Department of CS & E, IITD

CSV888 - Special Module on Database Systems

Chapter 5:

Web Services - Concepts, Functionality, Architecture

Instructor: S. Bhalla

[I] Web Services: Distributed Information Systems [2]

WSs

- Web Services form a support base for
  Distributed Information Systems (DISs)

Study

- Problems these try to solve

$\longleftarrow$ similar to DISs

Design Constrains $\longleftarrow$ new

outline

- [A] Limitations of Present Technologies
- [B] Web Services Techniques -
- Concepts -
  how do WSs tacke the Application Integration
- [C] Functionality - needed for distributed applications
- [D] WS Architecture -

$\longleftarrow$ Connection: Synchronous $\longleftarrow$ Asynchnonous

$\longleftarrow$ Access : low level $\longleftarrow$ high level
(not RPC but SOAP)

[I] Simple Slides [3]

- Centralized Systems $\longrightarrow$ DBMS (one system)

- Network $\longrightarrow$ Communication

- Distributed System $\longrightarrow$ Dist. O.S. (commands)

- Distributed Database System $\longrightarrow$ 1 database image
- Directory
- Recovery, Logging
- Concurrency Control
- Load Balancing
- Transactions
- Distributed Heterogenous Database System $\longrightarrow$

- Two or more DBMSs working under a common software system
- Federated Database System $\longrightarrow$ (use JDBC)

[I] Web Services: Definitions [4]

• Definition(1) [72,133] -

" An application accessible to other applications over the web" .

+ any thing that has a URL (?)

• Definition(2) [UDDI Consortium] [203] -

" Self-Contained, modular business applications that have open Internet-Oriented, Standard-based interfaces" .

+ Internet-Oriented $\longleftarrow$ standard;

+ open $\longleftarrow$ published interface, can be invoked across the web

- (?) modular business applications (?)

• Definition(3) [WWWC (W3C) Consortium] [212] -

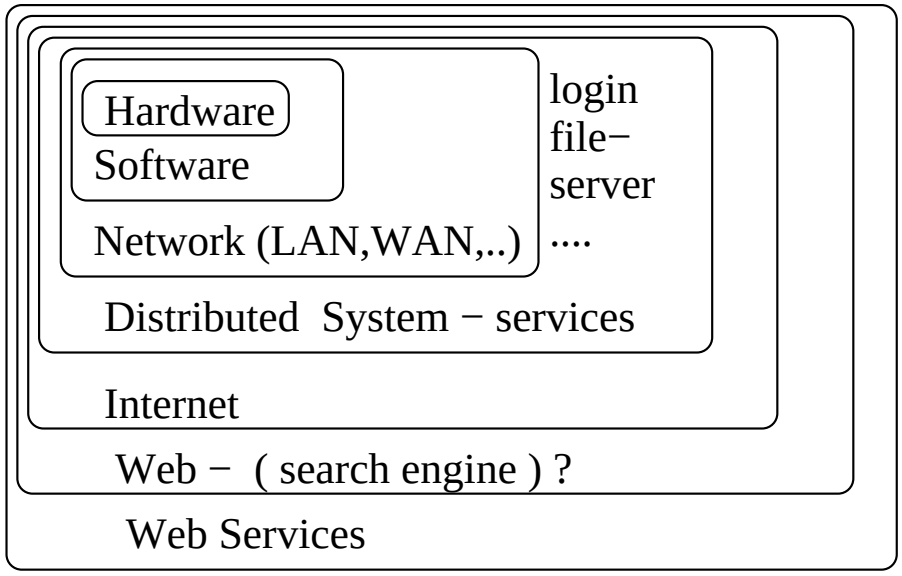
1. " A software application identified by a URI"

+ interfaces and bindings: Defined, Described and discovered using XML

2. WS uses XML-based messages exchanged via internet based protocols.

[I] Web Services [5]

- Large Scale B2B activity: Car maker $\leftrightarrow$ Parts supplier
- Large Scale B2C activity: Bank/Books/PCs $\leftrightarrow$ Customer
- Large Scale Application: System Upgrades $\leftrightarrow$ more Application integrations (Bottom-up design)



1. All levels (except Web) − directory
2. Homogeneous components − compatibility ?
   (software/hardware can connect with ease)

- Web Services $\longleftrightarrow$ external world connectivity

[I] Web Services: - [6]

• WSs (" services ") similar to middleware

⟵ up and running

⟵ Describe and advertise

⟵ possible to write client software
that can bind and interact with them.

• WSs are components

⟵ Can be integrated into complex distributed applications

• Definition (4): Webopedia : On-line technical dictionary -

" A standardized way of integrating web-based applications - using:

XML (tag the data),

SOAP (transfer data),

WSDL (description of services),

UDDI (list of services) open standard, over the internet

[I] WSs : usage [7]

• Service Oriented Paradigm in Application Development

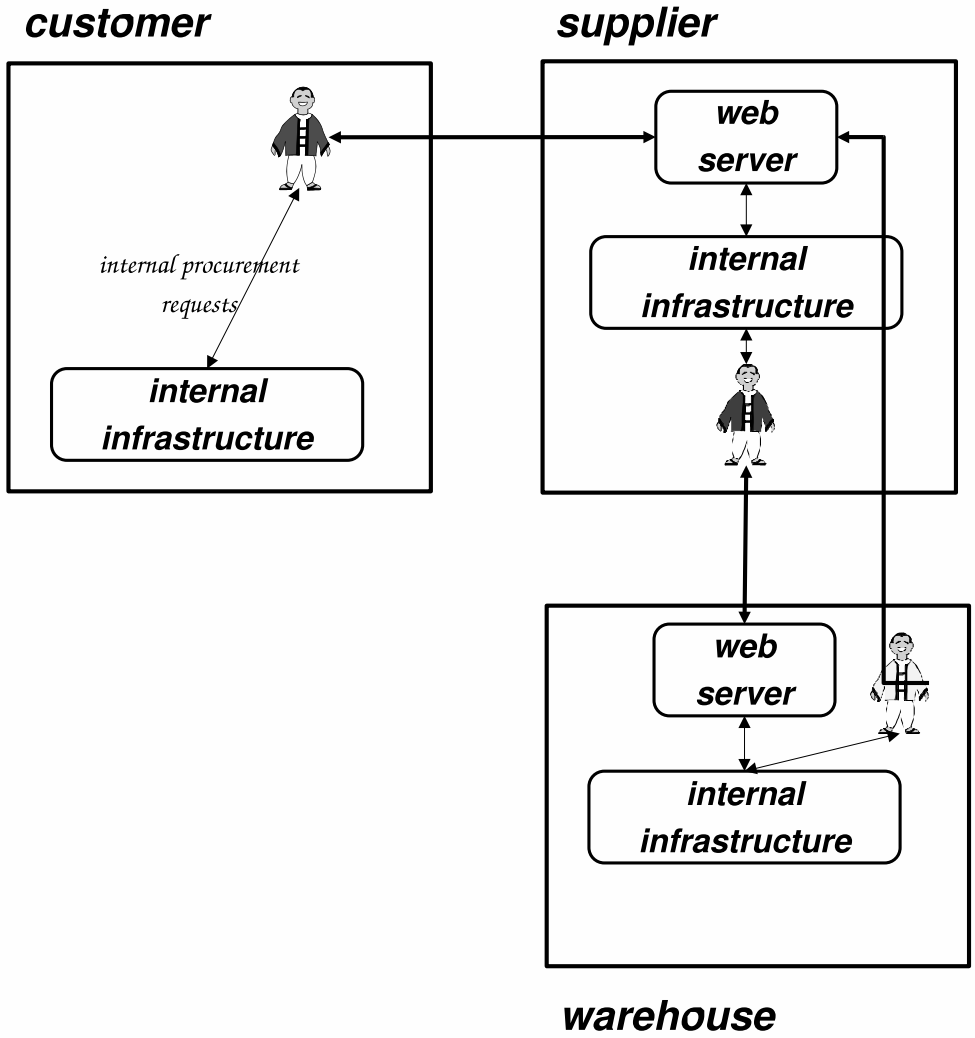
• Integration of serveral systems
+ autonomous,
+ heterogeneous

Example | Toyota Car Company and its part suppliers

• Business Process Span - need for automation

+ [At Present]: Employees access internal systems
+ Fill up web forms or send Emails across to order goods

[I] WSs based Systems [8]



B2B interactions occur by accessing Web pages, filling Web forms, or via email.

