Network Computing

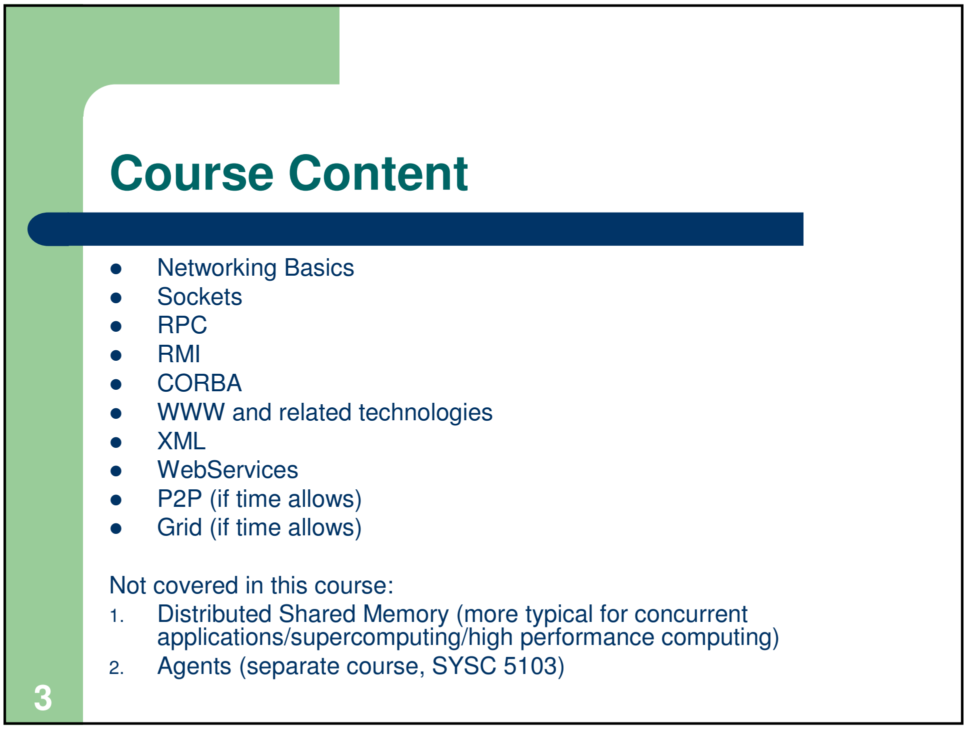
Course Information & Overview

Thomas Kunz
tkunz@sce.carleton.ca

*(slides provided by Thomas Kunz,
Qusay Mahmoud,
Roger Impey, Babak Esfandiari,
various textbooks)*

Goals

- This is not a theory course
- It is a SKILLS course: emphasis is on actual code development (see marking scheme/course requirements)
- Understand the basics of networking
- Learn about the different technologies that can be used to develop distributed applications
- Work effectively in groups to develop a significant distributed application
- If it is your first course in networking, you will develop an idea of where to go next



Course Content

- Networking Basics
- Sockets
- RPC
- RMI
- CORBA
- WWW and related technologies
- XML
- WebServices
- P2P (if time allows)
- Grid (if time allows)

Not covered in this course:

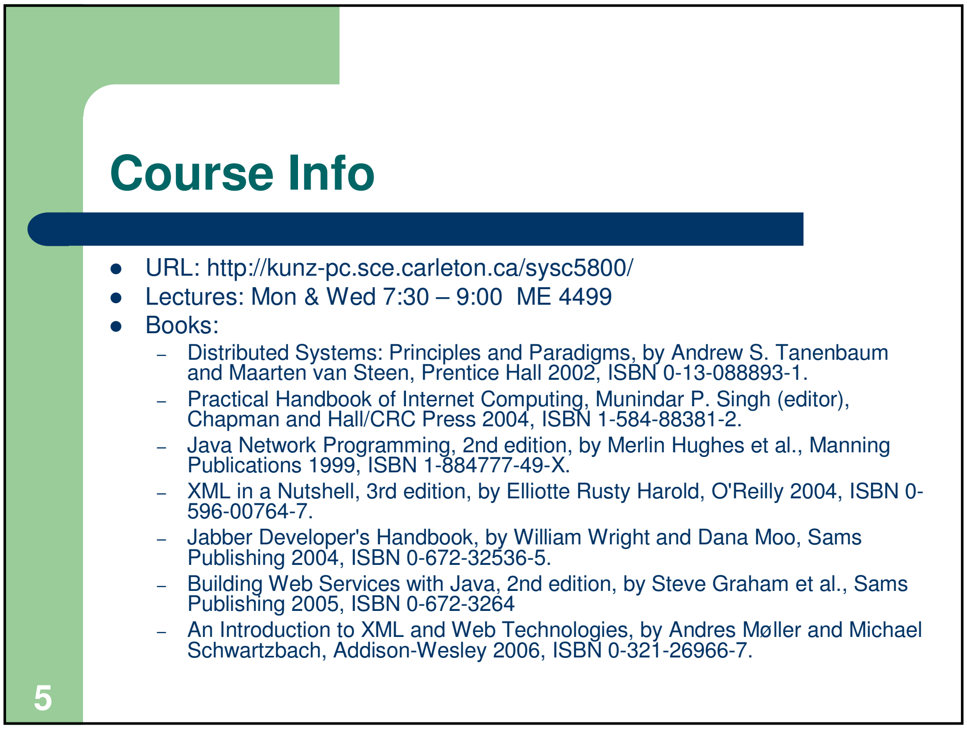
1. Distributed Shared Memory (more typical for concurrent applications/supercomputing/high performance computing)
2. Agents (separate course, SYSC 5103)

3

Limitations

- Broad set of related technical topics
- We will not be able to cover all topics in details
- For some topics, the course is more like an eye opener
- Do not expect to become an expert in every single topic that we will cover
- It is almost impossible to master all the technologies we will cover in the course

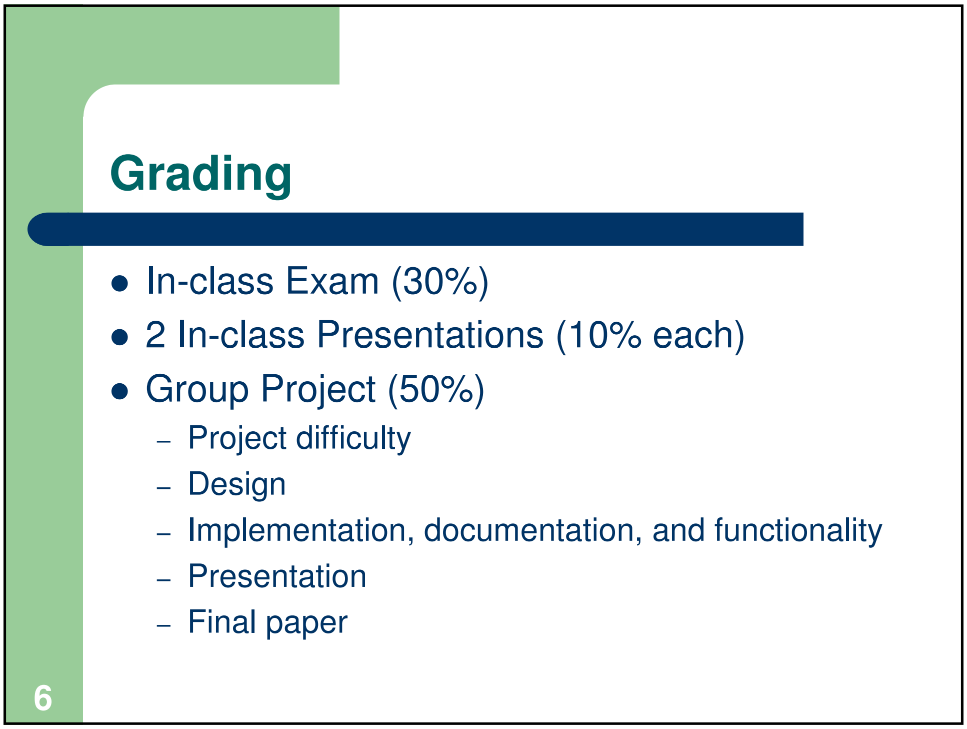
4



Course Info

- URL: http://kunz-pc.sce.carleton.ca/sysc5800/
- Lectures: Mon & Wed 7:30 – 9:00 ME 4499
- Books:
  - Distributed Systems: Principles and Paradigms, by Andrew S. Tanenbaum and Maarten van Steen, Prentice Hall 2002, ISBN 0-13-088893-1.
  - Practical Handbook of Internet Computing, Munindar P. Singh (editor), Chapman and Hall/CRC Press 2004, ISBN 1-584-88381-2.
  - Java Network Programming, 2nd edition, by Merlin Hughes et al., Manning Publications 1999, ISBN 1-884777-49-X.
  - XML in a Nutshell, 3rd edition, by Elliotte Rusty Harold, O'Reilly 2004, ISBN 0-596-00764-7.
  - Jabber Developer's Handbook, by William Wright and Dana Moo, Sams Publishing 2004, ISBN 0-672-32536-5.
  - Building Web Services with Java, 2nd edition, by Steve Graham et al., Sams Publishing 2005, ISBN 0-672-3264
  - An Introduction to XML and Web Technologies, by Andres Møller and Michael Schwartzbach, Addison-Wesley 2006, ISBN 0-321-26966-7.

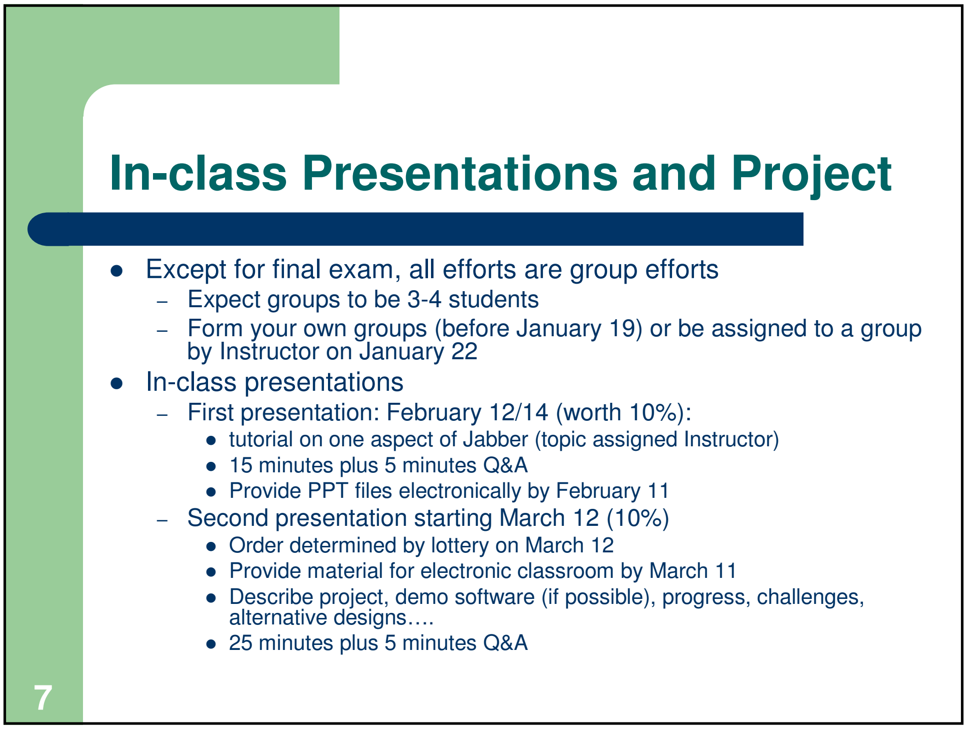
5



Grading

- In-class Exam (30%)
- 2 In-class Presentations (10% each)
- Group Project (50%)
  - Project difficulty
  - Design
  - Implementation, documentation, and functionality
  - Presentation
  - Final paper

