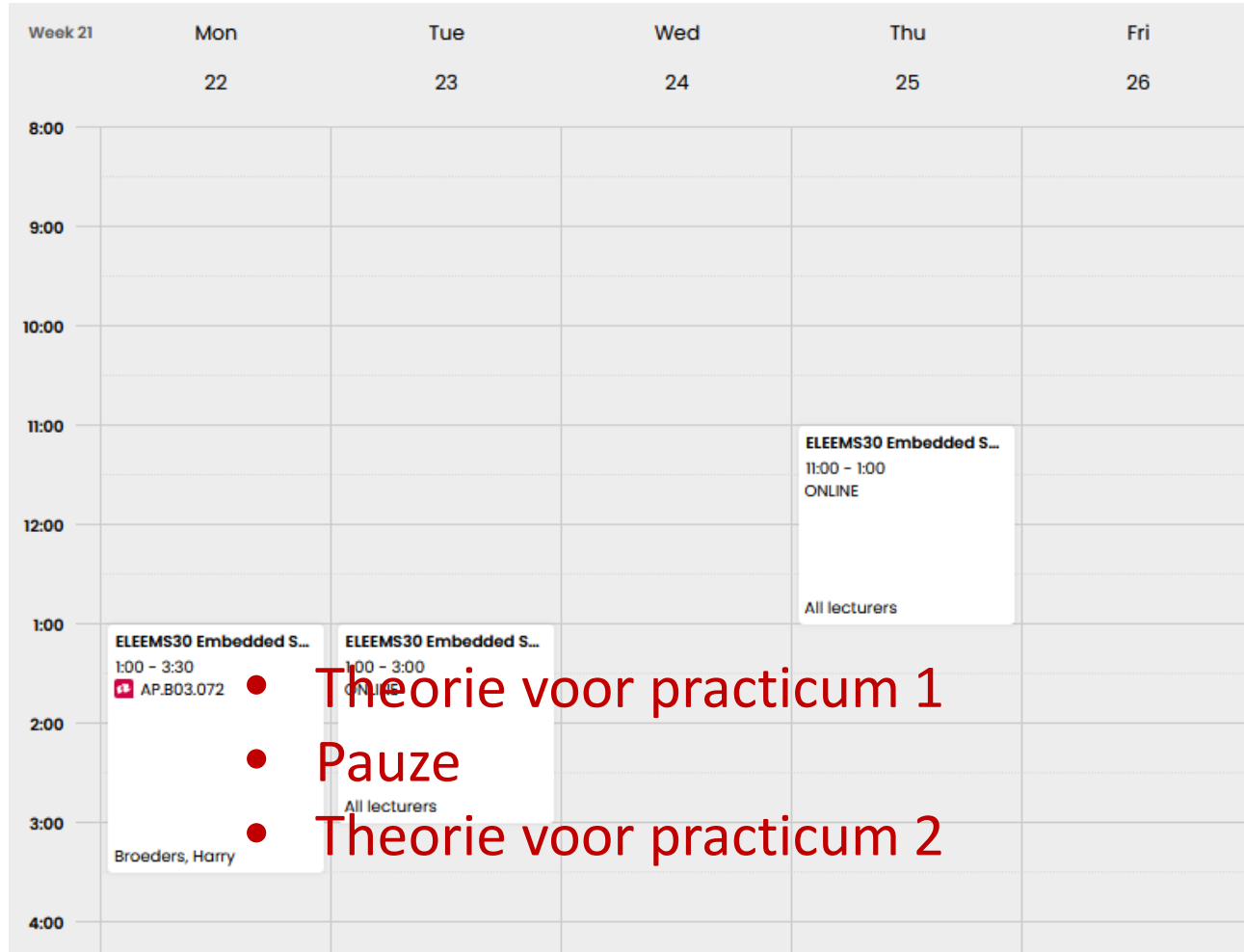
Planning week 5 t/m HT4

EMBEDDED SYSTEMS

Week	Werkvorm	Beschrijving
4.5	Les	1: Van C naar C++, 2: User-defined Data Type in C++
4.6	Les	1: Templates, 2: Overerving en polymorfisme
4.7	Les	1: Datastructuren, 2: Algoritmen, performance en big-O-notatie
4.8	Les	1: UML use cases en klassediagram, 2: UML sequentie-, toestands- en activiteitendiagram
T4	Opdracht 2	Deadline zondag 23.59 uur
HT4	Herkansing opdracht 1 en 2	Deadline zondag 23.59 uur