Copyright Springer Verlag Berlin Heidelberg 2004

[I] Conventional Middleware [9]

- Cross-organizational DISs -

(?) Where to put the middleware

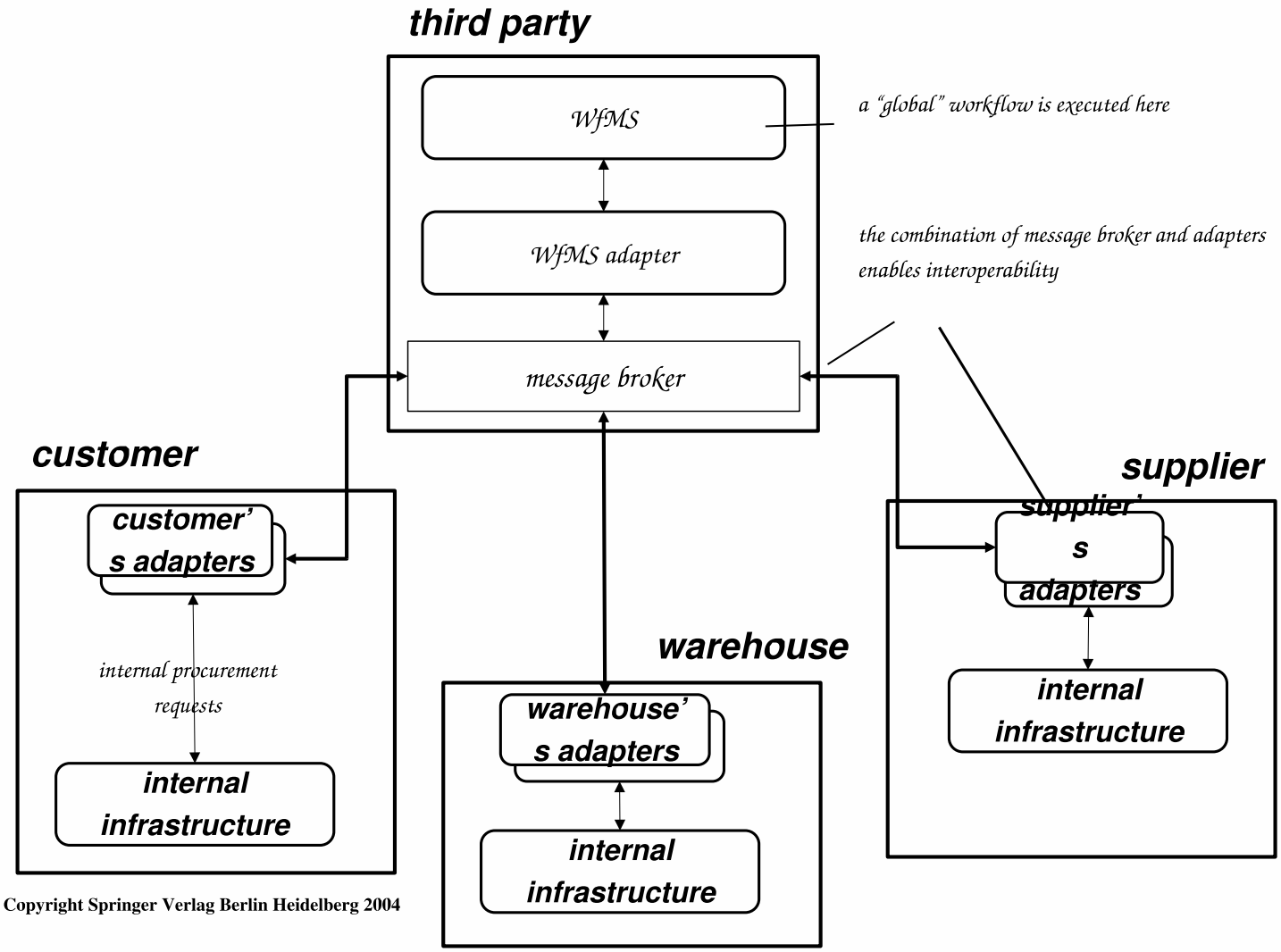
- Earlier, middleware was opted (chosen) by one company
- Many Companies must agree on co-operation

- Good idea: Few companies with close cooperation
- General Case: + lack of trust

+ autonomy
+ hiding internal business

- Centralized middleware hosted by one company

$\longrightarrow$ Not a possible solution



Copyright Springer Verlag Berlin Heidelberg 2004

[I] Point-to-Point [11]

- Customer - Supplier pair

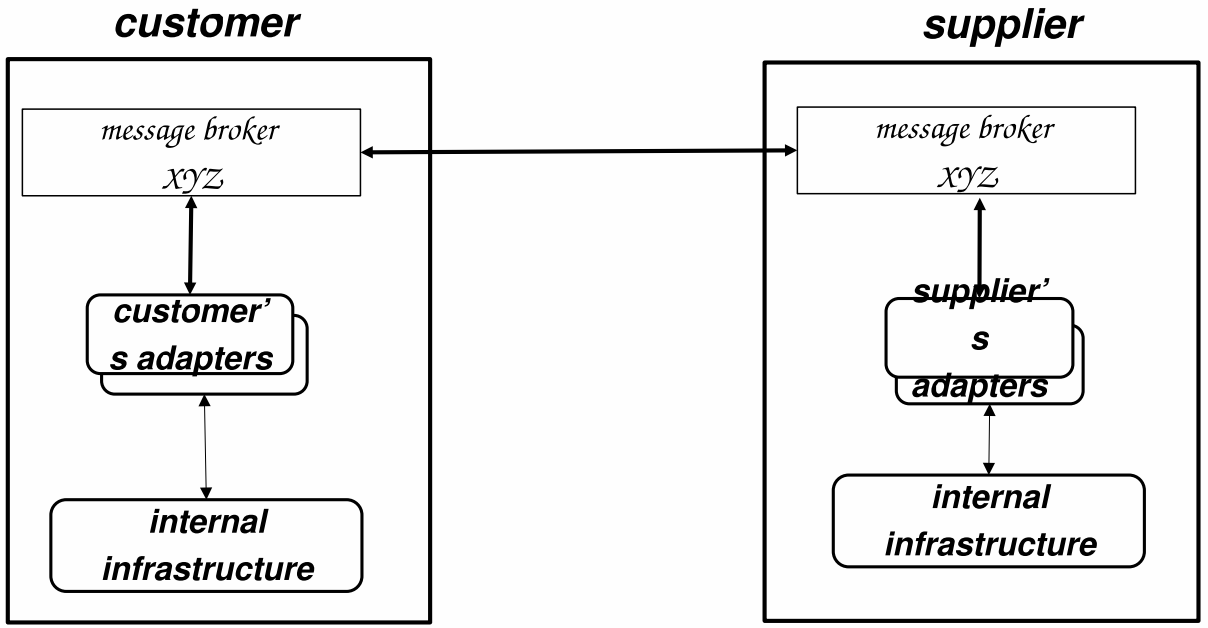
+ use common middleware
+ (advantage) Only intended customer can see the business data