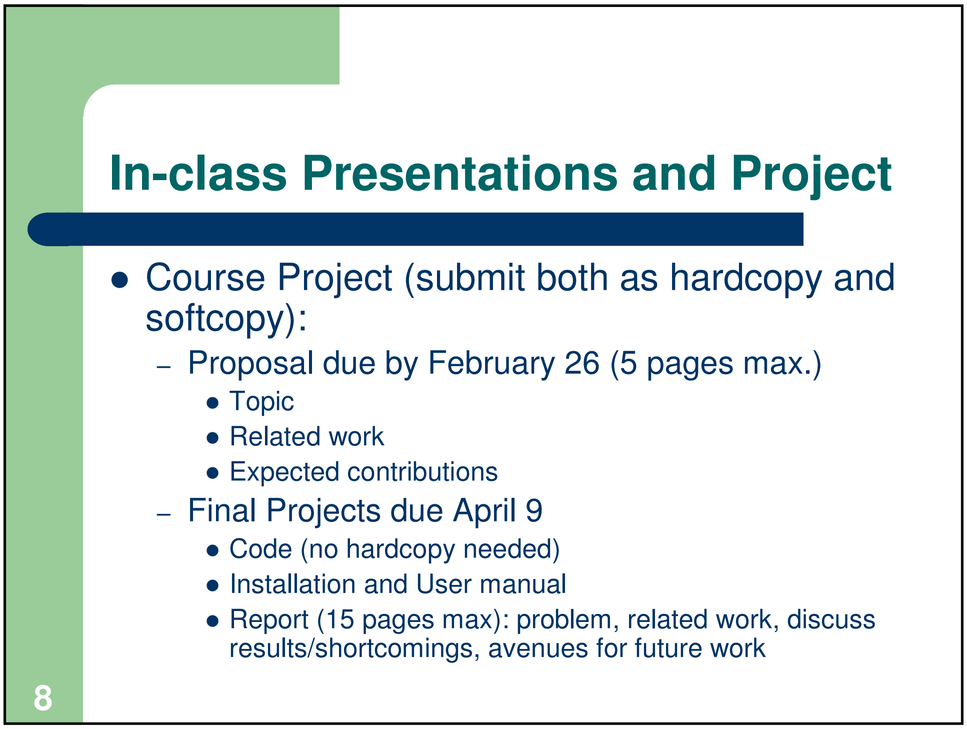
6



In-class Presentations and Project

- Except for final exam, all efforts are group efforts
  - Expect groups to be 3-4 students
  - Form your own groups (before January 19) or be assigned to a group by Instructor on January 22
- In-class presentations
  - First presentation: February 12/14 (worth 10%):
    - tutorial on one aspect of Jabber (topic assigned Instructor)
    - 15 minutes plus 5 minutes Q&A
    - Provide PPT files electronically by February 11
  - Second presentation starting March 12 (10%)
    - Order determined by lottery on March 12
    - Provide material for electronic classroom by March 11
    - Describe project, demo software (if possible), progress, challenges, alternative designs….
    - 25 minutes plus 5 minutes Q&A

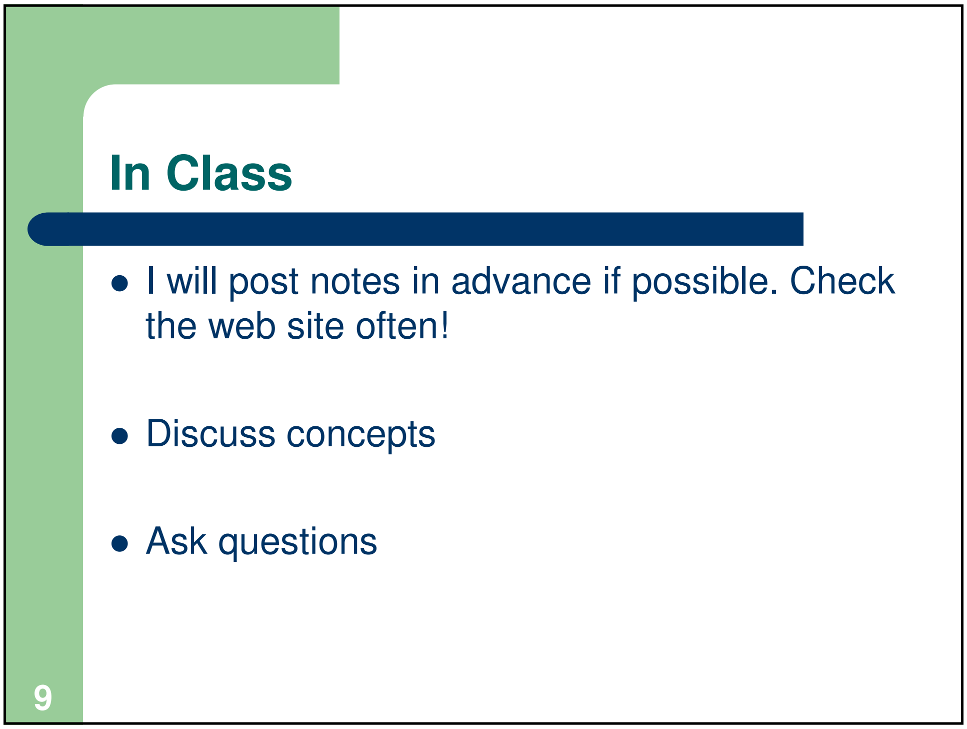
7



In-class Presentations and Project

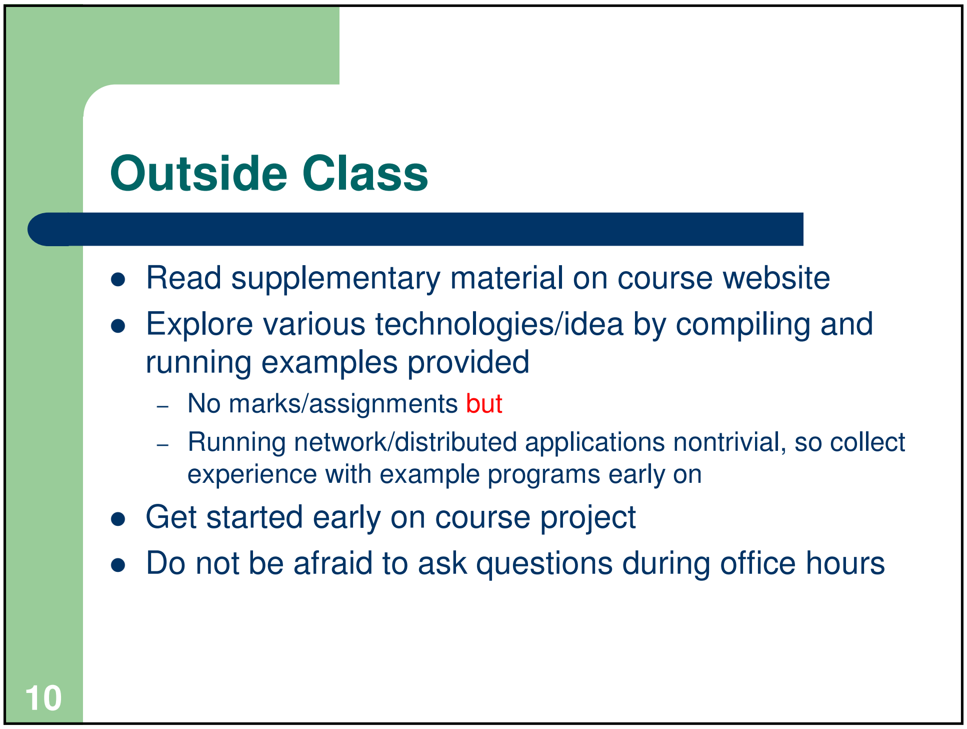
- Course Project (submit both as hardcopy and softcopy):
  - Proposal due by February 26 (5 pages max.)
    - Topic
    - Related work
    - Expected contributions
  - Final Projects due April 9
    - Code (no hardcopy needed)
    - Installation and User manual
    - Report (15 pages max): problem, related work, discuss results/shortcomings, avenues for future work

8



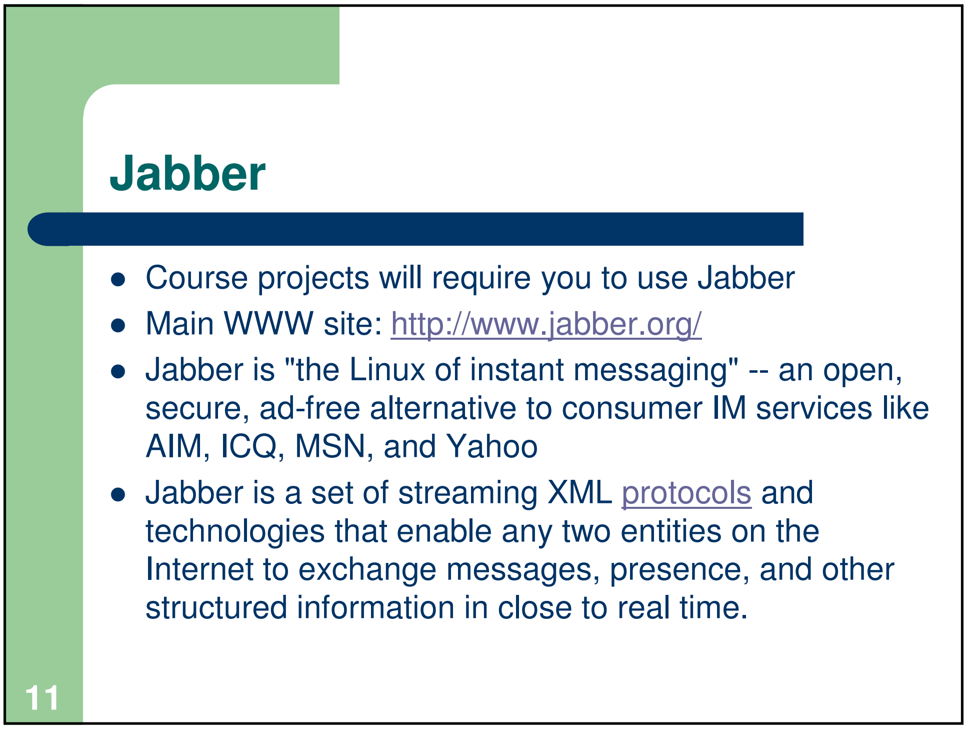
In Class

- I will post notes in advance if possible. Check the web site often!
- Discuss concepts
- Ask questions



Outside Class

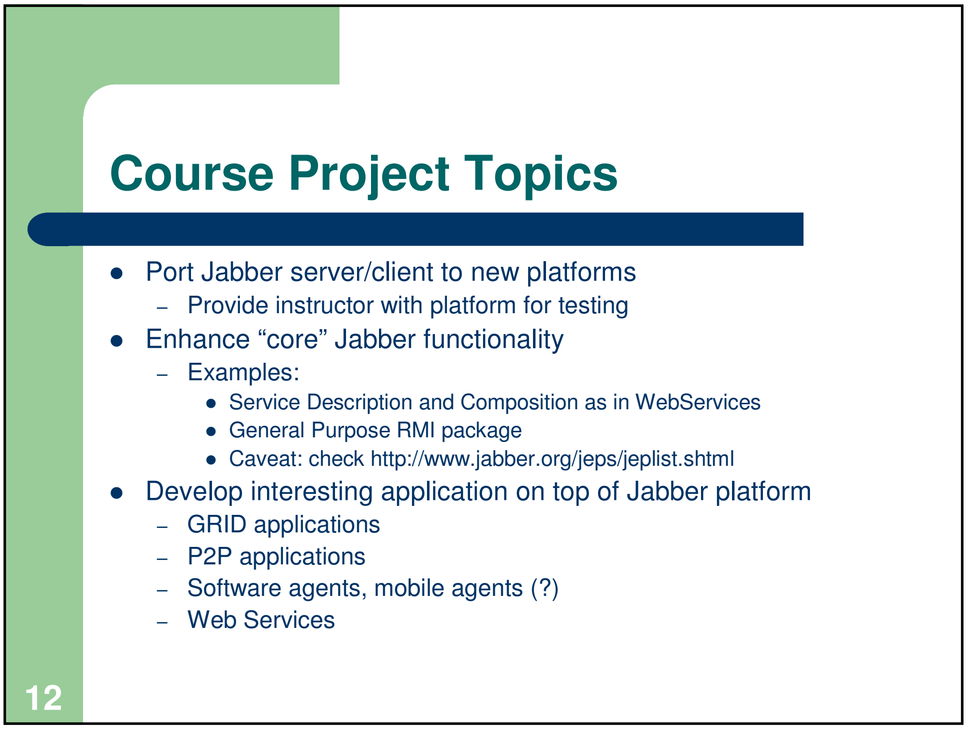
- Read supplementary material on course website
- Explore various technologies/idea by compiling and running examples provided
  - No marks/assignments but
  - Running network/distributed applications nontrivial, so collect experience with example programs early on
- Get started early on course project
- Do not be afraid to ask questions during office hours



Jabber

- Course projects will require you to use Jabber
- Main WWW site: http://www.jabber.org/
- Jabber is "the Linux of instant messaging" -- an open, secure, ad-free alternative to consumer IM services like AIM, ICQ, MSN, and Yahoo
- Jabber is a set of streaming XML protocols and technologies that enable any two entities on the Internet to exchange messages, presence, and other structured information in close to real time.

