Applicatie-Architecturen

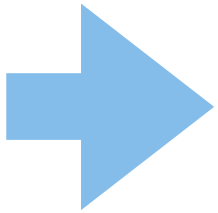
joost.vennekens@kuleuven.be

http://www.cs.kuleuven.be/~joost/DN/

Programmeren in het echt

Programming in the large

- Deel van groter geheel
- In teamverband
- Open opdracht
- Zelfstudie



- Inhoud: JEE
- Vorm: Project

Structuur vak

- Theorielessen:
  - grote lijnen
  - beperkt (6/12), aangevuld met zelfstudie (cursustekst en online)
  - http://www.cs.kuleuven.be/~joost/DN/
- Practicum:
  - inoefenen “in the small”

Project

- In groepjes van 2
- Tijdens practicumsessies **en daarbuiten**
- Deadline: **begin examenperiode**
- Op NetBeans 7.4
- Wekelijks verslag (wie, wat, hoe lang) volgens template

Evaluatie

- Op basis van
  - Project: werkt het én is het volgens de richtlijnen?
  - Verslagen
  - Werk tijdens practica
  - Mondelinge ondervraging over project

ECTS-fiche

Legende:
- Verplicht in fase
- Optioneel in fase
- Eerste semester
- Tweede semester
- Beide semesters
- Dit jaar
- Volgend jaar
- Wisselende jaren
- Extern
- Voorwaarden
- Docenten
- Onderwijstaal
- Duur
- Niveau

Applicatiearchitecturen (B-KUL-YI0915)

4.0 studiepunten | **Nederlands** | **45 uren** | **Eerste semester**

Vennekens Joost (coördinator) | **Van Haperen Anna** | **Vennekens Joost**

OC Elektronica-ICT - Campus De Nayer Sint-Katelijne-Waver

Toelichting | **Toelichting 2e examenkans**

De evaluatie van dit vak gebeurt op basis van het project dat de studenten -- na een grondige voorbereiding buiten de contactmomenten -- zullen implementeren tijdens de practicumzittingen. De docenten van dit vak zullen in samenspraak het resultaat van dit project beoordelen op basis van volgende criteria:

- Volledigheid: Zijn de studenten erin geslaagd om al de gevraagde functionaliteit te voorzien?
- Functionele correctheid: Gedraagt de software zich zoals gevraagd wordt?
- Ontwerp: Is de correct software ontworpen volgens de methodologie die in de cursus wordt aangeleerd?

Dit project wordt door de studenten uitgevoerd in groepjes van twee. Het globale project-punt kan voor een individuele student worden bijgesteld op basis van volgende criteria:

- Werk tijdens de practicumzittingen: Was de student steeds stipt aanwezig in de practicumzittingen en heeft hij zich tijdens deze contactmomenten op professionele wijze ingezet voor het project?
- Projectverslagen: Heeft de student tijdig de projectverslagen ingediend? Kwamen deze verslagen overeen met de vorderingen die de docent tijdens de practicumzittingen kon observeren? Heeft de student zich volgens deze verslagen voldoende ingezet voor het project?
- Mondelinge verdediging: Tijdens de examenperiode wordt een contactmoment voorzien waarop elke student individueel toelichting moet verschaffen over zijn/haar project bij de docent. Hierbij wordt de student geacht om elke ontwerp- of implementatiebeslissing die tijdens het project genomen is afdoende te kunnen toelichten. Daarnaast dient de student ook in staat te zijn om aan te geven hoe bepaalde wijzigingen aan de functionaliteit van de software het best zouden kunnen worden doorgevoerd, of om verschillende alternatieve aanpakken uit de cursus met elkaar te vergelijken binnen de context van dit concrete systeem.

Deze individuele criteria zullen resulteren in een factor waarmee de globale project-score vermenigvuldigd zal worden. Dit houdt in dat zelfs een perfekte score op het globale project niet noodzakelijk hoeft te leiden tot zelfs maar een voldoende individuele score, indien, bijvoorbeeld, de houding van de student tijdens de practicumzittingen niet voldoende professioneel was, de student geen realistische practicumverslagen heeft afgeleverd, of de student tijdens de mondelinge verdediging niet in staat was om de gemaakte keuzes op voldoende wijze te motiveren.

