Foundations of Software Engineering

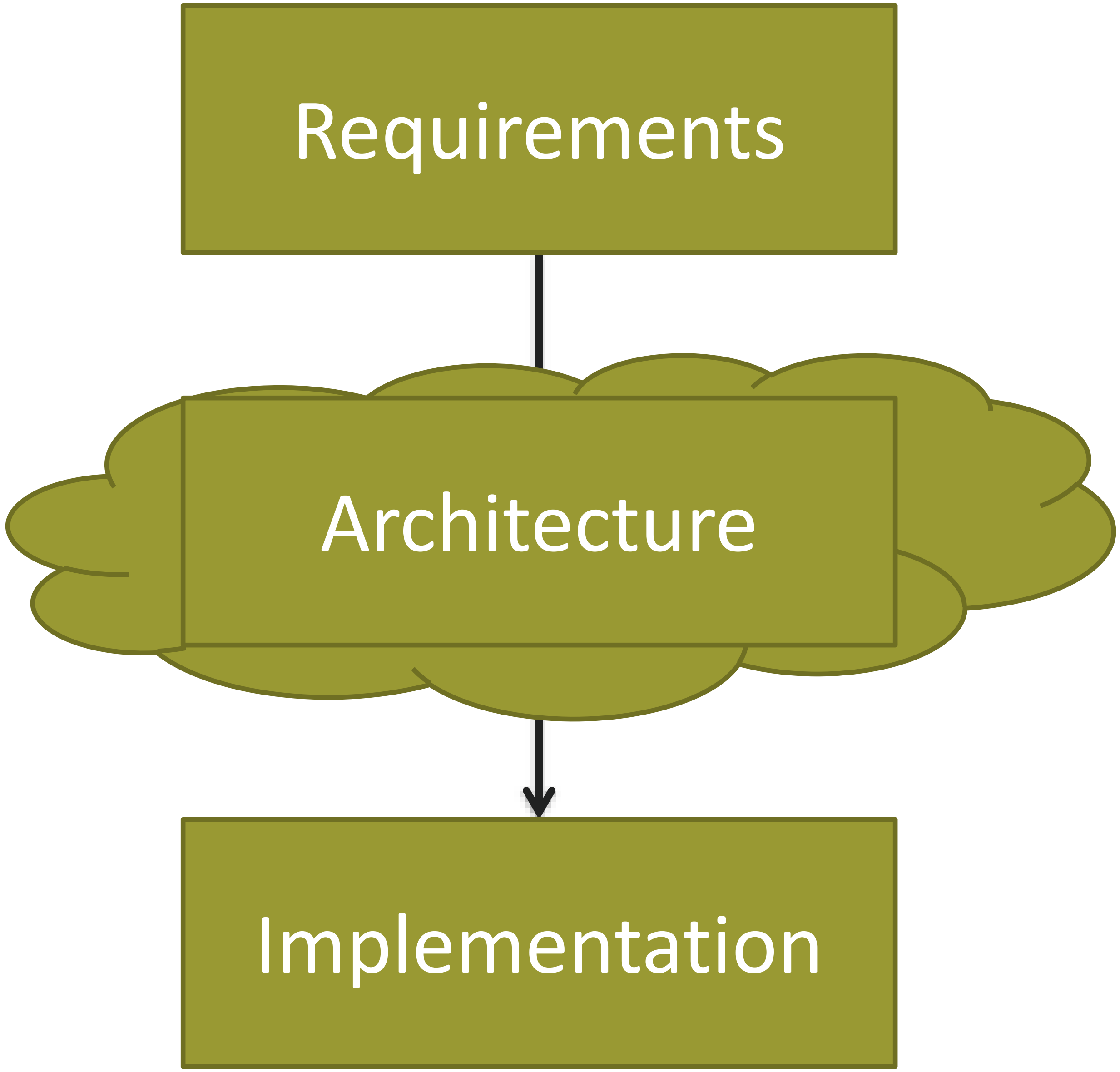
Lecture 9: Architecture Documentation, Patterns, and Tactics