Weekplanning week 5 t/m 8

Bijvoorbeeld:
Week 5




- Theorie voor practicum 1
- Pauze
- Theorie voor practicum 2

Weekplanning week 5 t/m 8

EMBEDDED SYSTEMS

Let op:
Week 6

Tue
30
ELEEMS30 Embedd... 9:00 - 11:30  PH.02.51
Broeders, Harry

Pieter de Hoochweg



Studiemateriaal vind je op de wiki:

EMBEDDED SYSTEMS

HR_ELEKTRO / Embedded / EMS30

Wiki

Create page

Clone wiki

EMS30 / Home

View

History

Edit

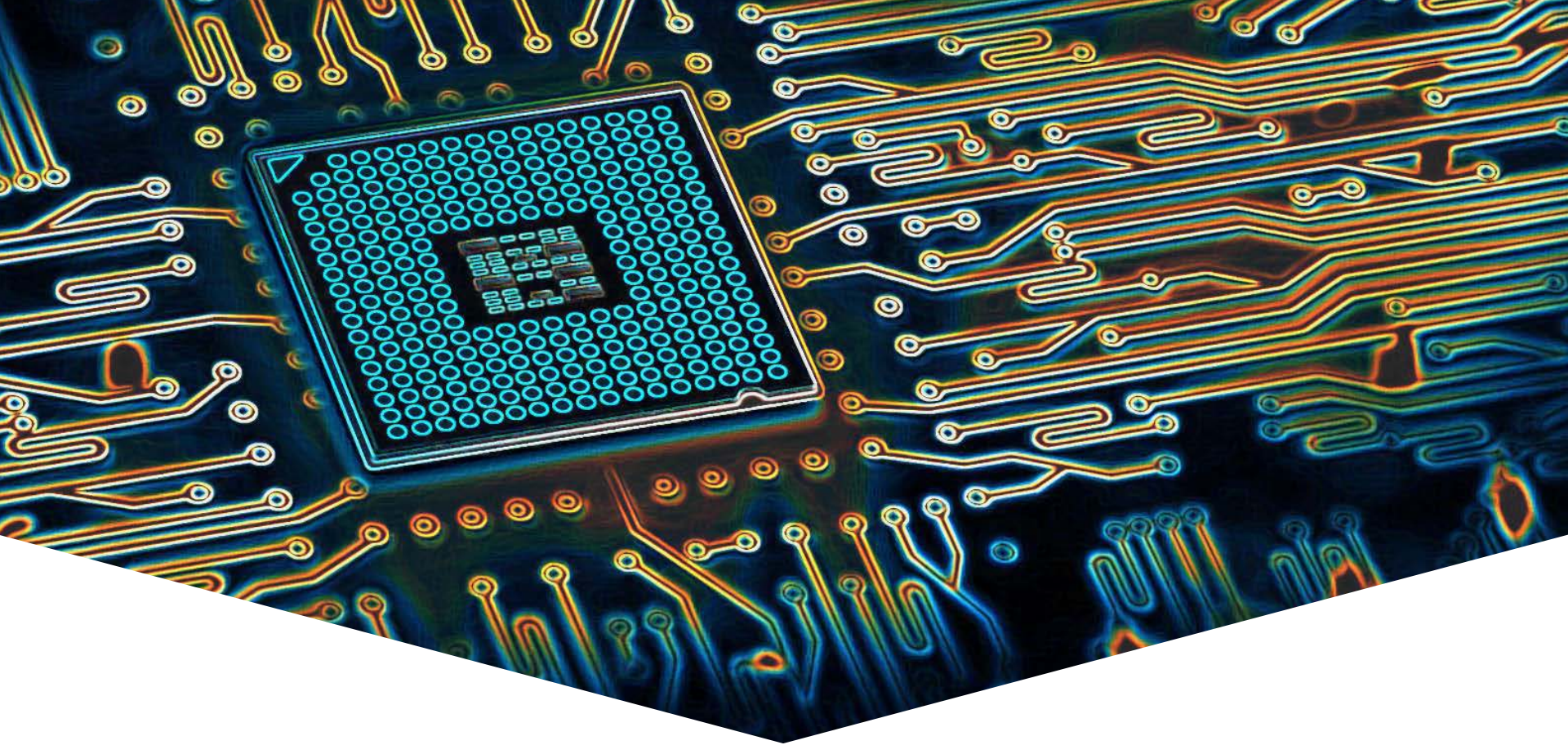
EMS30 - Embedded Systems 3

Studiemateriaal week 5 t/m 8

Harry Broeders. Dictaat Objectgeoriënteerd Programmeren in C++. 1ste ed. Hogeschool Rotterdam, 2020. URL: https://bitbucket.org/HR_ELEKTRO/cppprog. Dit dictaat wordt gebruikt bij de introductie in C++.