Inhoud

- Java Entreprise Edition (JEE) raamwerk
  - Glassfish Server
  - NetBeans IDE
- + inleiding .NET en ASP.MVC

Doel

Veel verschillende functionaliteit
in uniform geheel





BV: KU Loket

Doel

Eenheid in verscheidenheid

- Zelfde look
- Zelfde login
- Gedeelde gegevens
- Zelfde URLs
- Verschillende GUI
- Verschillende functionaliteit
- Verschillende gegevens

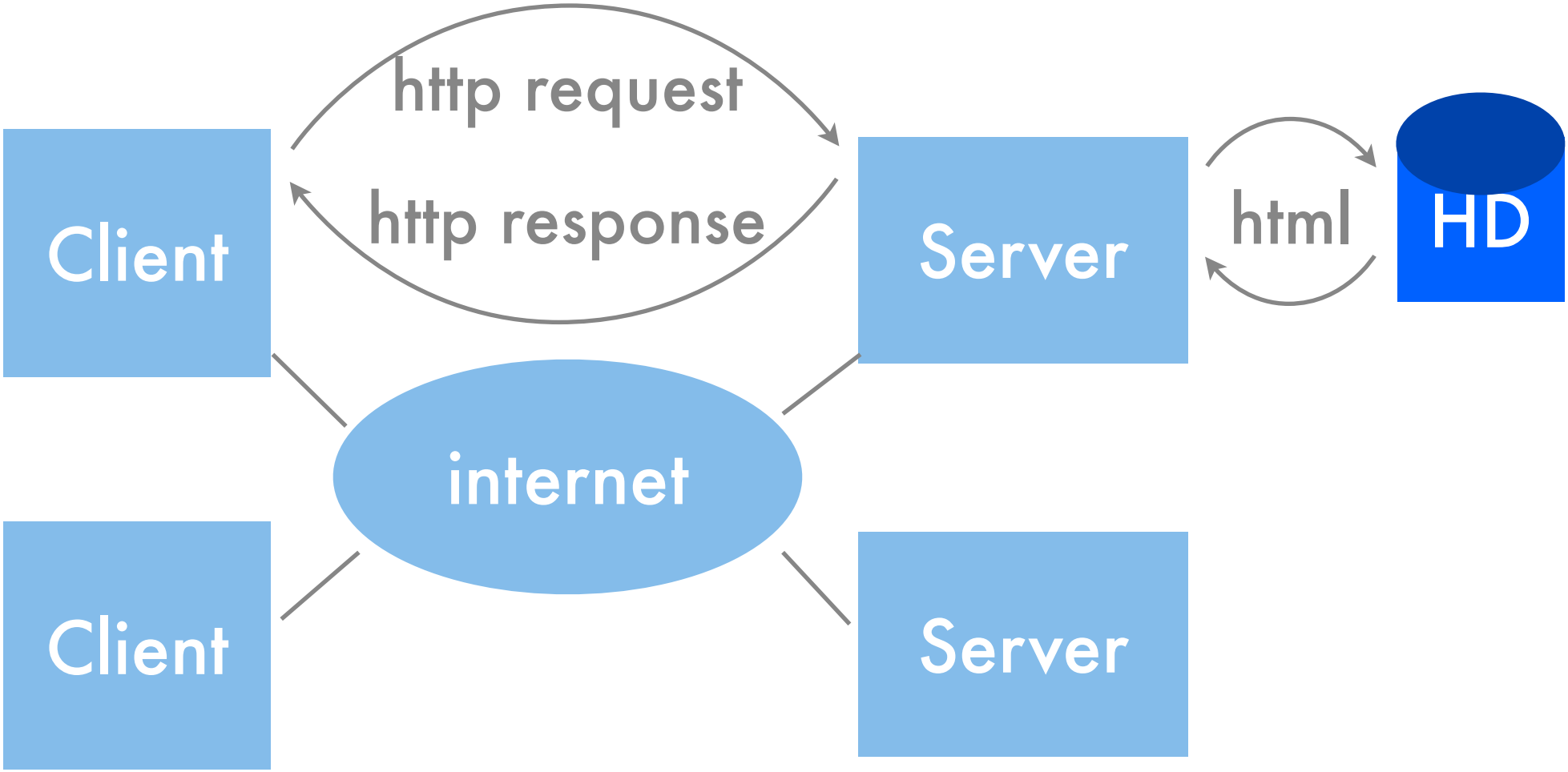
Nog meer verscheidenheid

- Verschillende programmeurs
- Verschillende omgevingen, machines, databanken, legacy software
- Web UI, native UI, webservices

