Distributed Information Systems

Motivation

- To understand the problems that Web services try to solve it is helpful to understand how distributed information systems evolved.
- While technology has changed, problems stayed the same.

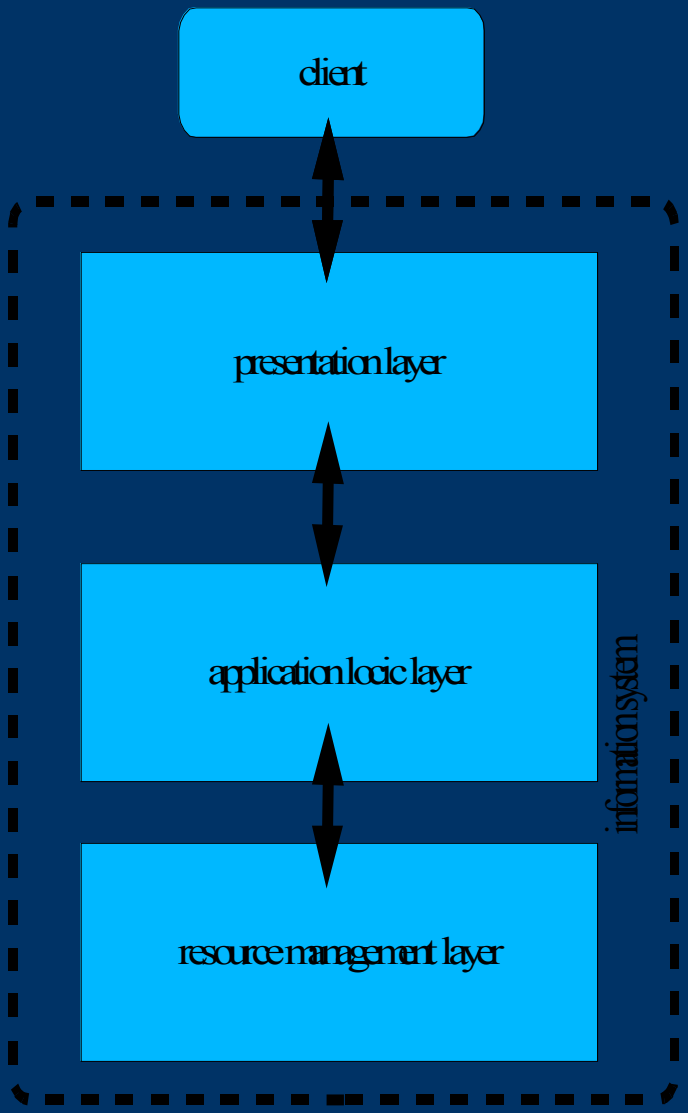
Overview

distributed Information Systems (IS)

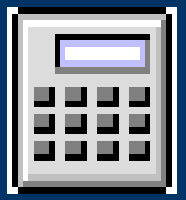
- design and related aspects
- architectures
- communication patterns
- scaling

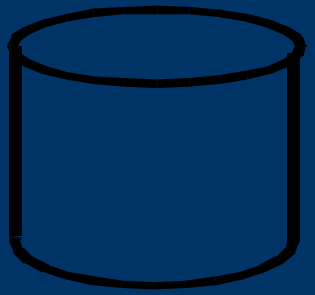
Design and related Aspects

Layers of an IS Example




```
<html>
...
</html>
```





Presentation Layer

- here is decided HOW data should appear to the user
- sometimes referred to as the client (not entirely true!!!)

Application Logic Layer

- Data Processing ('The actual Program')
- here the algorithms are implemented
- this Layer is often referred to as
  - services
  - business logic
  - business rules
  - server

Resource Management Layer

- deals with and implements different data sources of IS
- is the 'data layer' in a restricted interpretation (Database Management System)
- can also be an external system, which recursively uses other ISs

