Ohjelmistoarkkitehtuurit, syksy 2012

Harjoitus 3

26.-27.9.2012

1. Luennoilla käsiteltiin lyhyesti Internetin arkkitehtuuria. Selvitä tarkemmin, millainen on REST arkkitehtuurityyli. Millainen sovellusarkkitehtuuri on RESTFul?

REST:

- Asiakas-palvelin arkkitehtuuri
- Tilaton kommunikointi asiakkaan ja palvelimen välillä, kaikki palvelupyyntöön liittyvä tieto pyynnössä.
  Palvelimella ei säilytetä istunnon tilatietoa.
- Palvelut nähdään resursseina. Resursseilla on globaali identiteetti, jota käytetään palveluun viitattaessa, esim URI
- Välitetään resurssien esityksiä, mukana voi olla meta- ja ohjausdataa,
- Asiakkaalle voidaan välittää ohjelmakoodia suoritettavaksi
- Tiedot voidaan määritellä välimuistikelpoisiksi
- Yhdenmukainen liittymä asiakkaiden ja palvelinten välillä
- Kerrosrakenteisuus

Restful

Webbipalvelu joka on toteutettu REST periaatteiden mukaisesti.
Palvelua luonnehtii:

- Palvelun tunnus (URI)
- Kiinnitetty tiedonvälitystapa (internet media type), esim XML, JSON
- http-metodien käyttö operaatioiden määrittelyyn (GET,PUT,POST,DELETE)
- API hyperteksipohjainen

Lähteitä:
http://en.wikipedia.org/wiki/Representational_state_transfer
Taylorin kalvot “17 Applied Architectures”

2. Tutustu seuraaviin ohjelmistosuunnittelun *Anti-Patterneihin*. Selvitä kustakin ”anti-mallista” millaisesta ongelmasta on kysymys, miten se ilmenee, mitä seuraksia siitä on ja miten ongelmia voisi yrittää korjata?

a. Big Ball of Mud
b. Stovepipe System
c. Lava Flow
d. God Object
e. Dependency Hell

