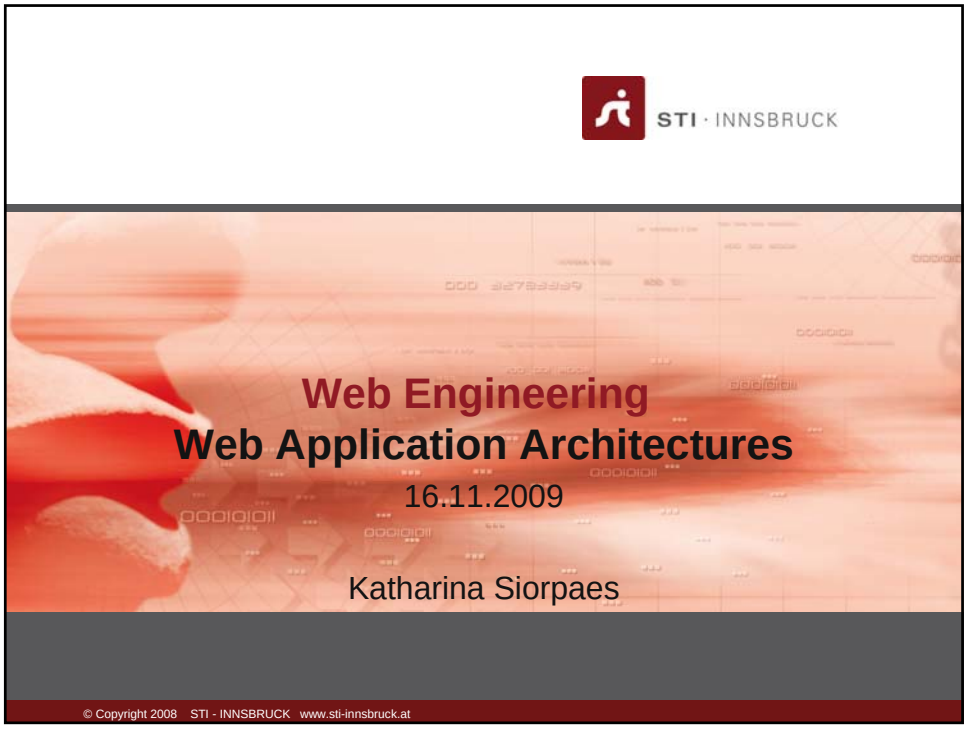


Overview

- Web Application Architectures
- Technology-aware design of Web applications

What is an architecture?

INTRODUCTION

Software Architectures

- “Architecture is defined [...] as the fundamental organization of a system, embodied in its components, their relationships to each other and the environment, and the principles governing its design and evolution.”(IEEE Architecture Working Group, P1471, 1999)
- Architectures describe structure
  - Components of software systems, their interfaces and relationships
  - static as well as dynamic aspects
  - blueprint of software system
- Architectures connect software development phases
  - requirements mapped iteratively to components and their relationships

Software Architectures

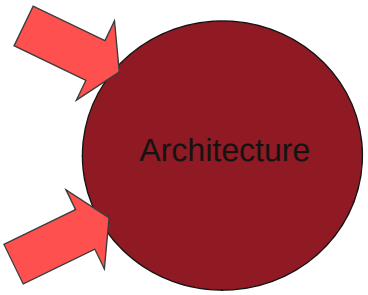
STI · INNSBRUCK

- "Architecture is the set of design decisions [...] that keeps its implementers and maintainers from exercising needless creativity."(Desmond F. D'Souza and Alan C. Wills, 1999)
- Architectures describe different viewpoints
  - conceptual view: entities of application domain and their relationships
  - process view: system runs, concurrency, synchronization
  - implementation view: software artifacts (subsystems, components, source code)
  - runtime view: components at runtime and their communication
- Architectures make systems comprehensible and controllable
  - structuring according to different viewpoints
  - enables communication between different stakeholders