Christian Kaestner

institute for SOFTWARE RESEARCH

Learning Goals

- Use notation and views to describe the architecture suitable to the purpose
- Document architectures clearly, without ambiguity
- Understand the benefits and challenges of traceability.
- Understand key parts of architectural process
- Use architectural styles and tactics for design decisions
- Make justified architectural decisions for new systems and within existing systems


Requirements
Architecture
Implementation

Architecture vs Object-Level Design

214 Review: Design

- Design process (analysis, design, implementation)
- Design goals (cohesion, coupling, information hiding, design for reuse, …)
- Design patterns (what they are, for what they are useful, how they are described)
- Frameworks and libraries (reuse strategies)

Levels of Abstraction

- Requirements
  – high-level "what" needs to be done
- Architecture (High-level design)
  – high-level "how", mid-level "what"
- OO-Design (Low-level design, e.g. design patterns)
  – mid-level "how", low-level "what"
- Code
  – low-level "how"

institute for SOFTWARE RESEARCH

Design vs. Architecture

Design Questions

- How do I add a menu item in Eclipse?
- How can I make it easy to add menu items in Eclipse?
- What lock protects this data?
- How does Google rank pages?
- What encoder should I use for secure communication?
- What is the interface between objects?

Architectural Questions

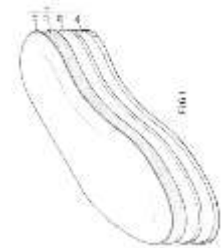
- How do I extend Eclipse with a plugin?
- What threads exist and how do they coordinate?
- How does Google scale to billions of hits per day?
- Where should I put my firewalls?
- What is the interface between subsystems?

Architecture Documentation & Views

Architecture Disentangled

Architecture as
structures and relations
(the actual system)

Architecture as
documentation
(representations of the system)