Joost Vennekens. Objectgericht ontwerpen. 1ste ed. De Nayer instituut, 2011. URL: <https://people.cs.kuleuven.be/%7Ejoost.vennekens/DN/OO-UML-cursus.pdf>. Dit dictaat wordt gebruikt bij de introductie in UML.

9



EMS30 Week 5 Les 1: Van C naar C++

Leerdoelen week 5 les 1.

In het **theorie**deel leer je wat:

- het verschil is tussen **objectgeoriënteerd** programmeren in **C++** in plaats van **gestructureerd** programmeren in C.

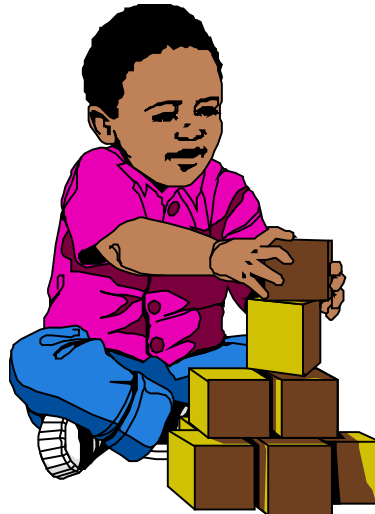
In het **practicum**deel leer je hoe:

- in C++ iets naar een **output**device (bijvoorbeeld je beeldscherm) kunt sturen en iets van een **input**device (bijvoorbeeld je toetsenbord) kunt inlezen;
- in C++ standaard classes (bijvoorbeeld **string**) kunt gebruiken.

Een stapje verder...

...met programmeren en ontwerpen.

- Van **gestructureerd** naar **objectgeoriënteerd**.
- **C++** is een uitbreiding op **C**:
 - alles wat in C kan, kan ook in C++.
 - veel wat in C kan, kan in C++ beter (I/O, struct, array, c-string enz).



C is het fundament
voor C++.
Zie inleiding dictaat.

Een stapje verder...

EMBEDDED SYSTEMS

...met programmeren en ontwerpen.

Waarom?

Dat is een lang verhaal...

Gestructureerde programmeertalen:

- ±1945 assembler, ±1957 FORTRAN, ±1960 Algol60, ±1972 C (1989 std ANSI C)

Software crisis:

- Software niet op tijd geleverd.
- Software duurder dan afgesproken.
- Software niet foutloos.

Idee voor de oplossing:

- Herbruikbare software componenten maken.
- Deze componenten gebruiken bij maken van grote uitbreidbare en onderhoudbare software systemen.

Objectgeoriënteerde programmeertalen:

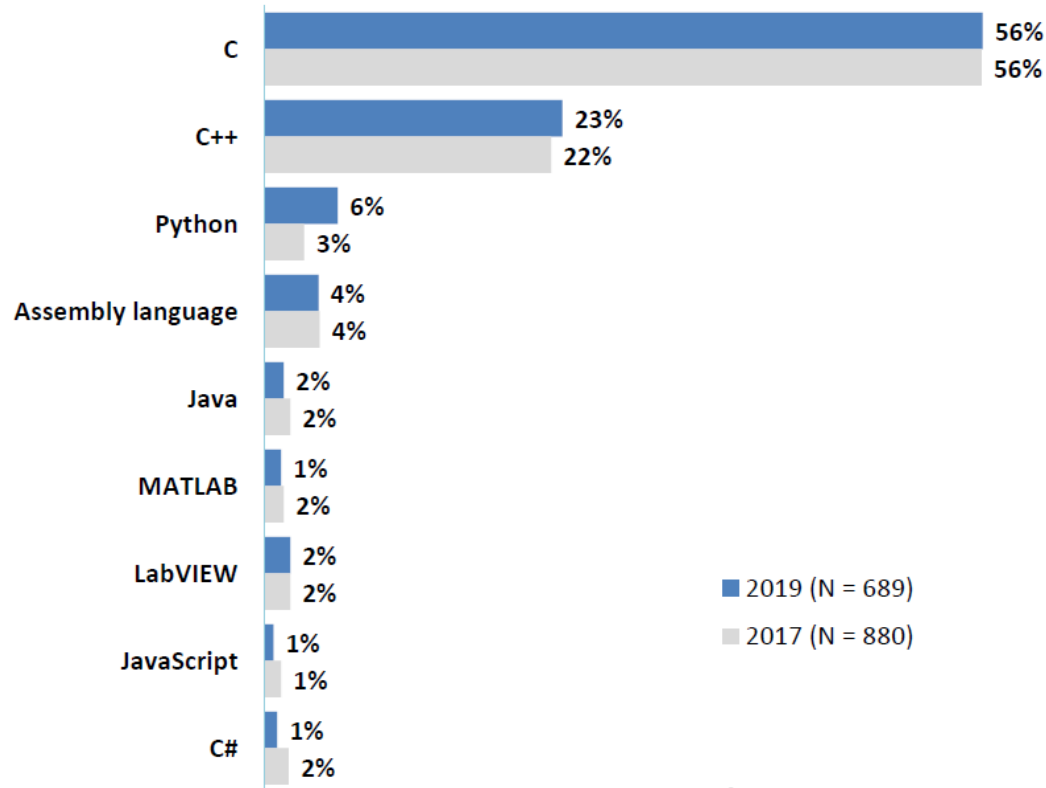
- ±1967 Simula, ±1976 Smalltalk, ±1983 C++ (1998 std C++), ±1995 Java (Sun/Oracle), ±2000 C# (Microsoft), ±2014 Swift (Apple).