- Problems: -

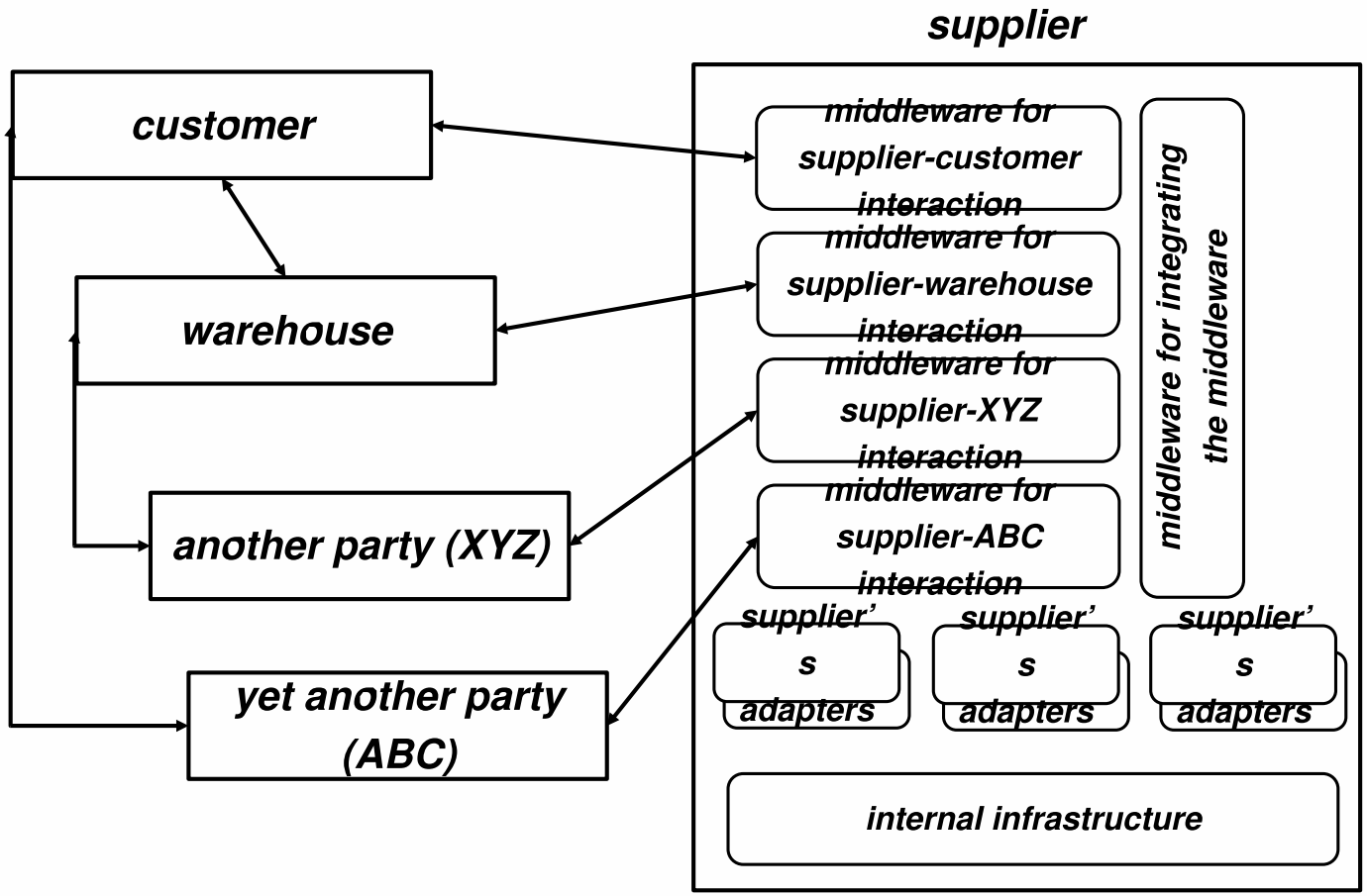
- A company interacts with many different partners

+ each may have a different platform

- Each company must support and maintain many middleware systems



Copyright Springer Verlag Berlin Heidelberg 2004



Copyright Springer Verlag Berlin Heidelberg 2004

[I] Point-to-Point Interaction [14]

• Interactions are long

+ earlier assumption - Transactions are short

• Interactions do not call a function

- may take a few hours or weeks

+ After the courier picks up the goods, may send despatch message

• Delays: Asynchronous exchanges

• Other Problems:

+ Trust domain is not same

+ authentication

+ encryption

• restriction - what other client can do

+ Companies severly restrict - what other clients can do

+ (?) What resources can be locked; (?) control over locking

[I] Service Oriented Paradigm [15]

Web Activities

- Every service avaiable on the web
ample
- Book store

Web Services

- Service in Software Sense
ample
- middleware

outline

- [A] Web Services: a software application
- [B] - published and stable program interface
- [C] - invokation made by a program
- [D] - usage same as middleware
- [E] - WSs are loosely coupled
- [F] Design: Services are autonoumous
- [G] [New Problems]: redesign middleware for peer-peer function

[I] Middleware Protocols [16]

• Redesign : Peer-to-peer and across companies

• Example P2P: Transaction Commit (2PC)

Design assumptions - do not hold in cross-organization interactions

+ Central Coordinator: Can not work in Cross-organization environments

+ Central Coordinator can lock resources at other systems

$\longrightarrow$ [X] Coordinator - There is lack of trust and confidence

[A1] $\longrightarrow$ 2PC to be redesigned: work in a fully distributed fashion

[A2] $\longrightarrow$ must be extended: allow flexibility in terms of locking resources

$\longrightarrow$ + All interactions and coordination protocols $\longleftarrow$ Redesign for WSs

[I] Middleware Protocols [17]

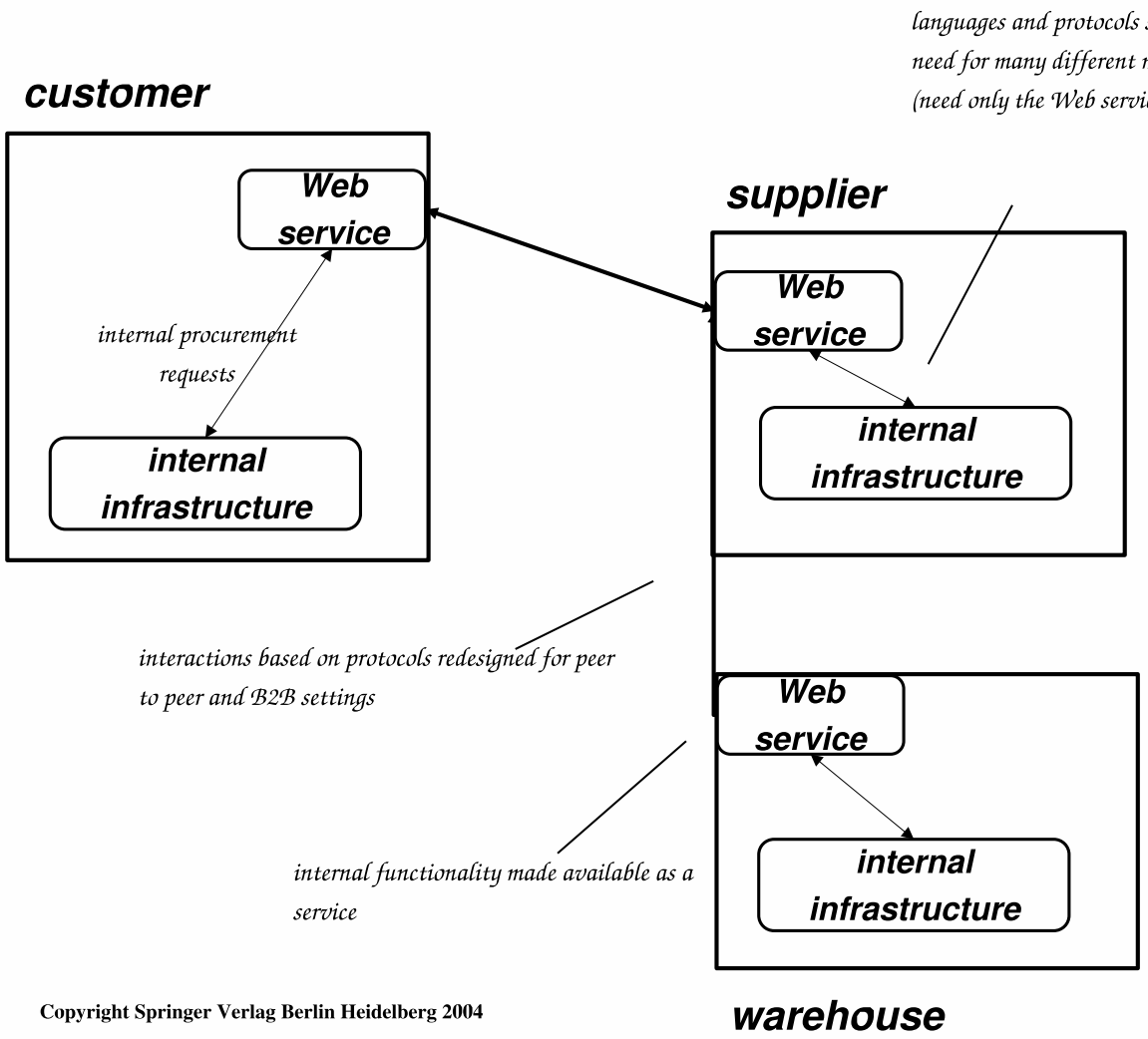
• Many properties provided by conventional middleware