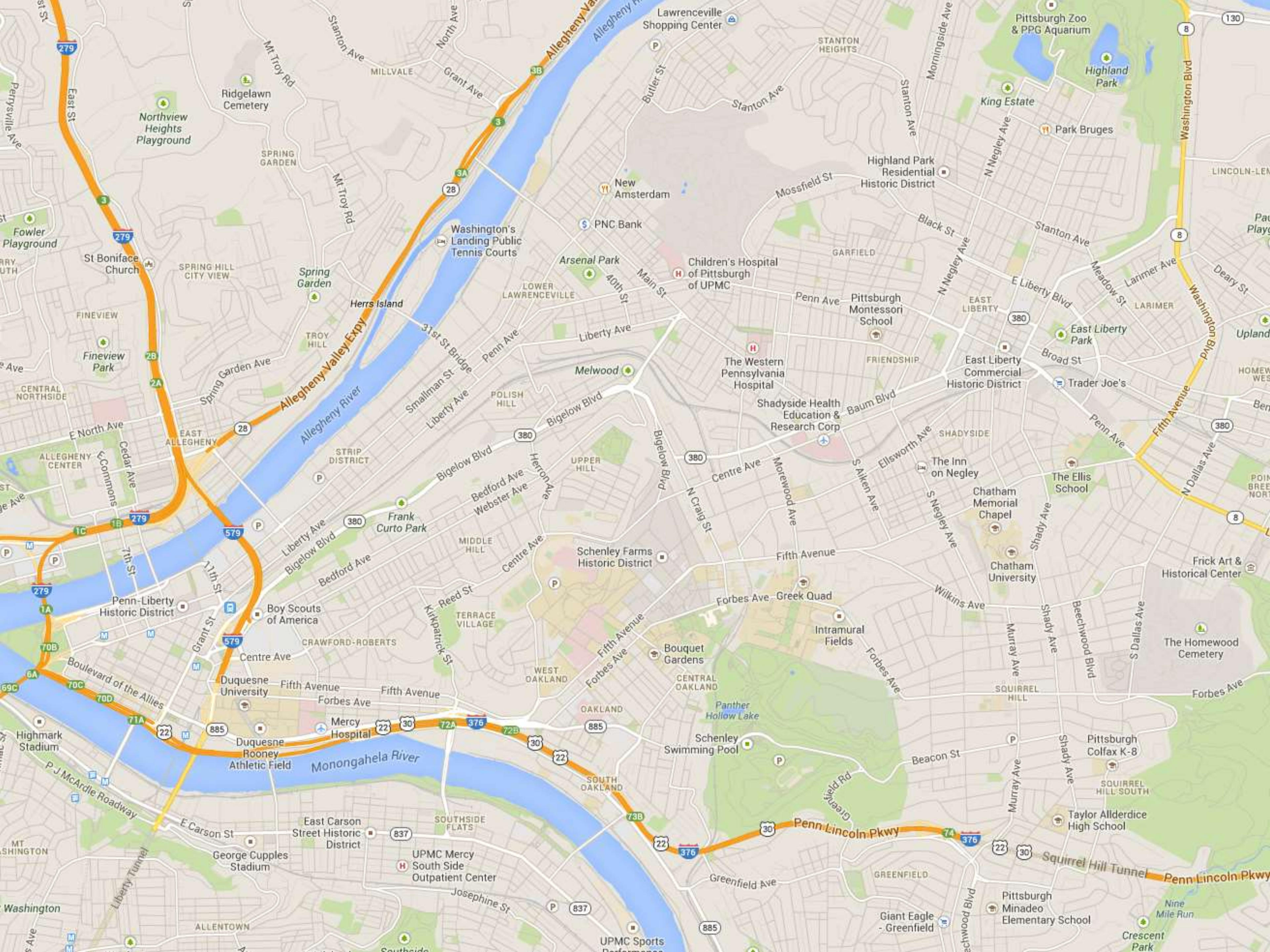


Architecture as (design) process
(activities around the other two)

