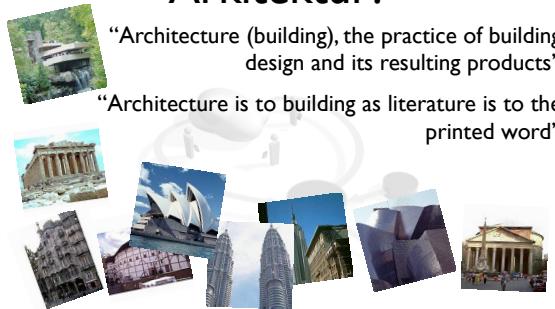


System Arkitektur

Vad är en arkitektur?
Har alla system en arkitektur?
Hur "designar" man en arkitektur?
Olika synsätt på arkitektur.
Mönster

— Design av Samverkande System —

Arkitektur?



"Architecture (building), the practice of building design and its resulting products"

"Architecture is to building as literature is to the printed word"

— Design av Samverkande System —

2

Systemarkitektur

- Fokus på mer eller mindre komplexa datorsystem
- Vi tittar på arkitektur för samverkande system
 - Centraliserade
 - Decentraliserade
 - Många medverkande
 - Gemensam återkopplingsmekanism



— Design av Samverkande System —

3

Vad är det?

"A software architecture is an abstraction of the run-time elements of a software system during some phase of its operation. A system may be composed of many levels of abstractions and many phases of operation, each with its own software architecture"

- En arkitektur är en specifikation av de komponenter som system består av och kommunikationen mellan dem
- Beskriver strukturen hos systemet på en hög nivå
- Ligger på en abstraktionsnivå där man kan se systemet i sin helhet
- Detaljer om implementationen är gömda
- Strukturen måste stödja både funktionella och icke-funktionella krav som finns på systemet
- Bör utvecklas tidigt i designprocessen

— Design av Samverkande System — 4



Icke-funktionella krav

- Verifierbara krav på system hur det ska fungera och inte vad systemet ska göra
 - Prestanda, tillförlitlighet, säkerhet, mm
- Designmål
 - Icke-funktionella krav som inte användaren direkt märker av
 - Svåra att mäta

— Design av Samverkande System — 5



Icke-funktionella krav

- Kapacitet
 - Nätverksprestanda
 - Throughput, overhead, bandwidth,...
 - Fördräningar, jitter,...
 - Nätverkseffektivitet
- Säkerhet
 - Rättigheter
 - Styrka identiteten
 - Icke förnekande
 - Integritet
 - Privatitet
 - Logging

— Design av Samverkande System — 6



Ickefunktionella krav

- Tillgänglighet
- Pålitlighet
- Svarstid
- Användbarhet
- Standarder
- Systemets arbetsmiljön
 - Hårdvara och mjukvara
 - Endast "open source"...
 - Fysikaliska miljö
- Konfigurerbart
- Lättunderhåll

— Design av Samverkande System — 7

Designmål...

- Framtidssäkert
 - Utökningsbart, modifierbart
- Produktbarhet
- Implementerbart inom budgetramarna?
- Portabilitet
- Interoperabilitet
- Återanvändningsbart
- Underhållbarhet
- Förståbarhet
- Implementerbart med befintlig teknik?
- Testbarhet

— Design av Samverkande System — 8

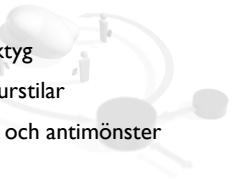
Olika perspektiv...

- Abstraktionsnivåer
 - Data, bearbetning, kopplingar
- System och delsystem
- Luckham, Vera & Meldal (1995), Three Concepts of System Architecture
 1. Object connection architecture
 2. Interface connection architecture
 3. plug and socket architecture
- Krutchen (1995), 4+1 View Model of Architecture
 1. Logical view – supports the functional requirements (objects and object classes)
 2. Process view – concurrency and synchronization
 3. Physical view - mapping the software onto the hardware
 4. Development view – software's static organization in its development environment
 5. Scenarios - instanser av tänkta användningar av systemet (upptäcka, validera och illustrera)

— Design av Samverkande System — 9

Bakgrund

Design...



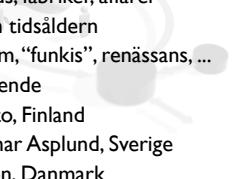
- Designverktyg
 - Arkitekturstilar
 - Mönster och antimönster

— Design av Samverkande System —

10

Systemarkitektur

Arkitekturstilar...