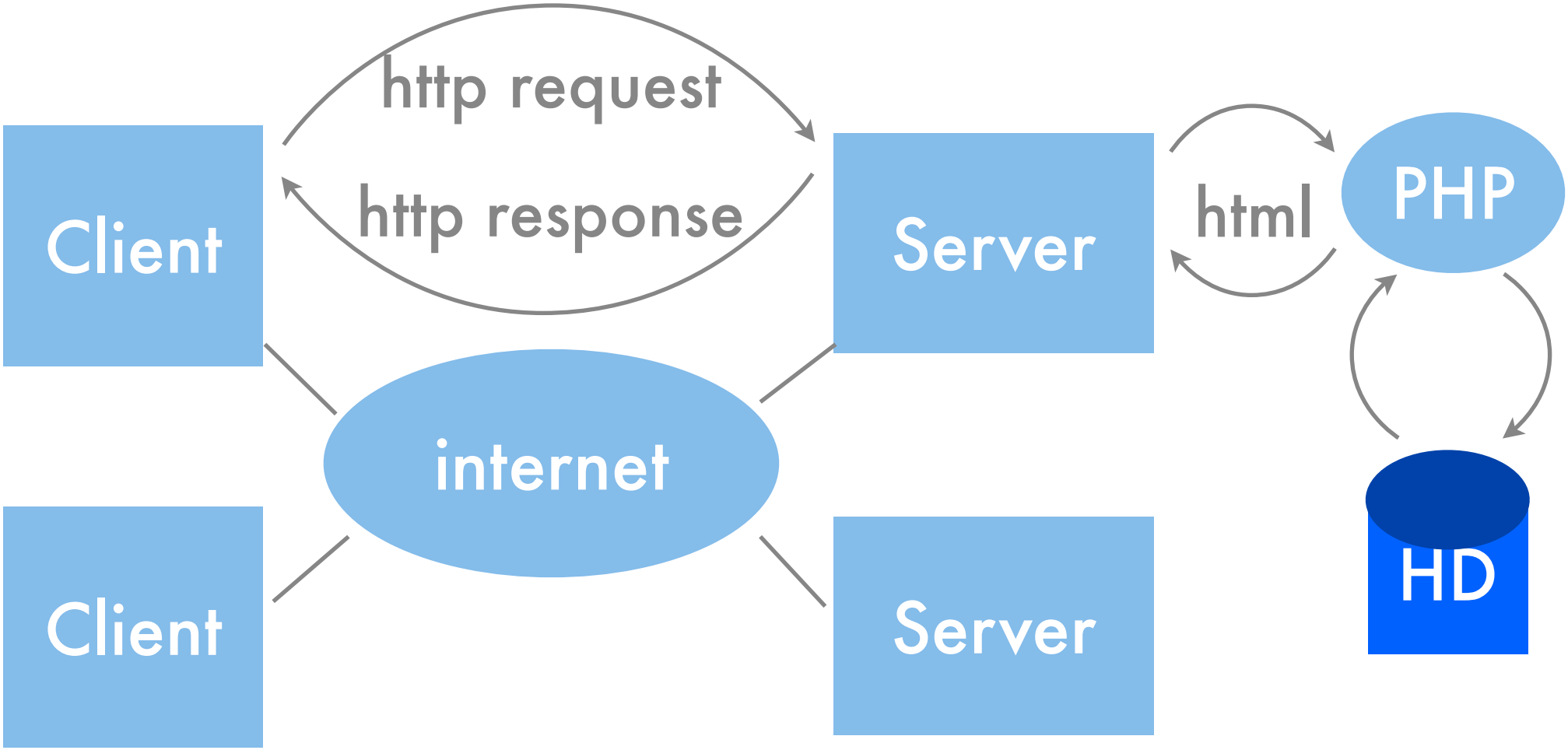
Applicatiearchitectuur

- Brengt daar eenheid in
  - Portaalfunctie
    - Web UI, native UI, webservices
  - Centraliseren van gedeelte functionaliteit
    - Security, logging, lokalisatie, configuratie, …
- Focussen op eigen “business logica”

