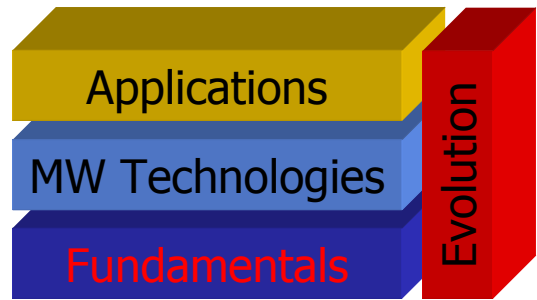


Summary

Mariano Cilia
cilia@informatik.tu-darmstadt.de

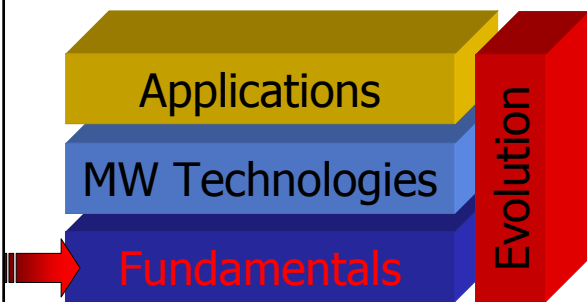
1

Building Blocks



2

Building Blocks



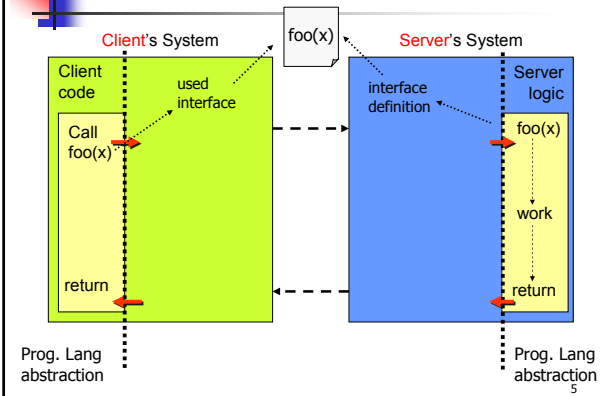
3

Communication Mechanisms

- Synchronous
- Asynchronous

4

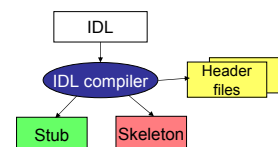
RPC - Abstraction



5

Remote Procedure Call (RPC)

- IDL: Interface definition
- IDL compiler produces
 - stub that is linked with caller
 - skeleton that is linked with the server
 - header files to be included in calling program



6

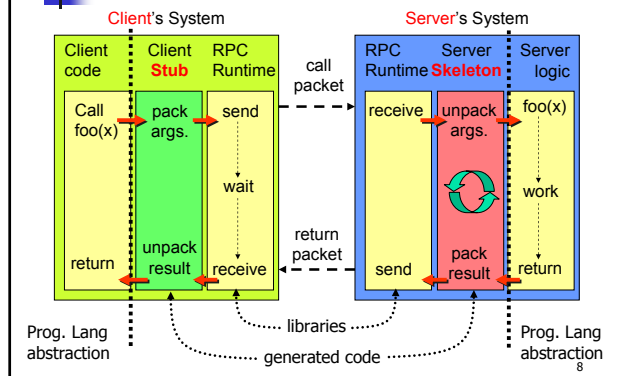
What are stubs and skeletons for?

- Clients call methods on a client stub
- Stub converts method call into network protocol
 - Marshals any method arguments
 - eXternal Data Representation (XDR) to standardize data representation
- Server object receives the method request through "server skeleton"
 - RPC runtime unmarshals method arguments, passes them to the implementation of the server
- Server object does its thing, returns a value
 - RPC runtime intercedes, marshals the result and send it back to the client stub
- Client stub unmarshals results, performs a regular method return to the client app

[Farley'04]