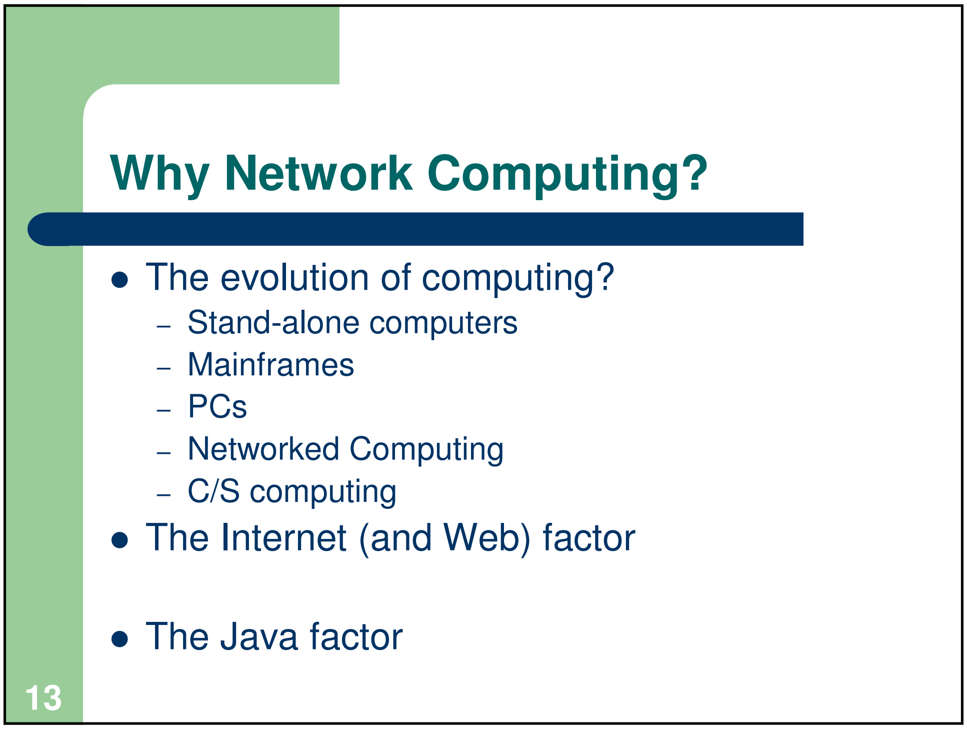
11



Course Project Topics

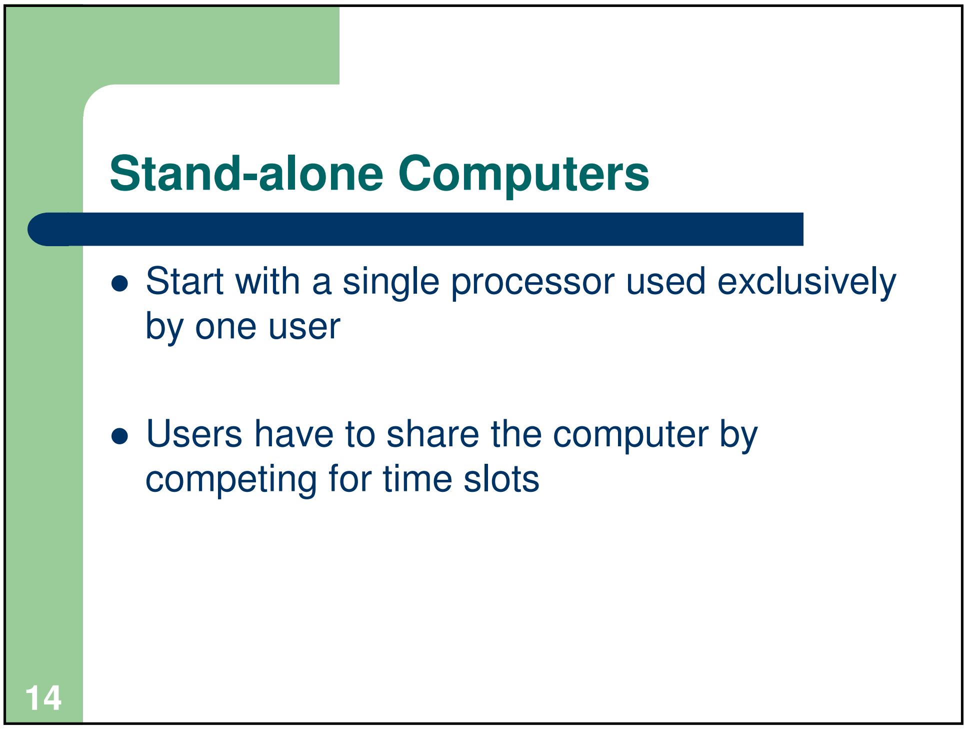
- Port Jabber server/client to new platforms
  - Provide instructor with platform for testing
- Enhance “core” Jabber functionality
  - Examples:
    - Service Description and Composition as in WebServices
    - General Purpose RMI package
    - Caveat: check http://www.jabber.org/jeps/jeplist.shtml
- Develop interesting application on top of Jabber platform
  - GRID applications
  - P2P applications
  - Software agents, mobile agents (?)
  - Web Services

12



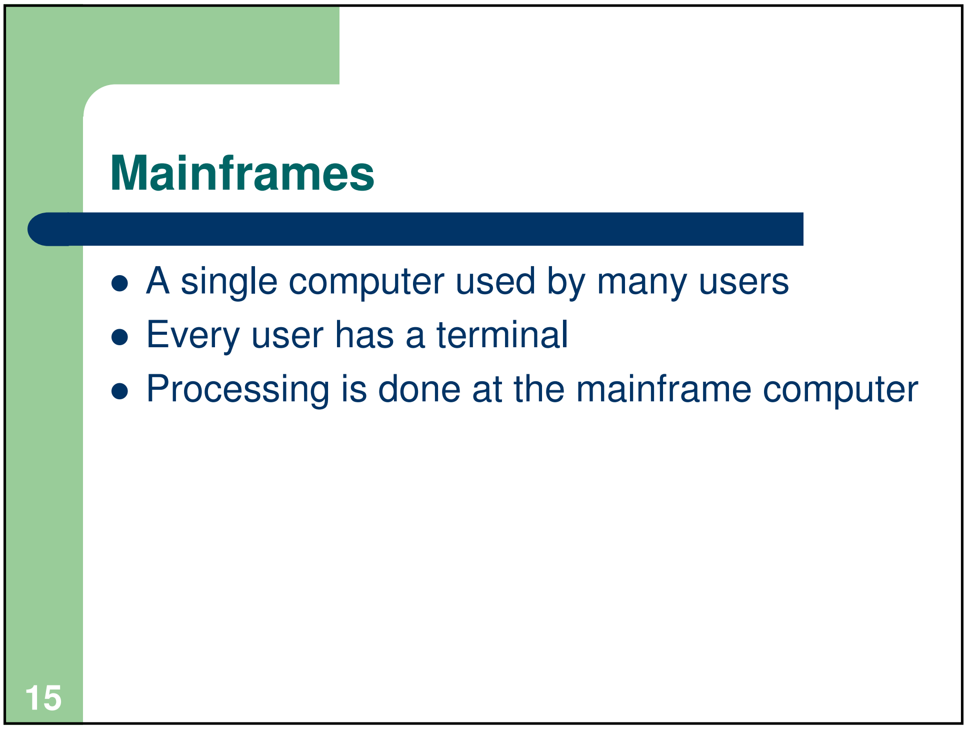
Why Network Computing?

- The evolution of computing?
  - Stand-alone computers
  - Mainframes
  - PCs
  - Networked Computing
  - C/S computing
- The Internet (and Web) factor
- The Java factor



Stand-alone Computers

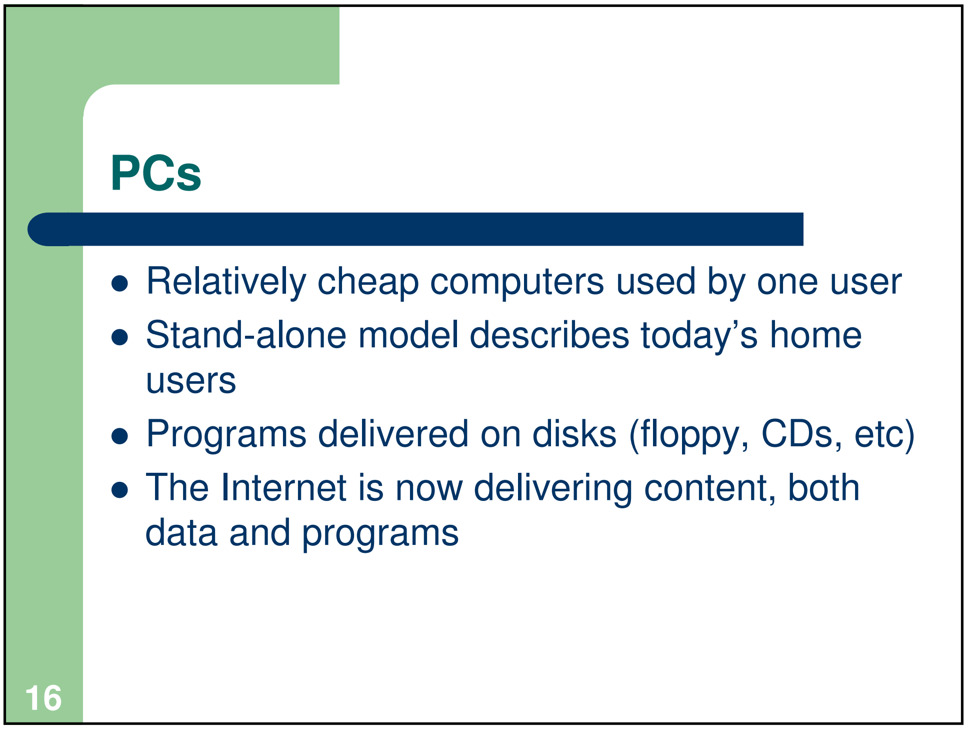
- Start with a single processor used exclusively by one user
- Users have to share the computer by competing for time slots



Mainframes

- A single computer used by many users
- Every user has a terminal
- Processing is done at the mainframe computer