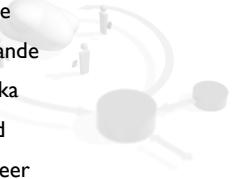
- Abstraktionsnivå
 - Stadsarkitekt, inredningsarkitekt
- Problem-/domänberoende
 - Bostadshus, fabriker, affärer
- "Mode" och tidsåldern
 - Modernism, "funkis", renässans, ...
- Personberoende
 - Alvar Aalto, Finland
 - Erik Gunnar Asplund, Sverige
 - Jorn Utzon, Danmark

— Design av Samverkande System —

11

Systemarkitektur

Arkitekturstilar...



- ...för nätverksbaserade system mer än client-server och P2P!
 - Data-flöde
 - Replekerande
 - Hierarkiska
 - Mobil kod
 - Peer-to-Peer

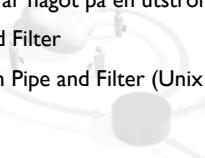
— Design av Samverkande System —

12

Arkitekturstilar...

Data-flöde

- Ström av data mellan komponenter (filter), filter läser från en inström och gör något med datat och producerar något på en utström
 - Pipe and Filter
 - Uniform Pipe and Filter (Unix pipes)



— Design av Samverkande System —

13

Arkitekturstilar...

Replekerande

- Replicated Repository (CVS)
 - Decentraliserade servrar (uppfattas som en av klienterna)
 - Distribuerade filssystem
 - Cache



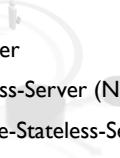
— Design av Samverkande System —

14

Arkitekturstilar...

Hierarkiska

- Client-Server
- Layered systems och Layered-Client-Server (TCP/IP)
- Client-Stateless-Server
- Client-Cache-Stateless-Server (NFS)
- Layered-Client-Cache-Stateless-Server (DNS)
- Remote Session (Telnet, ftp)
- Remote Data Access (Remote DB-server)



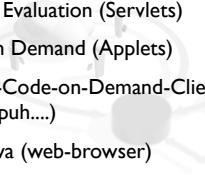
— Design av Samverkande System —

15

Mobil kod

- Virtual Machine (Ingen nätverksstil men basen för andra inom denna grupp)
 - Remote Evaluation (Servlets)
 - Code on Demand (Applets)
 - Layered-Code-on-Demand-Client-Cache-Stateless-Server (puh....)
 - HotJava (web-browser)
 - Mobile Agent
 - REV+CoD+VM

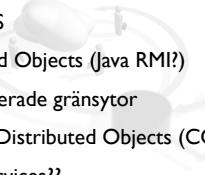
— Design av Samverkande System — 16



Peer-to-Peer

- Event-Based Integration
- C2
 - EBI+LCS
- Distributed Objects (Java RMI?)
 - Väldefinerade gränsytor
- Brokered Distributed Objects (CORBA, SOA, ...)
 - Web-services??
 - GRID-computing??

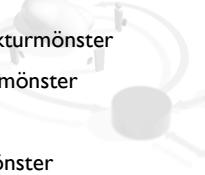
— Design av Samverkande System — 17



Designverktyg

- Mönster
 - Arkitekturmönster
 - Designmönster
 - Idiom
 - Antimönster

— Design av Samverkande System — 18



Mönster

“Mönster”

- “A pattern is a named nugget of instructive information that captures the essential structure and insight of a successful family of proven solutions to a recurring problem that arises within a certain context and system of forces.”
- "...every pattern we define must be formulated in the form of a rule which establishes a relationship between a context, a system of forces which arises in that context, and a configuration, which allows these forces to resolve themselves in that context."
- Alexandrian form & GoF form

— Design av Samverkande System — 19



Mönster

“Mönster”

- Identifierbara egenskaper hos mönster
 - Namn
 - Problem
 - Kontext
 - Vad som gör det giltigt, begränsningar, konflikter med andra
 - Lösningar - statiska relationer och dynamiska regler (instruktioner)
 - Exempel
 - Resulterande kontext (bra och dåliga saker)
 - Motiv
 - Relaterade och liknande mönster
 - Kända tillämpningar av mönstret i existerande system

— Design av Samverkande System — 20



Mönster

“Antipatterns”

- Represents a "lesson learned." Initially proposed by Andrew Koenig in the November 1995 C++ Report
- There are two notions of "anti-patterns."
 1. Those that describe a bad solution to a problem which resulted in a bad situation.
 2. Those that describe how to get out of a bad situation and how to proceed from there to a good solution.
- Typer av antimönster
 - Arkitektur – focus on the system-level and enterprise-level structure of applications and components.
 - Management – identify some of the key scenarios (in human communication) where these issues are destructive to software processes.
 - Utveckling – describe useful forms of software refactoring.

— Design av Samverkande System — 21