7

RPC Walk-through

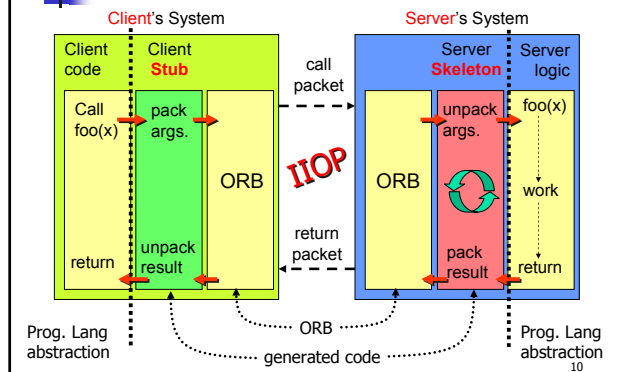


CORBA - RMI

- Remote Method Invocation (RMI)
- Language Interoperability
- IDL
 - language-independent code
- ORB as intermediary
- IIOP as communication protocol

9

CORBA



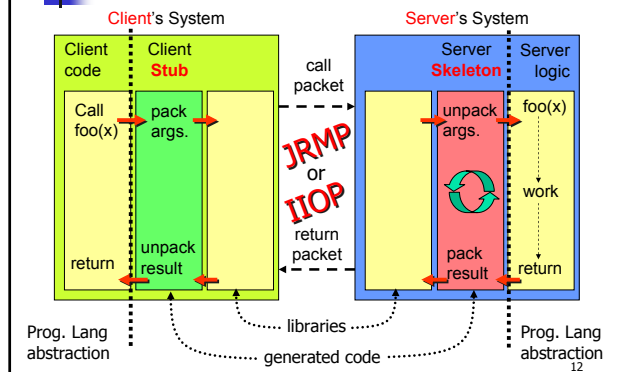
10

Java - RMI

- Set of Java interfaces for defining and using RMI objects
- RMI/JRMP default network protocol
 - Restricted to Java Objects
- RMI/IIOP allows interaction with CORBA objects (multi-lang)
- rmic
 - compiler that generates stub/skeleton

11

Java - RMI



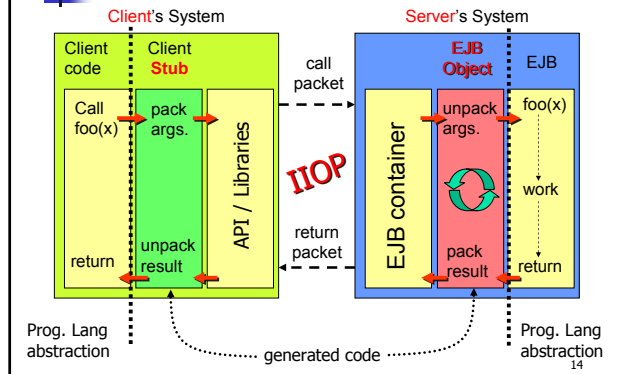
12

EJBs - RMI

- Logic encapsulated
- Not aware of communication protocol
- Wrapper/Skeleton is generated
 - EJBObject

13

EJBs - RMI



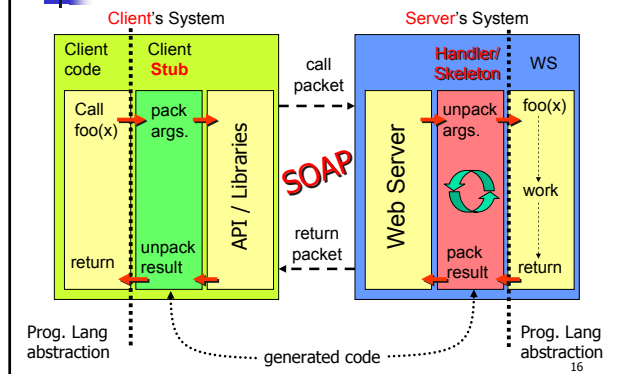
14

Web Services - SOAP

- Use of XML for data representation
 - platform-agnostic
- Language-independent
- Platform-neutral

15

Web Services - SOAP



16

Modes of Interaction

- Information flows from producer to consumer
- Other interaction modes
 - Initiator at the server (where the data changes)

		Initiator of Interaction	
		Consumer	Producer
Knowledge about Counterpart	Full	Request/Reply	One-to-One Message
	No	Anonymous Request/Reply	Event-based dissemination