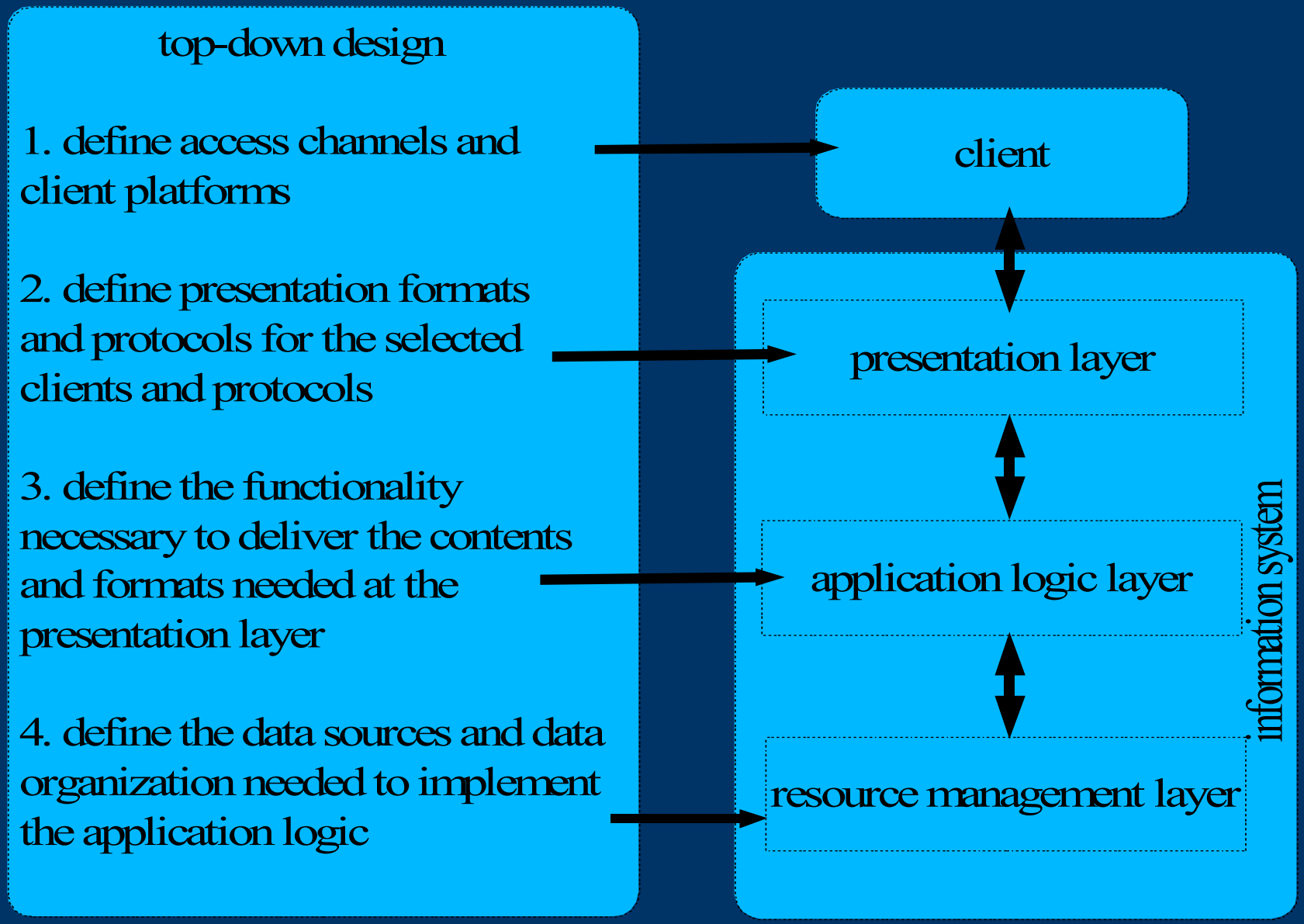
Designs of IS

- *top-down design*
- *bottom-up design*

top-down design

- starts with defining functionality desired by the client ('toplevel goals')
- implementation of application logic
- defining the resources needed by applictation logic

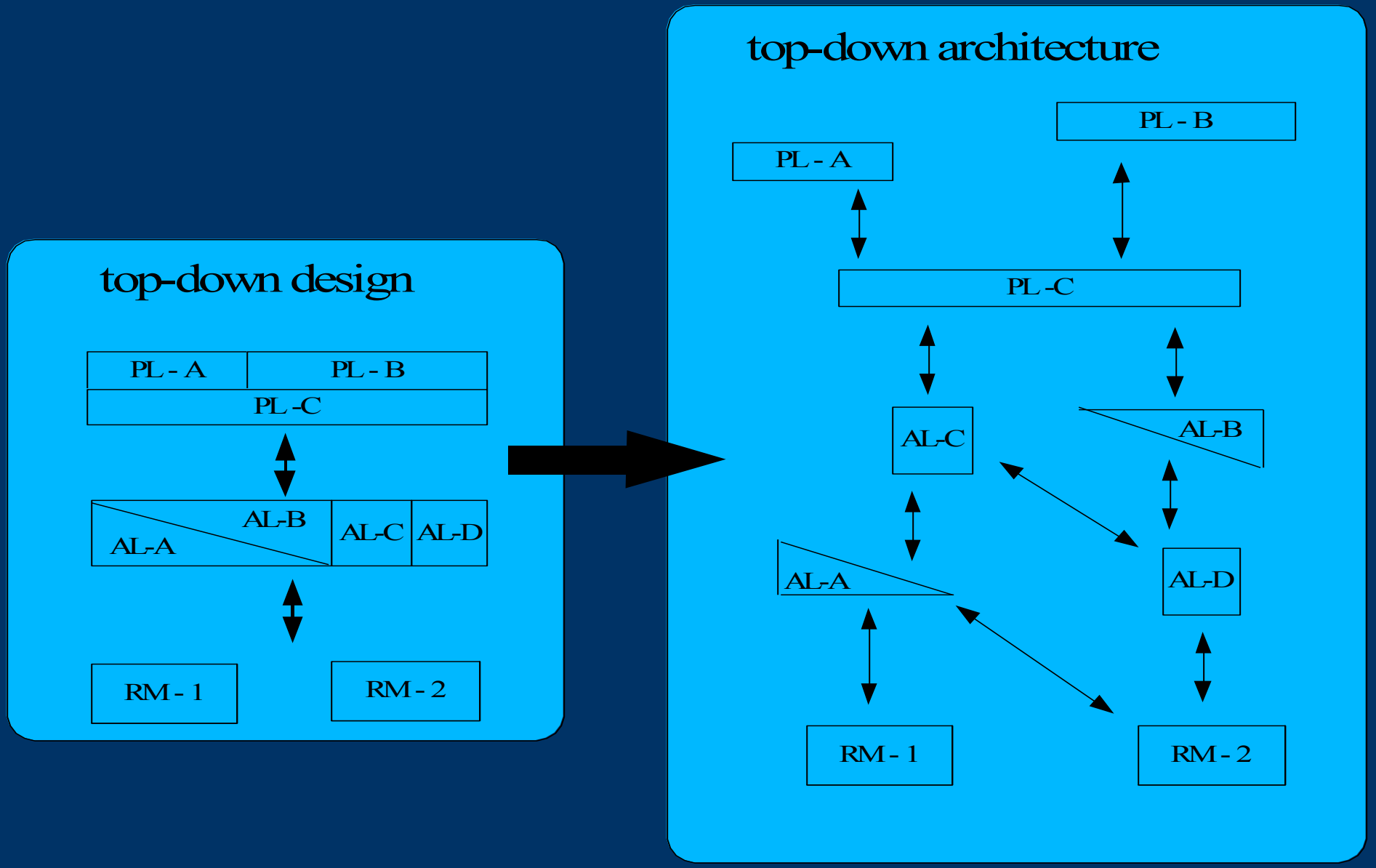
top-down [example]



top-down design

- usually created to run in homogenous environments
- way of distribution has to be specified
- results in tightly coupled components:
  - functionality of each component heavily depends on functionality of other components
  - design is component based, but components are not standalone

top-down design



advantages & disadvantages

- advantages:
  - design emphasises final goals of the system
  - can be optimized for: functional and non-functional(performance, availability,..) issues
- disadvantages
  - can only be designed from scratch
  - legacy systems cannot be integrated

