VEEBIRAKENDUSTE LOOMINE

MTAT.03.230 (6 EAP)

2. Loeng

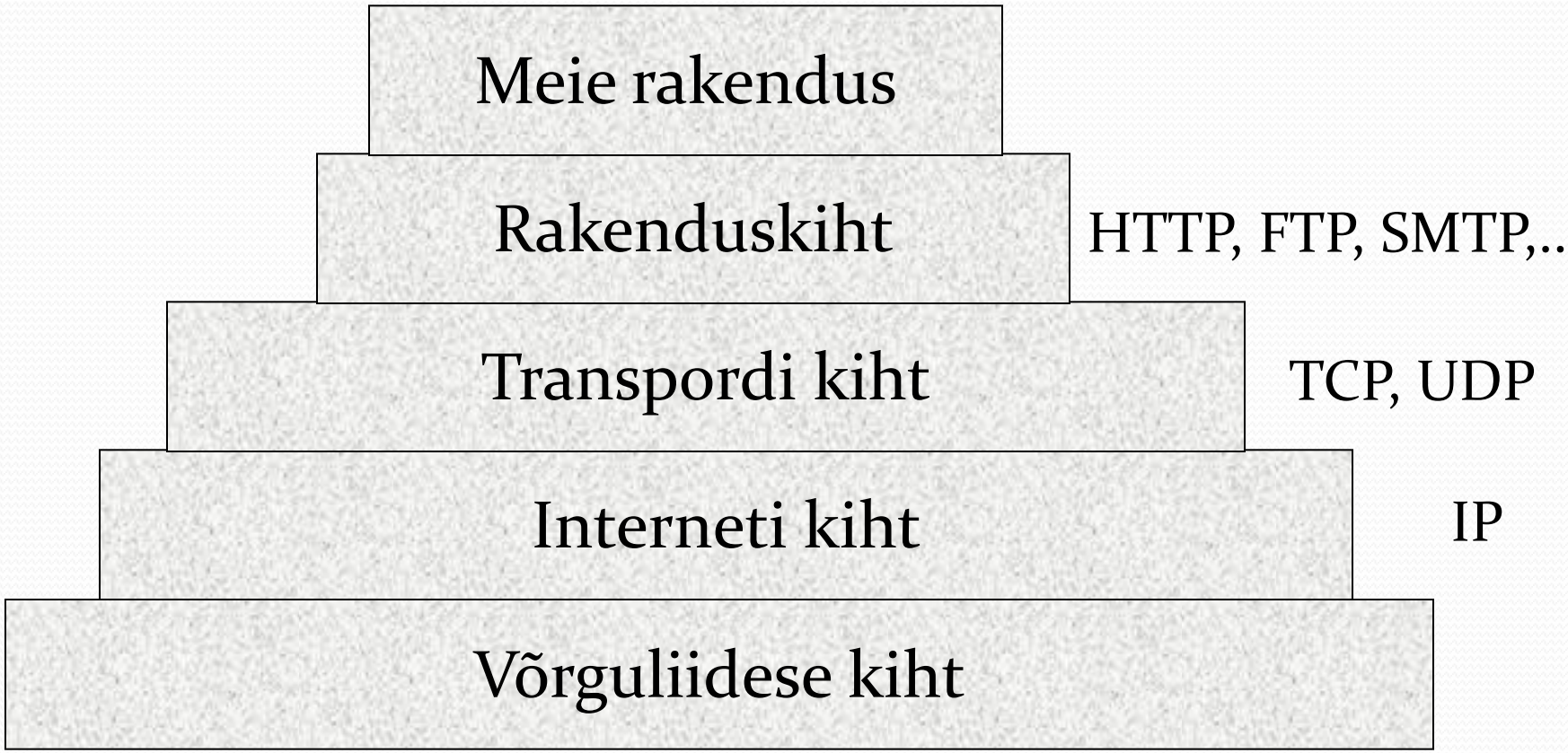
Helle Hein

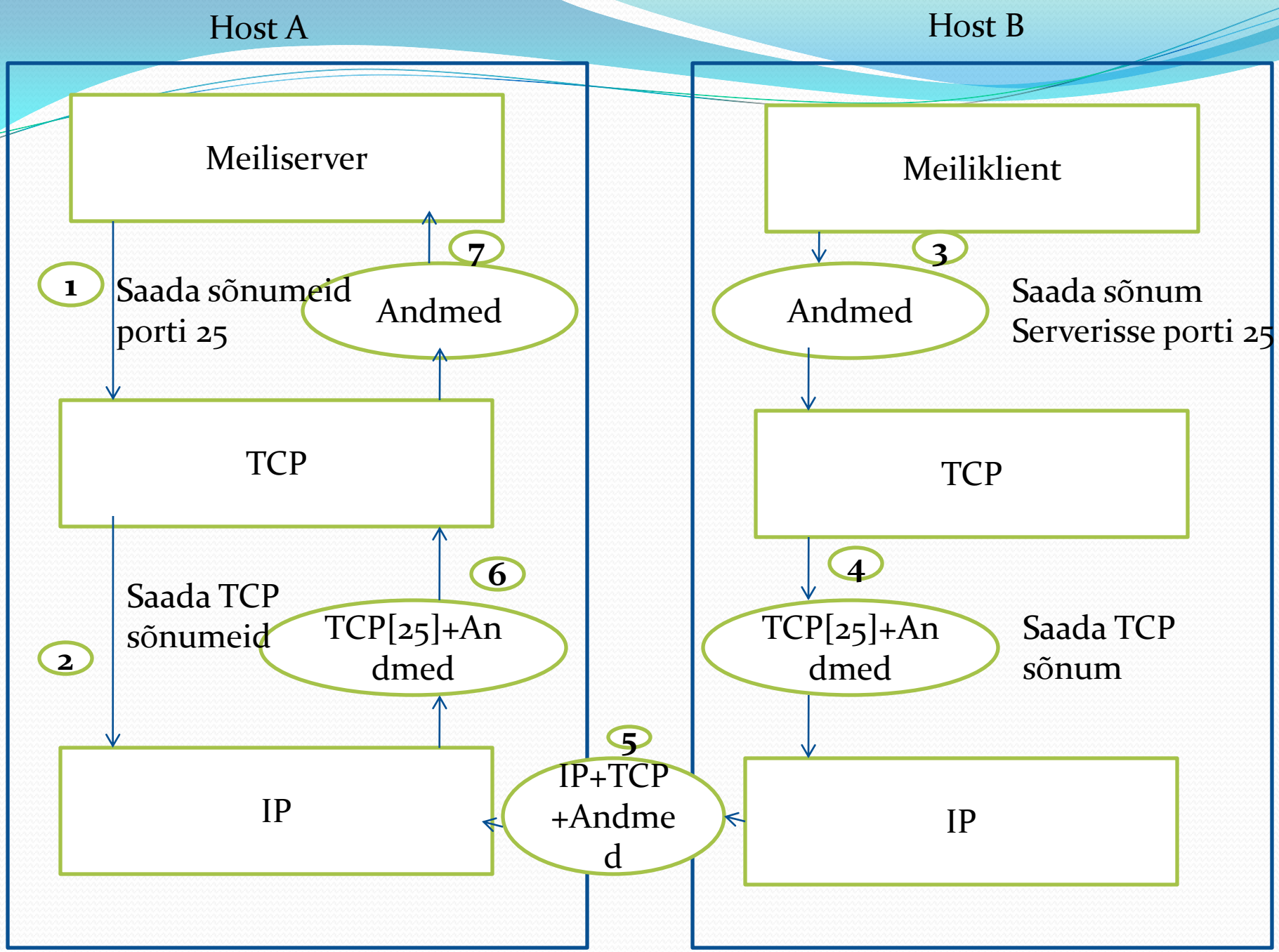
Teemad: HTTP, veebiserverid, mitmekihilised veebirakendused