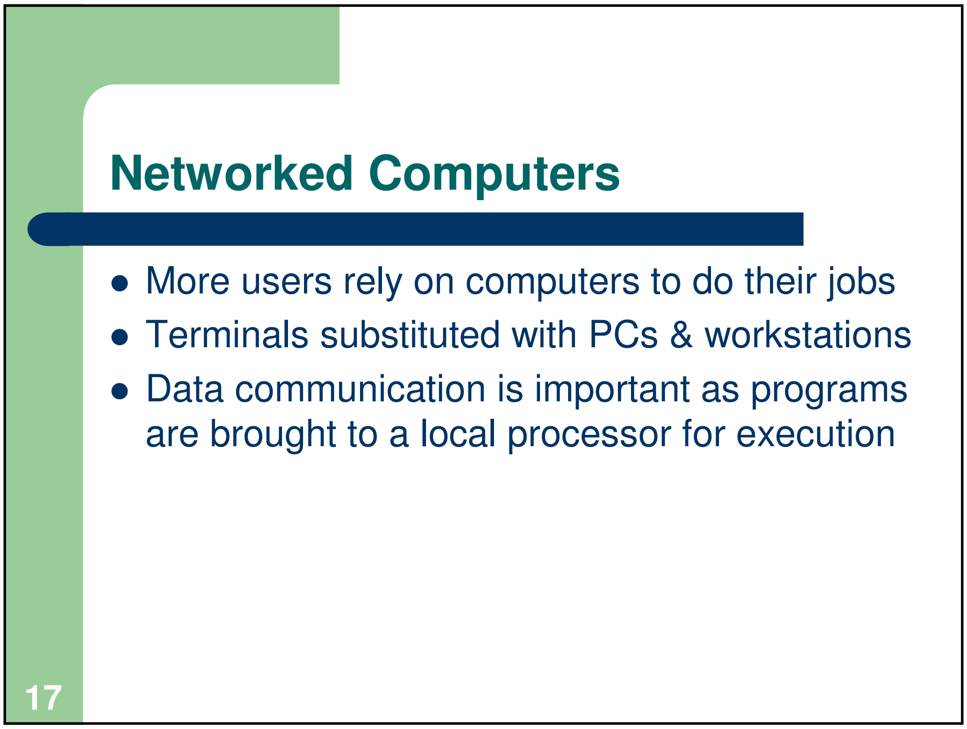
15



PCs

- Relatively cheap computers used by one user
- Stand-alone model describes today’s home users
- Programs delivered on disks (floppy, CDs, etc)
- The Internet is now delivering content, both data and programs

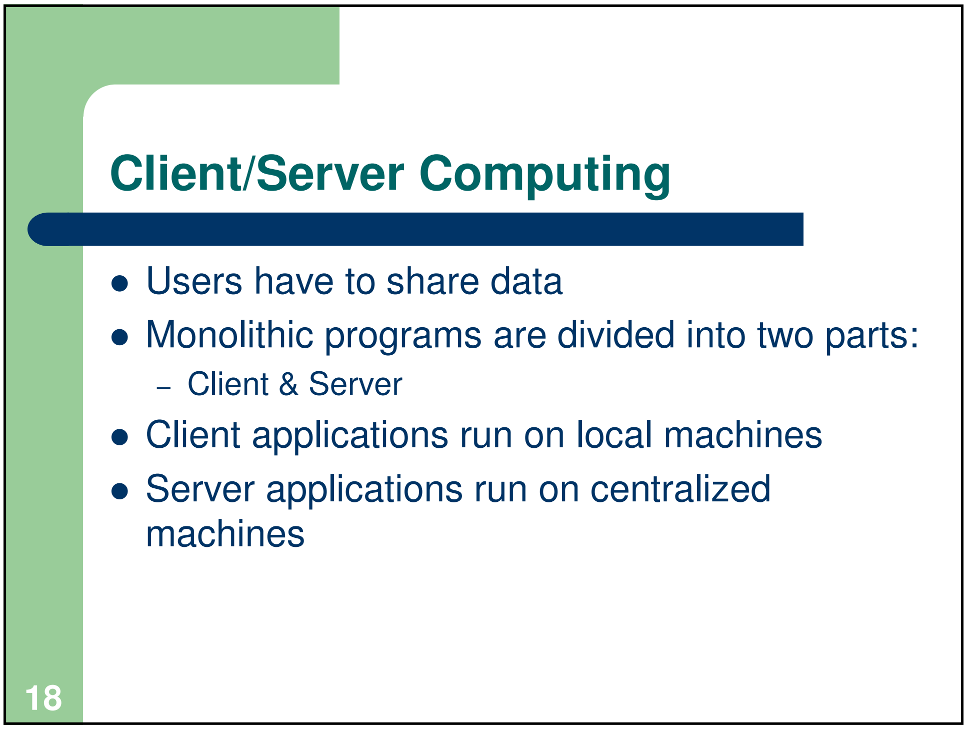
16



Networked Computers

- More users rely on computers to do their jobs
- Terminals substituted with PCs & workstations
- Data communication is important as programs are brought to a local processor for execution

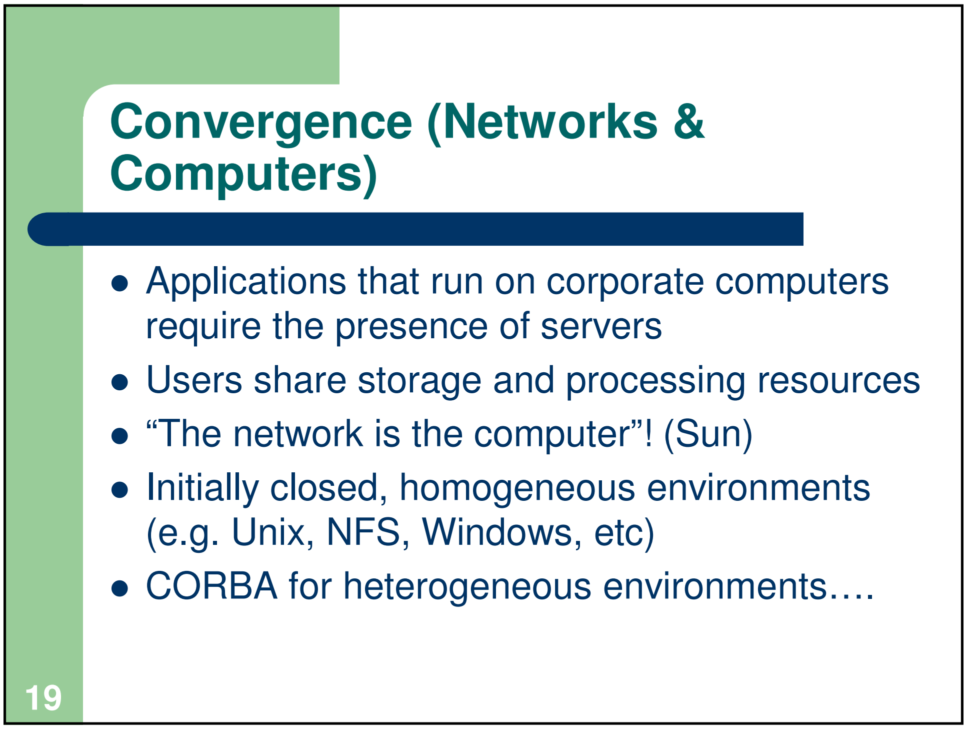
17



Client/Server Computing

- Users have to share data
- Monolithic programs are divided into two parts:
  - Client & Server
- Client applications run on local machines
- Server applications run on centralized machines

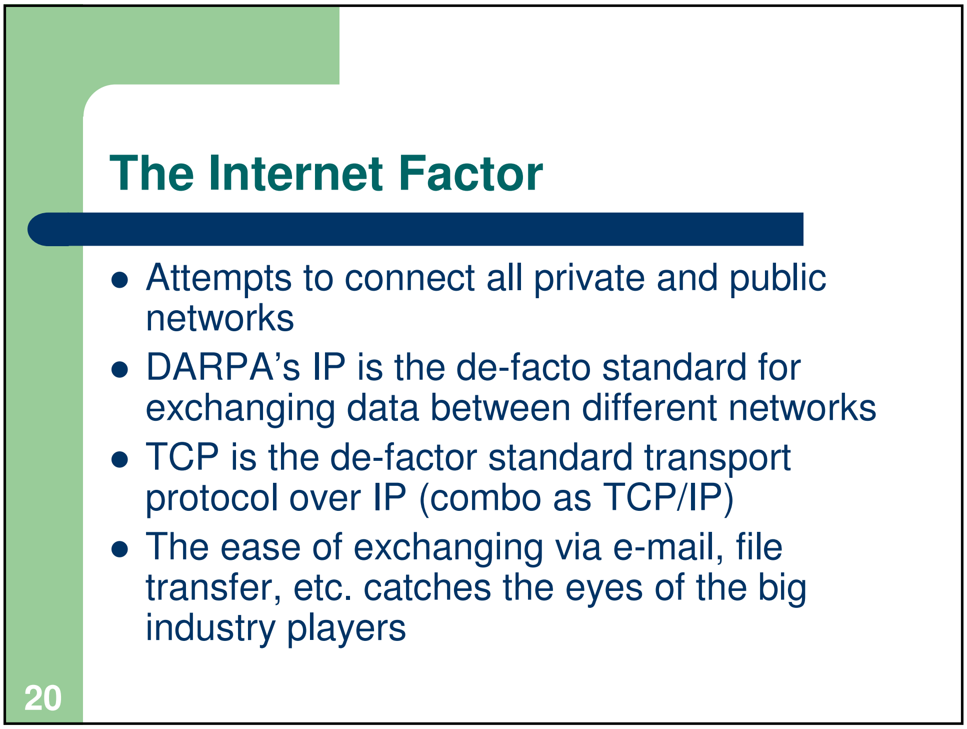
18



Convergence (Networks & Computers)

- Applications that run on corporate computers require the presence of servers
- Users share storage and processing resources
- “The network is the computer”! (Sun)
- Initially closed, homogeneous environments (e.g. Unix, NFS, Windows, etc)
- CORBA for heterogeneous environments….