today few ISs are designed purely top-down

bottom-up design

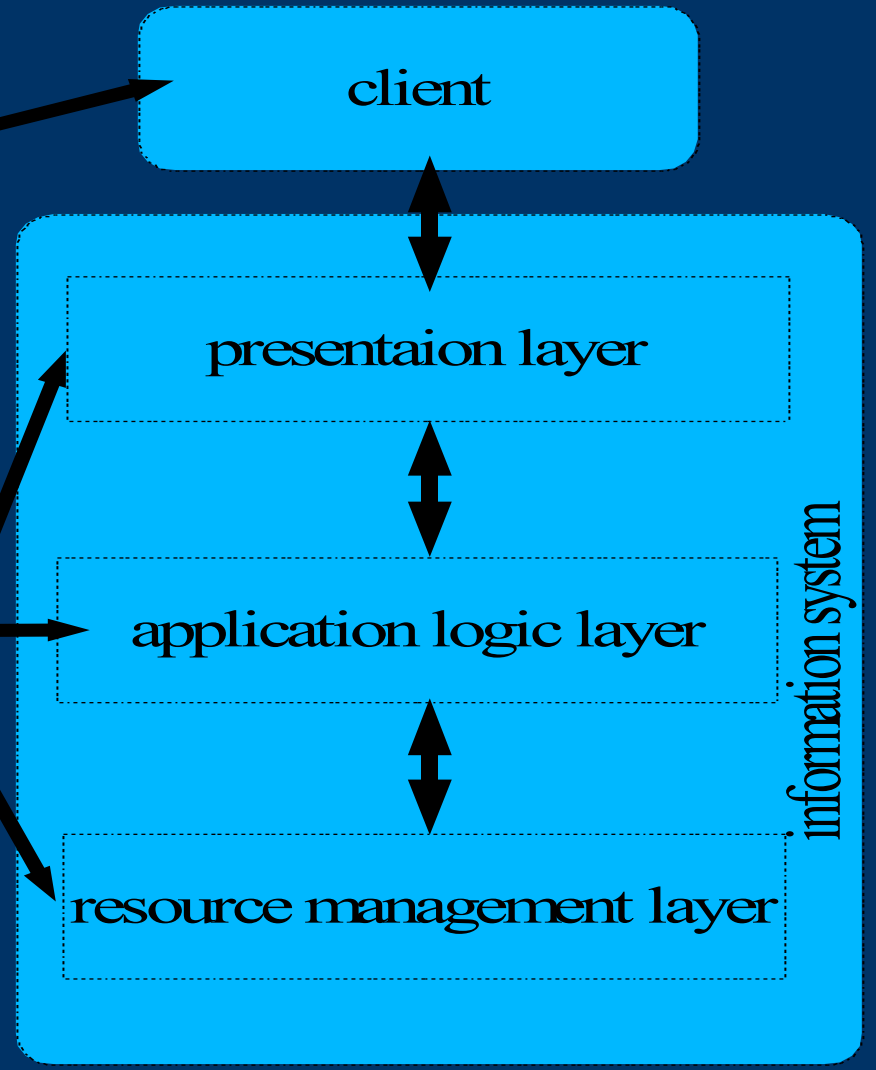
bottom-up design

1. define access channels and client plattforms

2. examine existing resources and the functionality they offer

3. wrap existing resources and integrate their functionality into a constistent interface

4. adapt the output of the application logic so that it can be used with the required access channels and client protocols

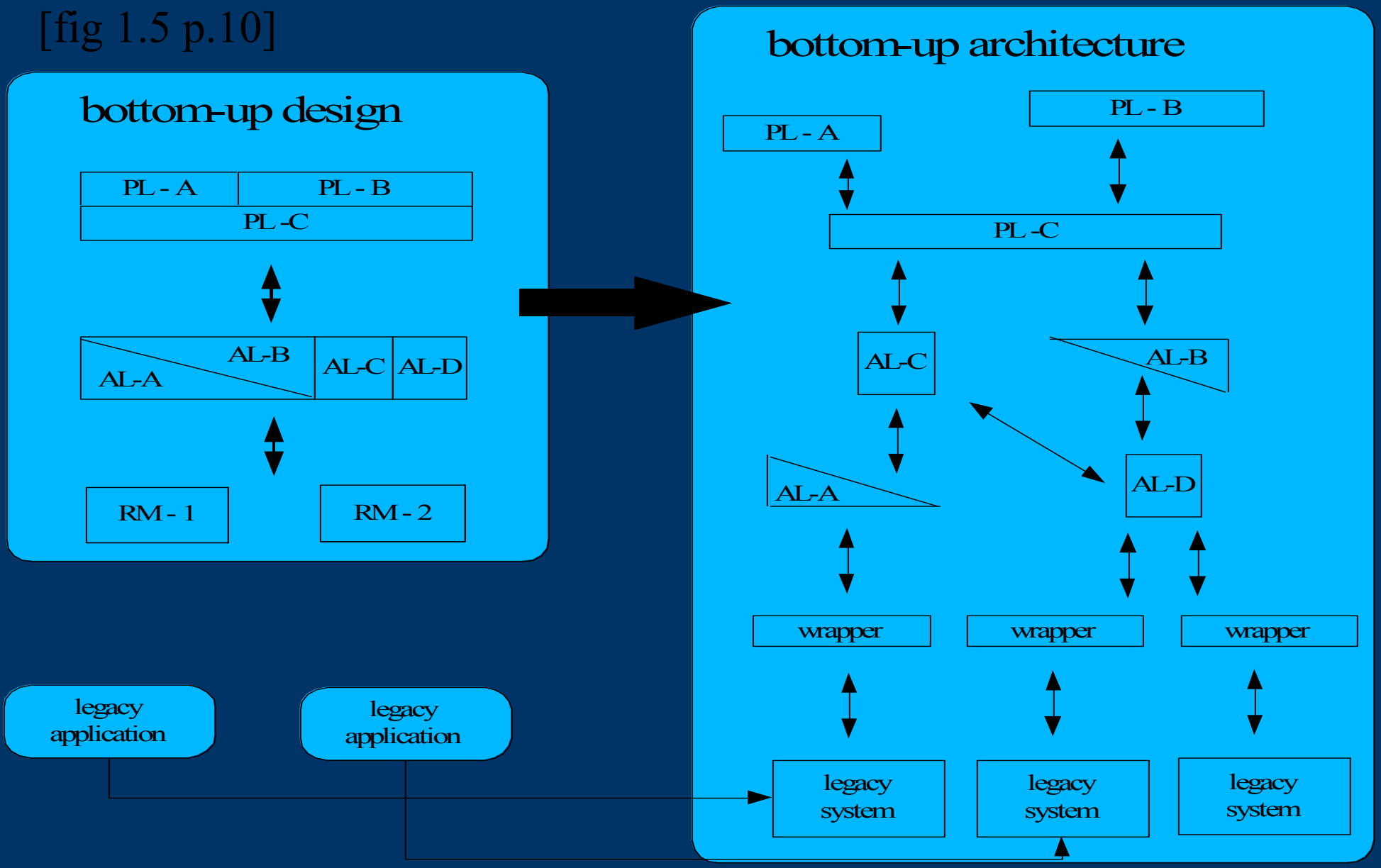


bottom-up design

- out of necessity rather than choice
- need to integrate legacy systems and/or applications
- results in loosely coupled systems
  - independent and
  - standolone components
- most distributed IS are result of a bottom-up design
- Web services can make those designs more efficient, cost-effective and simplier to design

bottom-up design

[fig 1.5 p.10]