+ reliability / + guaranteed delivery $\longrightarrow$ Centralized platforms: redesign

• Example: Deadlock Handling:
$\longrightarrow$ (centralized coordinator)

+ make a Transaction Wait For Graph (TWFG); or

+ send a probe to find the deadlocked list of transactions



Copyright Springer Verlag Berlin Heidelberg 2004

[I] Web Services middleware [19]

- Web Services need to be based on standards
- Interaction between companies (B2B integration problems) -

\+ P2P
\+ Through standard protocols
\+ There is no central middleware platform

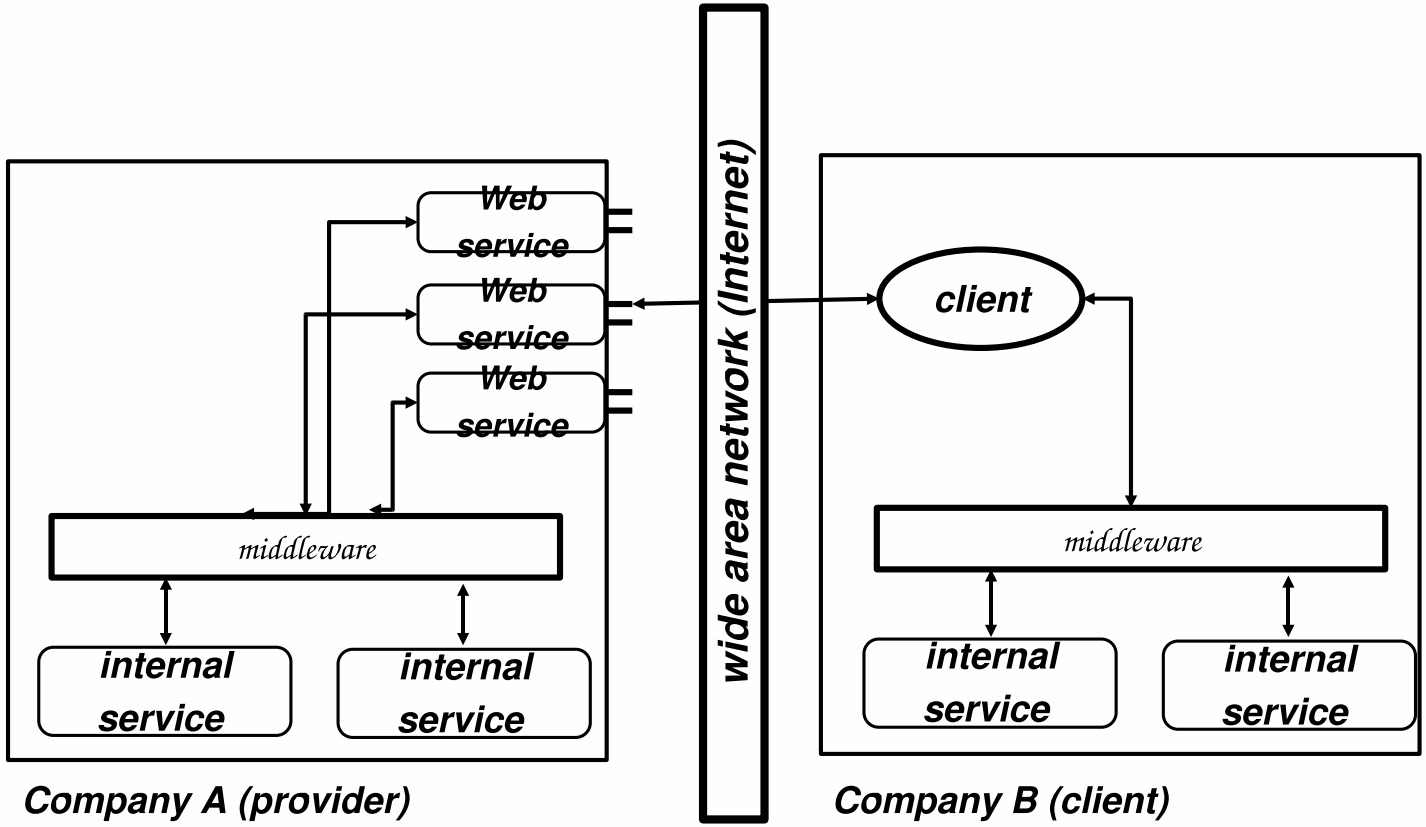
- WSs : Each party exposes its internal operations as WSs,

$\longrightarrow$ entry point to local information system
$\longrightarrow$ WS interface : functionality performed by internal system

- Web Services (WSs) use middleware for execution of P2P protocols

- Expose (interface is discoverable) -

1. WSs are sophisticated wrappers ;
2. Encapsulate one or more applications ;
3. provide a unique web access interface.



Copyright Springer Verlag Berlin Heidelberg 2004

[I] WSs Parts [21]