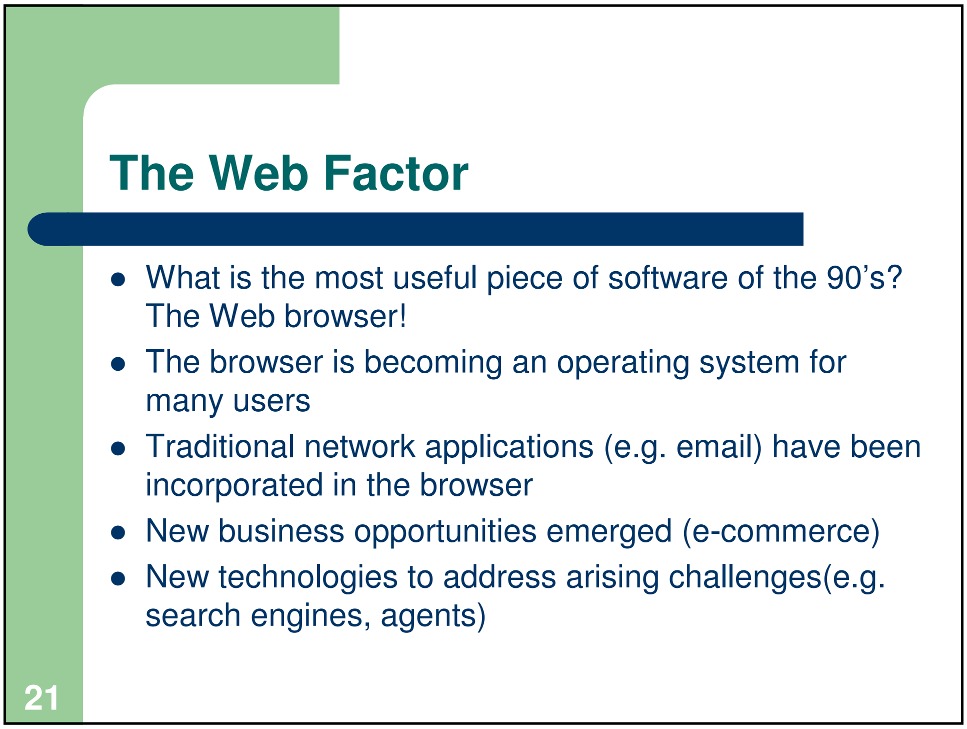
19



The Internet Factor

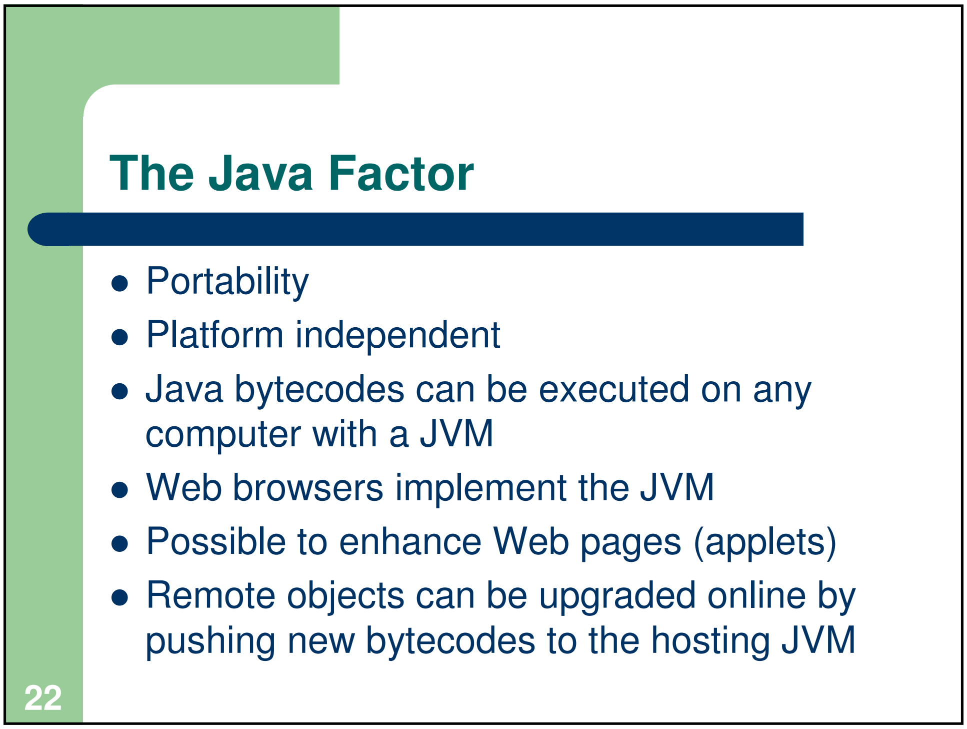
- Attempts to connect all private and public networks
- DARPA’s IP is the de-facto standard for exchanging data between different networks
- TCP is the de-factor standard transport protocol over IP (combo as TCP/IP)
- The ease of exchanging via e-mail, file transfer, etc. catches the eyes of the big industry players

20



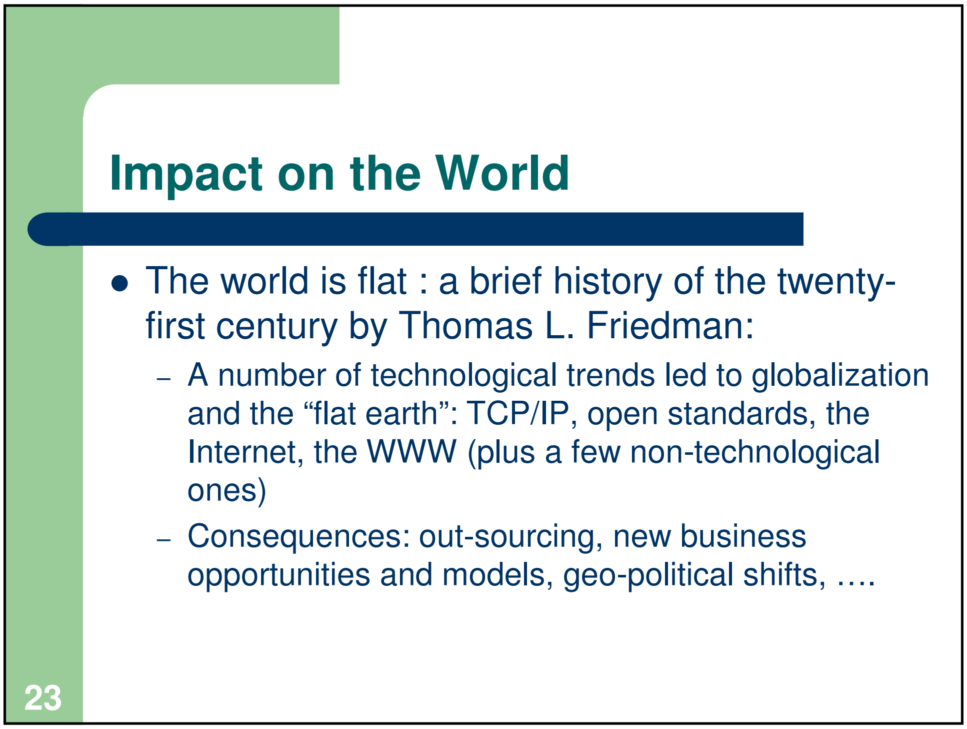
The Web Factor

- What is the most useful piece of software of the 90’s? The Web browser!
- The browser is becoming an operating system for many users
- Traditional network applications (e.g. email) have been incorporated in the browser
- New business opportunities emerged (e-commerce)
- New technologies to address arising challenges(e.g. search engines, agents)



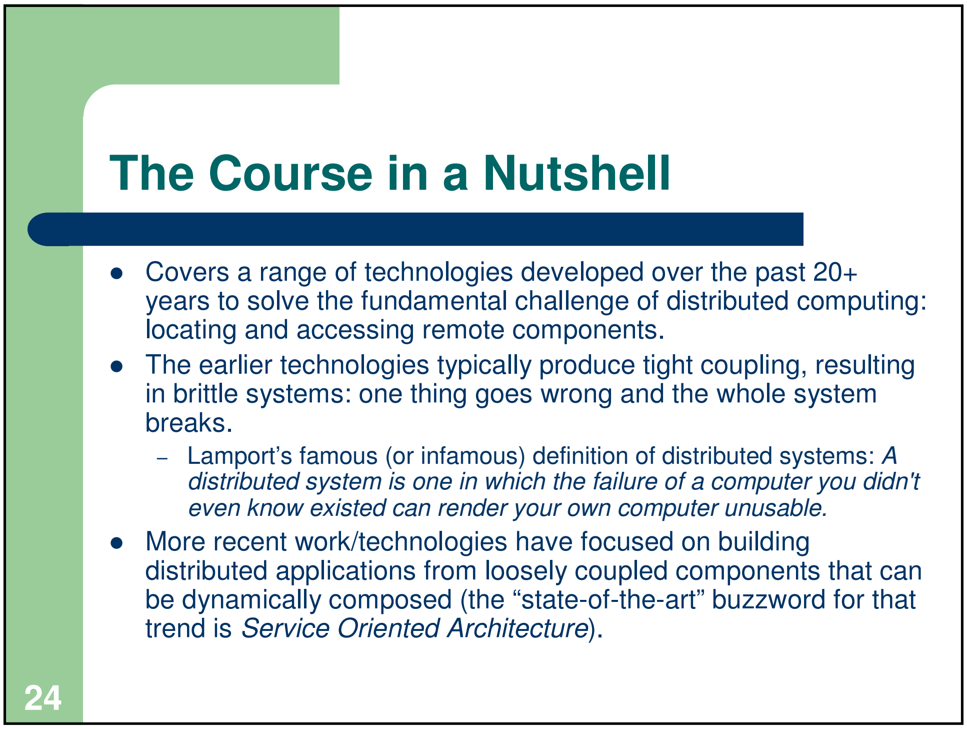
The Java Factor

- Portability
- Platform independent
- Java bytecodes can be executed on any computer with a JVM
- Web browsers implement the JVM
- Possible to enhance Web pages (applets)
- Remote objects can be upgraded online by pushing new bytecodes to the hosting JVM



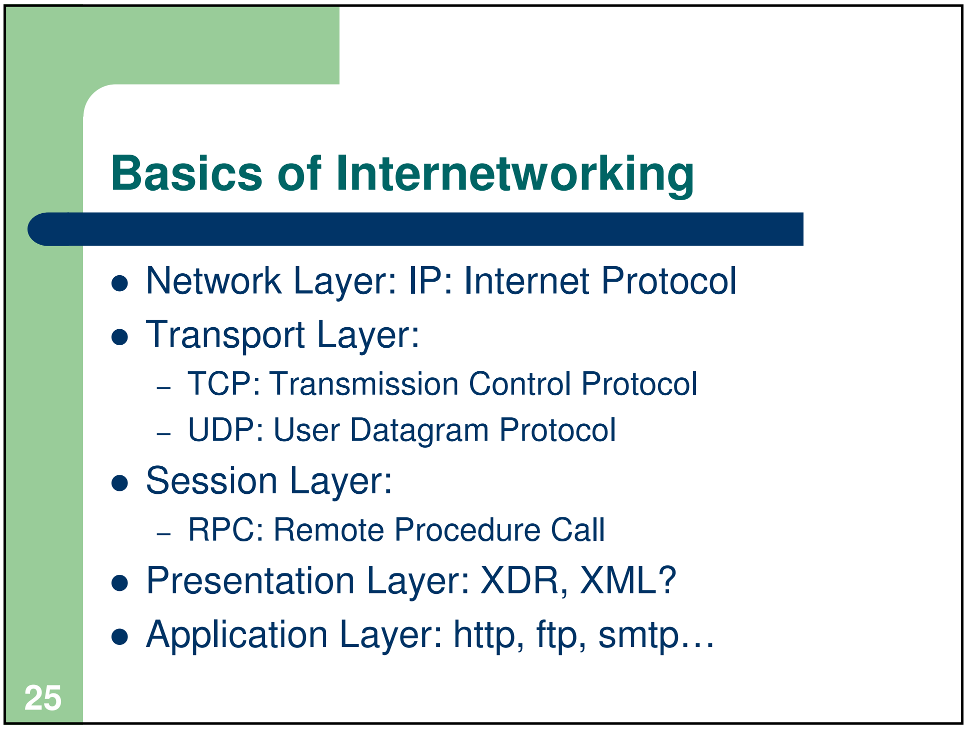
Impact on the World

- The world is flat : a brief history of the twenty-first century by Thomas L. Friedman:
  - A number of technological trends led to globalization and the “flat earth”: TCP/IP, open standards, the Internet, the WWW (plus a few non-technological ones)
  - Consequences: out-sourcing, new business opportunities and models, geo-political shifts, ….



The Course in a Nutshell

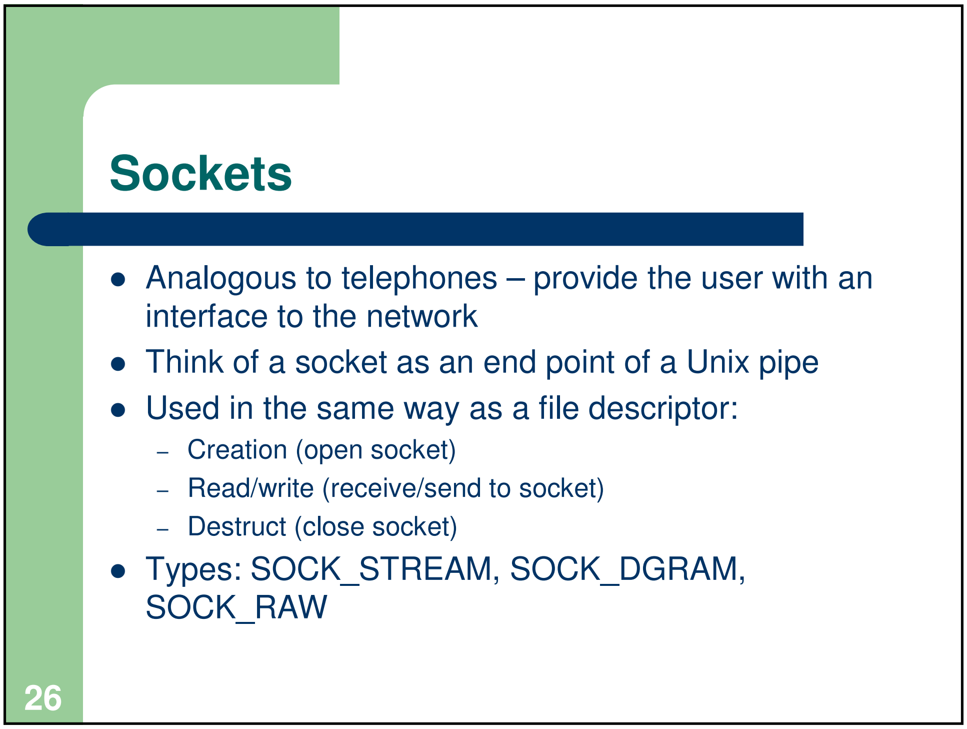
- Covers a range of technologies developed over the past 20+ years to solve the fundamental challenge of distributed computing: locating and accessing remote components.
- The earlier technologies typically produce tight coupling, resulting in brittle systems: one thing goes wrong and the whole system breaks.
  - Lamport’s famous (or infamous) definition of distributed systems: *A distributed system is one in which the failure of a computer you didn't even know existed can render your own computer unusable.*
- More recent work/technologies have focused on building distributed applications from loosely coupled components that can be dynamically composed (the “state-of-the-art” buzzword for that trend is *Service Oriented Architecture*).