HTTP - Hypertext Transfer Protocol

Mõnes allikas ka Hypertext Transport Protocol

VÕRGUKIHID


Meie rakendus
Rakenduskiht
HTTP, FTP, SMTP,..
Transpordi kiht
TCP, UDP
Interneti kiht
IP
Võrguliidese kiht

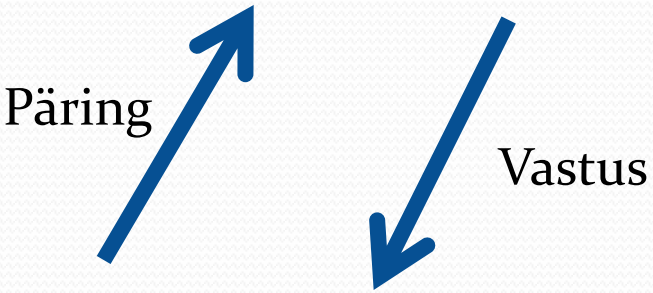

Host A
Host B
Meiliserver
Meiliklient
1
Saada sõnumeid porti 25
7
Andmed
3
Andmed
Saada sõnum Serverisse porti 25
TCP
TCP
6
Saada TCP sõnumeid
4
2
TCP[25]+Andmed
TCP[25]+Andmed
Saada TCP sõnum
5
IP+TCP+Andmed
IP
IP

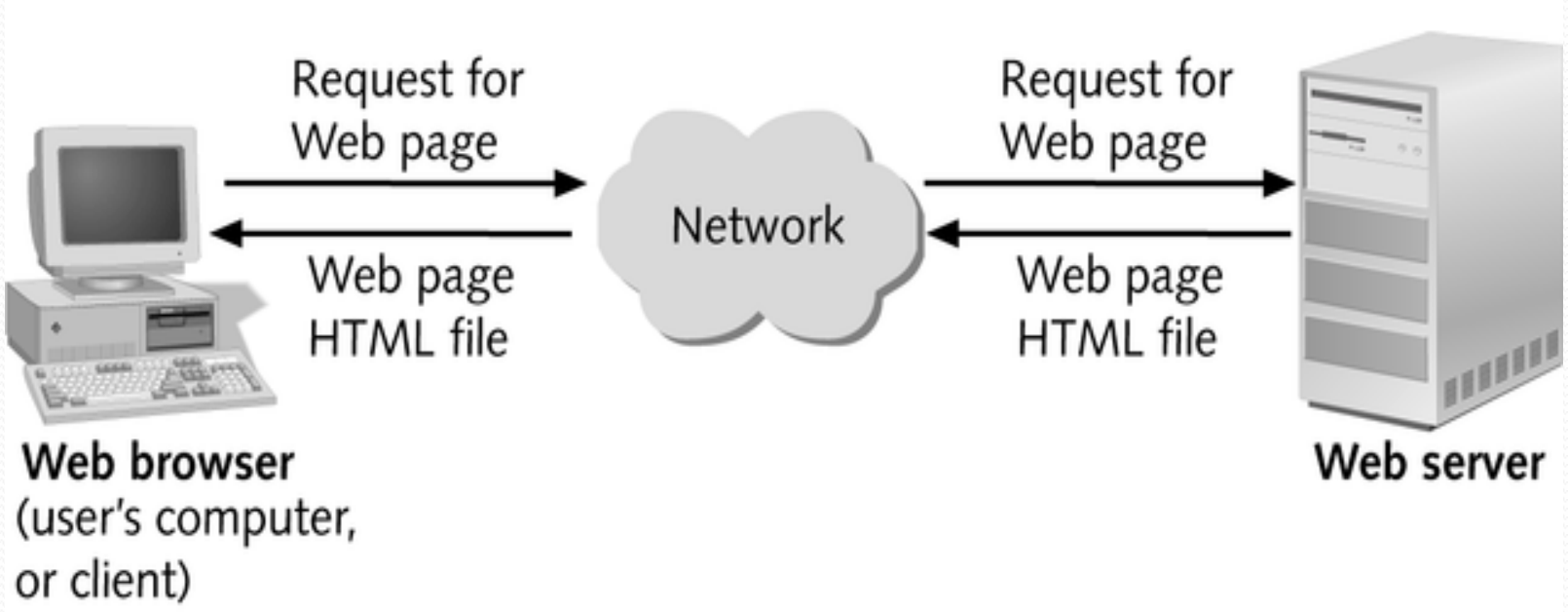
HTTP

Klient – server süsteem

Päring – Vastus mudel

Port: 80




Request for
Web page
Web page
HTML file
Network
Request for
Web page
Web page
HTML file
Web browser
(user's computer,
or client)
Web server

HTTP: päring

```
GET http://www.ut.ee/ HTTP/1.1
Accept: */*
Accept-Language: et
User-Agent: Mozilla/4.0 (compatible; MSIE 8.0; Windows
      NT 5.1; Trident/4.0; GTB6.3; .NET CLR 1.1.4322;
      .NET CLR 2.0.50727; InfoPath.2; .NET CLR
      3.0.4506.2152; .NET CLR 3.5.30729)
Accept-Encoding: gzip, deflate
Connection: Keep-Alive
Host: www.ut.ee
```

Päring

http://www.ut.ee/

HTTP päringu esimene rida

```
GET / HTTP/1.1
```

- Päringumeetod URI HTTP versioon

```
GET          /    HTTP/1.1
```

Enamus brausereid toetab HTTP/1.1

URI – Uniform Resource Identifier
Skeemid: http, ftp, telnet, mailto, https, file,
news, gopher, archie, wais ...
N: ftp://ftp.example.org/pub/file1.txt
http://www.iana.org/assignments/uri-schemes.html

Päringu meetodid

GET tagastada päritav leht – info URLis ;
POST tagastada päritav leht, lisaks saadan andmed – info kehas;

Teised meetodid:
HEAD –tagastab vaid päise;
OPTIONS – tagastab meetodite loetelu, mida saab kasutada antud ressursile ligipääsemiseks;
PUT – paigutada selle sõnumi keha serverisse ja omistada vastav URI paigutatud ressursile;
DELETE – tulevikus vastata selle päringuga sõnumile, et ressurssi ei ole;
TRACE – tagastab HTTP päringu sõnumi; kasutatakse testimiseks.