Developing Architectures
Influences on Architectures

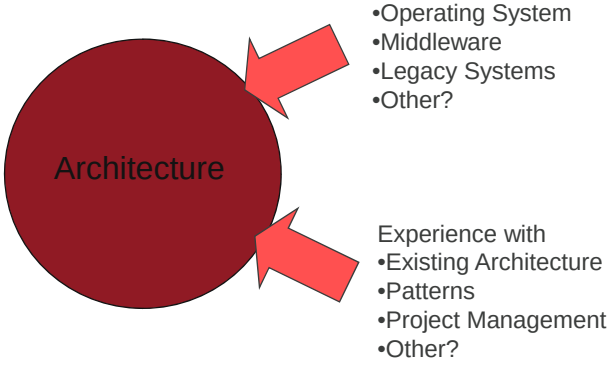
STI · INNSBRUCK

Functional Requirements
- Clients
- Users
- Other Stakeholders



Non-Functional Requirements
- Performance
- Scalability
- Reusability
- Other?

Developing Architectures
Influences on Architectures



Technical Aspects
•Operating System
•Middleware
•Legacy Systems
•Other?

Experience with
•Existing Architecture
•Patterns
•Project Management
•Other?

Developing Architectures

- Remember, requirements are always subject to change.
  - Organizational & Environment changes
  - Ambiguous requirements initially
- Thus, iterative approaches are the suggested means of development
  - Pro: Helps to mitigate design risks
  - Caution: Doesn't guarantee a good architecture

Patterns & Frameworks

STI · INNSBRUCK

- Patterns describe recurring design problems
- 3 types of patterns
  - Architecture patterns (e.g. MVC)
  - Design patterns (e.g. Publisher-Subscriber)
  - Idioms (e.g. Counted-Pointer in C++)
- They are a guideline, implementation must be grounded to the specific problem
- Patterns need to be “integrated” amongst each other!

Patterns & Frameworks

STI · INNSBRUCK

- Frameworks: another option to reuse existing architecture
  - something that provides you a frame to be filled!
- Reuse of existing software objects that just need to be properly configured
- Bound to a specific technology
  - Require training
  - High cost of switch
  - Level of customization not always acceptable

Specifics of Web apps

STI · INNSBRUCK

- High quality demands
  - Security
  - Extensibility
  - Adaptability
  - Stability
  - Performance
- Broad range of technical solutions that can be integrated.
  - Hard to evaluate quality demands (various components)
  - Hard to solve (where is the problem?)

Specifics of Web apps

STI · INNSBRUCK

- Inhomogeneity and immaturity of technical solutions
  - Fast product lifecycles
  - Many components: open source – quality?
  - Lack of standards
- Global requirements
  - Multi linguality
  - Cultural adaptation?
    - E.g. Google for Korea

WEB APPLICATION ARCHITECTURES

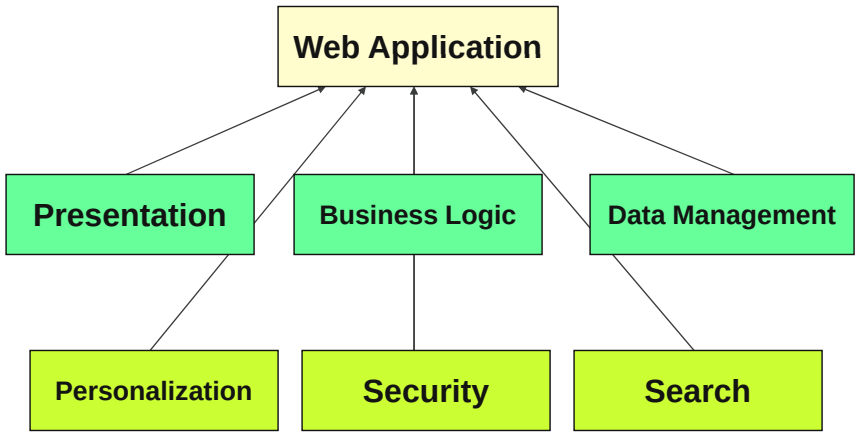
Architecture Types

- Layering Aspect
  - "Separation of concerns"
  - How many concurrent users are you serving?
  - Shared needs among multiple applications? (e.g., security)
- Data Aspect
  - What kind(s) of data are you delivering?
    - Structured vs. non-structured
    - On-demand vs. real-time
  - What are the bandwidth requirements?
    - Size & nature of data
    - Again, audience concerns