Basics of Internetworking

- Network Layer: IP: Internet Protocol
- Transport Layer:
  - TCP: Transmission Control Protocol
  - UDP: User Datagram Protocol
- Session Layer:
  - RPC: Remote Procedure Call
- Presentation Layer: XDR, XML?
- Application Layer: http, ftp, smtp…

25



Sockets

- Analogous to telephones – provide the user with an interface to the network
- Think of a socket as an end point of a Unix pipe
- Used in the same way as a file descriptor:
  - Creation (open socket)
  - Read/write (receive/send to socket)
  - Destruct (close socket)
- Types: SOCK_STREAM, SOCK_DGRAM, SOCK_RAW

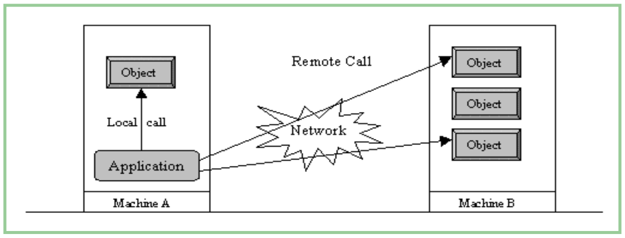
26

RPC

- Introduced by Birrell & Nelson (1984)
- Remote Procedure Calls allow a program to make use of procedures executing on a remote machine
  - If it doesn't sound OO, it's because it isn't OO!
- RPCs are based on sockets, and therefore dispense us from using them directly
- Remote Procedures could in principle be written in a different language than the clients.

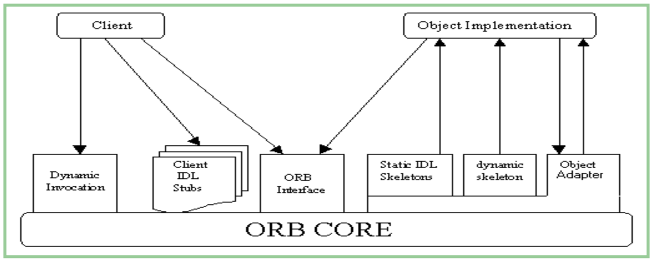
RMI (Remote Method Invocation)

- "RPC for Java": RMI is a core package of Java 1.1+
- The power of Java interfaces (no protocols)
- Methods can be invoked from JVMs, possibly running on remote hosts



CORBA

- Common Object Request Broker Architecture
- A specification for creating and using distributed objects
- It is not a programming language



CORBA vs. RMI

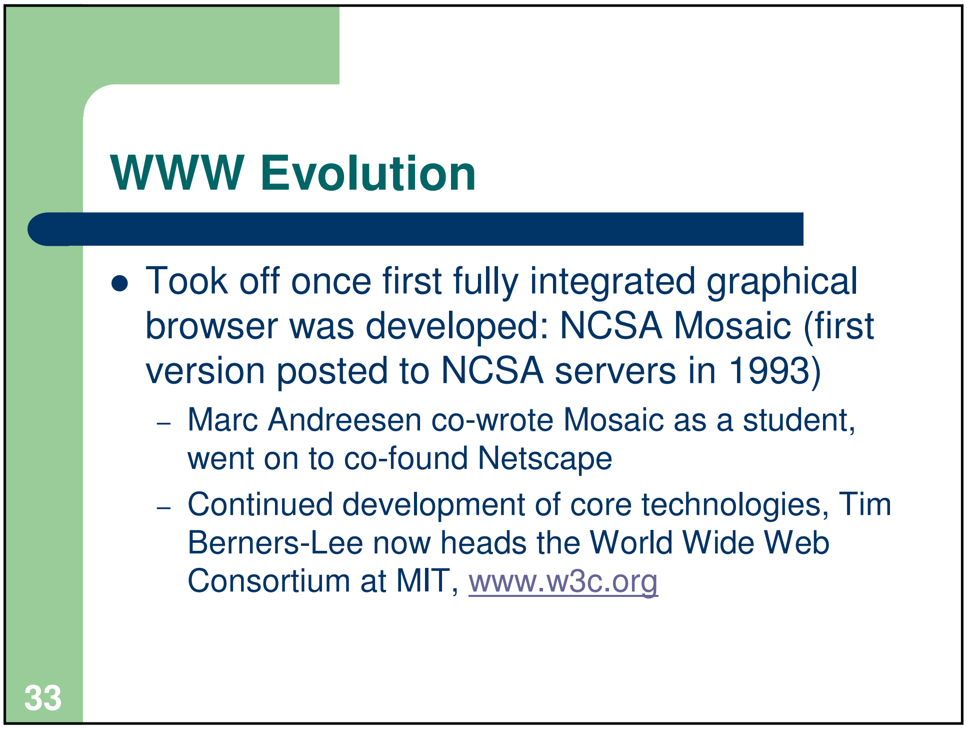
Interfaces in IDL	Interfaces in Java
Language-independent	Java-based
Heterogeneous language environment	Homogenous language environment
Garbage collector (No)	Garbage collector (Yes)
In, out, inout parameters	No such parameters
Pass by reference	Pass by copy (local objects), by reference to stub (remote objects)

The World Wide Web

- NOT invented by Academics or Industry ☺
- Invented by Tim Berners-Lee at CERN (see also http://www.zeltser.com/web-history/):
  - CERN is a meeting place for physicists from all over the world, who collaborate on complex physics, engineering and information handling projects.
  - Thus, the need for the WWW system arose "from the geographical dispersion of large collaborations, and the fast turnover of fellows, students, and visiting scientists," who had to get "up to speed on projects and leave a lasting contribution before leaving.“

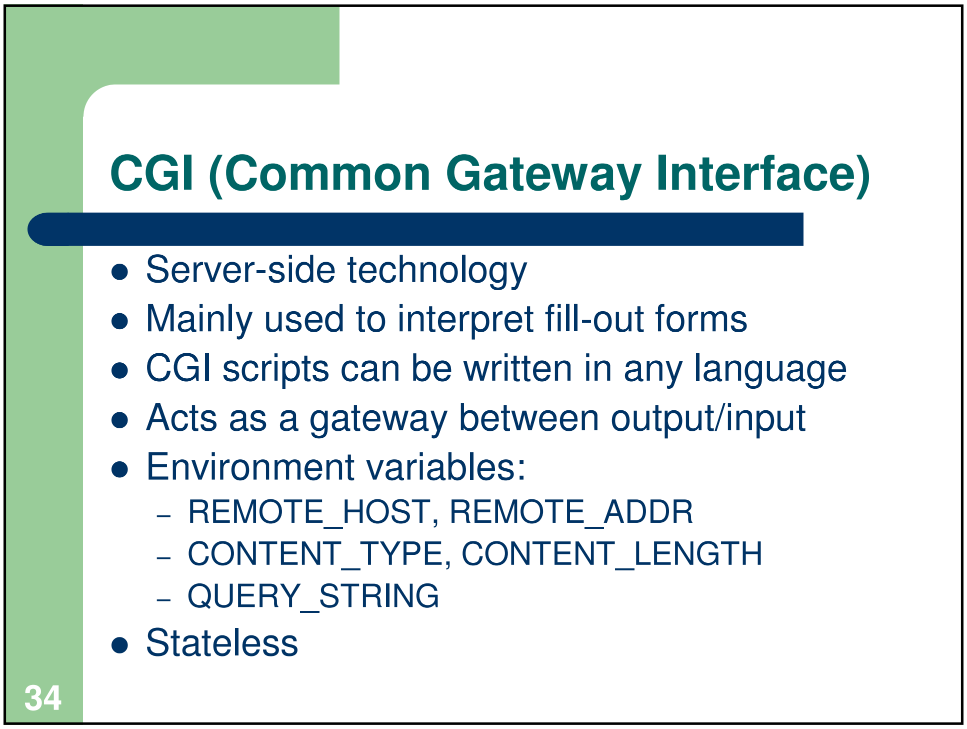
Major Components of WWW

- A uniform resource locator: URL
- A protocol: HTTP
  - The client: a Web browser
  - The server: the Web server
- A markup language: HTML
- Server-side dynamic generation of HTML documents: CGI, Servlets, ASPs, JSPs…
- Client-side rendering: Stylesheets, JavaScript, Java, Flash…



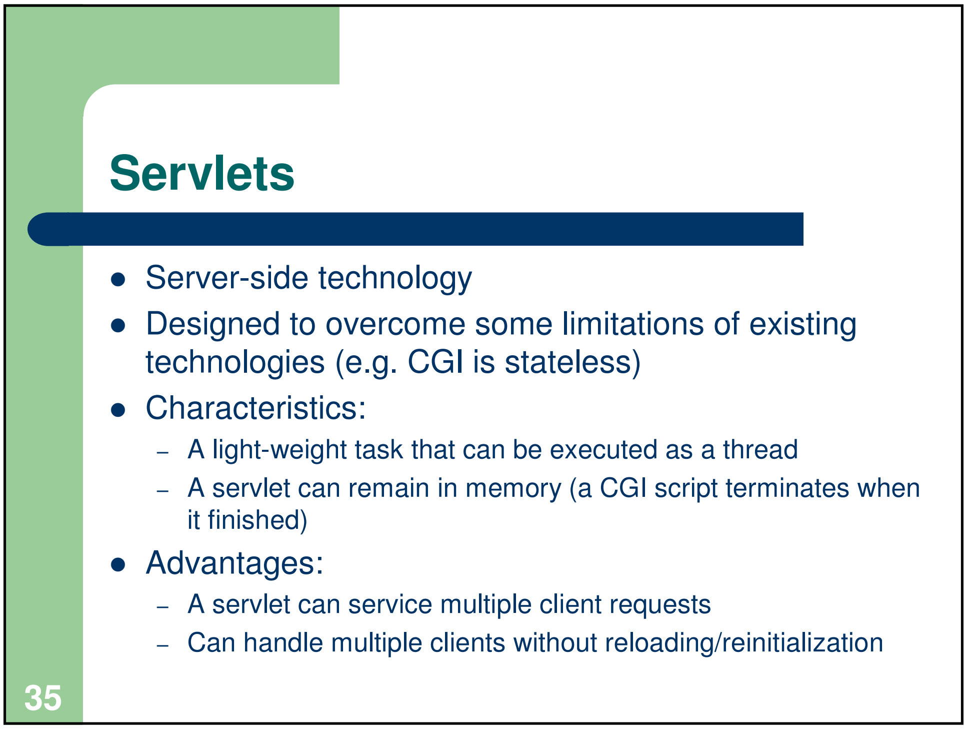
WWW Evolution

- Took off once first fully integrated graphical browser was developed: NCSA Mosaic (first version posted to NCSA servers in 1993)
  - Marc Andreesen co-wrote Mosaic as a student, went on to co-found Netscape
  - Continued development of core technologies, Tim Berners-Lee now heads the World Wide Web Consortium at MIT, www.w3c.org