- WSs wrappers: Hide heterogeniety

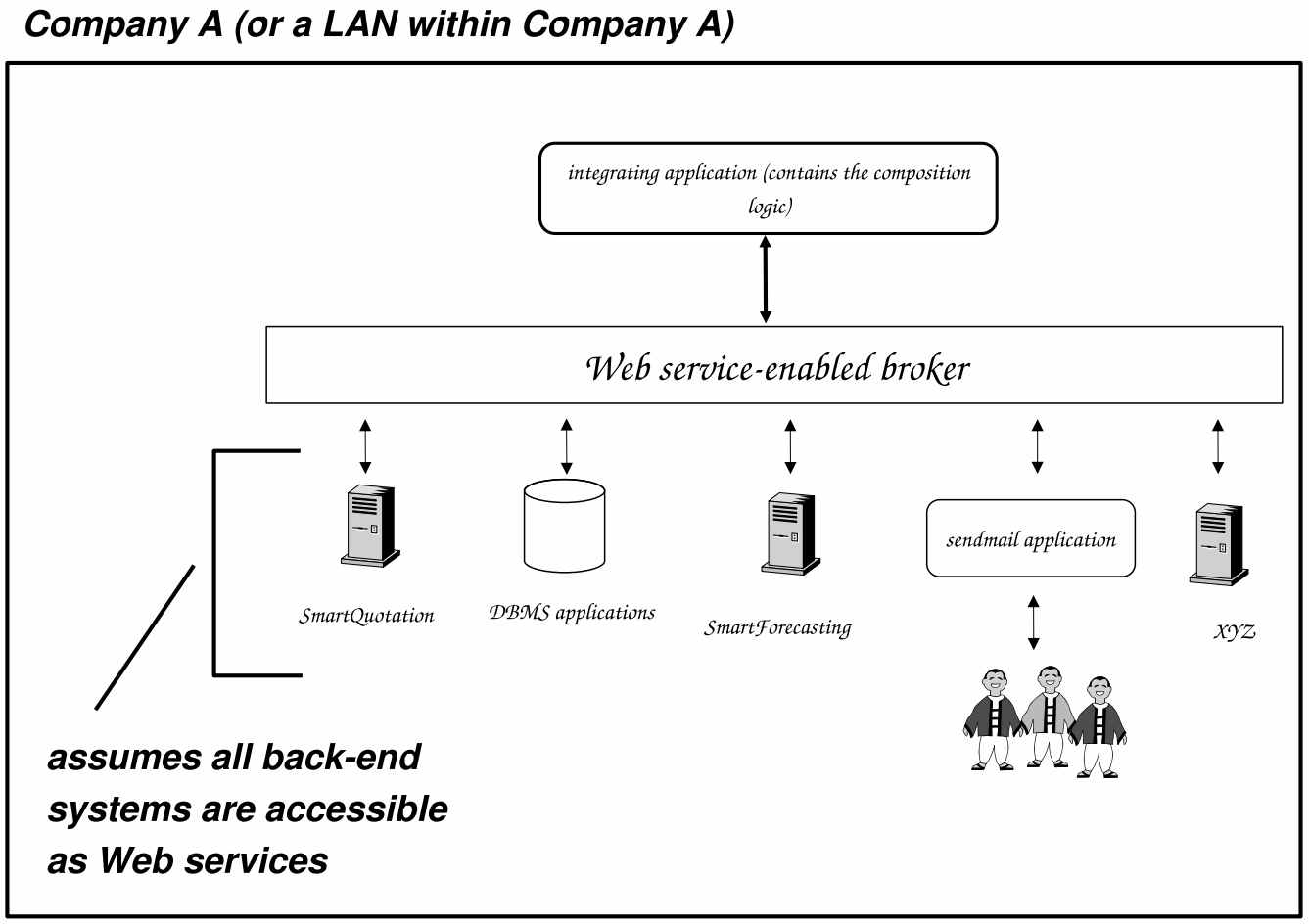
- clients -[see]- wrappers (standards) homogeneous

+ reduce difficulties of integration (standard components)

- Future Software applications

+ out of box with WS interfaces

- Even LAN based systems : WSs based ?



Copyright Springer Verlag Berlin Heidelberg 2004

[I] WSs Technologies: Service Description [23]

What

- What is Service

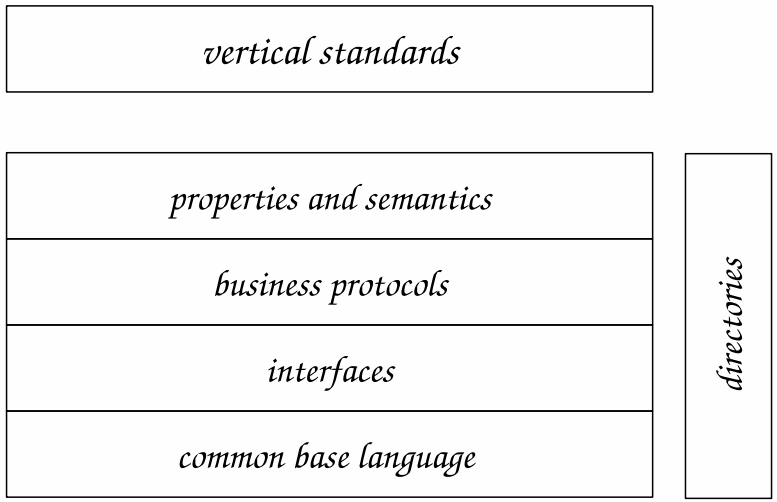
How

- How service - can be described ?

outline

- [A] Service Descriptions in Conventional Middleware
- [$\longrightarrow$] are known to programmers developing the clients
- [i] Sementics of different operations
- [ii] Order in which these will be invoked
- [iii] Other properties
- [$\longrightarrow$] Based on a) interfaces b) IDLs
- [+] IDLs automatically generate stubs and

contribute basis for dynamic binding and linking



Copyright Springer Verlag Berlin Heidelberg 2004

[I] Components [25]

• [1] Common Base Language

- common meta language for all aspects

+ XML

+ has a flexible sysntax

• [2] Interfaces (IDLs in case of middleware)

+ WSDL : need to specify URI of service

+ WSDL : How to invoke (Which transport protocol to use (HTTP?))

• [3] Business Protocols: WSs offer many operations

$\longrightarrow$ clients invoke operations

Example: Procurement - Customer + request a qoute;+ order;+ pay.

[Operations for a purpose] $\longrightarrow$ conversation

[Business Protocols]: $\longrightarrow$ rules for conversations

+ WSCL - WSs Conversation Language (being studied)

+ BPEL : Business Process Execution Language for WSs

[I] Components [26]

• [4] Properties and Semantics:

+ Functional Interface

+ Non-functional interface (new)

- Cost of service; - text (goods return policy);

$\longrightarrow$ UDDI used : How to organize and information

+ Universal Description, Discovery and Integration (UDDI)

• [5] Verticals

+ Vertical Standards define specific interfaces, protocols, ...

$\longrightarrow$ that WSs in each domain should support

Model: WSs $\longrightarrow$ Vertical Standards $\longleftarrow$ Client

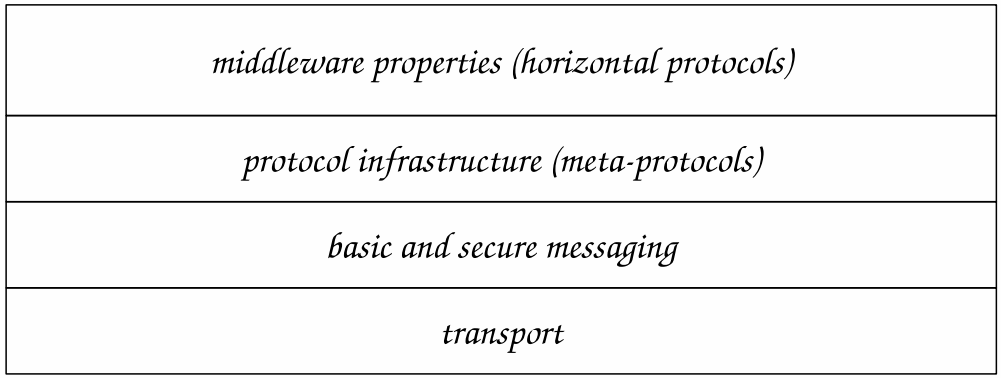
[I] Service Discovery [27]

- Clients : [interactions] [P2P fashion] : Directory Service
- APIs and protocols $\longrightarrow$ UDDI (publish and discover)

$\longrightarrow$ Concerned with static and dynamic binding

+ middleware properties (horizontal protocols)
+ protocols infrastructure (meta-protocols)
+ messaging (basic and secure)
+ Transport

[I] UDDI based Service Interactions [28]



Copyright Springer Verlag Berlin Heidelberg 2004

[I] UDDI based Service Interactions [29]

• 1. Transport:

WSs : Communication Network is hidden behind a transport protocol
+ use many (HTTP - most common)

• 2. Messaging:

WSs : standard way to format and package
+ use SOAP (Simple Object Access Protocol)
$\longrightarrow$ specifies a message template to add on the top
$\longrightarrow$ specifies transactional property and Security property (encryption)

• WS-Security: How to implement secure exchanges with SOAP
+ Additional Specifications and standards

[I] UDDI based Service Interactions [30]

- 3. Protocol Infrastructure (meta-protocols)
- Before an interaction: client and services

+ need to agree :protocols? :co-ordination? :messages ?

WS-Coordination: Specification

$\longrightarrow$ Standardize the meta-protocols

$\longrightarrow$ How to use the WSDL and SOAP for a task

- 4. Middleware (horizontal) protocols
- P2P protocols

Example: relibility + transaction properties

$\longrightarrow$ execute 2PC, Deadlock handling, ...

