System Arkitektur

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

Arkitektur?

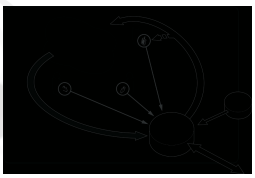
"Architecture (building), the practice of building design and its resulting products"

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

Bakgrund

Systemarkitektur

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



Systemarkitektur

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

Systemarkitektur

Ickefunktionella krav

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

Systemarkitektur

Ickefunktionella krav

- Kapacitet
  - Nätverksprestanda
    - Throughput, overhead, bandwidth,…
    - Fördröjningar, jitter,…
  - Nätverkseffektivitet
- Säkerhet
  - Rättigheter
  - Styrka identiteten
  - Icke förnekande
  - Integritet
  - Privathet
  - Loggning

Ickefunktionella krav

Systemarkitektur

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

Designmål…

Systemarkitektur

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

Olika perspektiv…

Systemarkitektur

- 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 syncronization
  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 användingar av systemet (upptäcka, validera och illustrera)

Design…

Bakgrund

- Designverktyg
  - Arkitekturstilar
  - Mönster och antimönster

Arkitekturstilar…

Systemarkitektur

- 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

Arkitekturstilar…

Systemarkitektur

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

Data-flöde

Arkitekturstilar…

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

Replekerande

Arkitekturstilar…

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

Hierarkiska

Arkitekturstilar…

- 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)

Mobil kod

Arkitekturstilar…

- 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

Peer-to-Peer

Arkitekturstilar…

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

Designverktyg

Mönster

- Mönster
  - Arkitekturmönster
  - Designmönster
  - Idiom
  - Antimönster

"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

"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

"Antipatterns"

Mönster

- 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.