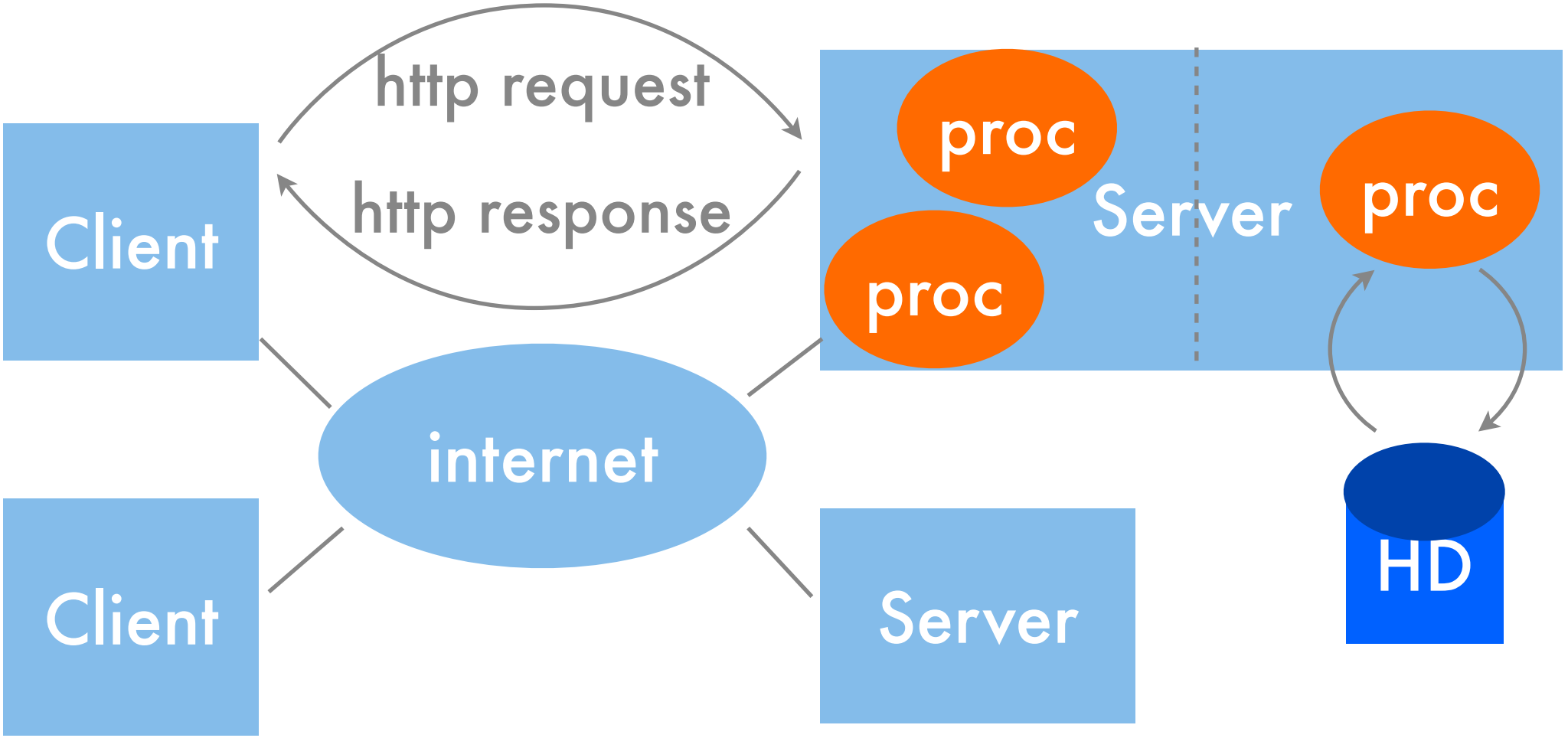
Traditionele webserver



Dynamische webserver



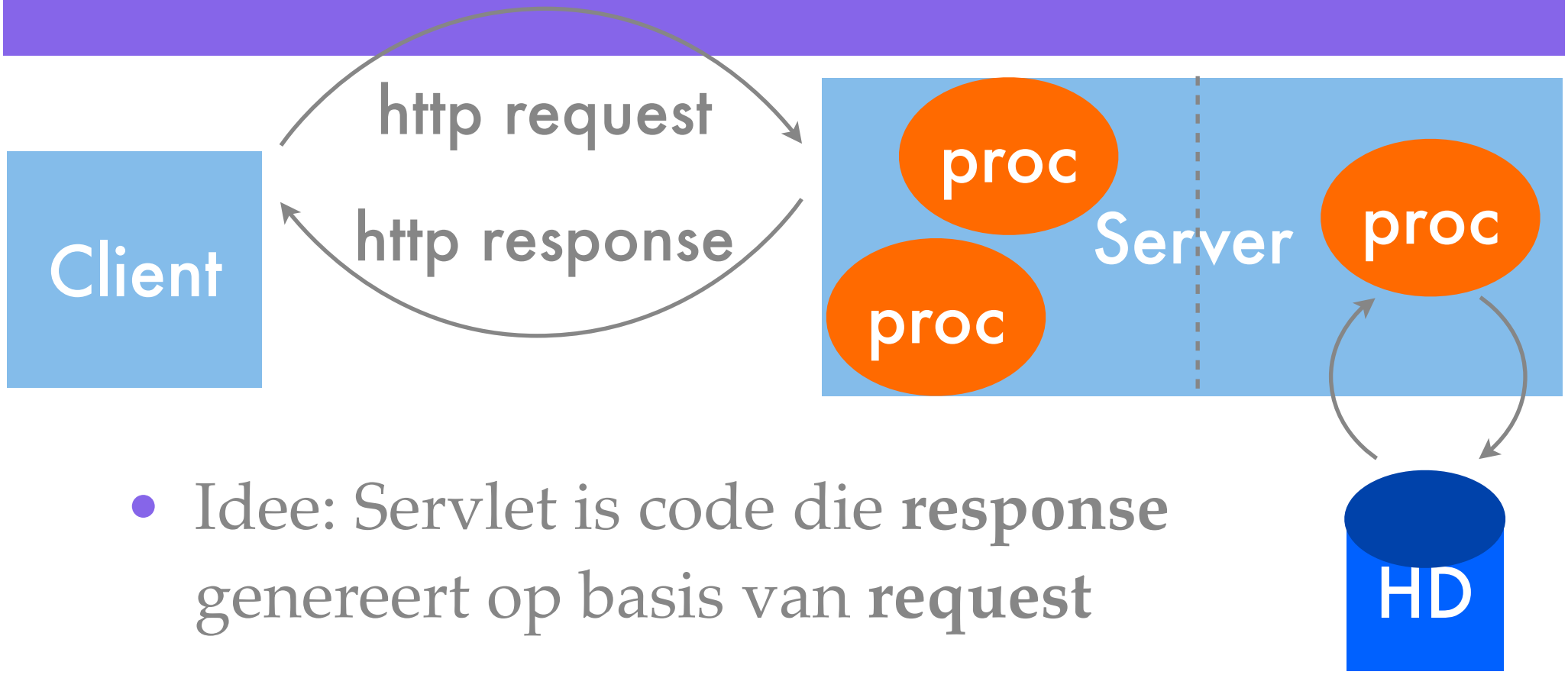
AA


http request
http response
Client
Client
internet
Server
proc
proc
Server
proc
HD

Voorbeeldje

- BierExpert
- Termen:
  - Web container, EJB container
  - Java Server Pages
  - Servlets
  - Web Deployment Descriptor

Servlet



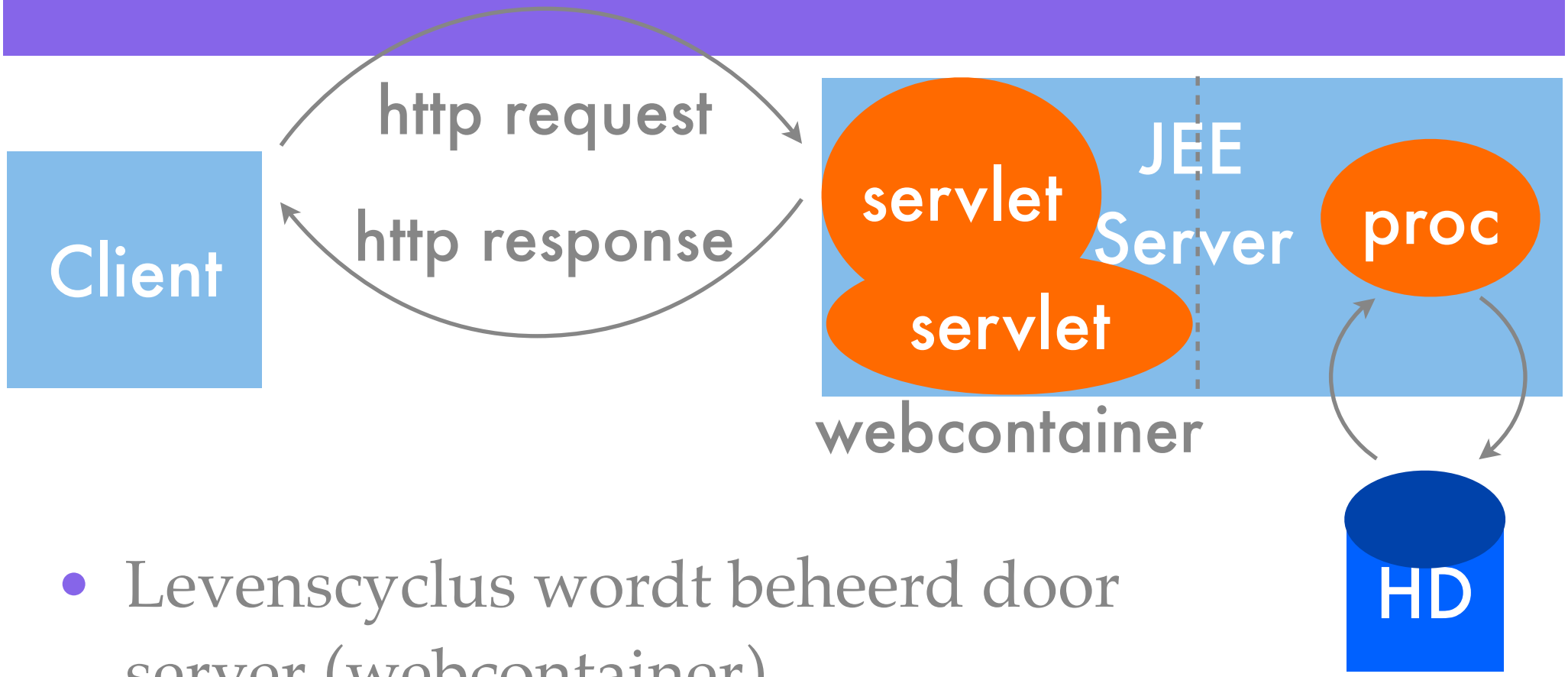
- Idee: Servlet is code die **response** genereert op basis van **request**
- Formeel: klasse die javax.servlet.Servlet implementeert