Architecture Types

- Web Platform Architecture (WPA)
  - Platform = Infrastructure
    - Hardware
    - Software modules & configurations
    - Choice of software platform (e.g., J2EE, .NET)
- Web Application Architecture (WAA)
  - Conceptual view of how key business processes and needs are separated & implemented
  - Often domain-specific
  - Greater complexity requires greater modularity

Example of a WAA



Generic Web (Platform) Architecture

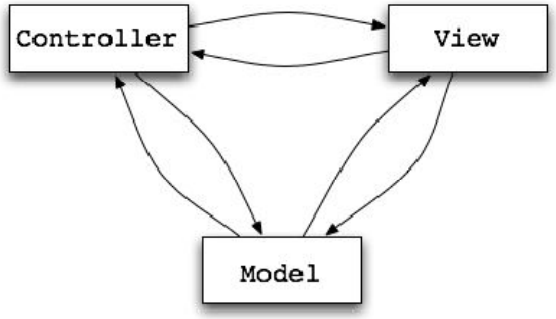
STI · INNSBRUCK

- The Web “platform” is based on
  - TCP/IP
  - HTTP
  - HTML
- It’s essentially a Client/Server architecture!
  - In term of patterns one of the simplest one
- But still thing can get complex…
  - Components on the network (firewall, proxy, load balancer)
  - Components in the intranet (Web server, application server, data base, legacy systems, web services)

Model View Controller

STI · INNSBRUCK

- Architectural Pattern from Smalltalk (1979)
- Decouples data and presentation
- Eases the development



Model View Controller

STI · INNSBRUCK

- Model
  - encapsulate application state
  - responds to state queries
  - exposes application functionality
  - notifies views of changes
- View
  - renders the models
  - requests updates from models
  - sends user interaction to controller
  - allows controller to select view
- Controller
  - defines application behavior
  - maps user actions to model updates
  - selects view for response
  - one for each functionality

Web Architectures: Specifics

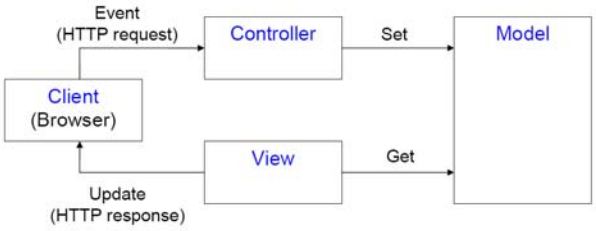
STI · INNSBRUCK

- Technological constraints
  - HTTP
- Broad variety of technical solutions
  - application servers, proxies, firewalls, legacy applications
  - checking of quality difficult
    - e.g., performance depends on various components, like database, network bandwidth, processor, memory, code, …
  - improvement of quality difficult
    - e.g., code performance may not change overall performance substantially
- Technical solutions inhomogeneous and immature
  - short product life cycles
  - missing standards impede component integration from different manufactures
  - many solutions are open source: continuity of development, extendibility, …
- Global access to Web applications
  - internationalization, cultural differences

Model-View-Controller 2 (MVC 2)

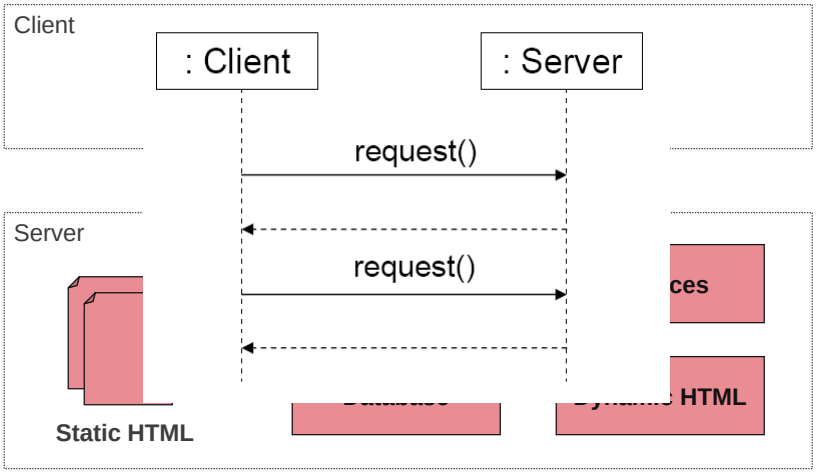
STI · INNSBRUCK

- **Adaptation of MVC for the Web**
  - stateless connection between the client and the server
  - notification of view changes
  - re-querying the server to discover modification of application's state