17

Messaging - MOM

- Asynchronous communications
- Models:
 - Point-to-Point
 - Pub/sub
 - Topic-based
 - Subject-based
 - Content-based
 - Concept-based

18

Messaging – Loosely coupling

- Dimensions of decoupling
 - Space
 - Components do not need to know peer address
 - Indirect addressing, implicit invocation, location transparency
 - Flow
 - Components are not blocked if they send a msg
 - Decouple communication (routing) from computation
 - Time
 - Components must not be available at the same time to communicate
 - e.g. Message queues

19

Messaging Infrastructure

- Compared w/databases

20

Messaging Infrastructure

- Compared w/databases

21

Semantic Data Exchange

- Messaging = is about data exchange!
- XML is the standard format
- Common interpretation basis for data
 - Use of ontologies
- Implicit modeling assumptions have to be made explicit
- Representation of data plus additional semantic metadata
- Integration of conversion functions
- Even for method invocation (in loosely-coupled apps) contextual information is fundamental

22

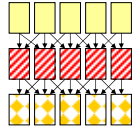
Building Blocks

23

Multi-tier: What is Involved?

24

A game of boxes and arrows



There is no problem in **system design** that cannot be solved by **adding a level of indirection**.
There is no **performance** problem that cannot be solved by **removing a level of indirection**.

- Each box represents a part of the system.
- Each arrow represents a connection between two parts of the system.
- The **more boxes**, the more modular the system: **more opportunities for distribution and parallelism**. This allows encapsulation, component based design, reuse.
- The more boxes, the more arrows: more sessions (connections) need to be maintained, **more coordination** is necessary. The system becomes **more complex** to monitor and manage.
- The more boxes, the greater the number of context switches and intermediate steps to go through before one gets to the data. Performance suffers considerably.
- System designers try to balance the capacity of the computers involved and the advantages and disadvantages of multiple layers.

25

Benefit of n-Tier Architecture

- Flexible distribution
 - communication among processes is via messages, location is functionally immaterial
- Flexible configuration
 - processes can be located to optimize for performance, availability, manageability, etc.
- Flexible control
 - independent control of the three functions, by varying number of threads in each process, relative speed can be tuned

26

Cost of n-Tier Architecture

- Communication overhead
 - message between processes 1500-15000 instructions, local procedure call 50 instr.
 - Even simple interactions require at least two round trips
 - (presentation server - workflow controller and workflow controller - transaction server)

27

J2EE Platform

- Multi-tier platform
 - J2EE has become widely successful
 - M\$ attempting to fight with .NET
- Specifications governed by Consortium
- EJBs
 - Component-based development
 - Encapsulation of business logic
 - Reuse
 - Container - Component model (inversion of control!)
 - Code portability (Application Verification Kit, AVK)
 - Application portability across multi-vendor J2EE app servers (20,000+ tests)
- Standardized benchmarks
 - SPECjAppServer

28

J2EE 1.4 – XML, Java and WS

- Convergence of XML and Java into Web Services
 - WS-I (basic profile for interoperability)
 - Java API for XML-based RPC (JAX-RPC)
 - Enterprise Web Services spec (JSR-109)
 - Lot of (automatic) code generation

29

J2EE - Outlook

- Research needed on self-tuning/adapting systems
- Caching appears everywhere
- There will be vendor consolidation
- EJB 3.0 focusing on ease of development JSR 220 (to appear in J2EE 1.5)
 - Home and remote interfaces generated automatically
 - No deployment descriptor produced by developer
- Web services becoming popular, even for intranet usage (being extended with TX, business process,...)

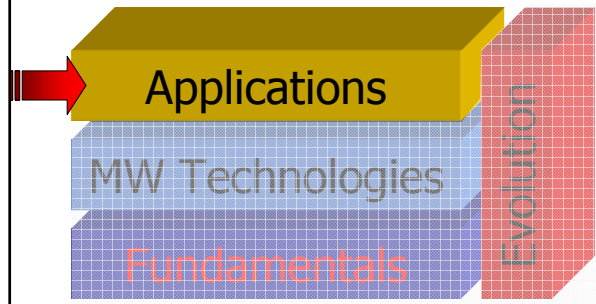
30

WS: Software Evolution

- NOT a software revolution
 - leverages your existing systems
 - builds on existing standards
 - does not require a new programming language
- SOAP model is not new
 - Similar in some respects to RPC of 20 years ago
- What's new: the added power of internet standards for remote object invocation
 - use of implementation-neutral message format
 - UDDI: a universal directory of available services
 - layered services for business aggregation