WS-Transaction: Specification

$\longrightarrow$ Based on WS-Coordination;

$\longrightarrow$ Defines - how to build a transaction Perspective

[I] WSs Architecture [31]

• 1. WSs Technologies (internal architecture)

$\longrightarrow$ internal middleware (similar to conventional middleware)

Request $\longrightarrow$ Basic system $\longrightarrow$ response

• 2. WSs Technologies (external architecture)

+ standardization efforts

+ Purpose: Integrate different WS components

[A] Centralized Brokers

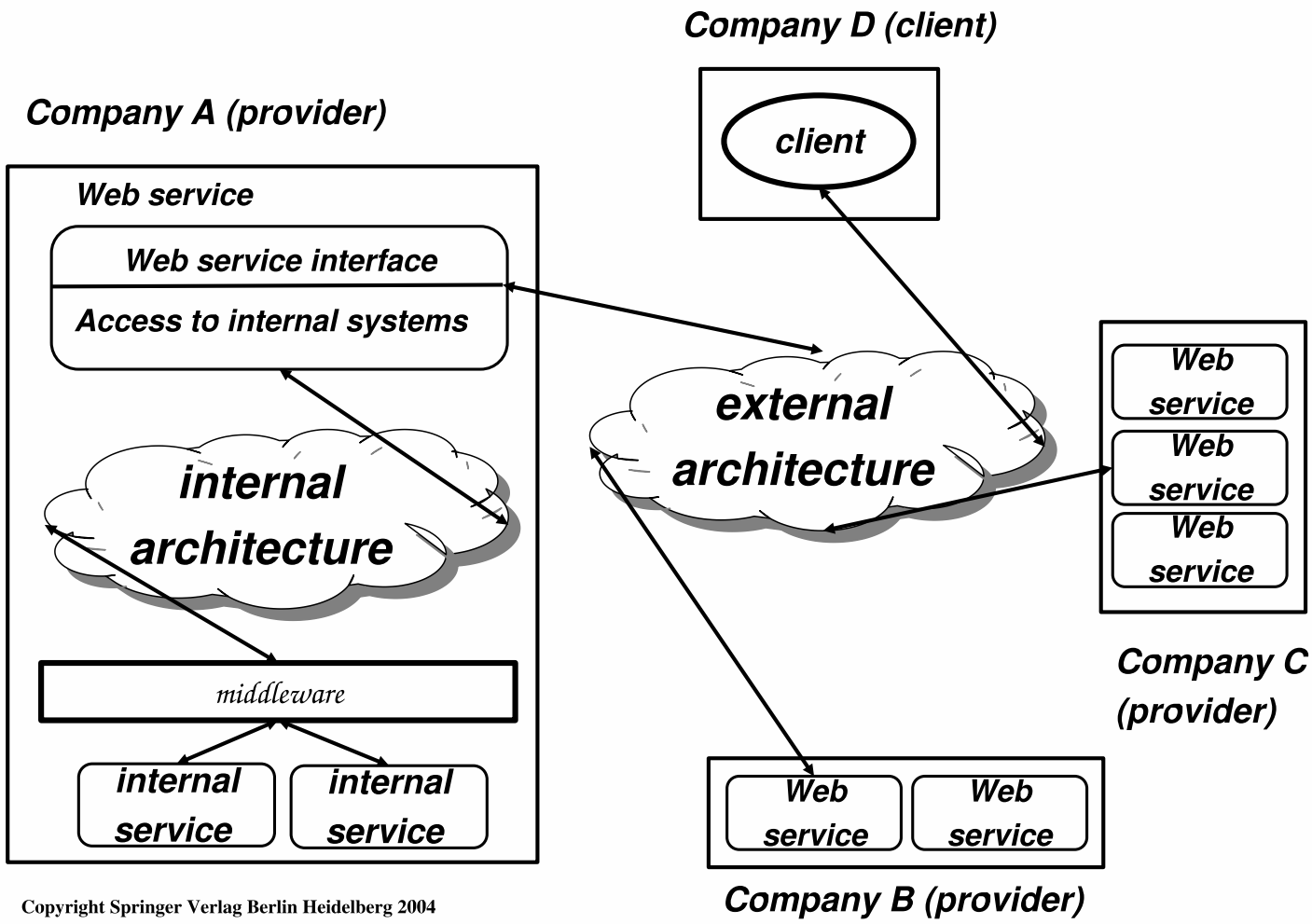
$\longrightarrow$ Route Messages + Provide properties for interactions:

logging; transactional guarantee; Name and directory; reliability

[B] Protocol Infrastructure : Implement P2P protocols

[C] Service Composition Infrastructure Tools:

+ definition and execution of composite services



Copyright Springer Verlag Berlin Heidelberg 2004

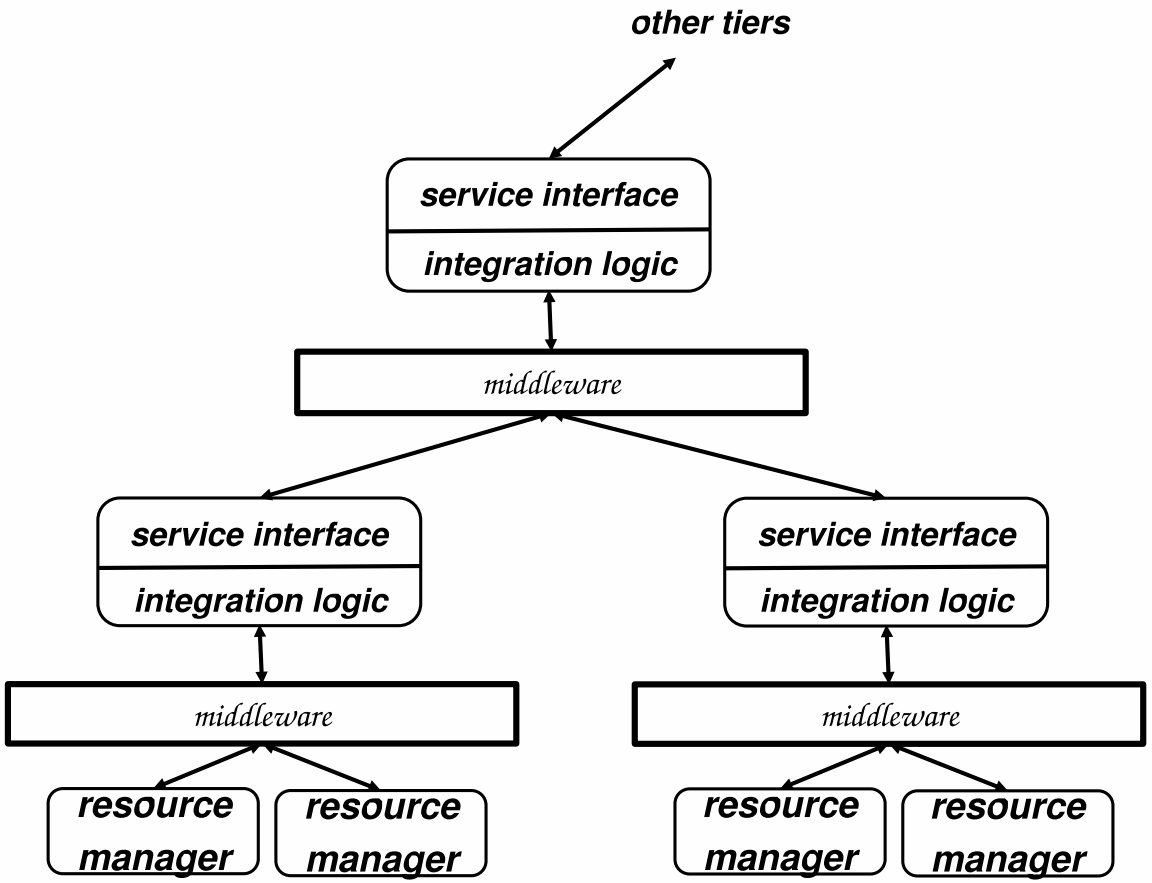
[I] Internal Architecture [33]