Bron: <https://www.publicdomainpictures.net/nl/view-image.php?image=6884&picture=kind-met-laptop>

... maar ook een kort verhaal

My *current* embedded project is programmed mostly in:



Bjarne Stroustrup:

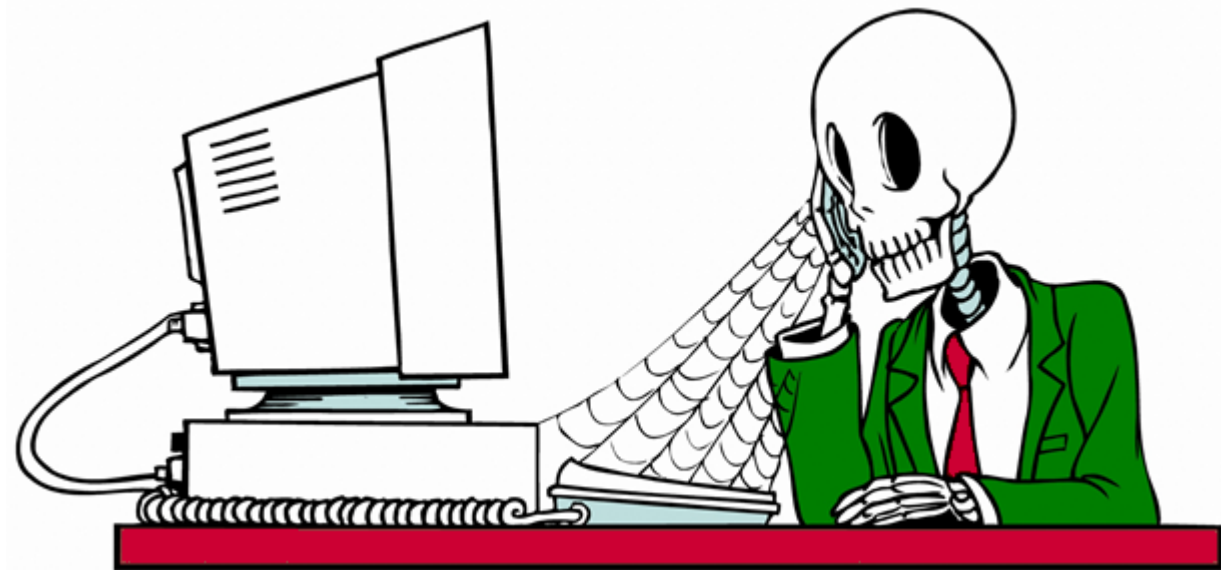
“C++ is a programming language that:

- is a **better** C,
- supports **data abstraction**,
- supports **object-oriented** programming,
- and supports **generic** programming.”



Bron: <https://www.watertechnology.com/people/4005396/coding-in-the-trenches-bjarne-stroustrup-morgan-stanley>

Welke software hoeft **nooit uitgebreid of veranderd** te worden?



Bron: <https://www.needpix.com/photo/1532331/skeleton-cobweb-spiderweb-computer-processor-pc-tun-laugh-cartoon>

Software die niemand (meer) gebruikt.