Päringu päiseväljad

Välja nimi: väärtus

Host: osa URList
User-Agent: info brauseri kohta
Accept: MIME tüübid, mis on vastuvõetavad
Accept-Language: keel vastuse keha jaoks
Accept-Encoding: info pakkimise kohta
Accept-Charset: info kasutatava märgisüsteemi kohta
Keep-Alive: aeg sekundites, kui kaua ühendus peab kestma

Täiendav info HTTP/1.1 jaoks:
ftp://ftp.rfc-editor.org/in-notes/rfc2616.txt

MIME - Multipurpose Internet Mail Extensions

text/html	HTML dokument
image/gif	pilt GIF formaadis
image/jpeg	pilt JPEG formaadis
text/plain	kujundamata tekst
…	

```
Accept: text/html,application/xhtml+xml,application/xml;
      q=0.9,*/*;q=0.8
```

q=num num∈(0, 1) - kvaliteet

Type	Meaning
application/msword	Microsoft Word document
application/octet-stream	Unrecognized or binary data
application/pdf	Acrobat (.pdf) file
application/postscript	PostScript file
application/vnd.ms-excel	Excel spreadsheet
application/vnd.ms-powerpoint	Powerpoint presentation
application/x-gzip	Gzip archive
application/x-java-archive	JAR file
application/x-java-vm	Java bytecode (.class) file
application/zip	Zip archive
audio/basic	Sound file in .au or .snd format
audio/x-aiff	AIFF sound file
audio/x-wav	Microsoft Windows sound file
audio/midi	MIDI sound file
text/css	HTML cascading style sheet
text/html	HTML document
text/plain	Plain text
text/xml	XML document
image/gif	GIF image
image/jpeg	JPEG image
image/png	PNG image
image/tiff	TIFF image
video/mpeg	MPEG video clip
video/quicktime	QuickTime video clip

Common HTTP 1.1 Request Headers

- Accept
  - Indicates MIME types browser can handle
  - Can send different content to different clients. For example, PNG files have good compression characteristics but are not widely supported in browsers. A servlet could check to see if PNG is supported, sending <IMG SRC="picture.png" ...> if it is supported, and <IMG SRC="picture.gif" ...> if not.
- Accept-Encoding
  - Indicates encodings (e.g., gzip or compress) browser can handle.

Common HTTP 1.1 Request Headers (Continued)

- Authorization
  - User identification for password-protected pages.
  - Instead of HTTP authorization, use HTML forms to send username/password and store info in session object. This approach is usually preferable because standard HTTP authorization results in a small, terse dialog box that is unfamiliar to many users.
  - Servers have high-level way to set up password-protected pages without explicit programming in the servlets.

Common HTTP 1.1 Request Headers (Continued)

- Connection
  - In HTTP 1.0, keep-alive means browser can handle persistent connection. In HTTP 1.1, persistent connection is default. Persistent connections mean that the server can reuse the same socket over again for requests very close together from the same client (e.g., the images associated with a page, or cells within a framed page).
  - Servlets can't do this unilaterally; the best they can do is to give the server enough info to permit persistent connections. So, they should set `Content-Length` with `setContentLength` (using `ByteArrayOutputStream` to determine length of output).
- Cookie
  - Gives cookies previously sent to client. Use `getCookies`, not `getHeader`..

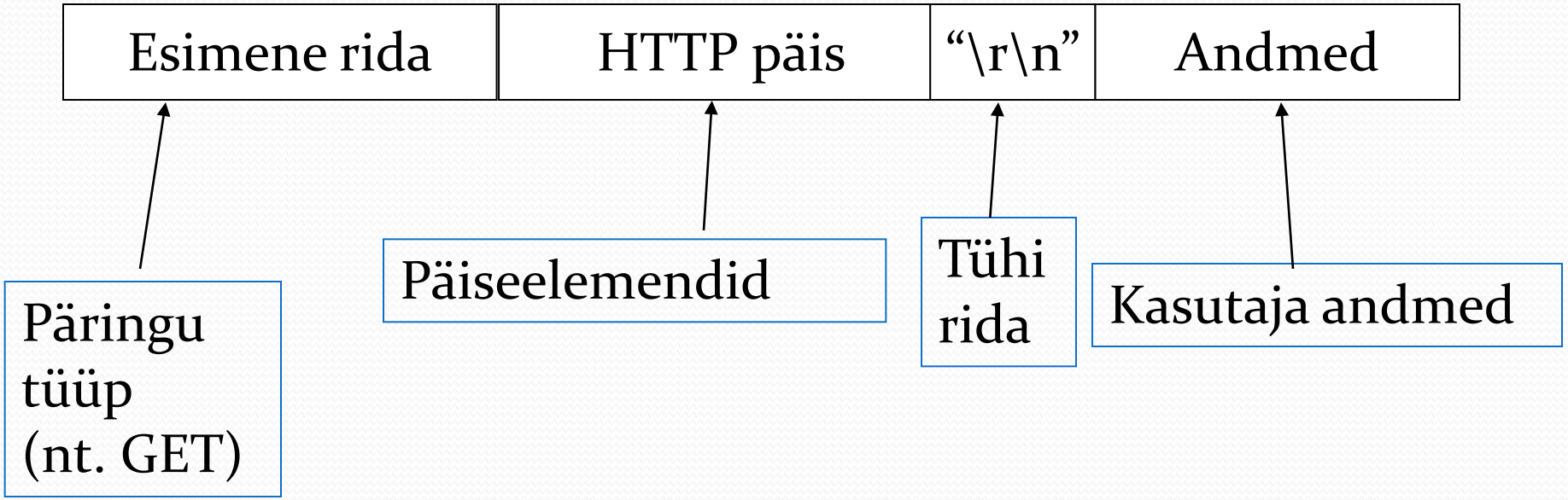
Common HTTP 1.1 Request Headers (Continued)

- Host
  - Indicates host given in original URL
  - This is a *required* header in HTTP 1.1. This fact is important to know if you write a custom HTTP client or telnet to a server and use the HTTP/1.1 version.
- If-Modified-Since
  - Indicates client wants page only if it has been changed after specified date
  - Don't handle this situation directly; implement `getLastModified` instead.

Common HTTP 1.1 Request Headers (Continued)

- Referer
  - URL of referring Web page
  - Useful for tracking traffic; logged by many servers
  - Can also be used to let users set preferences and then return to the page they came from
  - Can be easily spoofed, so don't let this header be your sole means of deciding (for example) how much to pay sites that show your banner ads.
- User-Agent
  - String identifying the browser making the request
  - Best used for identifying *category* of client
    - Web browser vs. I-mode cell phone, etc.
  - For Web applications, use other headers if possible
  - Again, can be easily spoofed.

HTTP: päring


Esimene rida
HTTP päis
“\r\n”
Andmed
Päringu tüüp (nt. GET)
Päiseelemendid
Tühi rida
Kasutaja andmed

HTTP: vastus

```
HTTP/1.1 200 OK
Date: Sun, 14 Feb 2010 15:17:29 GMT
Server: Apache/2.2.3 (CentOS)
X-Powered-By: PHP/5.1.6
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-
check=0, pre-check=0
Pragma: no-cache
Vary: Accept-Encoding
Content-Length: 14507
Connection: close
Content-Type: text/html

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
Transitional//EN"
      "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html;
charset=iso-8859-15" /> ...
```

HTTP: vastus

Koosneb:

- seisundi reast, nt `HTTP/1.1 200 OK`
- ühest või mitmest päiseväljast;
- tühjast reast;
- sõnumi kehast.

HTTP vastuse seisundikoodid

1 – informatsioon

2 – edukas

- 200 - OK
- 202 - Accepted

3 – ümbersuunamine

- 301 – Moved

4 – kliendiviga

- 400 – Bad Request
- 401 – Unauthorized
- 403 – Forbidden
- 404 – Not found

5 – serveri viga

- 500 – Internal Server Error
- 502 – Service Overloaded
- …

HTTP vastus - päised

- Date
- Server
- Set-Cookie
- Pragma
- Content-Length
- Content-Type
- Location
- …

Common HTTP 1.1 Response Headers

- Cache-Control (1.1) and Pragma (1.0)
  - A no-cache value prevents browsers from caching page. Send both headers or check HTTP version.
- Content-Encoding
  - The way document is encoded. Browser reverses this encoding before handling document.
- Content-Length
  - The number of bytes in the response.
  - Use `ByteArrayOutputStream` to buffer document before sending it, so that you can determine size.