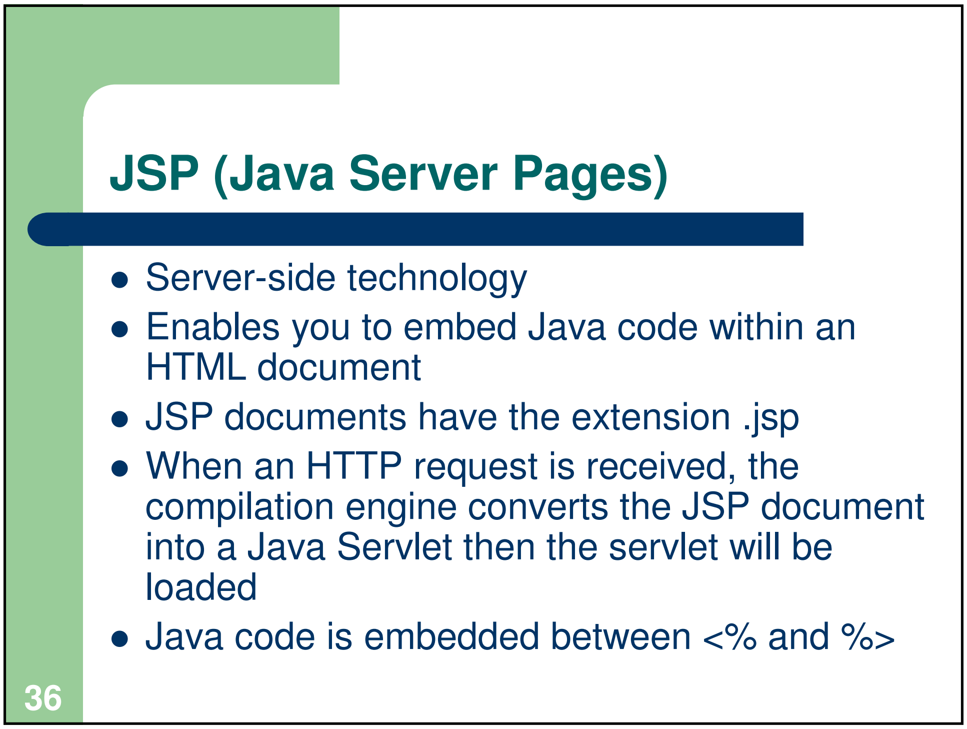
CGI (Common Gateway Interface)

- Server-side technology
- Mainly used to interpret fill-out forms
- CGI scripts can be written in any language
- Acts as a gateway between output/input
- Environment variables:
  - REMOTE_HOST, REMOTE_ADDR
  - CONTENT_TYPE, CONTENT_LENGTH
  - QUERY_STRING
- Stateless



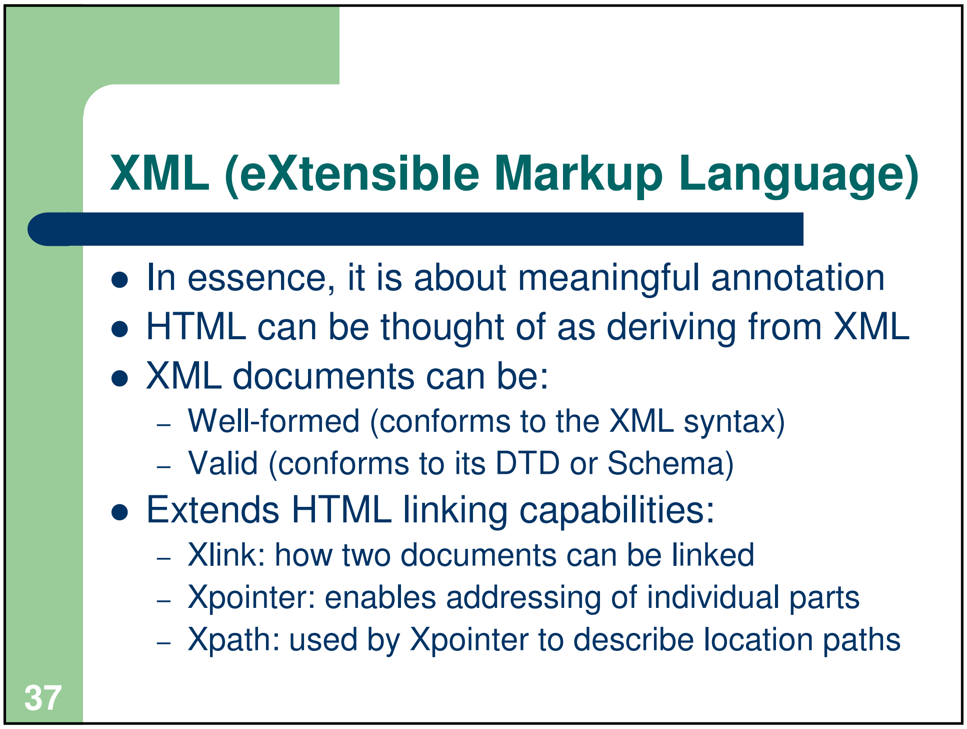
Servlets

- Server-side technology
- Designed to overcome some limitations of existing technologies (e.g. CGI is stateless)
- Characteristics:
  - A light-weight task that can be executed as a thread
  - A servlet can remain in memory (a CGI script terminates when it finished)
- Advantages:
  - A servlet can service multiple client requests
  - Can handle multiple clients without reloading/reinitialization



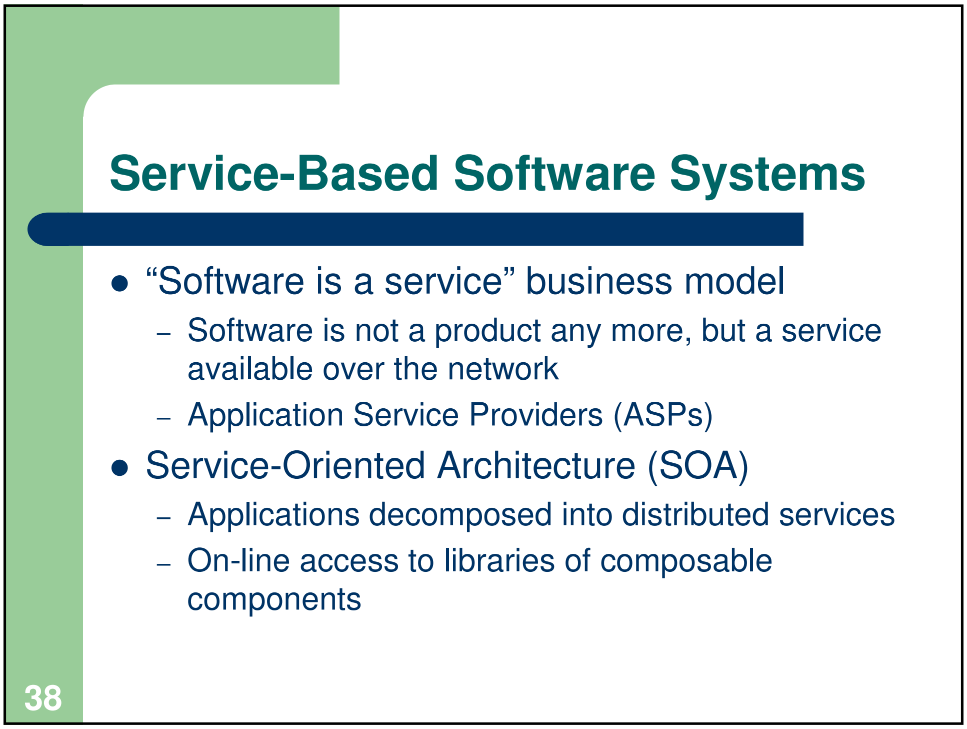
JSP (Java Server Pages)

- Server-side technology
- Enables you to embed Java code within an HTML document
- JSP documents have the extension .jsp
- When an HTTP request is received, the compilation engine converts the JSP document into a Java Servlet then the servlet will be loaded
- Java code is embedded between <% and %>



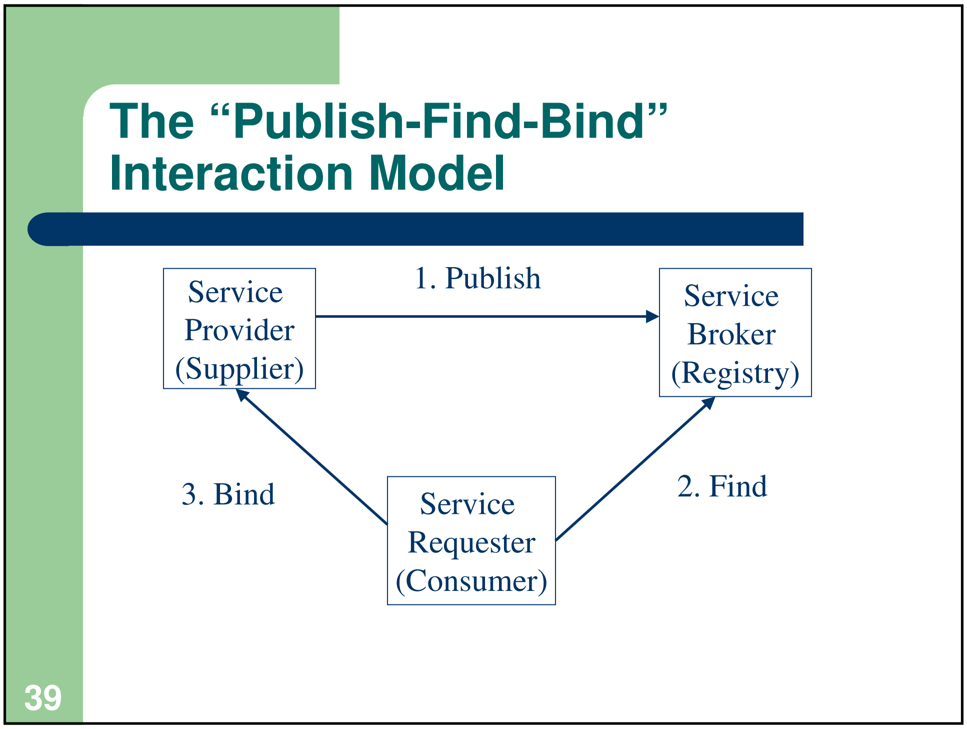
XML (eXtensible Markup Language)

- In essence, it is about meaningful annotation
- HTML can be thought of as deriving from XML
- XML documents can be:
  - Well-formed (conforms to the XML syntax)
  - Valid (conforms to its DTD or Schema)
- Extends HTML linking capabilities:
  - Xlink: how two documents can be linked
  - Xpointer: enables addressing of individual parts
  - Xpath: used by Xpointer to describe location paths