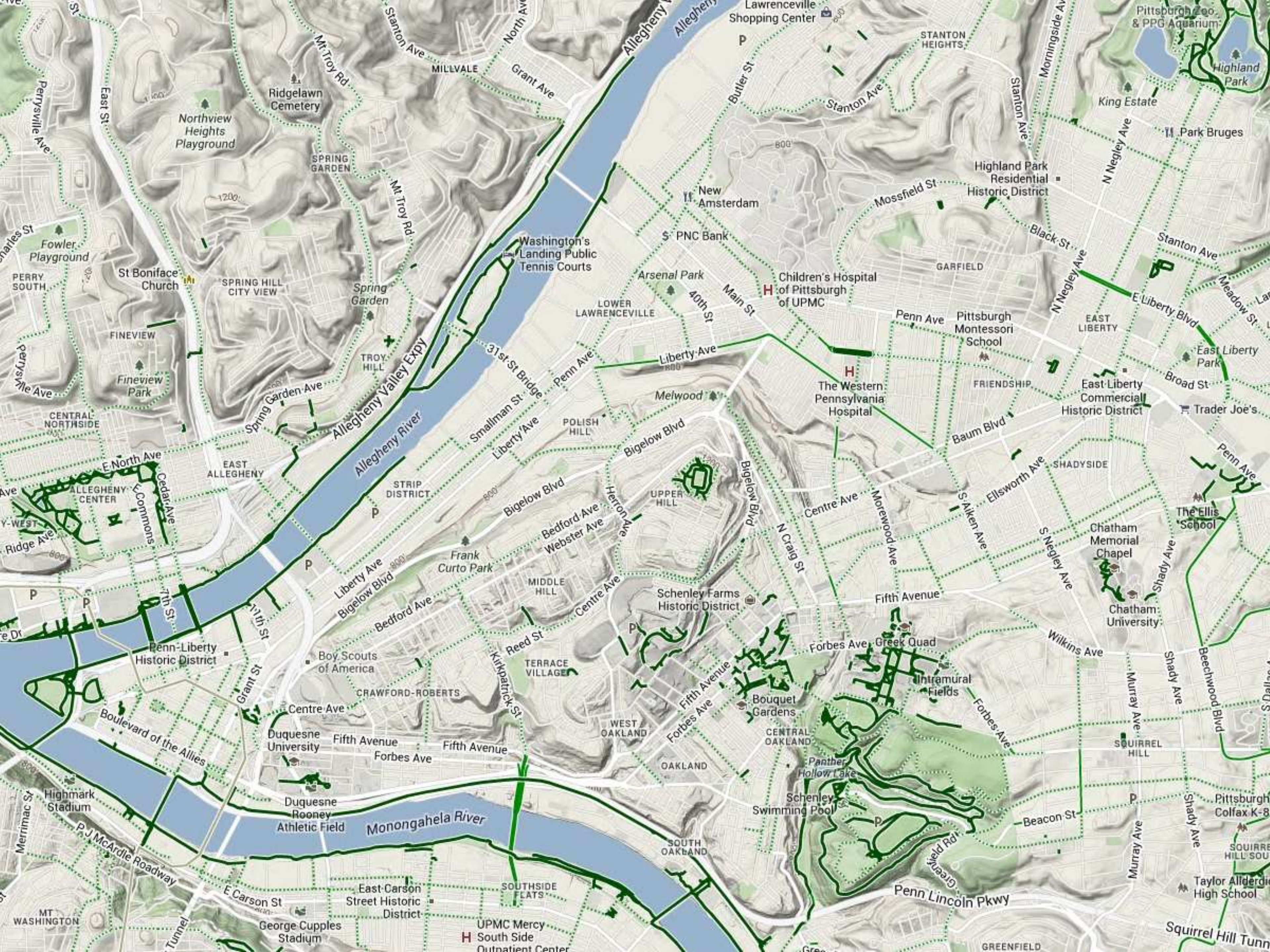
institute for
SOFTWARE
RESEARCH

Why Document Architecture?