javax.servlet.Servlet

Method Summary

void	**destroy**() Called by the servlet container to indicate to a servlet that the servlet is being taken out of service.
ServletConfig	**getServletConfig**() Returns a ServletConfig object, which contains initialization and startup parameters for this servlet.
java.lang.String	**getServletInfo**() Returns information about the servlet, such as author, version, and copyright.
void	**init**(ServletConfig config) Called by the servlet container to indicate to a servlet that the servlet is being placed into service.
void	**service**(ServletRequest req, ServletResponse res) Called by the servlet container to allow the servlet to respond to a request.

Servlet



- Levenscyclus wordt beheerd door server (webcontainer)
- Zorgt dat er (hoogstens 1) instantie is
- Kunnen samenwerken in ketting

HttpServlet

- javax.servlet.http.HttpServlet implementeert javax.servlet.Servlet

Method Summary	
protected void	**doDelete**(HttpServletRequest req, HttpServletResponse resp) Called by the server (via the service method) to allow a servlet to handle a DELETE request.
protected void	**doGet**(HttpServletRequest req, HttpServletResponse resp) Called by the server (via the service method) to allow a servlet to handle a GET request.
protected void	**doOptions**(HttpServletRequest req, HttpServletResponse resp) Called by the server (via the service method) to allow a servlet to handle a OPTIONS request.
protected void	**doPost**(HttpServletRequest req, HttpServletResponse resp) Called by the server (via the service method) to allow a servlet to handle a POST request.

Voorbeeldje

- BierKeuze1.java

Java Server Pages

HTML + scriptlets + JSP tags

- Makkelijker om HTML uitvoer te genereren
- Voorbeeldje: resul.jsp
- Wordt on-demand vertaald naar servlet