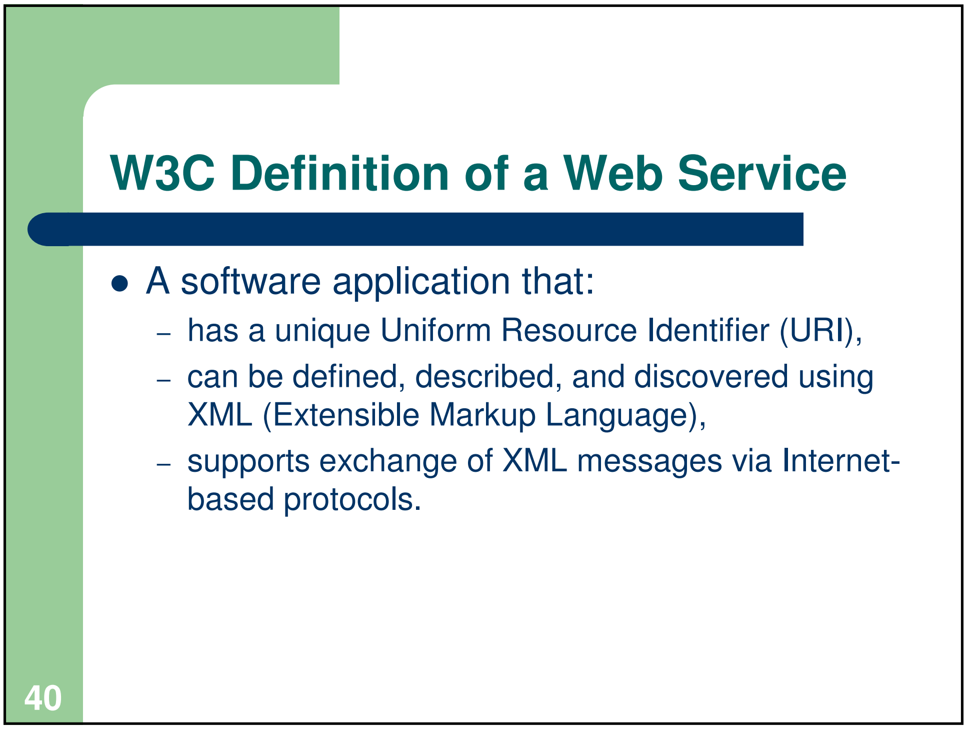


Service-Based Software Systems

- “Software is a service” business model
  - Software is not a product any more, but a service available over the network
  - Application Service Providers (ASPs)
- Service-Oriented Architecture (SOA)
  - Applications decomposed into distributed services
  - On-line access to libraries of composable components

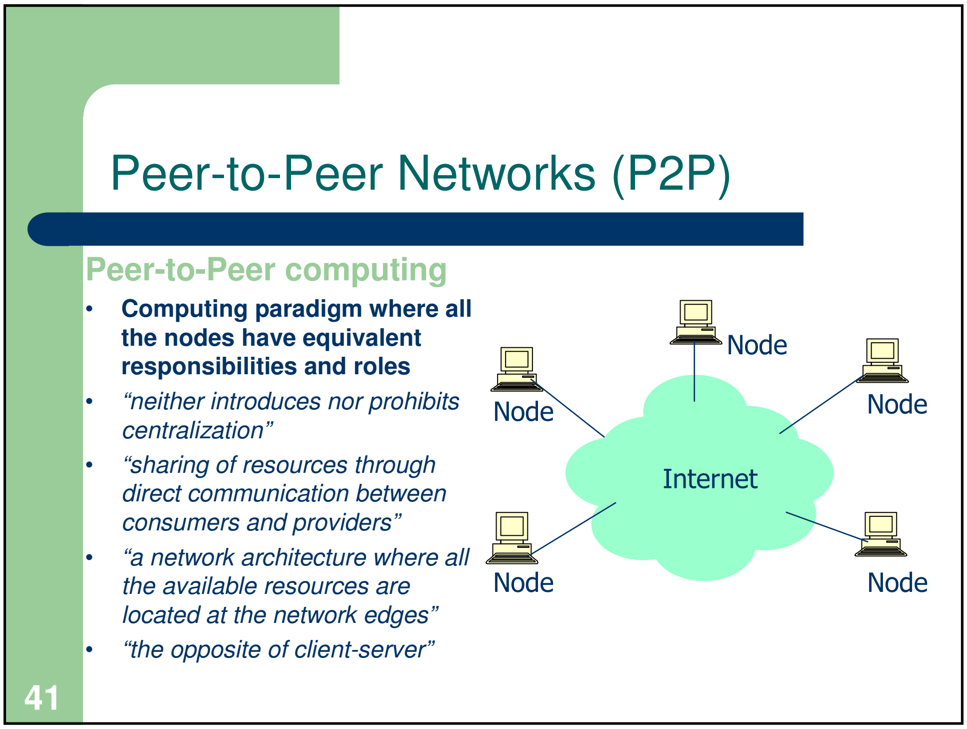


The “Publish-Find-Bind” Interaction Model



W3C Definition of a Web Service

- A software application that:
  - has a unique Uniform Resource Identifier (URI),
  - can be defined, described, and discovered using XML (Extensible Markup Language),
  - supports exchange of XML messages via Internet-based protocols.

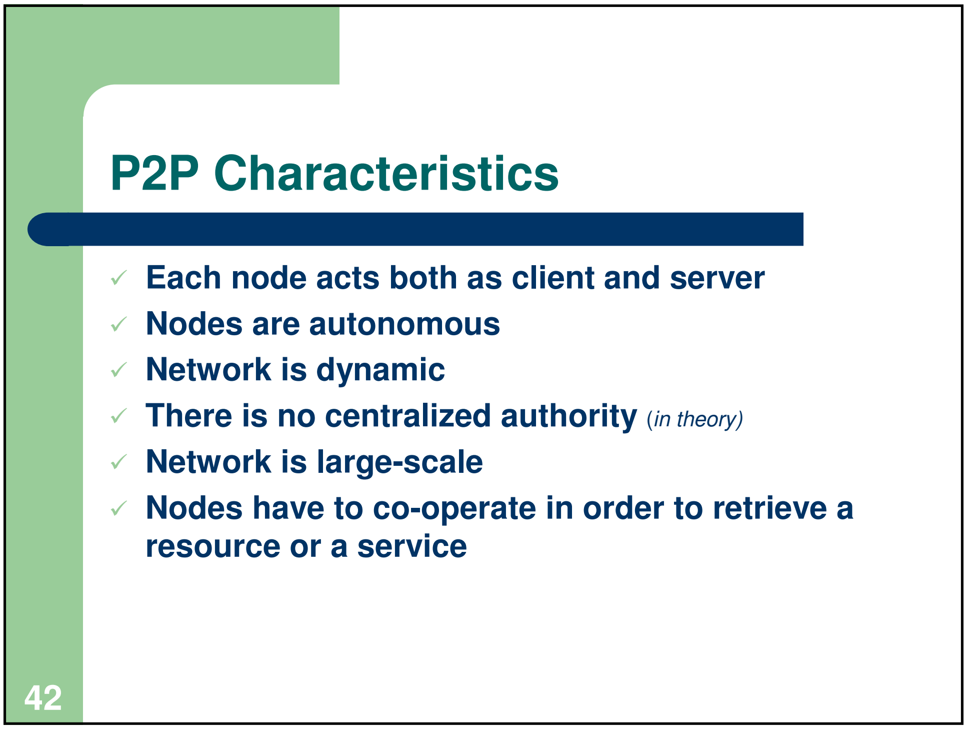


Peer-to-Peer Networks (P2P)

Peer-to-Peer computing

- **Computing paradigm where all the nodes have equivalent responsibilities and roles**
- *“neither introduces nor prohibits centralization”*
- *“sharing of resources through direct communication between consumers and providers”*
- *“a network architecture where all the available resources are located at the network edges”*
- *“the opposite of client-server”*

41



P2P Characteristics

- ✓ **Each node acts both as client and server**
- ✓ **Nodes are autonomous**
- ✓ **Network is dynamic**
- ✓ **There is no centralized authority** *(in theory)*
- ✓ **Network is large-scale**
- ✓ **Nodes have to co-operate in order to retrieve a resource or a service**

42

Examples of P2P Systems

- **Napster**
- **KaZaA**
- **Gnutella**
- **FreeNet**
- **NeuroGrid**
- **Chord, CAN, Tapestry**
- **JXTA**

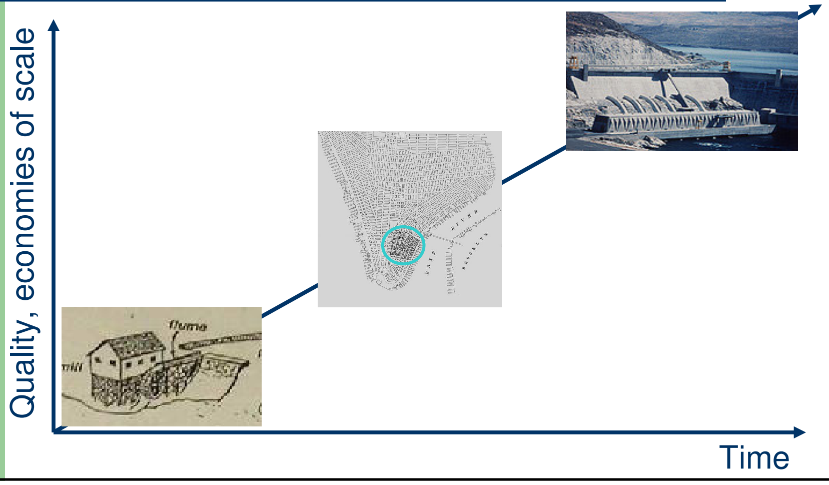
43

A Computing Grid

- Decouple production and consumption
  - Enable on-demand access
  - Achieve economies of scale
  - Enhance consumer flexibility
  - Enable new devices
- On a variety of scales
  - Department
  - Campus
  - Enterprise
  - Internet

44

The (Power) Grid: On-Demand Access to Electricity



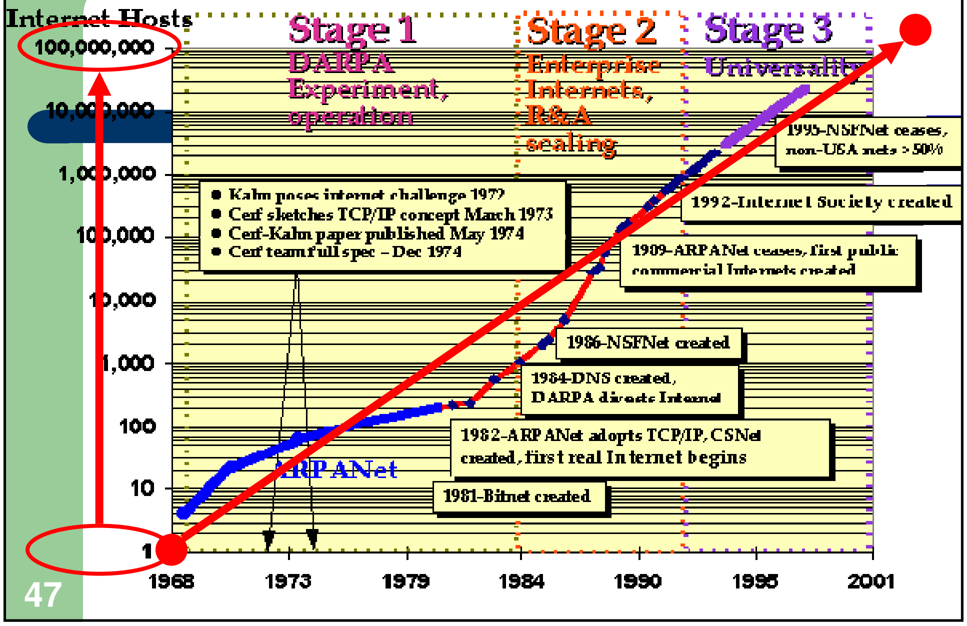
45

Not Exactly a New Idea …

- “The time-sharing computer system can unite a group of investigators …. one can conceive of such a facility as an … intellectual public utility.”
  - Fernando Corbato and Robert Fano, 1966
- “We will perhaps see the spread of ‘computer utilities’, which, like present electric and telephone utilities, will service individual homes and offices across the country.”
  - Len Kleinrock, 1967

46

But Things are Different Now …



- Kahn poses internet challenge 1972
- Cerf sketches TCP/IP concept March 1973
- Cerf-Kahn paper published May 1974
- Cerf team full spec – Dec 1974

1995-NSFNet ceases, non-USA nets > 50%

1992-Internet Society created

1909-ARPANet ceases, first public commercial Internets created

1986-NSFNet created

1984-DNS created, DARPA divests Internet

1982-ARPANet adopts TCP/IP, CSNet created, first real Internet begins

1981-Bitnet created

47