Common HTTP 1.1 Response Headers (Continued)

- Content-Type
  - The MIME type of the document being returned.
  - Use `setContentType` to set this header.
- Expires
  - The time at which document should be considered out-of-date and thus should no longer be cached.
  - Use `setDateHeader` to set this header.
- Last-Modified
  - The time document was last changed.
  - Don't set this header explicitly; provide a `getLastModified` method instead.

Common HTTP 1.1 Response Headers (Continued)

- Location
  - The URL to which browser should reconnect.
  - Use `sendRedirect` instead of setting this directly.
- Refresh
  - The number of seconds until browser should reload page. Can also include URL to connect to.
- Set-Cookie
  - The cookies that browser should remember. Don't set this header directly; use `addCookie` instead.
- WWW-Authenticate
  - The authorization type and realm needed in Authorization header.

HTTP: vastus

Success/Failure

Arv vahemikust 100 kuni 599

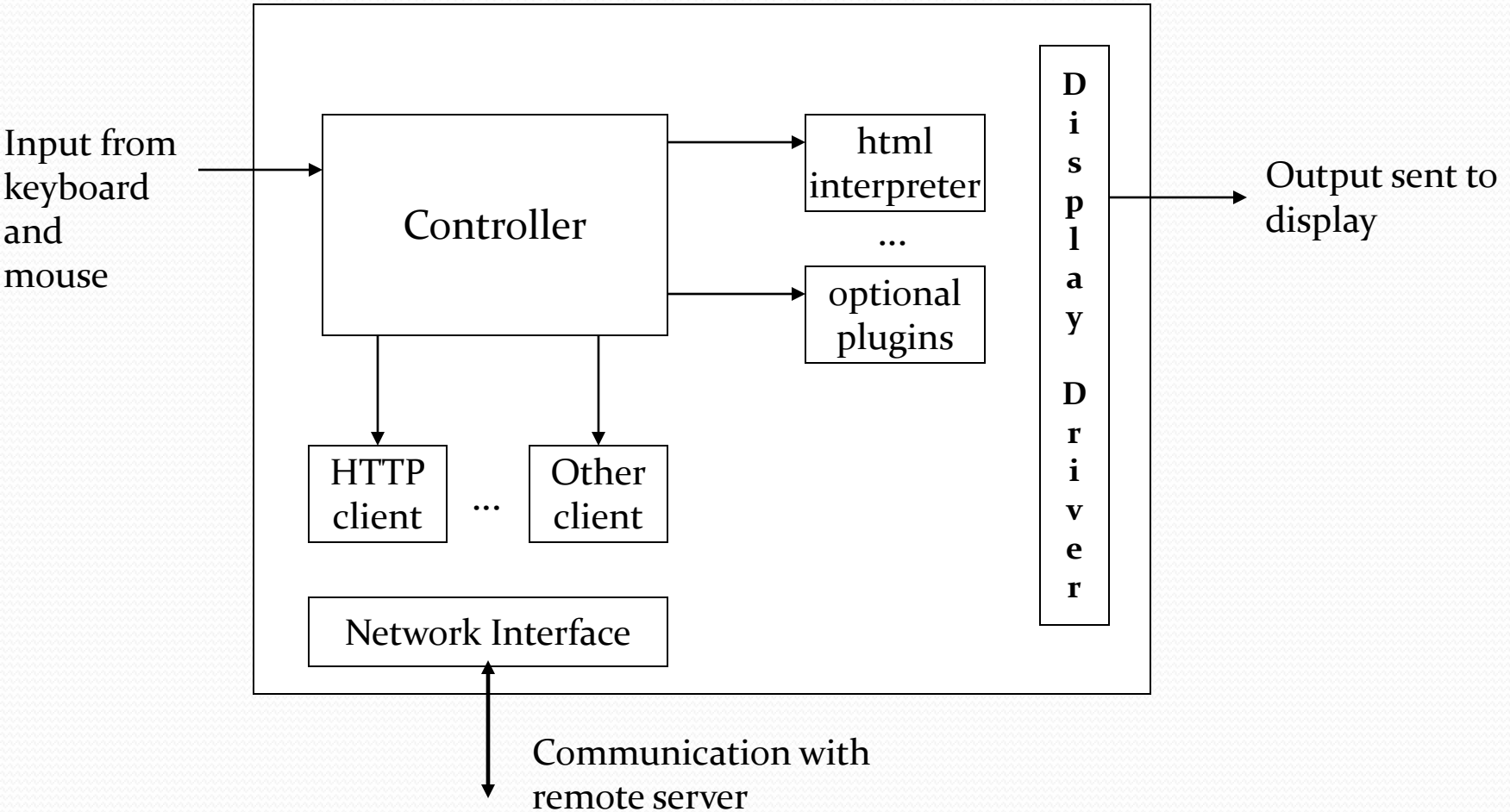
Päiseelemendid, nt sisu tüüp

Tühi rida

Andmed

WWW – Brauseri arhitektuur

Dr. Ralph Moseley slaid



Brauserite põhifunktsioonid

Teha HTTP päringuid brauseri kasutaja nimel

Seisundirea sõnumid:

- Resolving host www.example.org …
- Connecting to www.example.org …
- Waiting for www.example.org …
- Transferring data from www.example.org …
- Done
- …

Brauserite põhifunktsioonid

Peale kasutaja URL sisestust tuleb brauseril täita terve rida ülesandeid:

- formaatida URL ümber kehtivaks;
- konverteerida serveri nimi kasutades DNS sobivaks IP aadressiks;

- luua TCP ühendus nõutud veebiserveriga;
- saata HTTP päring üle TCP ühenduse ja oodata serveri vastust;

- näidata dokumenti, mis sisaldub vastuses; kui dokument ei ole plain-text, vaid HTML, siis renderdada dokument: paigutada tekst ja graafika sobivalt brauseri aknasse, luua tabelid, kasutada fonte, värve, jne.

URL

http – skeemi URL koosneb mitmest osast:

http://www.example.org:12345/a/b/c/d.txt?t=win&s=chess#para5

www.example.org:12345	domeen:port
/a/b/c/d.txt	path
t=win&s=chess	query string
#para5	fragment identifier

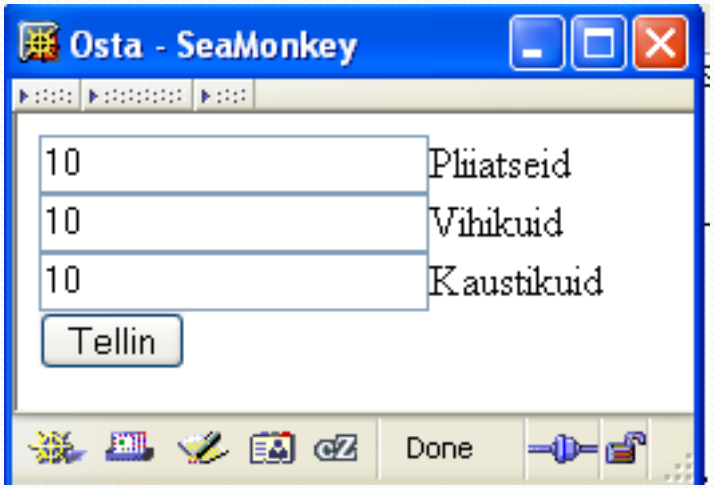
IP vs nimi
nslookup

HTTP: andmed

- **enctype=** ”application/x-www-form-urlencoded”

```
<form method= "post"
      action=http://www.pliiats.ee/osta.html enctype=
      ”application/x-www-form-urlencoded”>
  <input name= "pl" type= "text" value = "10">Pliiatseid
      </input>
  <input name= "vi" type= "text" value = "10">Vihikuid
      </input>
  <input name= "ka" type= "text"  value = "10">
      Kaustikuid</input>
  <input type="submit" value= "Tellin">
</form>
```