- WSs: (a) Simple ; (b) Composite | N-Teir Architecture |
- WSs invoke internal services

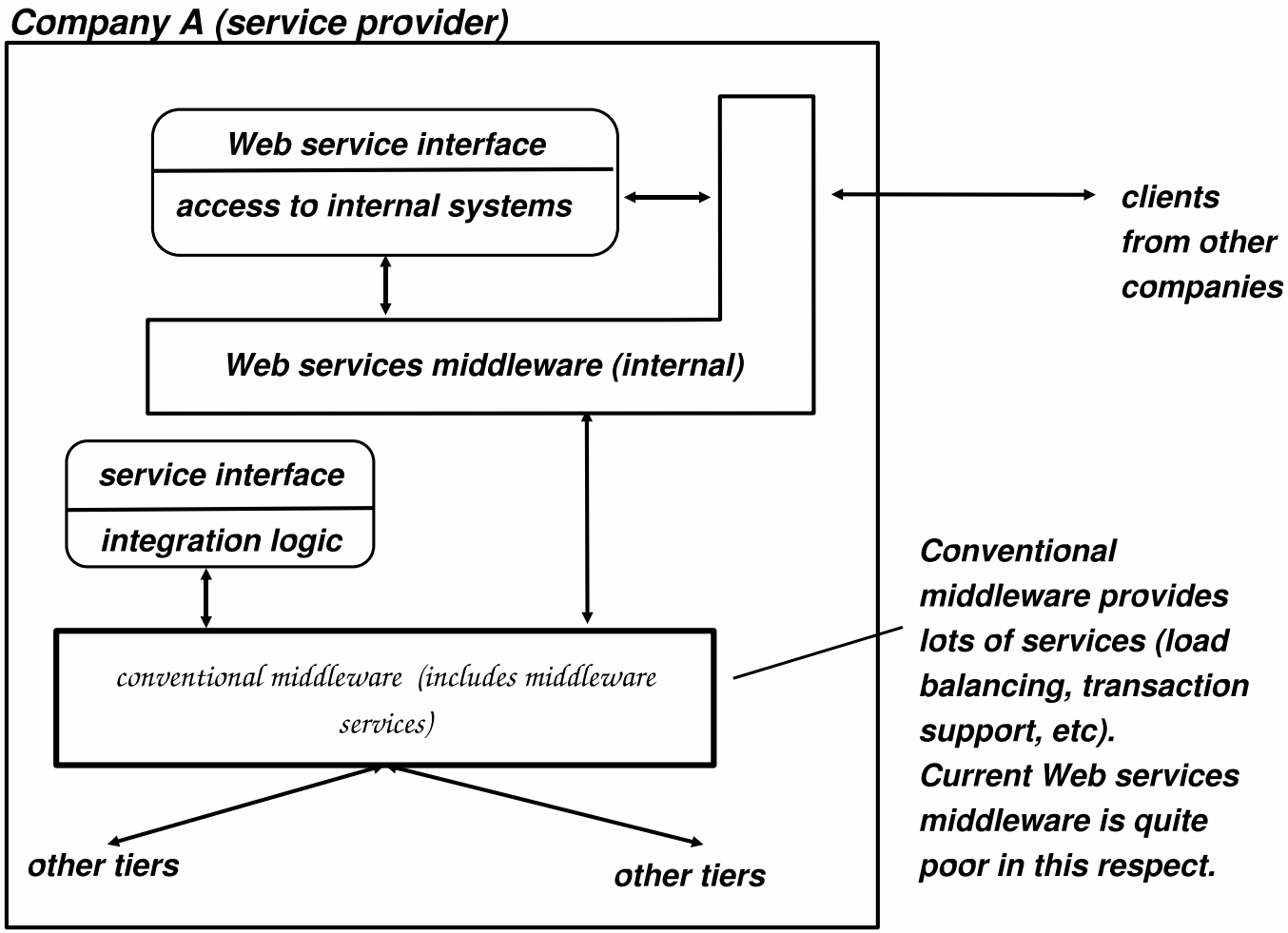
$\longrightarrow$ implement the needed application logic

$\longrightarrow$ Collect results

- Cause overheads
- package and unpackage messages



Copyright Springer Verlag Berlin Heidelberg 2004



Copyright Springer Verlag Berlin Heidelberg 2004

[I] External Architecture of WSs [36]

- 1. Centralized Brokers
- 2. Protocol Infrastructure
- 3. Service Composition Infrastructure
- Example:

- Implementation of Name and Directory Service

(?) Where should this middleware reside

- [A] Peer-to-peer : participants cooperate to provide service
- [B] Intermediatories and brokers act as the necessary middleware

[I] External Architecture of WSs [37]

Main Issue

- (?) provide the degree of reliability and trust required for industrial strength systems

Main Assumption

- Can find a trusted and reliable site
$\longrightarrow$ Name and Directory service

outline

- [A] Such servers
$\longleftarrow$ Part of WSs middleware Infrastructure
- [B] $\longrightarrow$ $\longrightarrow$ Participants and a part of middleware : may reside at different sites.
- [C] Standardized WSs broker: only one (so far)
$\longrightarrow$ Name and Directory service
- [D] Most of reference papers on Architecture
$\longrightarrow$ Name and Directory service
- [E] WSs discovery $\longrightarrow$ part of External middleware

[I] WSs External Architecture [38]

- Service Providers:

$\longrightarrow$ create WSs
$\longrightarrow$ define interfaces for invoking them
$\longrightarrow$ generate service descriptions
$\longrightarrow$ publish service in service registry

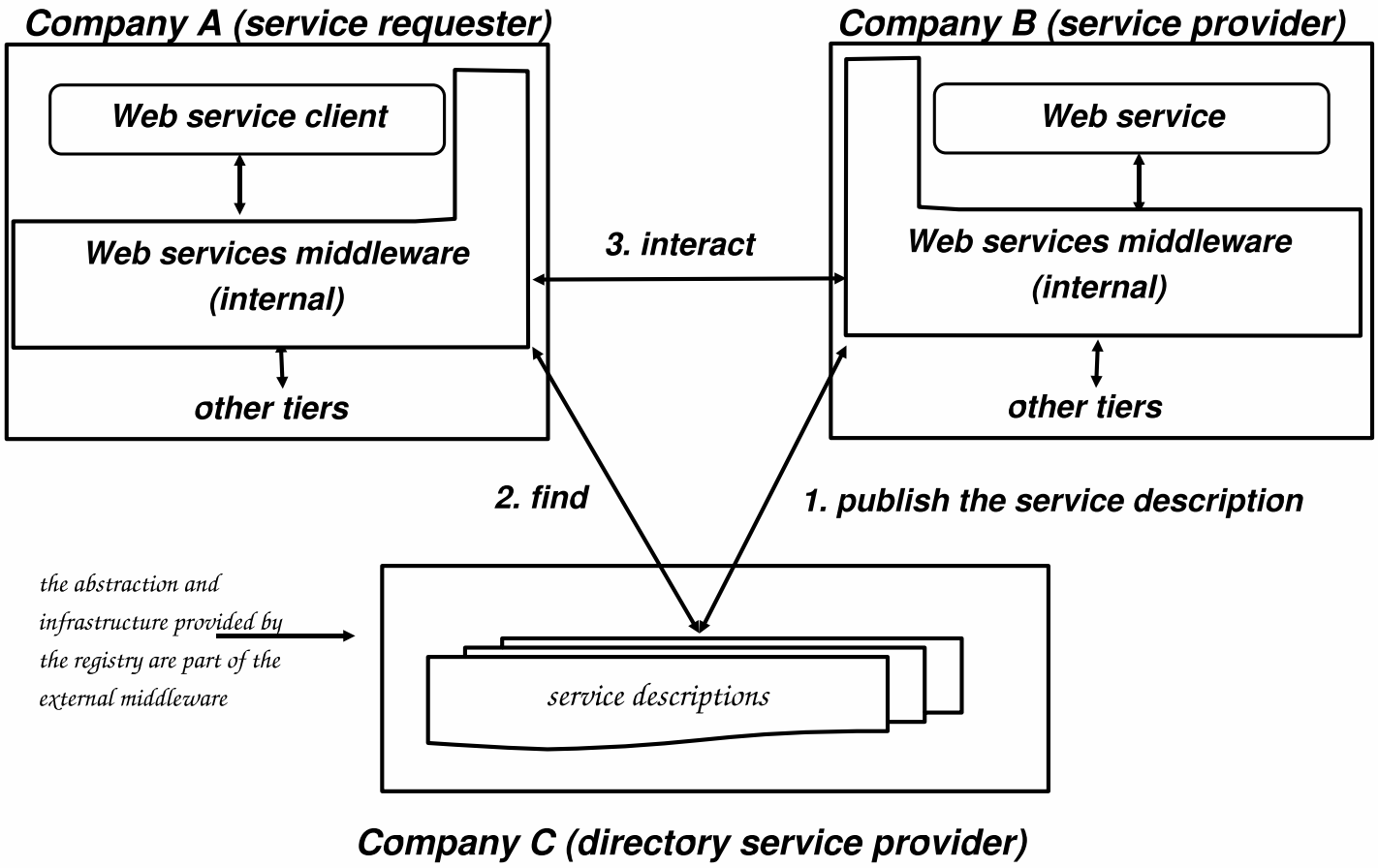
- Service Registry: catalogues and searches when service request arrives