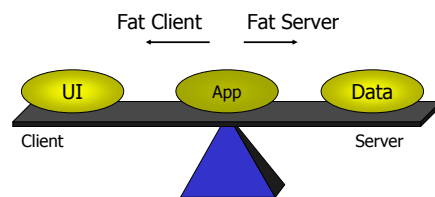
31

Building Blocks



32

Client/Server Balance



33

What drives adoption of WS?

- Cost savings
- Early adopters are building applications now because they recognize obvious cost savings in automating business processes
- Flexibility in programming and integration

34

Adoption of Web Services

amazon.com

- www.amazon.com/gp/aws/landing.html

Google

- www.google.com/apis/

DHL

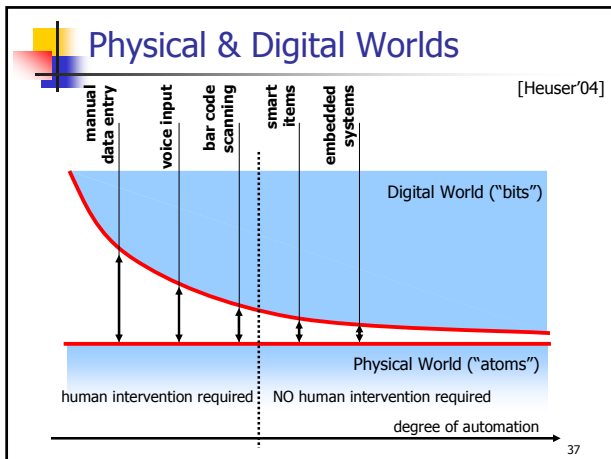
- www.expediteship.com/Default.aspx?tabid=62

35

EAI & B2B

- MW is about application integration
- EAI deals with (tightly) integration
 - EAI refers to software systems that facilitate the integration of heterogeneous, coarse-grained apps
 - More control over MW and processes
- B2B deals with loosely integration
 - Different approaches
 - Semantic heterogeneity
 - Web services
 - Protocols, business coordination (workflow)
- = Messaging + Semantic Data Exchange + Workflow + Web Services

36

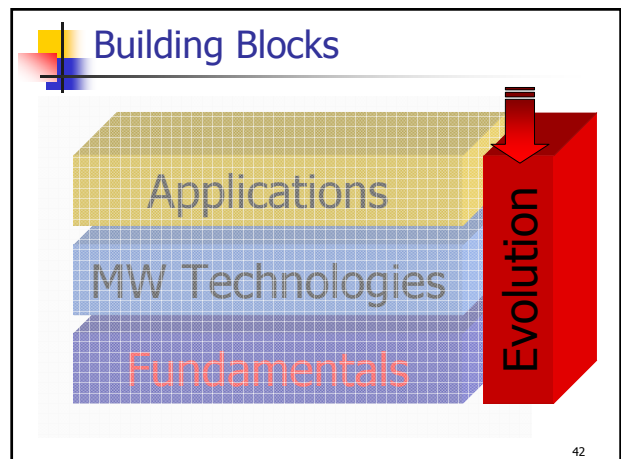


- ### Bar codes
- Understanding bar code fundamentals ensures successful business solutions
 - Linear symbologies perfect for real time lookup applications
 - 2D offers
 - Higher information density (less space)
 - More security
 - Robust error correction
 - Portable databases
 - Bar code print quality important

- ### RFID
- Convenience
 - Efficiency
 - Less human interaction required
 - Read and write capability
 - Portable database
 - Uniquely identifies every physical object

- ### RFID - Middleware/Infrastructure
- From the software point of view
 - Data is no longer static: streams of data!
 - Data exchange across enterprise boundaries:
 - Data is generated along the supply chain
 - Transactions (workflows, etc)
 - Privacy aspects (who consumes which data)
 - Data needs to be disseminated proactively
 - From data producers to data consumers
 - 1-to-many communications
 - Traditional Middleware (Request/reply) does not fit
 - Event handling infrastructure is required