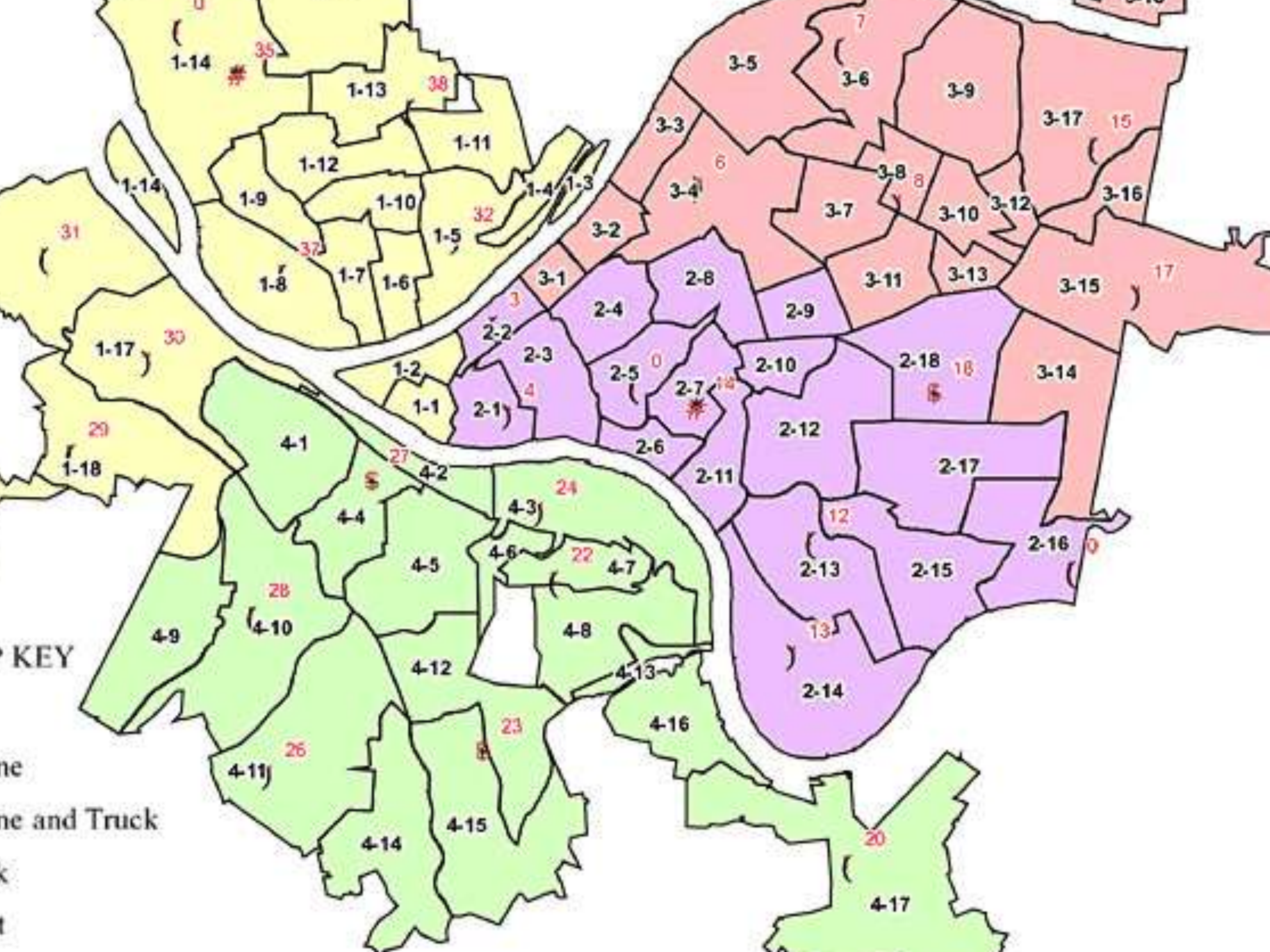
- Blueprint for the system
  - Artifact for early analysis
  - Primary carrier of quality attributes
  - Key to post-deployment maintenance and enhancement
- Documentation speaks for the architect, today and 20 years from today
  - As long as the system is built, maintained, and evolved according to its documented architecture
- Support traceability.