pl=10&vi=10&ka=10

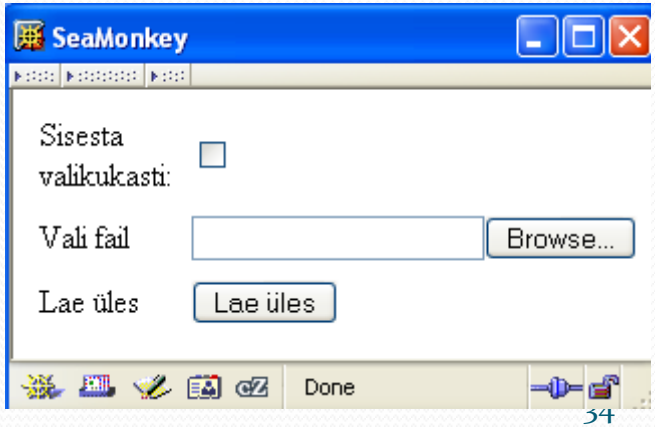


HTTP: andmed

- **`enctype=`** `”multipart/form-data”`
- Suured andmemahud
- Binaarsed andmed
- Iga vormi element eraldi
- Content-Disposition päis väärtusega `form-data`
- Vormi muutuja `name` atribuudiga

HTTP: andmed

```
<form method="post" enctype="multipart/form-data"
action= "www.example.org" >
  <table cellpadding="5" cellspacing="0" border="0">
      <tr> <td>Sisesta valikukasti:</td>
        <td><input type="checkbox"
      name="input_check"/></td> </tr>
      <tr> <td>Vali fail</td> <td><input type="file"
      name="file1"value="Vali"/></td> </tr>
      <tr> <td>Lae üles</td> <td><input
      type="submit"/></td> </tr>
  </table>
</form>
```



HTTP: andmed

```
-----------------------------7d01ecf406a6
Content-Disposition: form-data;
name="input_check"

on

-----------------------------7d01ecf406a6--
Content-Disposition: form-data; name="file1";
filename="C:\Inetpub\wwwroot\Upload\pic.gif"
Content-Type: image/gif
Content-Transfer-Encoding: binary
```

Faili pic.gif sisu…

```
-----------------------------7d01ecf406a6--
```

HTTP piirangud

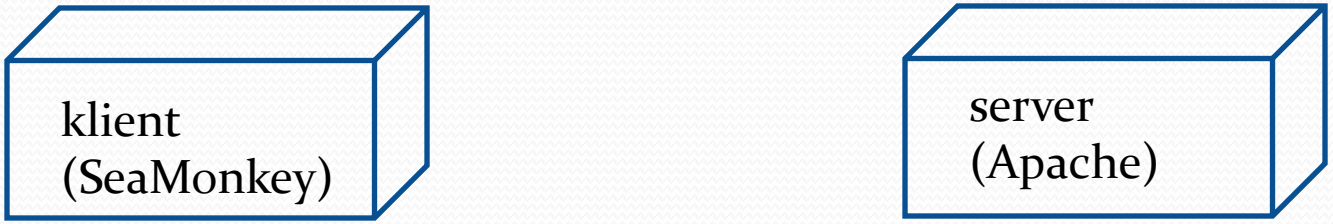
- Olekuta, ei ole sisseehitatud mehhanismi klientide jälgimiseks;
- Puudub seansihaldamine;
- Puuduvad sisseehitatud turvamehhanismid.

Veebiserver

Veebiserveri põhiülesanne:

Võtta klientidelt vastu HTTP päringuid ja tagastada sobiv allikas (kui võimalik) HTTP vastuses.

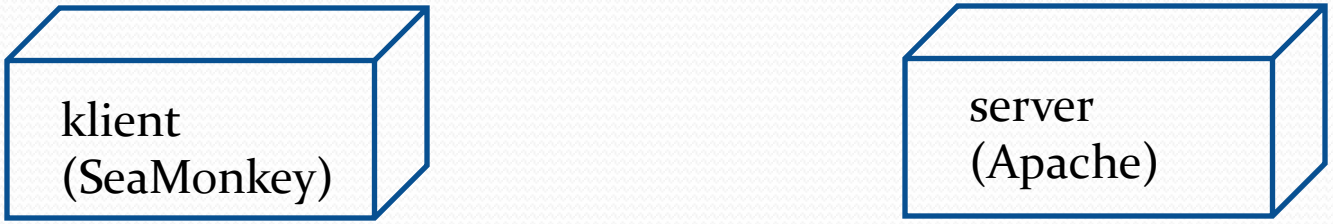
- Kliendi arvuti kasutaja sisestab URL:




```
https://www.eesti.ee/index.html
```