Objectgeoriënteerd programmeren is een **nieuwe** manier van denken ...

... over hoe we **code** en **data** in een computerprogramma kunnen **structureren**.

Programmeer **paradigma's**:

- **imperative** (C, Pascal).
- **functional** (LISP, Haskell).
- **logic** (Prolog).
- **object oriented** (C++, Java, C#).
- **generic** (ADA, C++).

De manier van probleem oplossen die gebruikt wordt bij de objectgeoriënteerde programmeertalen en ontwerpmethoden lijkt vaak op de manier van probleem oplossen die mensen in het **dagelijks leven** ook gebruiken.

Ik wil mijn oma een bosje bloemen sturen.



- object
- message + arguments
- receiver's responsibility
- method (information hiding)

<http://www.pngall.com/?p=33394>

Wat is het verschil tussen een **message** en een **functie**?

- Een message heeft een bepaalde **receiver**.
- De **method** die bij de message hoort is **afhankelijk** van de receiver.
- De receiver van een message kan ook tijdens **run-time** worden bepaald.

Dynamic binding between the message (function name) and method (code).

Waarom weet ik zoveel van mijn bloemiste?



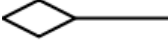

- class
- instance (object)
- hierarchy
- inheritance (base and derived)

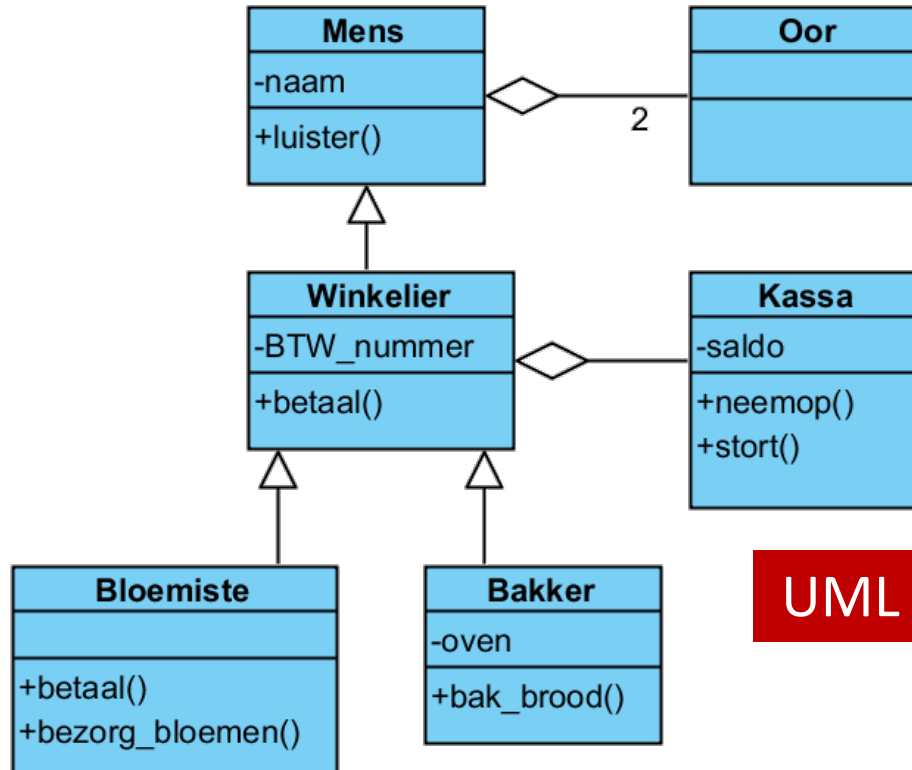
Bron: https://pages.rediff.com/photos/preview/phoolwala/5987/4/florist_girl

- Zoek in **class** van het **receiver** object.
- Als daar geen method is zoek dan in de **base class** van de **class** van het receiver object.
- Als daar geen method is zoek dan in de **base class** van de **base class** van de **class** van het receiver object.
- Enzovoort.

Een method uit de base class kan **overriden** worden door een method in een derived class.