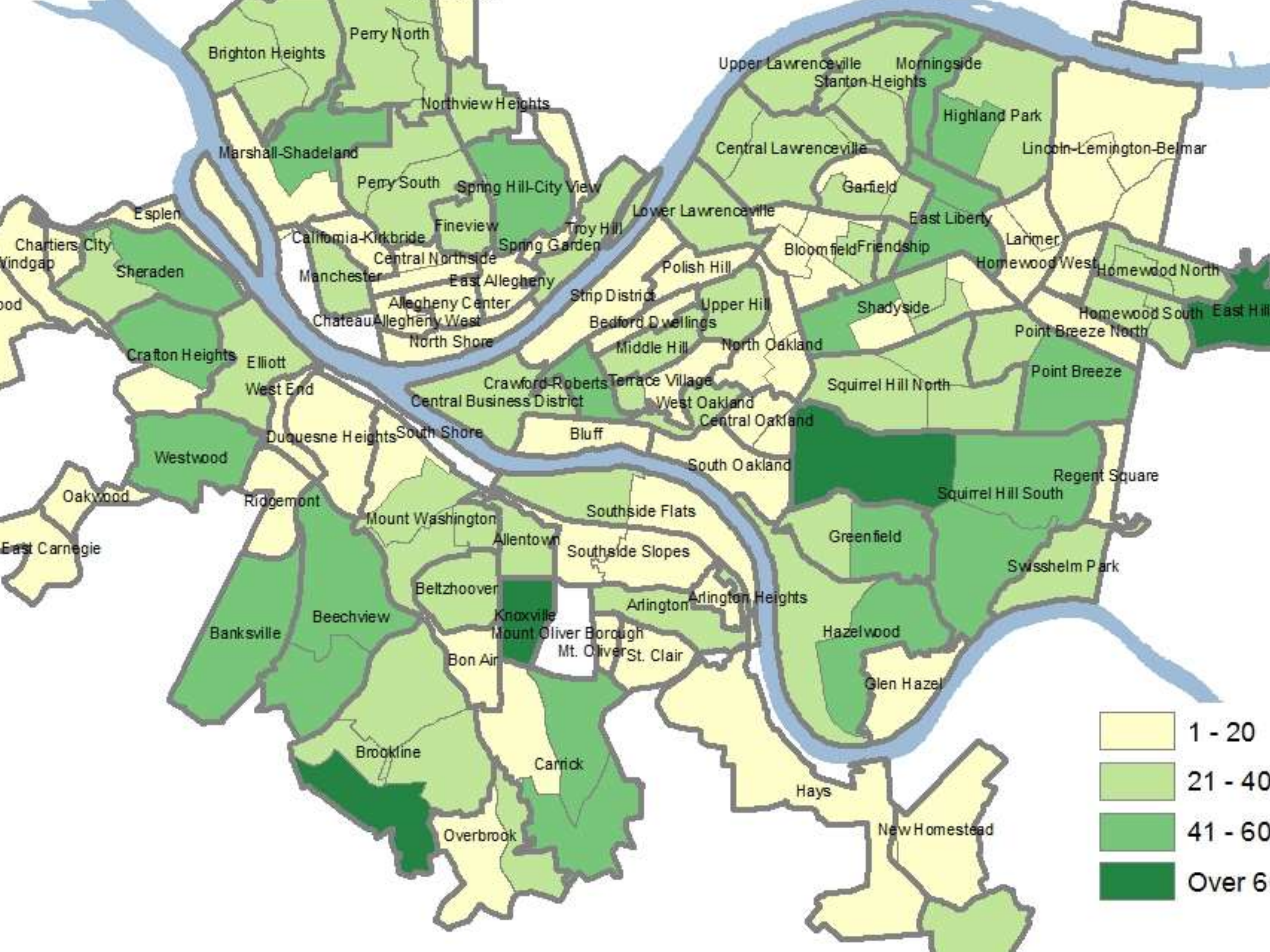
Lawrenceville Shopping Center
Pittsburgh Zoo & PPG Aquarium
Highland Park
King Estate
Park Bruges
Stanton Heights
Millvale
Ridgelawn Cemetery
Northview Heights Playground
Spring Garden
Fowler Playground
St Boniface Church
Spring Hill City View
Spring Garden
Herrs Island
Fineview
Fineview Park
Troy Hill
Central Northside
East Allegheny
Allegheny Center
Washington's Landing Public Tennis Courts
New Amsterdam
PNC Bank
Arsenal Park
Lower Lawrenceville
Children's Hospital of Pittsburgh of UPMC
Highland Park Residential Historic District
Garfield
East Liberty
Pittsburgh Montessori School
East Liberty Park
Larimer
Upland
Lincoln-Lem
Friendship
The Western Pennsylvania Hospital
East Liberty Commercial Historic District
Trader Joe's
Melwood
Polish Hill
Shadyside Health Education & Research Corp
Shadyside
The Inn on Negley
The Ellis School
Chatham Memorial Chapel
Chatham University
Strip District
Upper Hill
Frank Curto Park
Middle Hill
Schenley Farms Historic District
Frick Art & Historical Center
Penn-Liberty Historic District
Boy Scouts of America
Terrace Village
Crawford-Roberts
Greek Quad
Intramural Fields
The Homewood Cemetery
Duquesne University
West Oakland
Bouquet Gardens
Central Oakland
Oakland
Panther Hollow Lake
Squirrel Hill
Highmark Stadium
Mercy Hospital
Duquesne Rooney Athletic Field
Monongahela River
Allegheny River
Schenley Swimming Pool
Pittsburgh Colfax K-8
Squirrel Hill South
South Oakland
Taylor Allderdice High School
Southside Flats
E Carson Street Historic District
UPMC Mercy South Side Outpatient Center
George Cupples Stadium
Mt Washington
Allentown
Greenfield
Pittsburgh Minadeo Elementary School
Giant Eagle - Greenfield
Nine Mile Run
Crescent Park
UPMC Sports Performance
Allegheny Valley Expy
Penn Lincoln Pkwy
Squirrel Hill Tunnel
Liberty Tunnel
Boulevard of the Allies
Fifth Avenue
Forbes Ave
Centre Ave
Bigelow Blvd
Liberty Ave
Penn Ave
Baum Blvd
E Liberty Blvd
Stanton Ave
Washington Blvd
Greenfield Ave
Beacon St
Wilkins Ave
Murray Ave
Shady Ave
Beechwood Blvd
S Dallas Ave
N Negley Ave
S Negley Ave
N Craig St
Morewood Ave
S Aiken Ave
Ellsworth Ave
Herron Ave
Bedford Ave
Webster Ave
Kirkpatrick St
Reed St
Smallman St
31st St Bridge
40th St
Main St
Butler St
Mossfield St
Black St
Broad St
Larimer Ave
Meadow St
Morningside Ave
Stanton Ave
Grant Ave
Stanton Ave
North Ave
Mt Troy Rd
Spring Garden Ave
E North Ave
Cedar Ave
E Commons
East St
Perrysville Ave
7th St
11th St
Grant St
P J McArdle Roadway
E Carson St
Josephine St
Greenfield Rd
Deary St
Fifth Avenue
Penn Ave
N Dallas Ave
279
579
376
380
28
885
837
22
30
8
130