Veebiserver

- Serveri nimele seatakse vastavusse IP aadress DNS kaudu

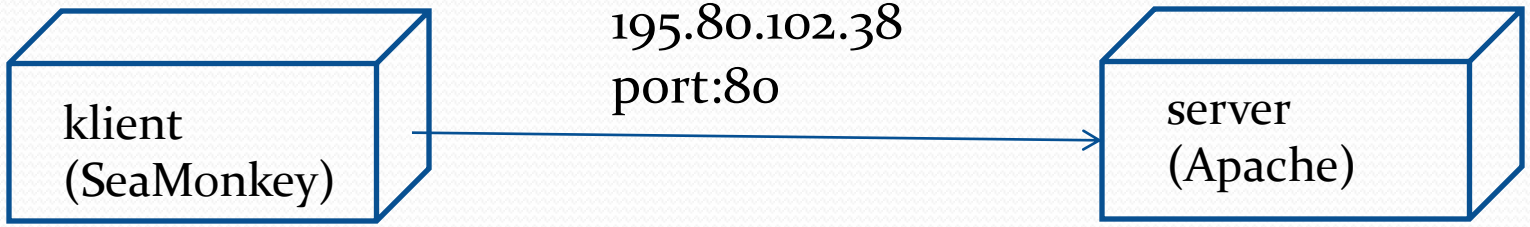


`https://www.eesti.ee/index.html`

195.80.102.38

Veebiserver

- Klient ühendab end serveriga kasutades IP aaddressi ja pordi numbrit

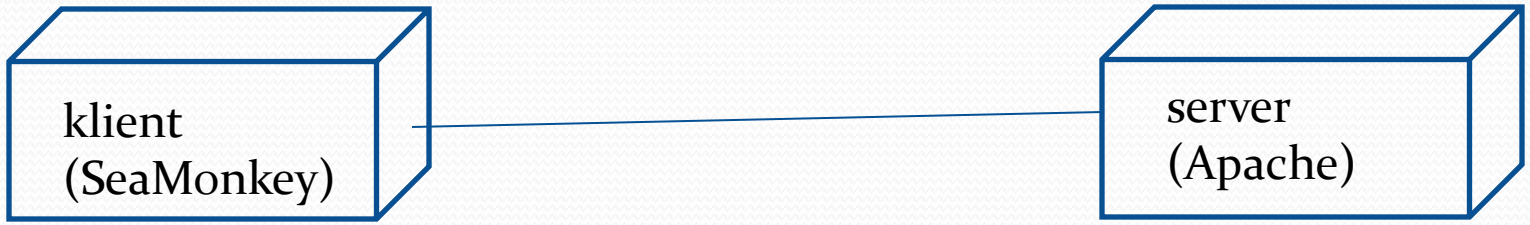


`https://www.eesti.ee/index.html`

195.80.102.38

Veebiserver

- Klient teeb kindlaks nõutud faili ja tee selleni



`https://www.eesti.ee/index.html`

Veebiserver

- Klient saadab HTTP päringu serverile

`https://www.eesti.ee/index.html`

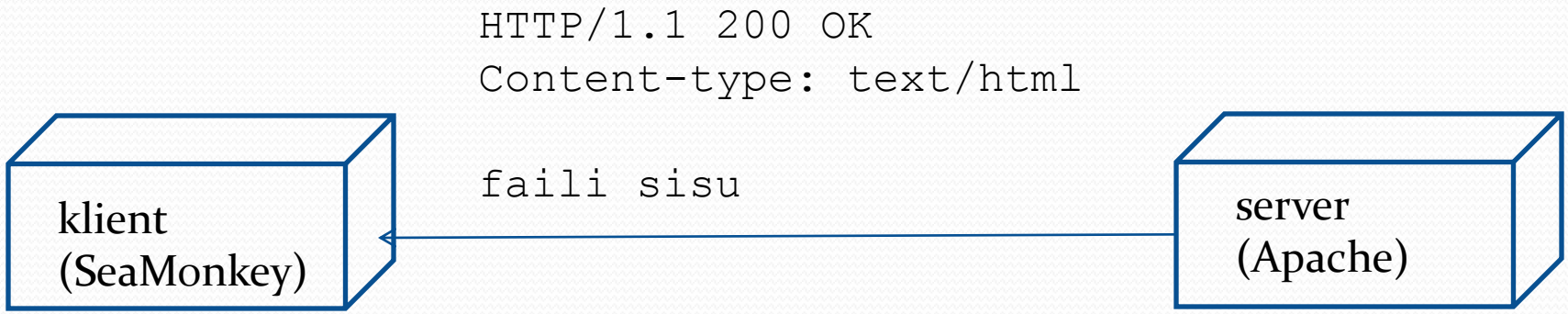
Veebiserver

- Server teeb kindlaks, millist faili saata

`https://www.eesti.ee/index.html`

Veebiserver

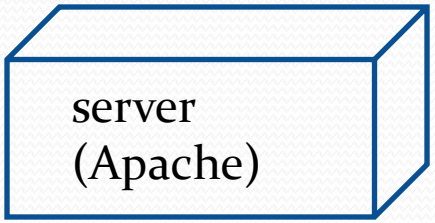
- Server saadab vastuse koodi ja dokumendi



`https://www.eesti.ee/index.html`

Veebiserver

- Ühendus katkestatakse



Veebiserver

Veebiserveri põhiülesanne:

võtta klientidelt vastu HTTP päringuid ja tagastada sobiv allikas (kui võimalik) HTTP vastuses.

Sammud selle töö täitmiseks:

1. Server käivitab TCP tarkvara ja ootab ühenduse päringuid ühe või mitme pordiga.
2. Kui ühenduse soov on tulnud, siis server pühendub selle ühenduse alamülesandele.
3. Alamülesanne loob TCP ühenduse ja võtab vastu HTTP päringu.
4. Alamülesanne kontrollib Host päisevälja, et teha kindlaks, milline „virtuaalne host“ peab selle päringu saama ja kaasab tarkvara selle hosti jaoks.

Veebiserver

5. Virtuaalse hosti tarkvara seab vastavusse päringu URI serveri ressursiga.
6. Kui ressurss on fail, siis hosti tarkvara teeb kindlaks faili MIME tüübi ja loob HTTP vastuse, mis sisaldab faili vastuse kehas.
7. Kui ressurss on programm, siis hosti tarkvara käivitab programmi varustades teda päringu informatsiooniga ja tagastab tulemuse HTTP vastuses.
8. Server hoiab alles informatsiooni päringu ja vastuse kohta (IP aadress ja vastuse seisundikood) tekstifailis.
9. Kui TCP ühendus on veel olemas, serveri alamülesanne jätkab ühenduse jälgimist kuni kindlaksmääratud aja möödumiseni, klient saadab uue päringu või suleb ühenduse.

Veebiserver

• Veebiserverid töötlevad palju päringuid korraga nagu töötaks palju servereid korraga.

• Virtual host – mitu hosti nime võib seada vastavusse ühe IP aadressi.

• Veebiserverid on kirjutatud erinevates programmeerimiskeeltes ja skriptikeeltes.

Veebiserverite ajalugu

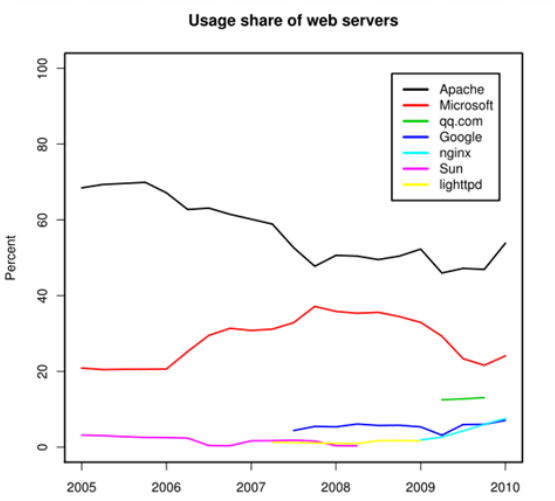
- NCSA http veebiserver
- Apache 1995
- IIS

Veebiserverid (sept. 2008)

TootjaServer		Majutatud saite	Protsent
Apache	Apache	91,068,713	50.24
Microsoft	IIS	62,364,634	34.4
Google	GWS	10,072,687	5.56
lighttpd	lighttpd	3,095,928	1.71
nginx	nginx	2,562,554	1.41
Oversee	Oversee	1,938,953	1.07
Teised -		10,174,366	5.61
Kokku -		181,277,835	100.00

Jaanuar 2010 (Wikipedia)

Vendor	Product	Web Sites Hosted (millions)	Percent
Apache	Apache	111	54%
Microsoft	IIS	50	24%
Igor Sysoev	nginx	16	8%
Google	GWS	15	7%
lighttpd	lighttpd	1	0%



Mitmekihiline infosüsteem

- Andmebaas (AB)
  - Andmed, seosed,...
- Reeglistik (ÄL - äriloogika)
  - Kuidas, kas, miks, ...
- Kasutajaliides (KL)

`http://kodu.neti.ee/~aulis/Opingud/AulisMag/`

Mitmekihiline infosüsteem

- 1 - kihiline
  - Kõik ühes programmis
- 2 - kihiline
  - AB ja ÄL vs KL
  - AB vs ÄL ja KL

Mitmekihiline infosüsteem

- 3- kihiline
  - AB/ÄL/KL
- n - kihiline
  - AB jaotatud mitmeks
  - ÄL jaotatud mitmeks
  - KL jaotatud mitmeks

Mitmekihiline veebiinfosüsteem

Ülesanded:

–HTML lehtede toimetamine kliendi brauserisse

– Kasutaja seansi haldamine

– Dünaamiliste HTML lehtede tootmine

– HTML lehtede infoga varustamine

…

• Kasutada sobivaid keeli, raamistikke, tehnoloogiaid iga kihi jaoks

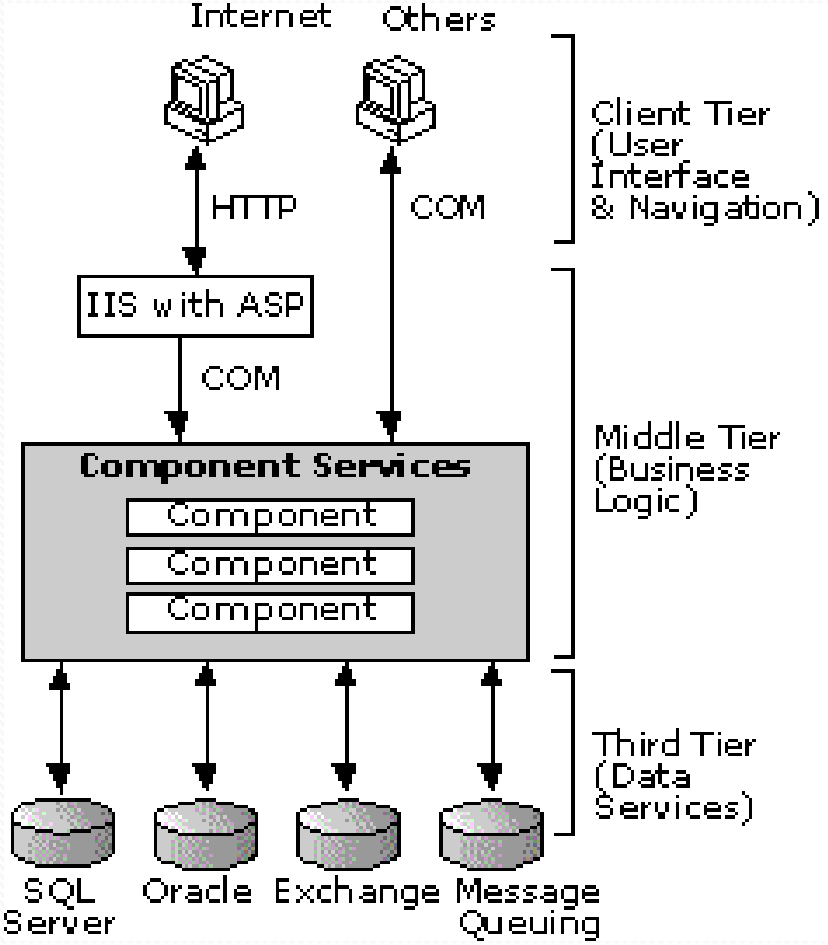
• Kihid võivad olla füüsiliselt eraldatud

Mitmekihiline veebiinfosüsteem

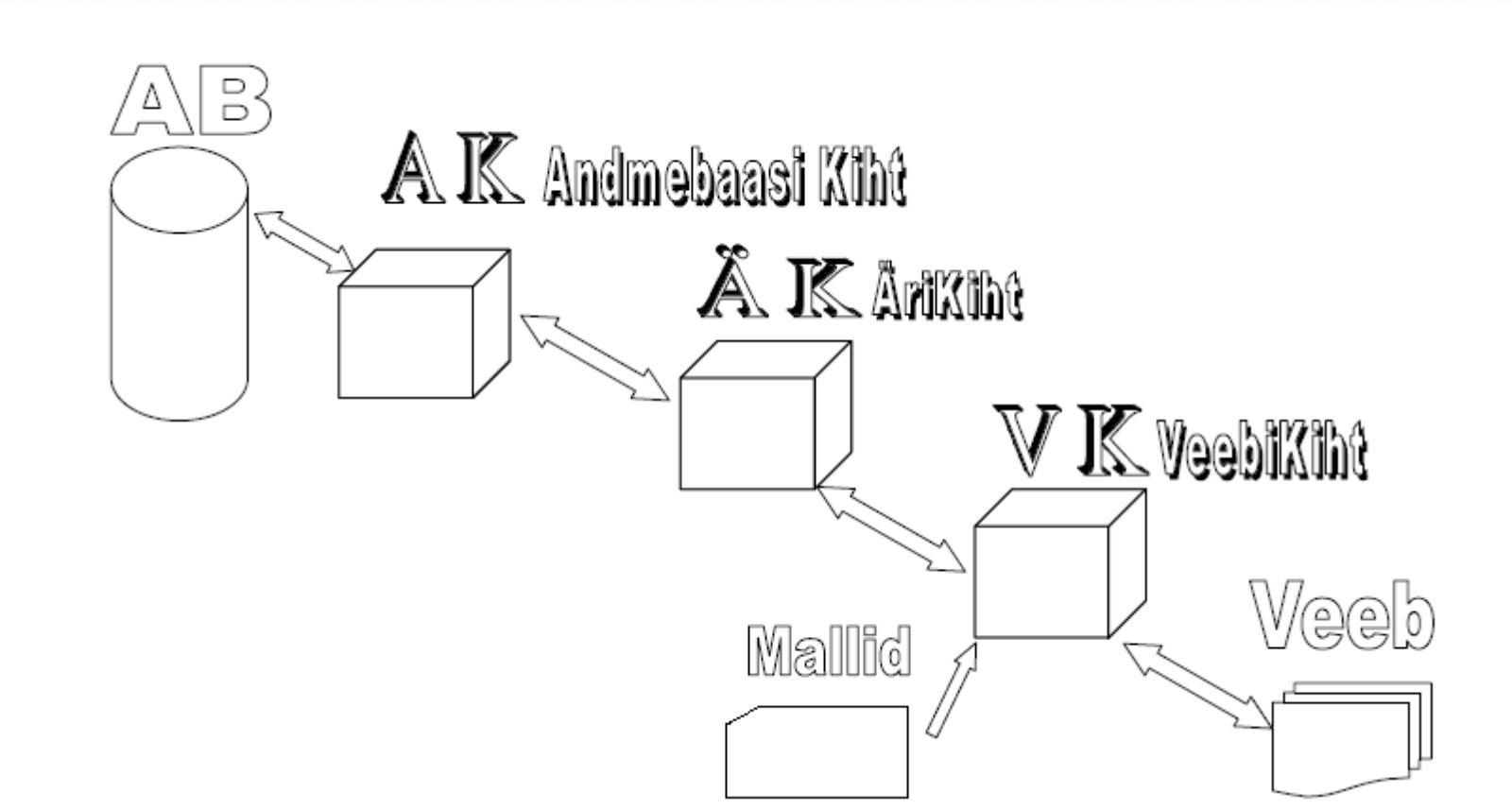
- 1 - kihiline
  - Programm suhtleb kasutajaga
  - Töötleb kasutaja poolt sisestatud andmeid
  - Haldab andmeid
- 2 - kihiline
  - Andmebaas
  - Kasutajaliides

Mitmekihiline veebiinfosüsteem

- 3 - kihiline
  - Kasutajaliides (client tier)
  - Äriloogika kiht (middle tier)
  - Andmebaasi kiht (third tier)