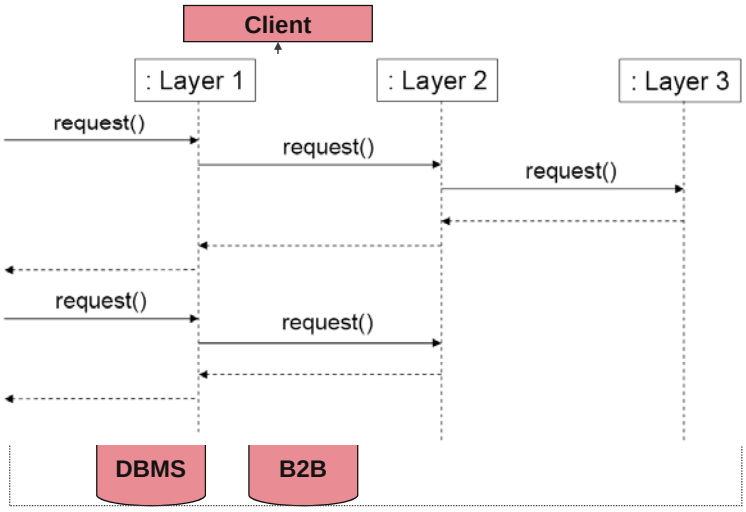


Client/Server (2-Layer)

STI · INNSBRUCK



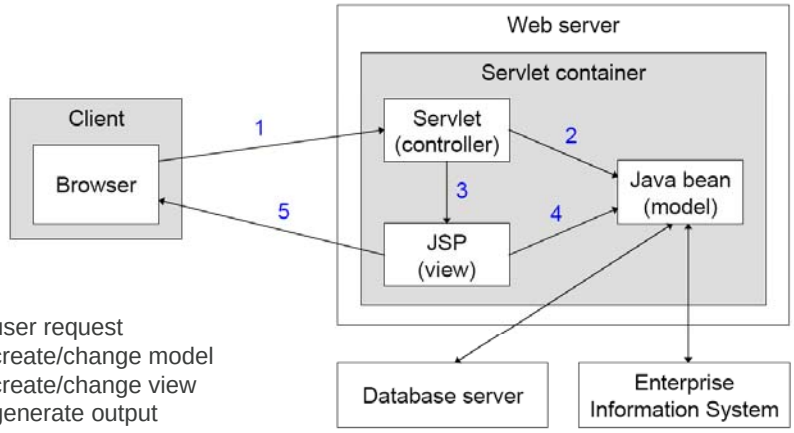
N-Layer Architectures



Why an N-Layer Architecture?

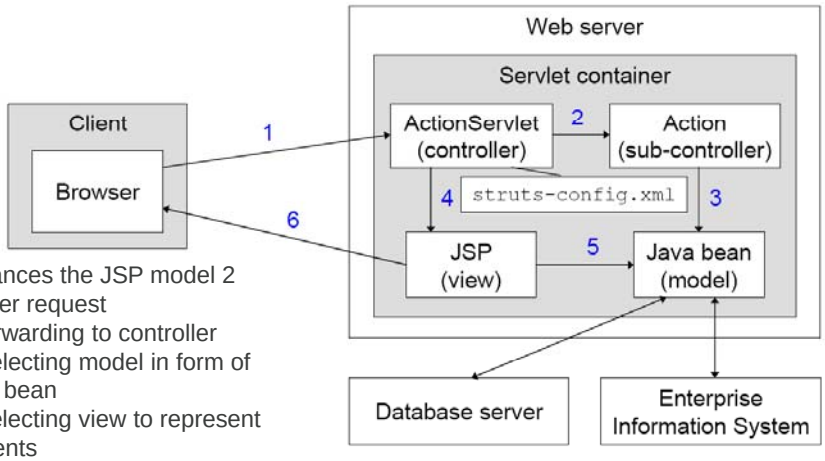
- Separating services in business layer promotes re-use among applications
  - Loose-coupling – changes reduce impact on overall system.
  - More maintainable (in terms of code)
  - More extensible (modular)
- Trade-offs
  - Needless complexity
  - More points of failure

JSP-Model-2 Architecture



1. user request
2. create/change model
3. create/change view
4. generate output
5. server response

Struts Architecture



Enhances the JSP model 2

1. user request
2. forwarding to controller
3. Selecting model in form of Java bean
4. Selecting view to represent contents
5. generate output
6. server response

WRAP-UP

Things to keep in mind (or summary)

- Good design of architecture is crucial
- You can leverage on patterns and frameworks
  - Both have advantages and disadvantages
- Design is constrained on Web “infrastructure”
- MVC is the most commonly used pattern

Bibliography

- Mandatory reading
  - Chapter 4

Questions?