Lawrenceville Shopping Center
Stanton Heights
Pittsburgh Zoo & PPG Aquarium
Highland Park
Millvale
Stanton Ave
North Ave
Mt Troy Rd
Grant Ave
Allegheny Valley Expy
Allegheny River
Butler St
Morningside Ave
King Estate
Park Bruges
Ridgelawn Cemetery
Northview Heights Playground
East St
Perrysville Ave
Spring Garden
New Amsterdam
Mossfield St
Highland Park Residential Historic District
N Negley Ave
Washington's Landing Public Tennis Courts
PNC Bank
Black St
Stanton Ave
Fowler Playground
Perry South
St Boniface Church
Spring Hill City View
Arsenal Park
Children's Hospital of Pittsburgh of UPMC
Garfield
Lower Lawrenceville
40th St
Main St
Penn Ave
Pittsburgh Montessori School
East Liberty
E Liberty Blvd
Meadow St
Fineview
Troy Hill
31st St Bridge
Liberty Ave
East Liberty Park
Fineview Park
Spring Garden Ave
Smallman St
The Western Pennsylvania Hospital
Friendship
East Liberty Commercial Historic District
Broad St
Trader Joe's
Melwood
Central Northside
Polish Hill
Baum Blvd
Bigelow Blvd
Shadyside
E North Ave
East Allegheny
Allegheny Center
Cedar Ave
E Commons
Strip District
Upper Hill
Centre Ave
Ellsworth Ave
S Aiken Ave
Morewood Ave
N Craig St
The Ellis School
Ridge Ave
Herron Ave
Bedford Ave
Webster Ave
Frank Curto Park
Chatham Memorial Chapel
S Negley Ave
Shady Ave
Middle Hill
Schenley Farms Historic District
Fifth Avenue
Chatham University
7th St
11th St
Reed St
Wilkins Ave
Penn-Liberty Historic District
Boy Scouts of America
Terrace Village
Forbes Ave
Greek Quad
Intramural Fields
Murray Ave
Beechwood Blvd
Grant St
Crawford-Roberts
Kirkpatrick St
Bouquet Gardens
Centre Ave
West Oakland
Central Oakland
Boulevard of the Allies
Duquesne University
Fifth Avenue
Forbes Ave
Squirrel Hill
Oakland
Panther Hollow Lake
Highmark Stadium
Duquesne Rooney Athletic Field
Monongahela River
Schenley Swimming Pool
Beacon St
Pittsburgh Colfax K-8
Merrimac St
P J McArdle Roadway
Greenfield Rd
South Oakland
East Carson Street Historic District
Southside Flats
E Carson St
Penn Lincoln Pkwy
Taylor Allderdice High School
Mt Washington
George Cupples Stadium
UPMC Mercy South Side Outpatient Center
Greenfield
Squirrel Hill Tunnel