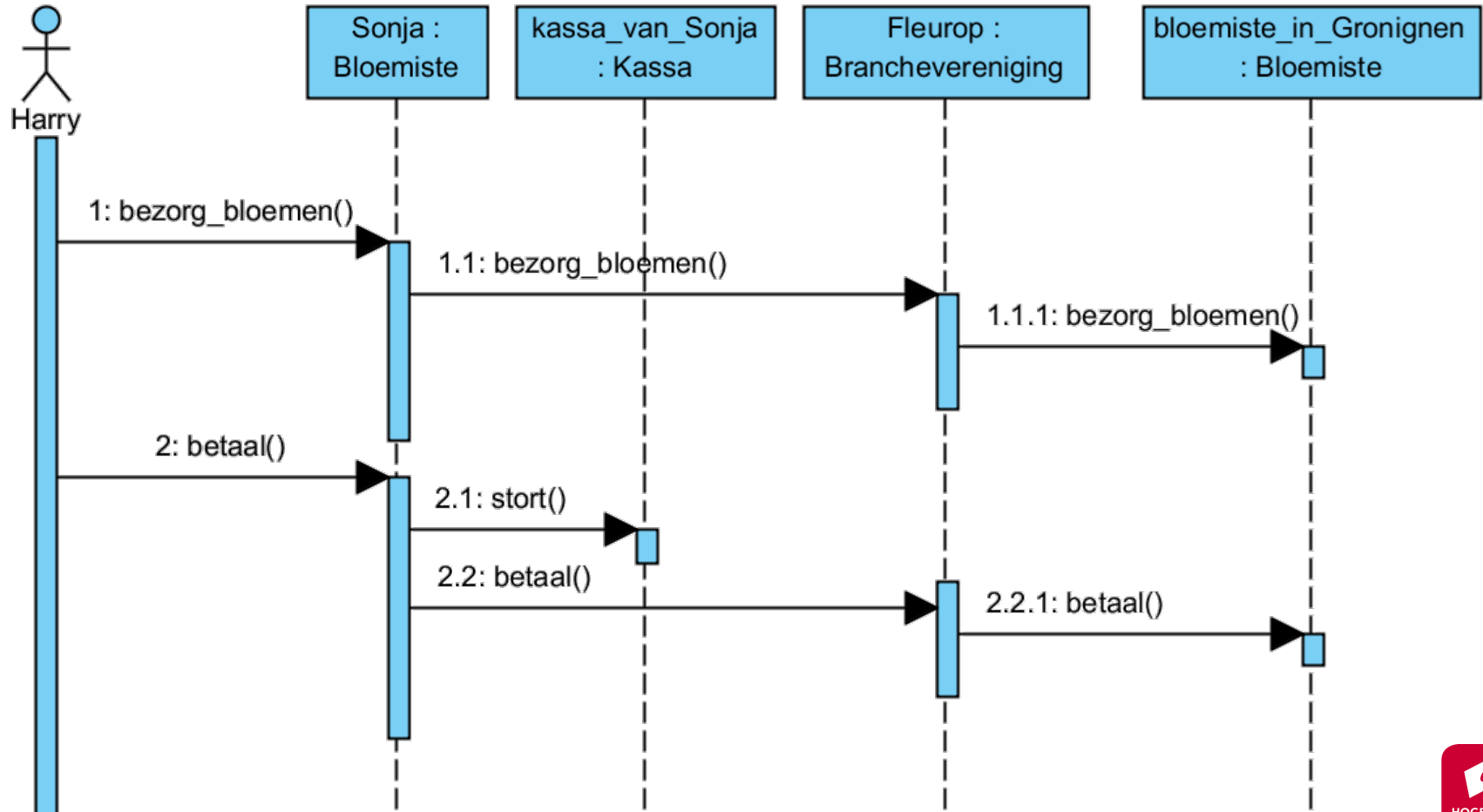
Verband tussen classes

- **Aggregation** = heeft een (of meer) 
- **Inheritance** = is een (speciaal soort) 



UML klassendiagram

UML sequentiediagram



Steeds meer abstractie

Functions.

- Avoid duplicating code.
- Information hiding.

Modules.

- Data and information hiding.

User-defined data types (UDT's).

- Instantiation.

Generic functions en generic UDT's.

- Templates.

Classes.

- Messages.
- Inheritance.
- Polymorphism.

Doel van object oriëntatie

- Construeren van **herbruikbare software componenten**.
- Gebruiken van deze componenten bij het construeren van grote **aanpasbare** en **uitbreidbare** systemen.



Software IC

Huiswerk:
Bestudeer hoofdstuk 2 t/m 2.1
van het dictaat.

Volgende les...

Definiëren van een **UDT** in C++ d.m.v. een **class**.

Aan de slag!

Aan de slag met [Opdrachten Week 5 Les 1.pdf](#)



<https://www.modekoninginmaxima.nl/mode-maxima-mode-algemeen/nieuwjaarsontvangsten-2020-dag-1>