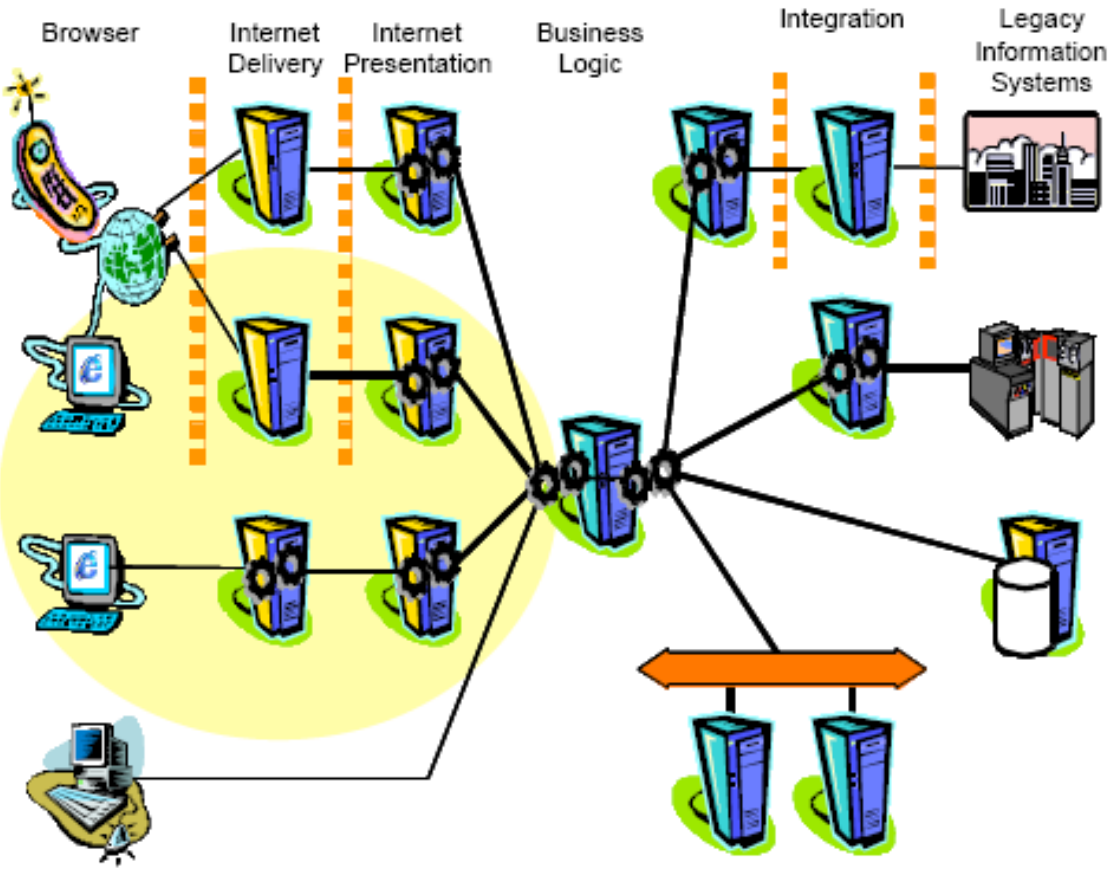


Mitmekihiline veebiinfosüsteem



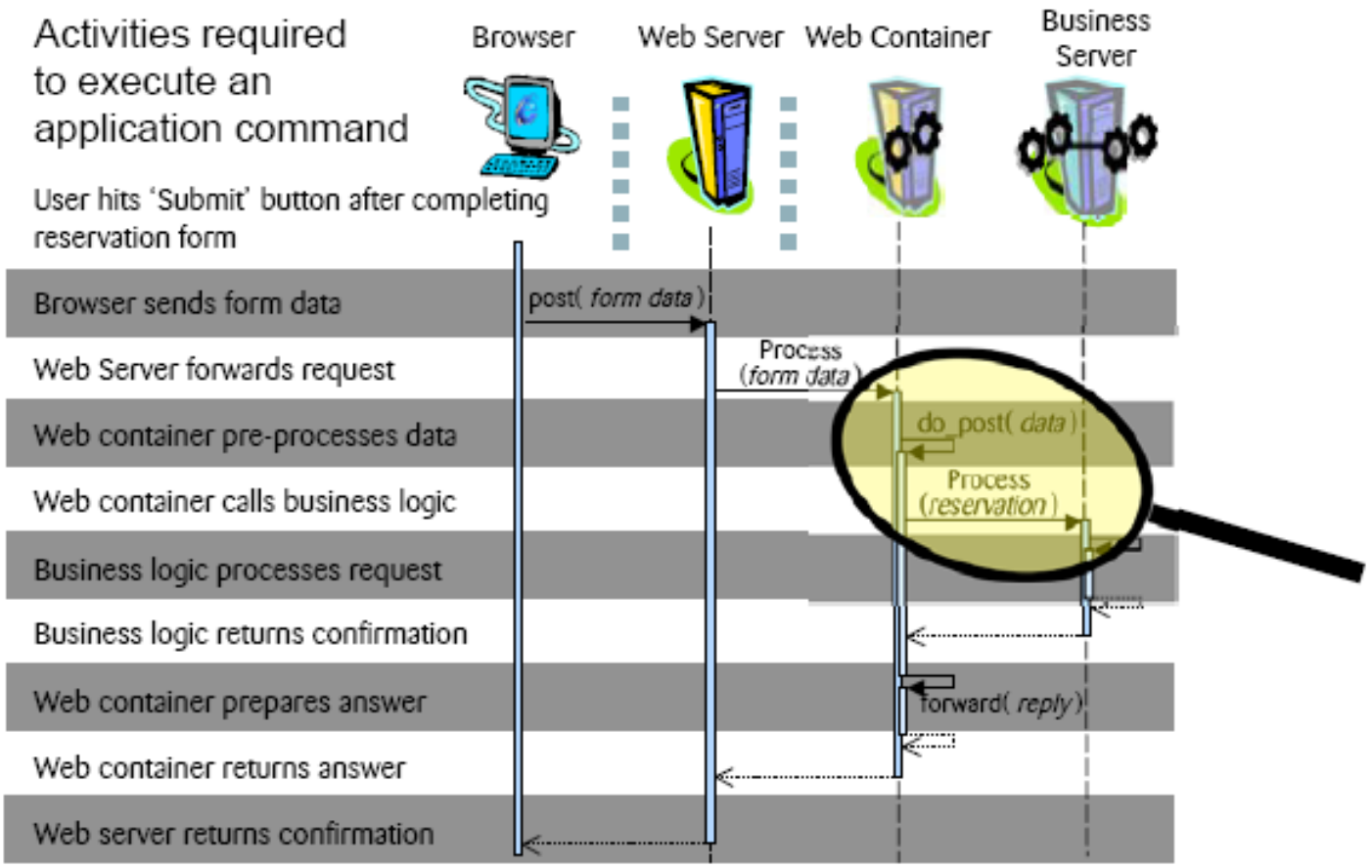
Aulis Sibola

Layered Multi-tier Web Architectures



Wolfgang Emmerich http://sse.cs.ucl.ac.uk

Interaction of the different tiers



Wolfgang Emmerich http://sse.cs.ucl.ac.uk

Mitmekihiline veebiinfosüsteem

Mallide kasutamise eelised:

– Eraldavad kujunduse programmeerimisest

– Kujundaja ja arendaja saavad töötada sõltumatult

– Brauserite eri versioonide jaoks erinevad mallid

Mitmekihilised veebirakendused

Komponentide kasutamise eelised:

– Tegevused on atomaarsed skripti seisukohalt

– Scriptid on ainult liimiks eri komponentide vahel

– Kombineerib omavahel rangelt tüübitud keeled nõrgalt tüübitud keeltega

Mitmekihilised veebirakendused