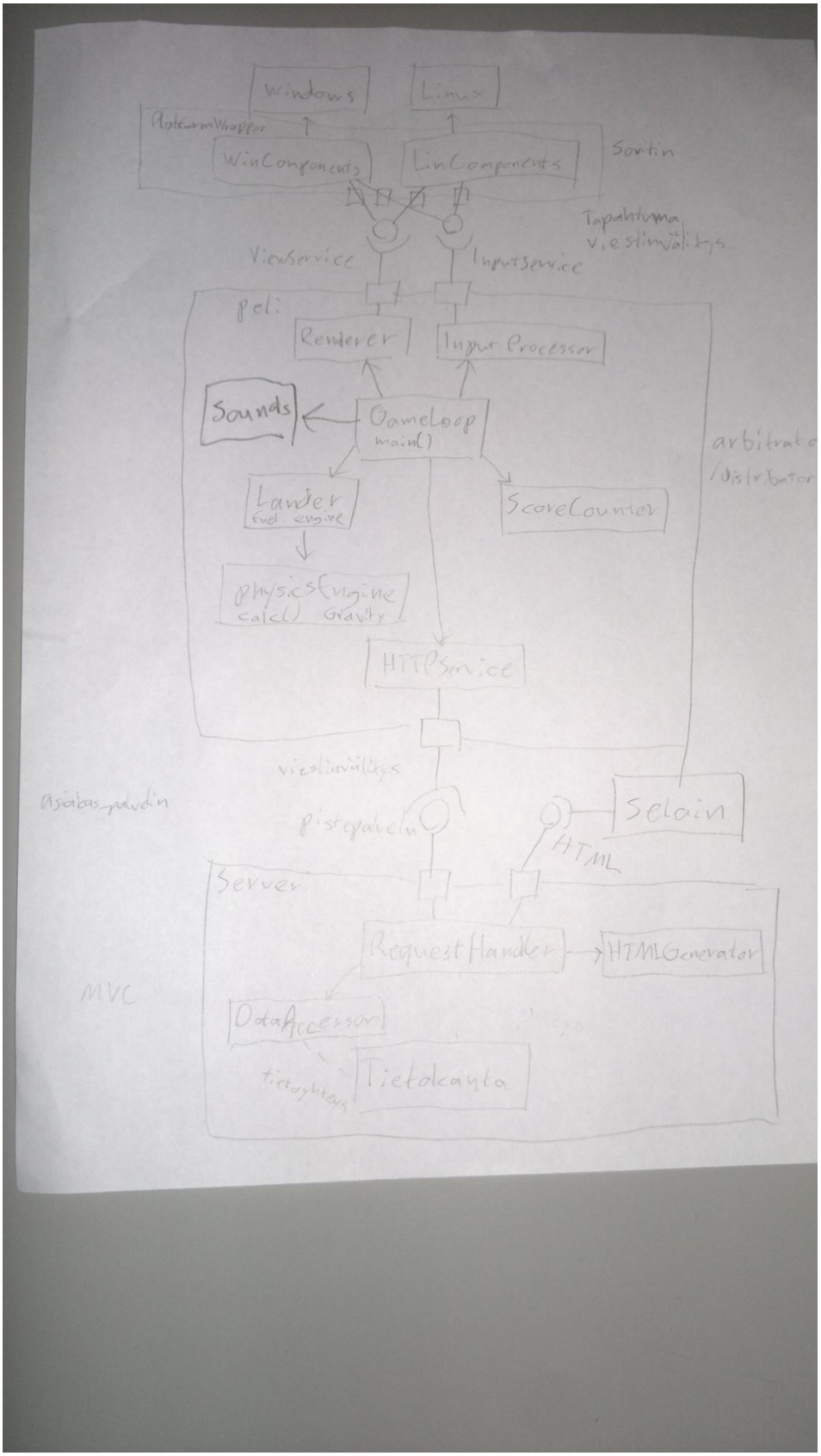
a) http://en.wikipedia.org/wiki/Big_ball_of_mud
b) http://sourcemaking.com/antipatterns/stovepipe-system
c) http://sourcemaking.com/antipatterns/lava-flow
d) http://sourcemaking.com/antipatterns/the-blob
http://en.wikipedia.org/wiki/God_object
e) http://en.wikipedia.org/wiki/Dependency_hell

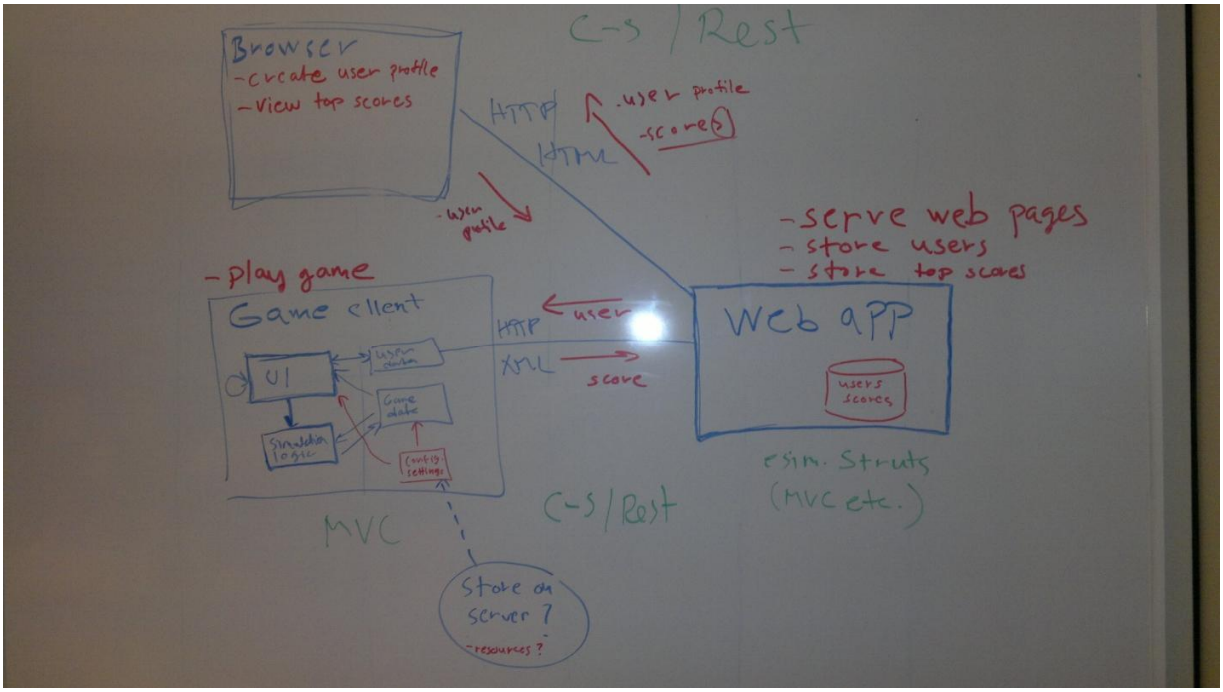
3. Mitkä UML:n kaaviotyypit sopivat mielestäsi
a) *skenaarionäkymän*,
b) *loogisennäkymän*,
c) *prosessinäkymän*,
d) *kehitysnäkymän*,
e) *fyysisen näkymän*
kuvaamiseen?
Perustele. (Lista UML 2:n kaaviotyypeistä löytyy esimerkiksi osoitteesta:
http://www.agilemodeling.com/essays/umlDiagrams.htm.)