$\longrightarrow$ Answer to - Where to locate and how to invoke

- Service Requester: Can bind to service provider

by invoking the service (directory/Service registry)

- Address and Interface details are known apriori by the requester



Copyright Springer Verlag Berlin Heidelberg 2004

[I] External Architecture [40]

- Figure 5.13 [shows] $\longrightarrow$ service discovery

- as the only component of the WSs middleware

- (?) TM features of the TP monitor

- (?) Services offered by CORBA (Object Broker)

- Research Problems

- 1. How to build centralized brokers at well known locations
- 2. Provide high degree of efficiency
- 3. A broker to mediate business interactions

- may not be acceptable to companies

$\longrightarrow \longrightarrow$ Two approaches - [A.] Centralized [B.] P 2 P

[I] [A.] Consider a Centralized TM [41]

Concept

- Imagine centralized broker for WSs

Design

- Implementation -
similar to brokers in conventional middleware

outline

- [A] Technically feasible
- [B] Difficult in practice
- [$\longrightarrow$] requires a standard way of running transactions (accepted by everyone) -
so that transaction semantics are not violated
- [C] Transactional Semantics
$\longrightarrow$ depend on middleware platform
- [D] Standardize transactional semantics across middleware tools

[I] [A.] Consider a Centralized TM [42]

Idea

- under preparation - WS-Transaction Specification

Design

- Middleware platforms will follow
a common transactional interface

outline

- [A] Assumption - All participants trust the broker

- [B] Assumption ? (O.K.) in restricted settings
- [C] Few Examples :
Trusted brokers in on-lone shopping
- [Examples] - Yahoo ! Lycos !

- [E] Situation not same as directory service

- [F] More research going on

[I] [B.] Consider a P2P model TM [43]

- Each service requester its own TM

$\longrightarrow$ on transaction execution - its own TM executes

- Requires $\longrightarrow$ standardized transactional interactions

+ same as centralized solution

- Functionality provided by a P2P solution (?)

may be a subset of earlier systems

- Present Trends: P2P model

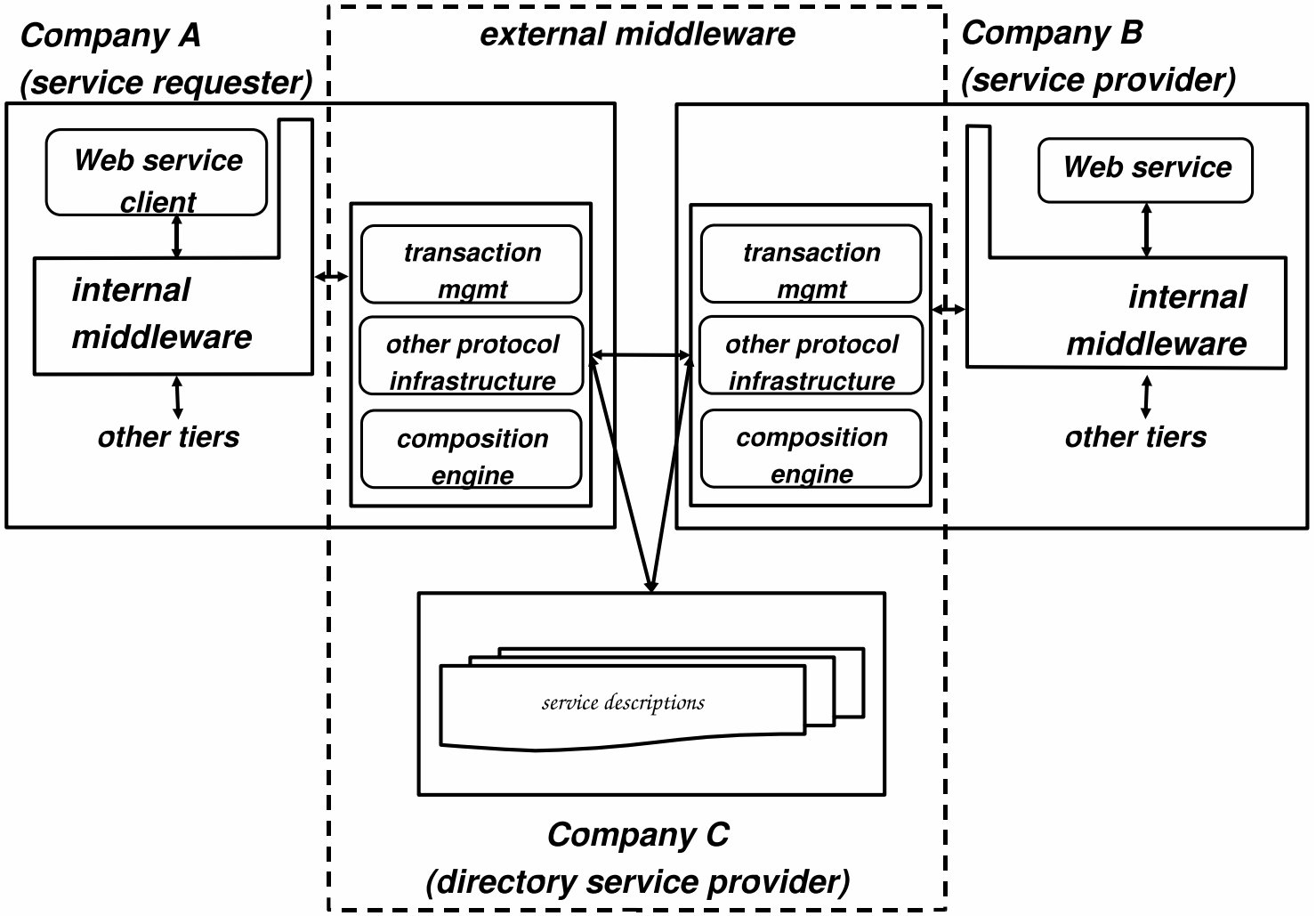
[I] Service Composition Tools [44]

- Can be centralized

- Implementations are often properietary and confidential

- Infrastructure $\longrightarrow$ will be provided by the service provider

+ [P2P model] not by a Third party



Copyright Springer Verlag Berlin Heidelberg 2004

[I] Summary [46]

• 1. WSs are being used as -
Sophisticated wrappers over conventional middleware platforms

• 2. WSs comprise additional Tier

• 3. Allow middleware services to be invoked as WSs

• WSs are defined by -
• [A.] Internal Architecture
+ Connection and local information system
- supported by internal middleware
• [B.] External Architecture
+ How to discover and interact
- supported by external middleware
- relies on standards
- Configure WSs landscape
- perform cross-organization interactions across internet