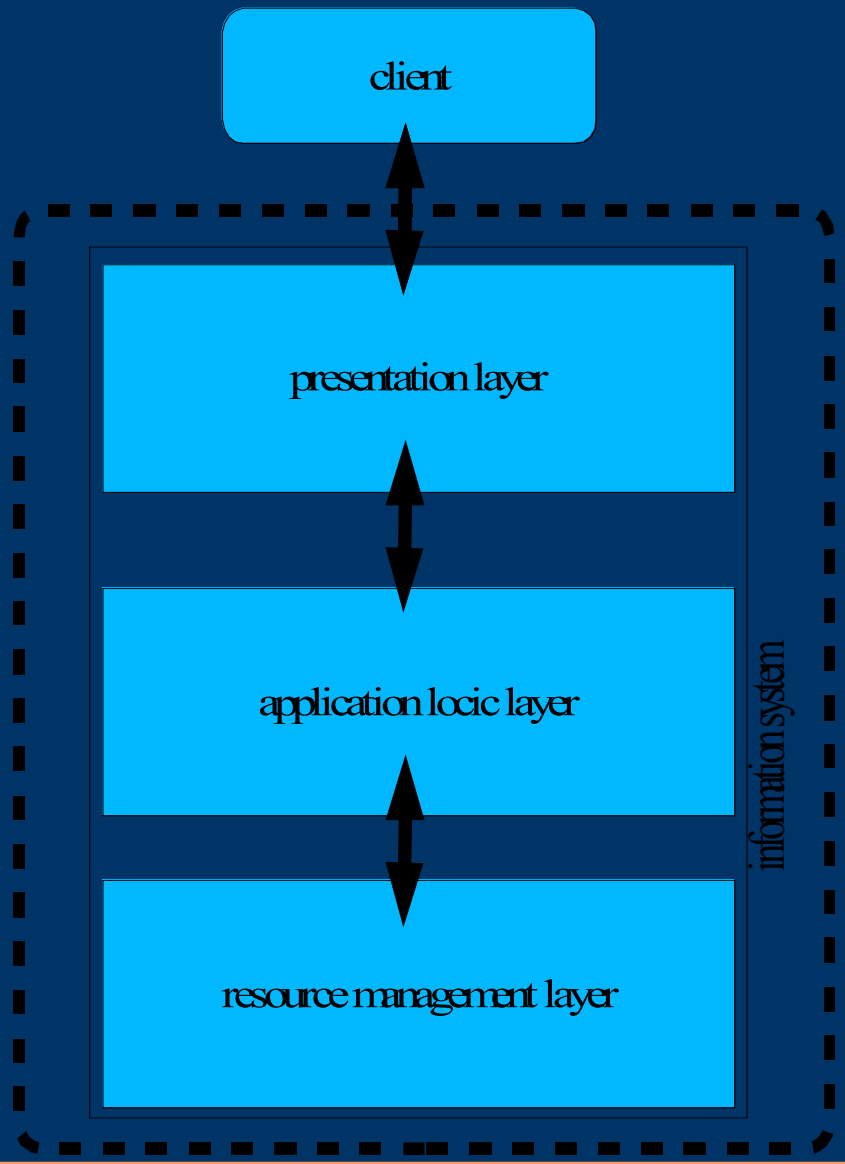
Architecture of an Information System - 4 types:

- *1 – tier*
- *2 – tier*
- *3 – tier*
- *n – tier*

1 – tier Architectures

- were used decades ago..
- monolithic Information Systems
- presentation, application logic, and resource management were merged into a single tier
- many of these 'old' Systems are still in use!

Design of 1 – tier Architecture



1 – tier Architecture

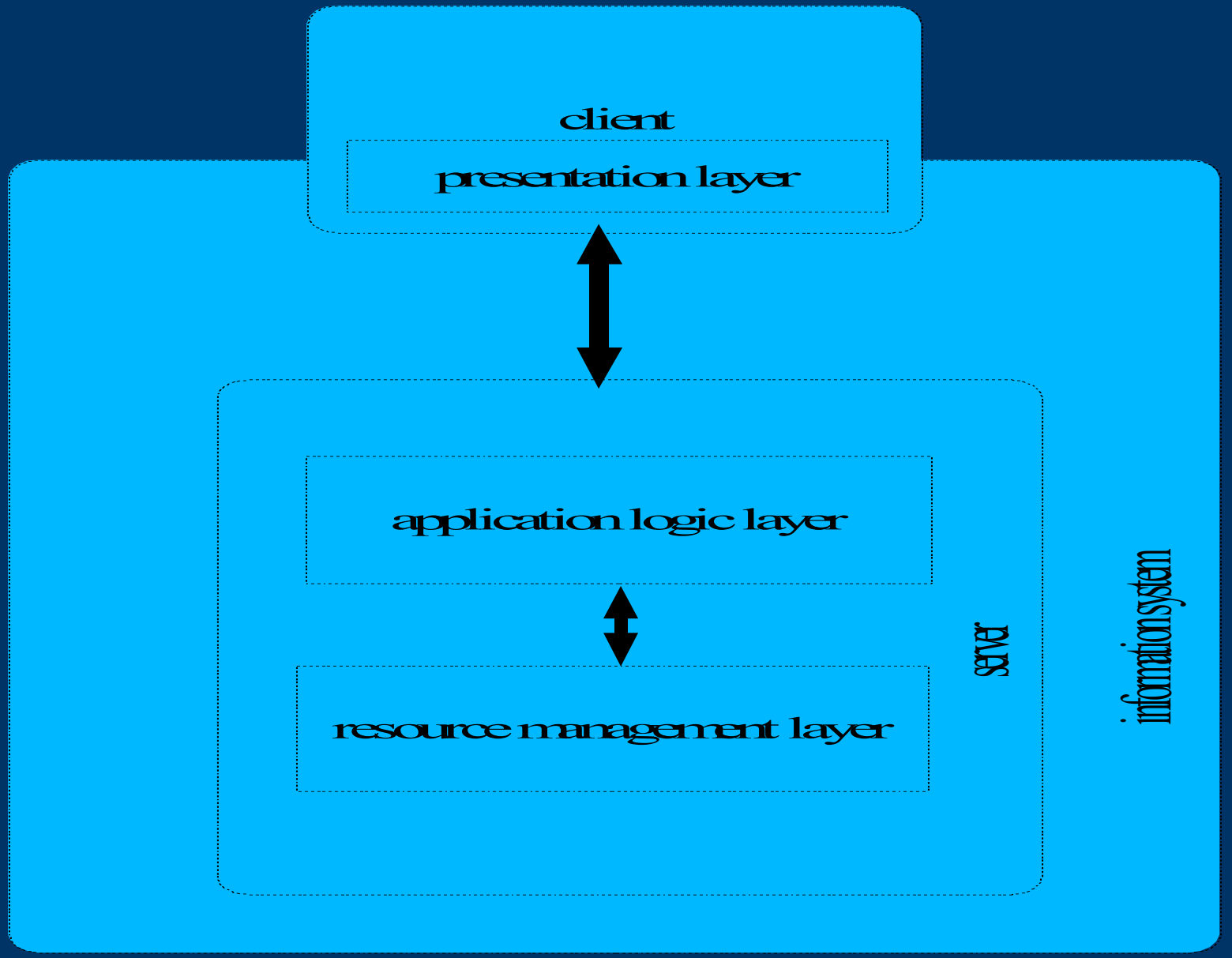
advantages:

- easy to optimize performance
- no context switching
- no compatibility issues
- no client developement, maintainance and deployment cost

disadvantages:

- monolithic pieces of code (high maintainance)
- hard to modify
- lack of qualified programmers for these systems

2 - tier Architectures



2 - tier Architectures

- separation of presentation layer from other 2 layers (app + resource)
- became popular as 'server/client' systems
- thin clients/fat clients
- RPC (Remote Procedure Call)
- API (Application Program Interface)
- need for standardization

advantages & disadvantages

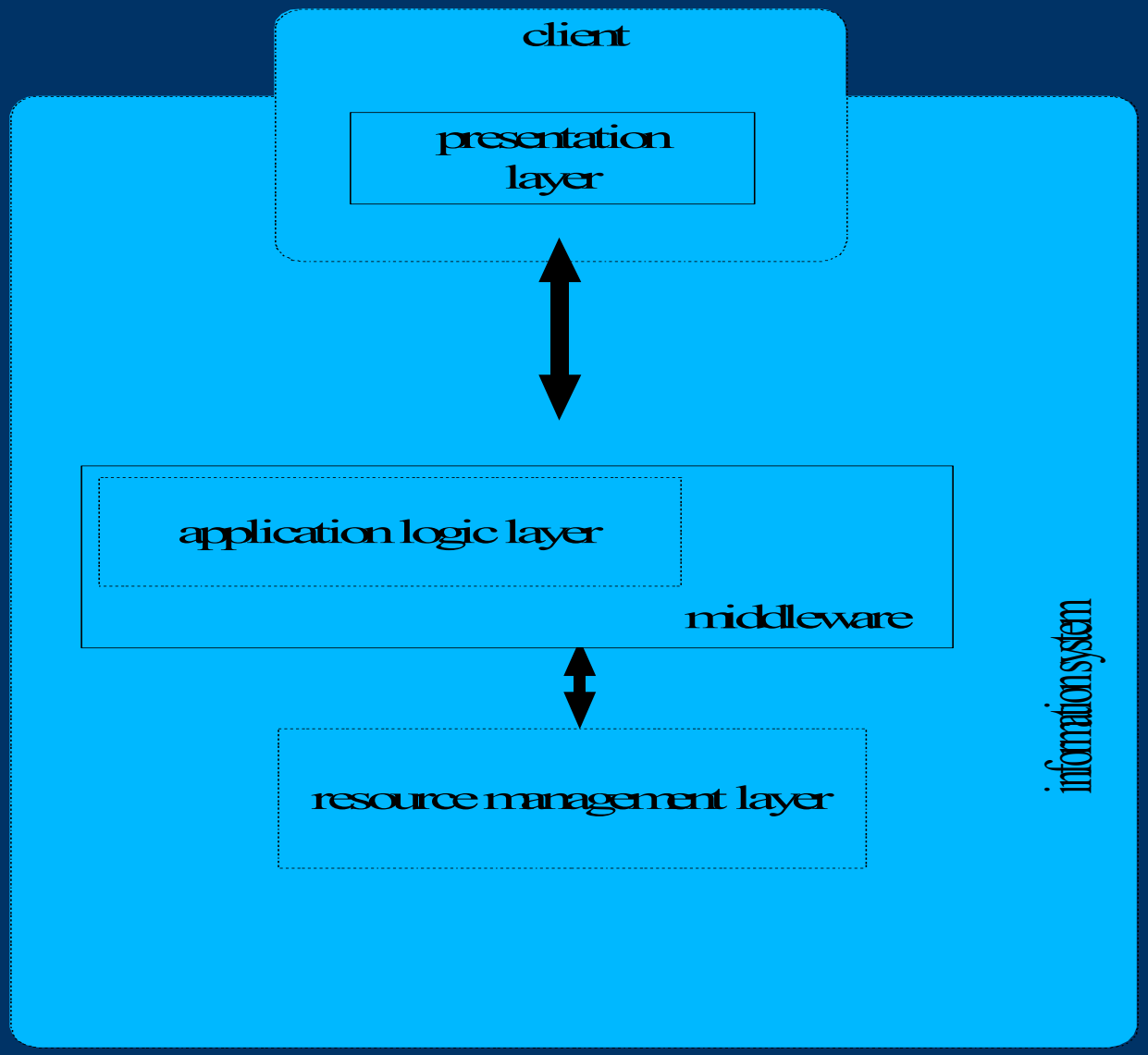
advantages

- portability
- no need for context switches or calls between component for key operations

disadvantages

- limited scalability
- legacy problems (blown up clients)