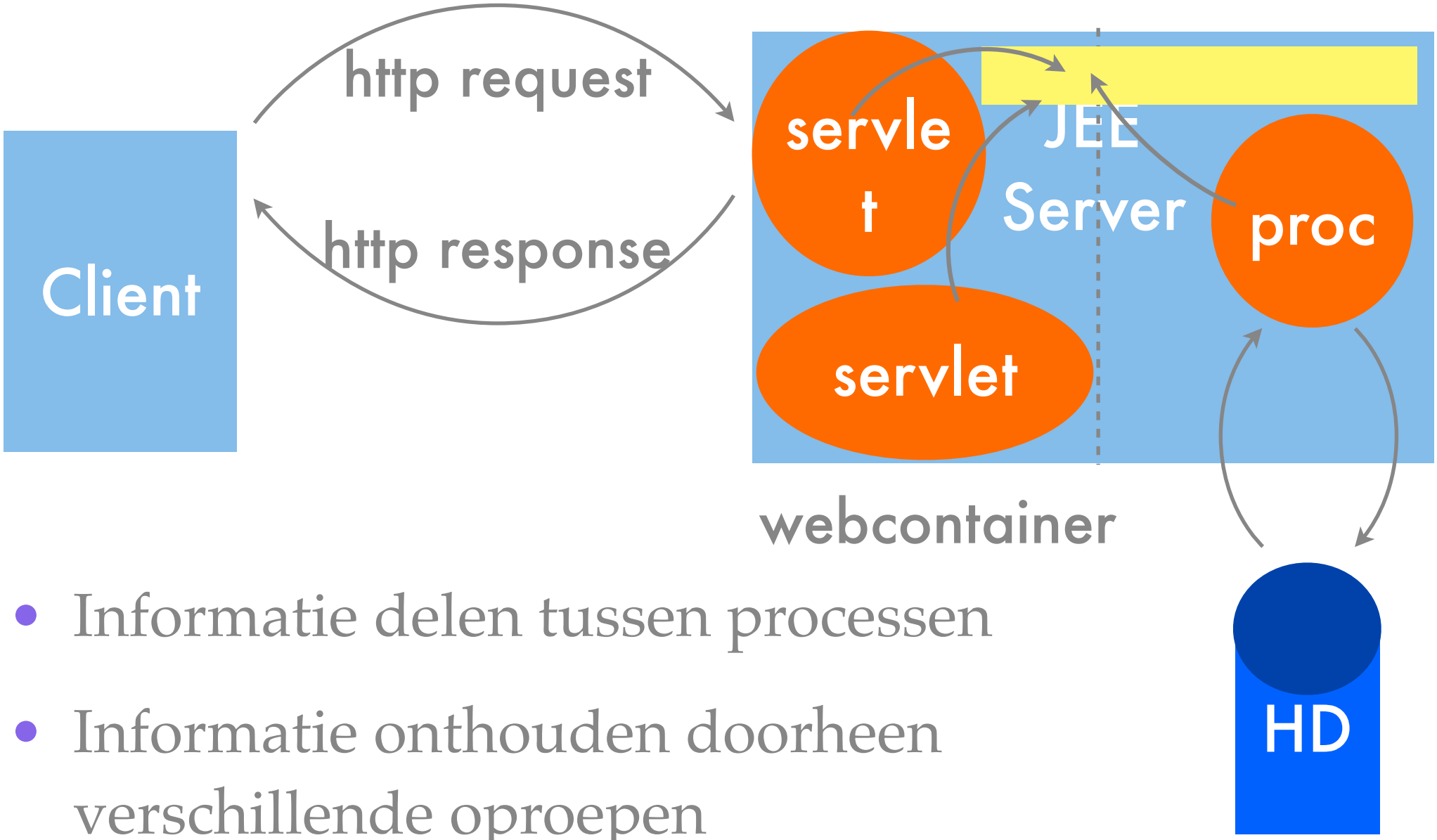
JSP Tags

- Scriptlets zijn nogal ongebruiksvriendelijk
  - (voor webdesigners)
- Oplossing: complexiteit verbergen in HTML tags
- JSP Standard Tag Library

Web deployment descriptor

- Voorbeeldje: web.xml
- Configureert o.a.
  - Servlet mappings
  - Foutenboodschappen
  - (Koppeling met DBen)

Gedeelde geheugen



- Informatie delen tussen processen
- Informatie onthouden doorheen verschillende oproepen

Gedeelde geheugen

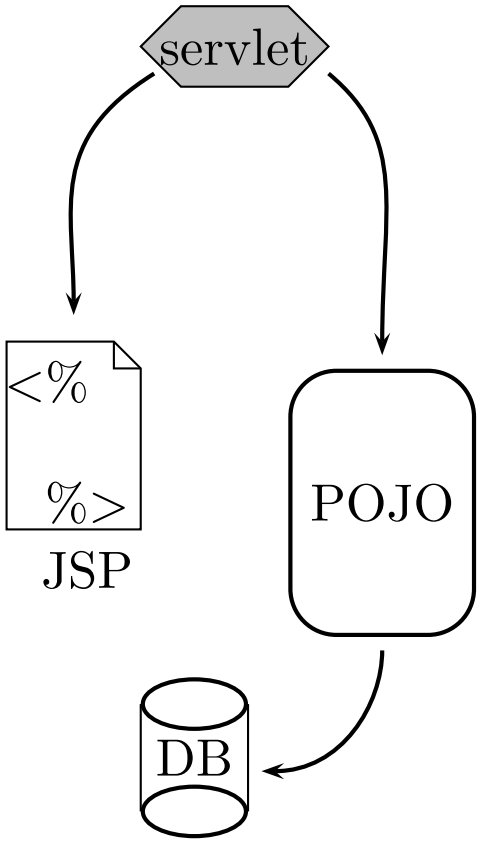
- Servlet
- Request
- Sessie
- Applicatie

```
//Op Request-niveau
   List lijst;
   lijst = (List) request.getAttribute("naam_van_object");
   request.setAttribute("naam_van_object", lijst);
//Op Sessie-niveau
   HttpSession session = request.getSession(true);
   lijst = (List) session.getAttribute("naam_van_object");
   session.setAttribute("naam_van_object", lijst);
//Op Context-niveau
   lijst = (List)getServletContext().getAttribute("naam_van_object
   getServletContext().setAttribute("naam_van_object", lijst);
```