STI · INNSBRUCK

Web Engineering
Technology-aware design of Web Applications

16.11.2009

Katharina Siorpaes

Motivation

STI · INNSBRUCK

- Experts often have a limited view on their area of expertise
  - Web designer:

  Colorful, playful pages but user might not find the required information easily.
  - Researchers:

  Logically structured navigation but content representation is rather boring.
  - Developpers:

  Creation of Web applications without knowledge in HCI.
- An integrated view is required!

Evolutionary view on Web design

- HTML has many disadvantages compared to „real“ hypertext systems
- HTML is misused oftentimes: e.g. Long documents while one node should only be a small portion of content.
- HTML mixes hypertext, document structure, and layout
- HTML is text-centered, other media often only applied as link targets.
- Simplicity is the reason for the success.

Developper view

- Programmable Web
  - Dynamicity through forms with browser-side or server-side processing.
  - Active Server Pages
  - Java Server Pages
  - Java Applets
  - Plug-ins for the dynamic representation of multimedia contents, e.g. Flash
- Distributed programming
  - Interprocess communication
  - Remote procedure call
  - Remote method invocation
  - CORBA
  - Publish-subscribe architectures
  - Web services

Presentation design

- What is presented?
  - Content
  - Navigation
    - Where am I?
    - Where was I?
    - Where can I go?
- How is it presented?
  - Problem: authors describe look not semantics.
  - Example <b> vs.

Interaction design

- Interaction design involves
  - Visual, dynamic, functional Elements of a Web application

Interaction design: links

STI · INNSBRUCK

- Links can have different semantics, even when they look alike.
- Semantics:
  - Unclarity of meaning
- Navigation:
  - Single or multiple targets?
  - Target within or outside of website?
  - Real navigation or will only the text be changed on the current website?
- Download:
  - Which document is retrieved?
  - Do I have the required tools?
- Process:
  - Will it be possible to undo?

Interaction design: complex activities

STI · INNSBRUCK

- Multi-stage processes must be visualized.
- Comparison hypertext and business process

	Hypertext	Business process
Navigation	User chooses the order of pages. Page is visited by clicking a link.	Process defines which activity is next. Often sequential; can be more complex.
Leaving website / activity	A page is left but the status of the page doesnt change. „Ending“ a page is not possible	Leaving an activity: ensure that it is completely ended. Ending is a part of the process definition.
Uptaking / undoing	Going back to a page. Re-loading page.	Status is re-taken; undo has to be explicitly called.

Functional design

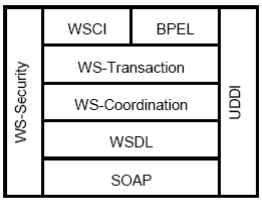
STI · INNSBRUCK

- Web application
  - Interaction of various components (DB, workflow systems)
  - Enterprise Application Integration
- Integration is required on various levels:
  - Data
  - Application
  - Process
- Achieved through using communication paradigms and middleware

Functional design

STI · INNSBRUCK

- Web applications in the future
- Across enterprise boundaries
- Composition of services
- Building on open standards



Sustainability

STI · INNSBRUCK

- Trend: various devices! Not only browsers, but PDAs, mobile phones, etc.
- Problem for sustainability: new characteristics of devices
  - Smaller displays, more colors, new usage philosophy, new application area...
- Solution from the Web
  - Optimize apps for target platforms

Sustainability

STI · INNSBRUCK

- Situation-centered applications
  - Adaptation to a new situation on all layers
    - Presentation
    - Navigation
    - Functionality
- Device independent applications
  - End devices
    - Support various markup languages
    - Varying performance
- Reusability

Questions?

STI · INNSBRUCK