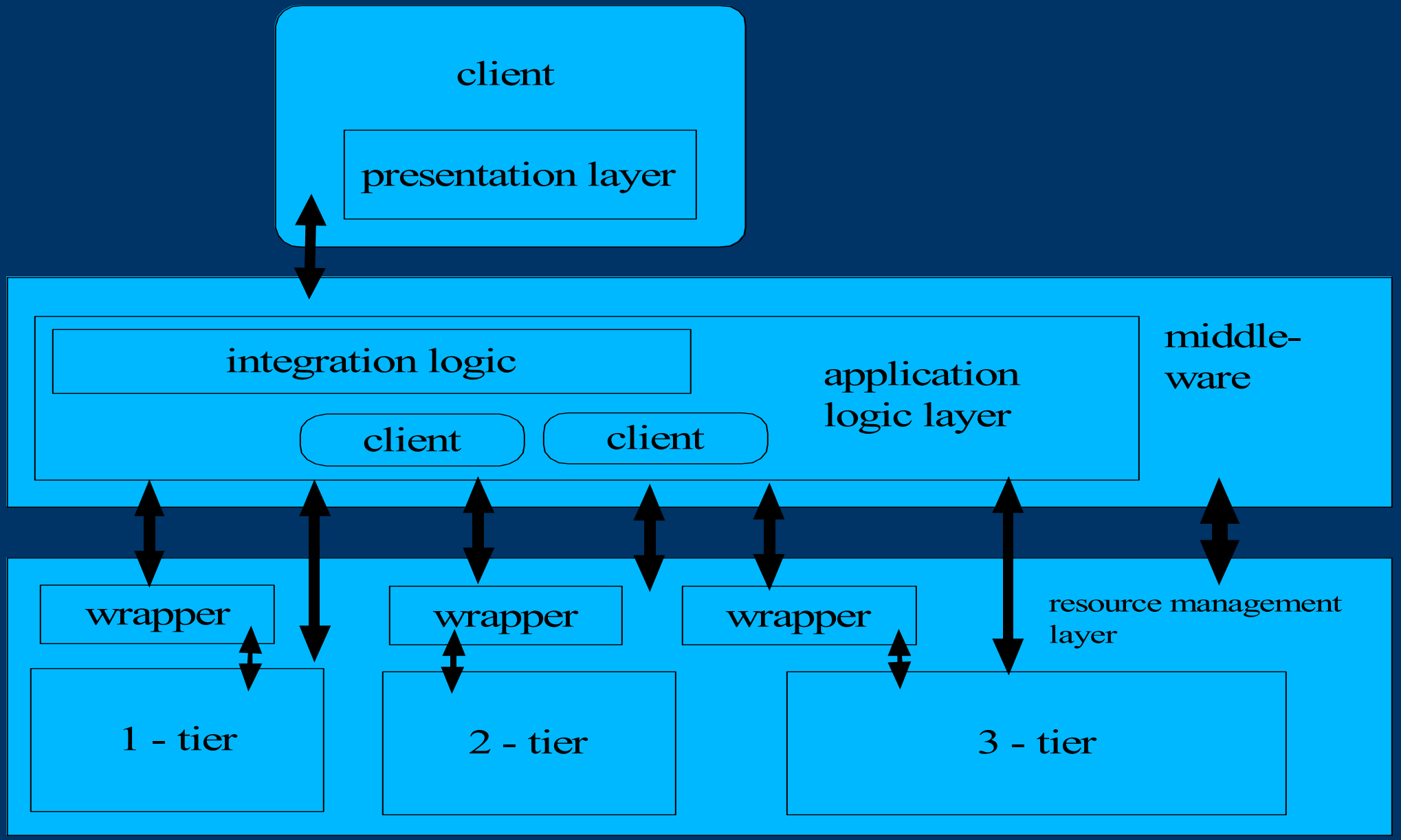
3 - tier Architectures



3 - tier Architectures

- can be achieved by separating RM (resource management) from application logic layer
- additional middleware layer between client and server
  - integration logic
  - application logic
- lead to the introduction of clear RM layer interfaces
- good at dealing with intgration of different resources

3 - tier


client
presentation layer
integration logic
client
client
application
logic layer
middle-
ware
wrapper
wrapper
wrapper
resource management
layer
1 - tier
2 - tier
3 - tier

advantages & disadvantages

advantages

- scalability by running each layer on a different server
- scalability by distributing AL (application logic layer) across many nodes
- additional tier for integration logic
- flexibility

disadvantages

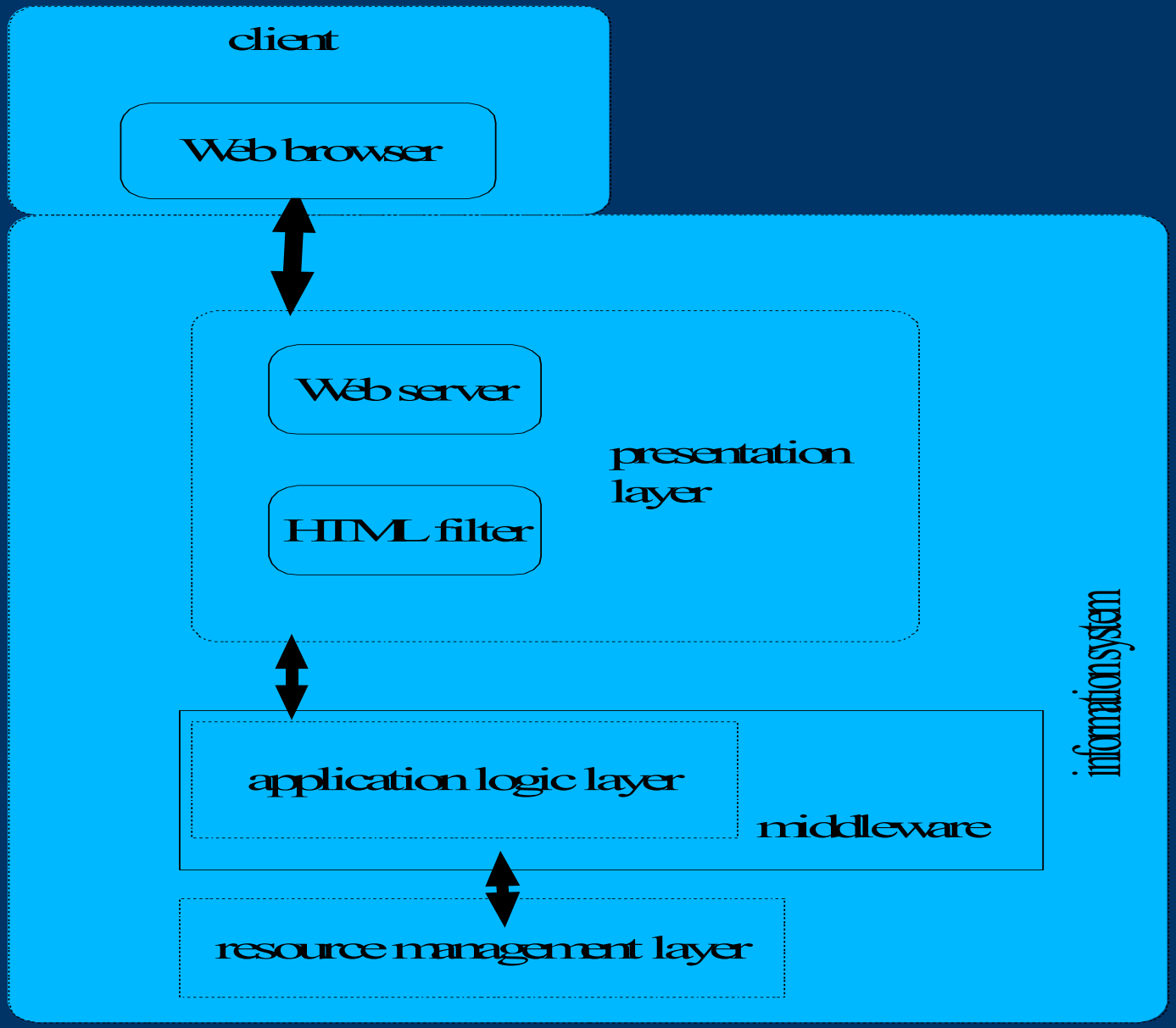
- performance loss if distributed over the internet
- problem when integrating different 3 – tier systems