- ### Physical & Digital Worlds
- Understand (on field) the problem and then select appropriate solution
 - Ergonomics, working range, ease of use impact success
 - Trend: integration of physical and digital worlds
 - Digital representation
 - Web presence

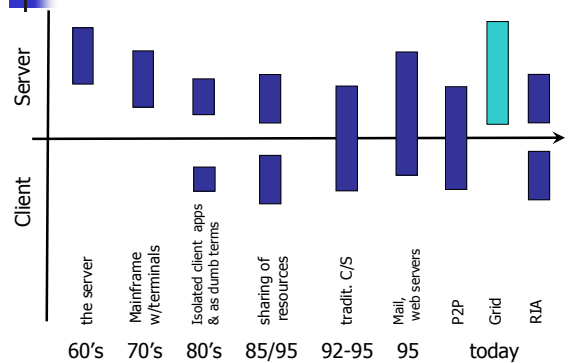


MW Evolution

- TP-Monitors are still as important as they have been in the past decades but they have become components in larger systems and hidden behind additional layers intended for EAI and WS
 - Like RPC, the functionality of TP-Monitors is starting to migrate to the low levels of the infrastructure and becoming invisible to the developer
- CORBA is being replaced by other platforms although its ideas are still being used and copied in new systems
 - CORBA suffered from 3 developments that changed the technology landscape:
 - the quick adoption of Java and the JVM
 - the Internet and the emergence of the Web
 - the raise of J2EE and related technologies to an almost de-facto standard for MW

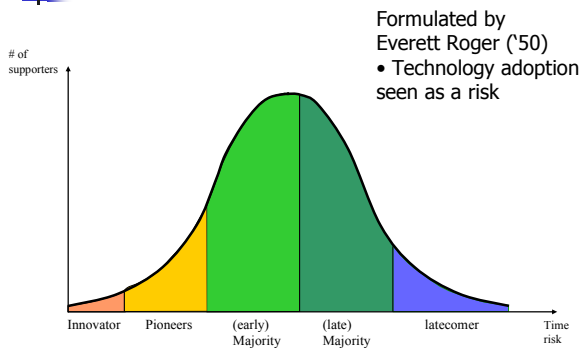
43

Technology Evolution



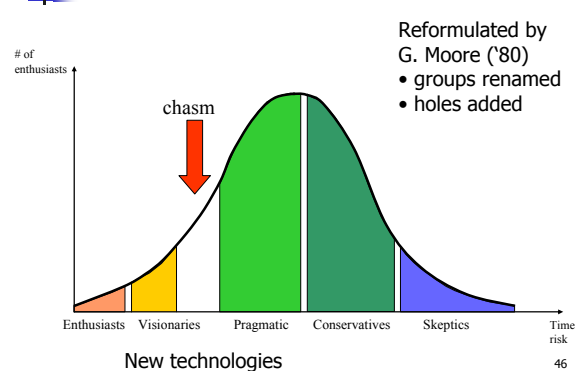
44

Technology Adoption Cycle



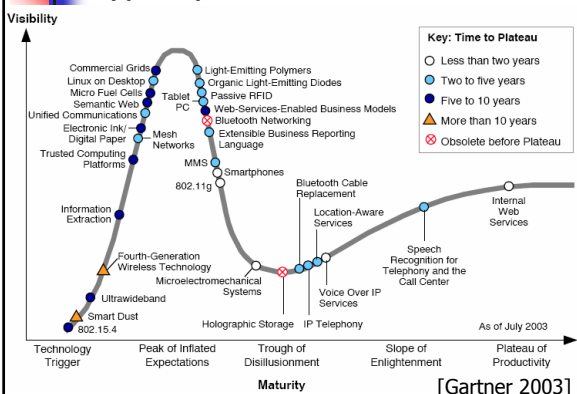
45

Technology Adoption Cycle (2)



46

Hype Cycle



Review of Objectives

- What you should get out of this course
 - a basic understanding of large system architecture
 - an understanding of the **technological building blocks** used in the implementation of large, modern systems
 - an overview of the state of the art in **middleware**

50