Artikkelista Applying 4+1 View Architecture with UML 2 (http://www.sparxsystems.com/downloads/whitepapers/FCGSS_US_WP_Applying_4+1_w_UML2.pdf) löytyy hyvä analyysi kaavioiden sopivuudesta.

4. Aloitteleva pieni peliyritys haluaa toteuttaa ensimmäisenä tuotteenaan seuraavanlaisen pelin: pelissä on tavoitteena ohjata laskeutuja (kuumoduuli) turvallisesti Kuun pinnalle (katso visualisointi Taylorin kirjan luentodiat 16 ja 17 esityksessä *11_Visualizing_Software_Architectures.ppt*). Laskeutujaan vaikuttaa Kuun vetovoimakenttä, jonka kumoavana voimana toimii laskeutujan laskeutumismoottorin työntövoima. Jos laskeutuja tömähtää kuukamaraan liian suurella nopeudella, se tuhoutuu. Polttoainetta on vain rajallinen määrä. Pelaajan laskeutumissuorituksesta annetaan pisteitä siten, että mitä vähemmän aikaa ja polttoainetta laskeutumiseen kuluu, sitä korkeammat pisteet. Peliin liittyy oleellisena osana web-palvelu, johon pelaajat voivat luoda itselleen oman profiilin ja jonne peli tallettaa kunkin pelaajan parhaat tulokset pelissä. Pelaajat voivat käydä web-selaimella katsomassa omia ja muiden tuloksia.
Peliyrityksen työntekijät tuntevat parhaiten Java-teknologiat, ja heidän ajatuksenaan on toteuttaa varsinainen peliosuus PC:llä toimivana Java-ohjelmana. Tarkoitus on, että peliä voi pelata Windowsissa ja Linuxissa.
Laadi alustava arkkitehtuuri pelille ja web-palvelulle. Millaisia komponentteja, konnektoreja ja arkkitehtuurityylejä/-malleja arkkitehtuurissa käytetään?

Pari ehdotusta seuraavilla sivuilla:


Windows
Linux
PlatformWrapper
WinComponents
LinComponents
Sovitin
Tapahtuma viestinvälitys
ViewService
InputService
peli
Renderer
Input Processor
Sounds
GameLoop
main()
arbitrato
Lander
fuel engine
ScoreCounter
physicsEngine
calc() Gravity
HTTPService
viestinvälitys
Asiakasohjelma
pistepalvelu
Selain
HTML
Server
RequestHandler
HTMLGenerator
MVC
DataAccessor
tietohaku
Tietolanta


C-S / Rest
Browser
- create user profile
- view top scores
HTTP
HTML
- user profile
- scores
- user profile
- play game
Game client
UI
User data
Game data
Simulation logic
Config settings
MVC
HTTP
user
XML
score
Web app
users
scores
- serve web pages
- store users
- store top scores
esim. Struts
(MVC etc.)
C-S / Rest
Store on server?
- resources?