n -tier

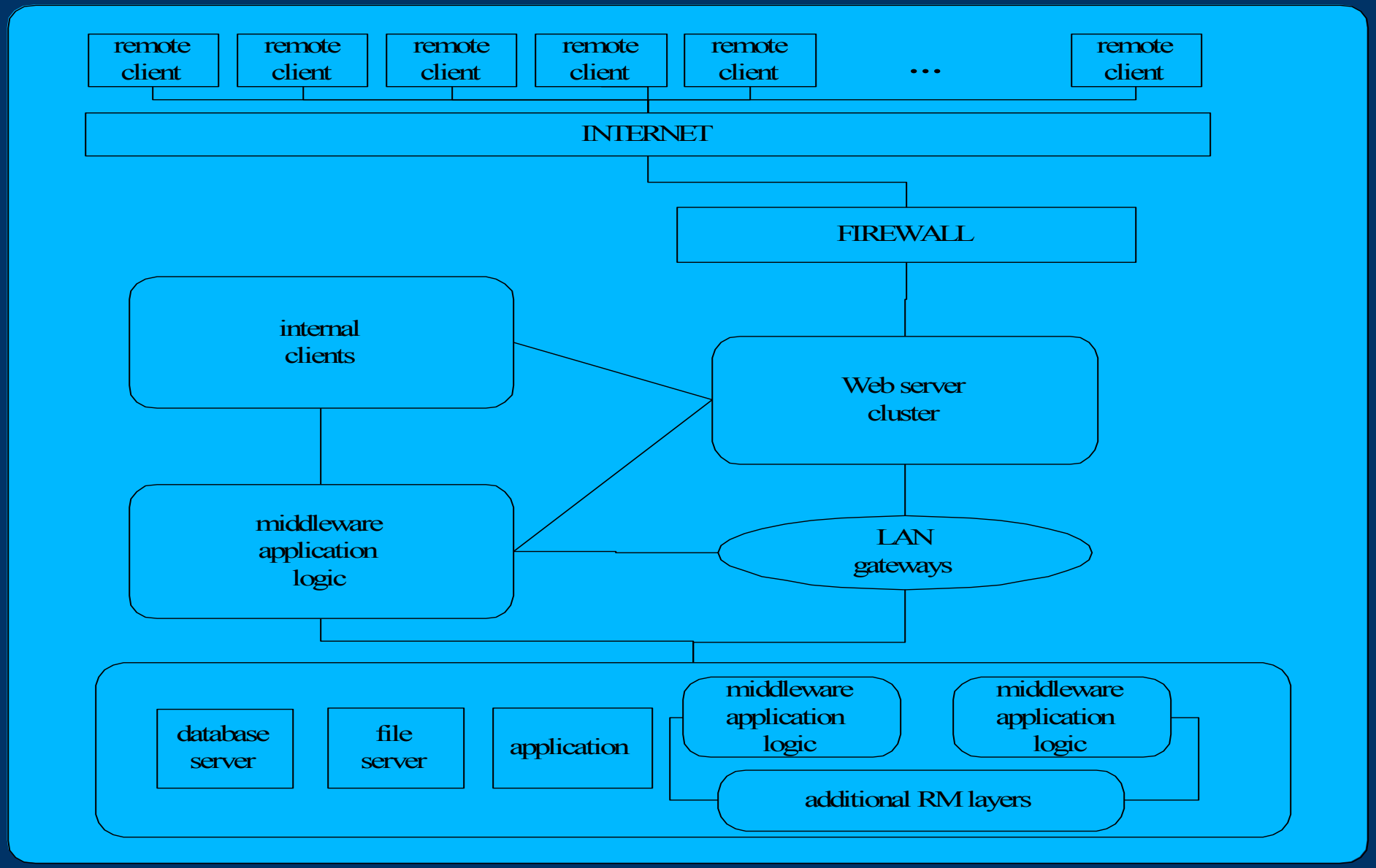
2 cases of n – tier

- systems linked with added connectivity through the internet
- resource layer is a full fledged 2 - or 3 - tier system

n -tier


client
Web browser
Web server
HTML filter
presentation layer
application logic layer
middleware
resource management layer
information system

n - tier


remote client
remote client
remote client
remote client
remote client
...
remote client
INTERNET
FIREWALL
internal clients
Web server cluster
middleware application logic
LAN gateways
database server
file server
application
middleware application logic
middleware application logic
additional RM layers

advantages & disadvantages

advantages

- better scalability
- higher fault tolerance
- higher throughput for less cost

disadvantages

- too much middleware involved
- redundant functionality
- difficulty and cost of developement

gains and losses

with growing number of tiers one gains:

- flexibility
- functionality
- possibilities for distribution

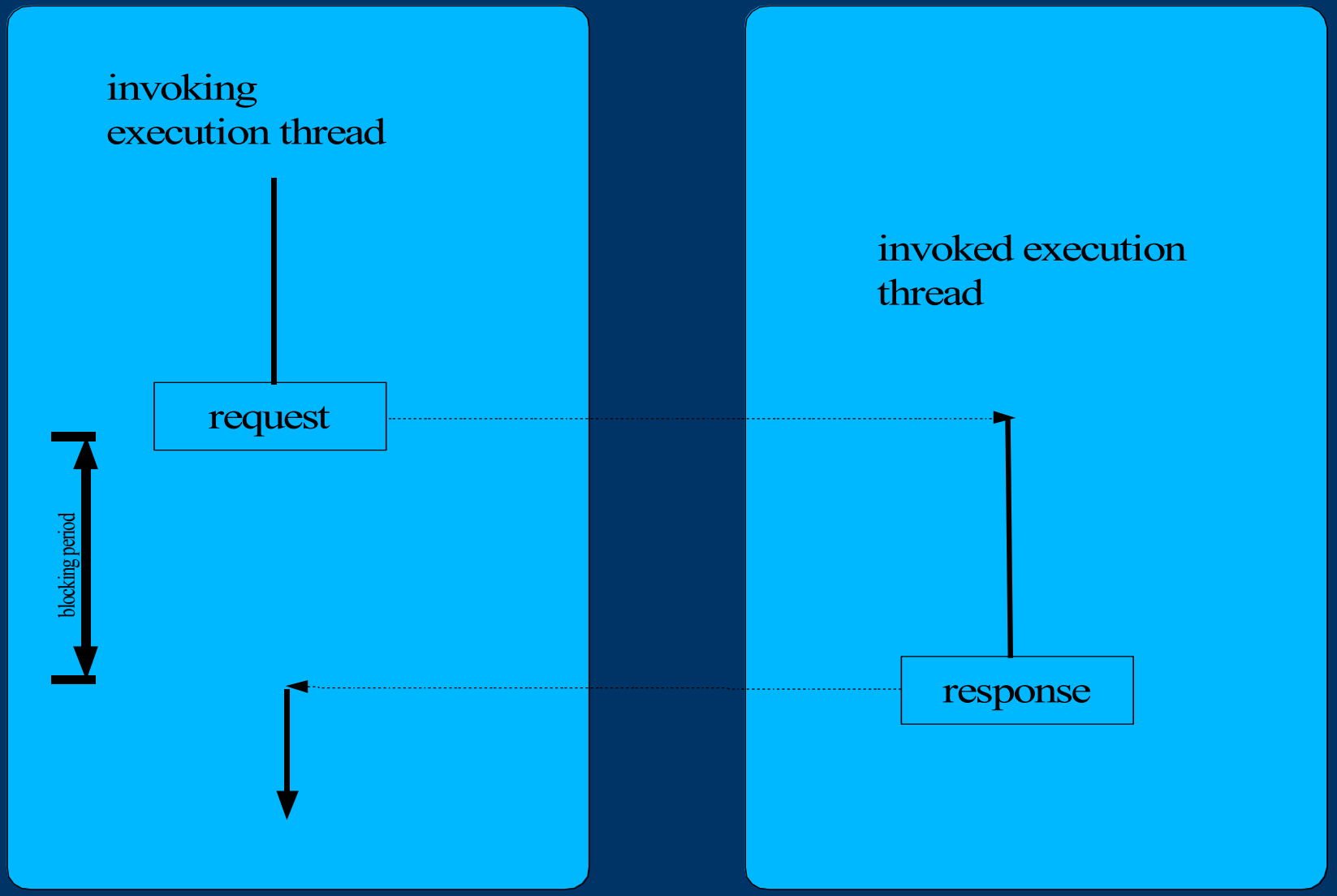
but:

- each tier increases communication costs
- complexity rises
- higher complexity of management and tuning

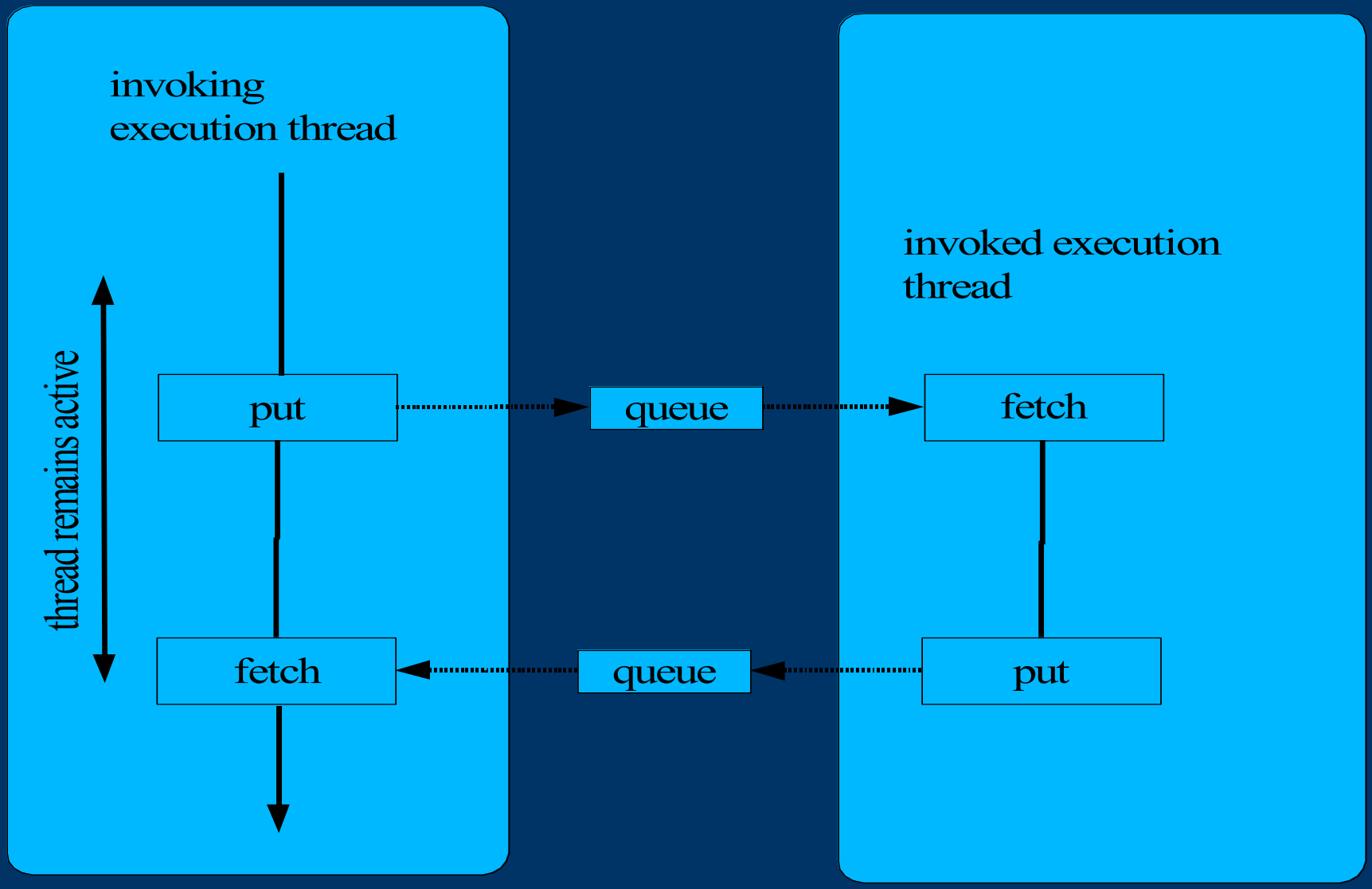
communication in an IS between distributed layers/tiers

- synchronous interactions
- asynchronous interactions

synchronous interactions (blocking)



asynchronous interactions (non blocking)



scaling multi tier systems

6 steps

- understand the application environment
- categorize your workload
- determine the components most impacted
- select scaling techniques to apply
- apply the techniques
- reevaluate
- .. and hope its better :)

what do scaling techniques improve?

scaling technique	increase capacity/ speed	improve efficiency	shift/reduce load
use faster machine	x		
create machine cluster	x		
use a special machine	x	x	
segment the work load		x	x
batch request		x	
aggregate user data		x	
manage connections		x	
cache data & requests		x	x

use faster machine

- increases the ability to do more work in a unit of time by processing tasks more rapidly
- applies to almost all parts of the system (from edge servers to database server)

create cluster of machines

- services more client requests. improves response time through parallelism
- applies to Web presentation server, Web application server, directory and security servers

use special machines

- improves efficiency of a component by using a special purpose machine, which is optimized for a specific function
- applies to egde server, Web presentation server, directory and security servers, the network and the Internet firewall

segment the workload

- splits up workload into managable chunks to obtain more predictible response times
- applies to Web representation server, Web application server, the data server and the network

batch requests

- reduces number of requests by defining new ones that combine multiple requests
- applies to Web presentation server, Web application server, directory and security servers, existing business applications and database

aggregate user data

- allows rapid access to large customer data controlled by existing system applications by aggregating distributed customer data into a costomer information service
- applies to the Web presentation server, Web application server and the network

manage connections

- minimizes number of connections and eliminates overhead of setting up connections by sharing a pool of preestablished connections between the layers
- applies to Web presentation server, Web aplication server and the database.

cache

- improves performance and scalability and response time by buffering data flows and reducing consumption of resources
- applies to the edge server, Web presentation server, Web application server, network, existing business applications and the database.

summary

- layers of an IS
- designs of distributed IS
- evolution of architectures and concepts
- scaling techniques