MVC patroon

VIEW

Verantwoordelijk voor de presentatie.
deel met gebruikersinput, die dan doorgestuurd wordt naar de Controller.



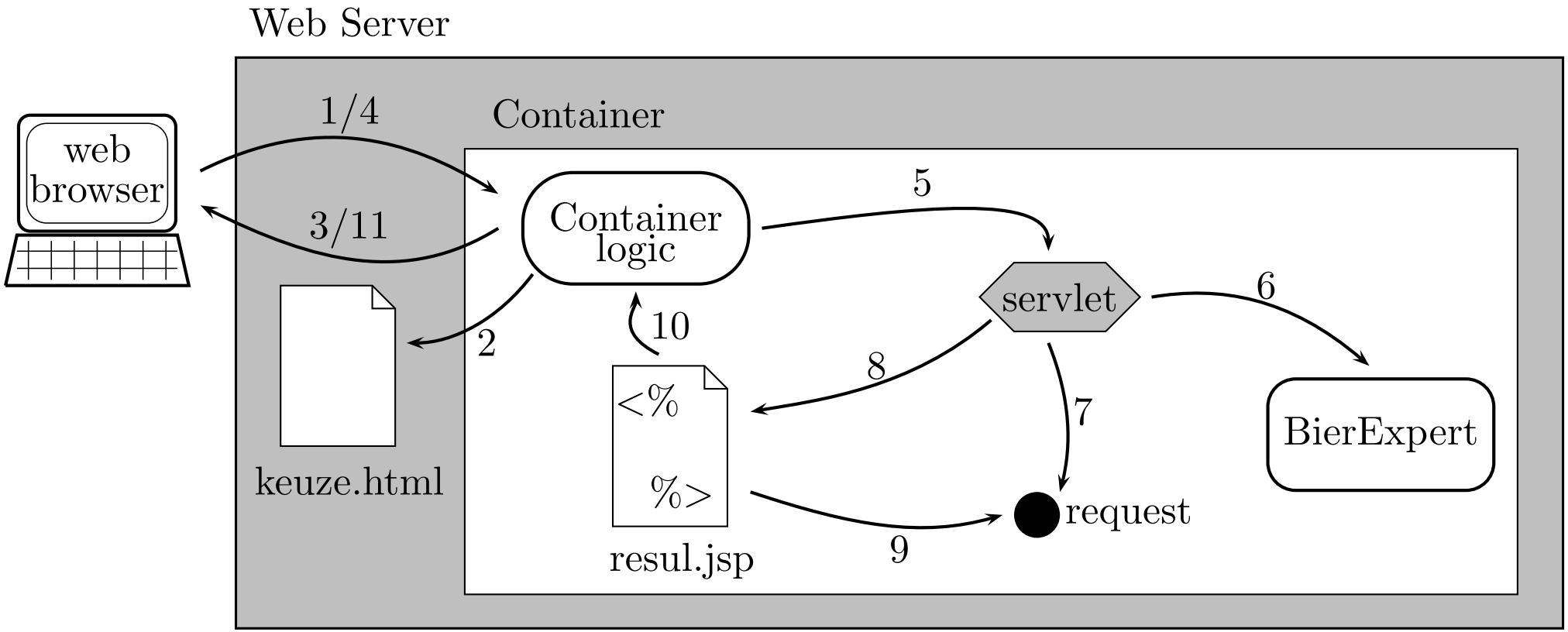
CONTROLLER

gebruikersinput uit de request naar Model
nieuwe toestand van Model naar View

MODEL

reële toestand en business logica (regels die op deze data werken).
in contact met de databank, (via een ander object)

Voorbeeld


Web Server
Container
web browser
1/4
3/11
Container logic
5
servlet
6
BierExpert
2
10
8
7
keuze.html
<%
%>
resul.jsp
9
request