KEY
ne
ne and Truck
k
t
1-1
1-2
1-3
1-4
1-5
1-6
1-7
1-8
1-9
1-10
1-11
1-12
1-13
1-14
1-17
1-18
2-1
2-2
2-3
2-4
2-5
2-6
2-7
2-8
2-9
2-10
2-11
2-12
2-13
2-14
2-15
2-16
2-17
2-18
3-1
3-2
3-3
3-4
3-5
3-6
3-7
3-8
3-9
3-10
3-11
3-12
3-13
3-14
3-15
3-16
3-17
4-1
4-2
4-3
4-4
4-5
4-6
4-7
4-8
4-9
4-10
4-11
4-12
4-13
4-14
4-15
4-16
4-17
3
4
6
7
8
12
13
14
15
17
18
20
22
23
24
26
27
28
29
30
31
32
35
37
38


North Hills
West Deer
Harris
Tarentum
Franklin Park
McCandless
Hampton
Sewickley
Ross
Springdale
Fox Chapel
Shaler
Oakmont
Coraopolis
Emsworth
on
Plum
Etna
Sharpsburg
Bellevue
Ohio River
Millvale
Allegheny River
Penn Hills
Perry South
Sheraden
Robinson
Pittsburgh
Crafton
Churchill
Monroeville
tte
Allentown
Turtle Creek
Carnegie
Knoxville
Monongahela River
Braddock
Trafford
East Pittsburgh
Scott
Mount Lebanon
Duquesne
South Fayette
West Mifflin
Whitehall
McKeesport
Upper St. Clair
White Oak
Pleasant Hills
Bethel Park
Liberty
Jefferson Hills



Perry North
Brighton Heights
Upper Lawrenceville
Morningside
Stanton Heights
Northview Heights
Highland Park
Marshall-Shadeland
Central Lawrenceville
Lincoln-Lemington-Belmar
Perry South
Spring Hill-City View
Garfield
Esplen
Lower Lawrenceville
East Liberty
Fineview
Troy Hill
California-Kirkbride
Chartiers City
Windgap
Spring Garden
Larimer
Central Northside
Bloomfield
Friendship
Sheraden
Polish Hill
Homewood West
Homewood North
Manchester
East Allegheny
Strip District
Allegheny Center
Upper Hill
Shadyside
Chateau
Allegheny West
Bedford Dwellings
Homewood South
East Hill
Point Breeze North
North Shore
North Oakland
Middle Hill
Crafton Heights
Elliott
Point Breeze
Crawford-Roberts
Terrace Village
Squirrel Hill North
West End
Central Business District
West Oakland
Central Oakland
Duquesne Heights
South Shore
Bluff
Westwood
South Oakland
Regent Square
Squirrel Hill South
Oakwood
Ridgemont
Southside Flats
Mount Washington
Greenfield
Allentown
East Carnegie
Southside Slopes
Swisshelm Park
Beltzhoover
Arlington Heights
Arlington
Knoxville
Beechview
Banksville
Mount Oliver Borough
Hazelwood
Mt. Oliver
St. Clair
Bon Air
Glen Hazel
Brookline
Carrick
Hays
New Homestead
Overbrook
1 - 20
21 - 40
41 - 60
Over 6

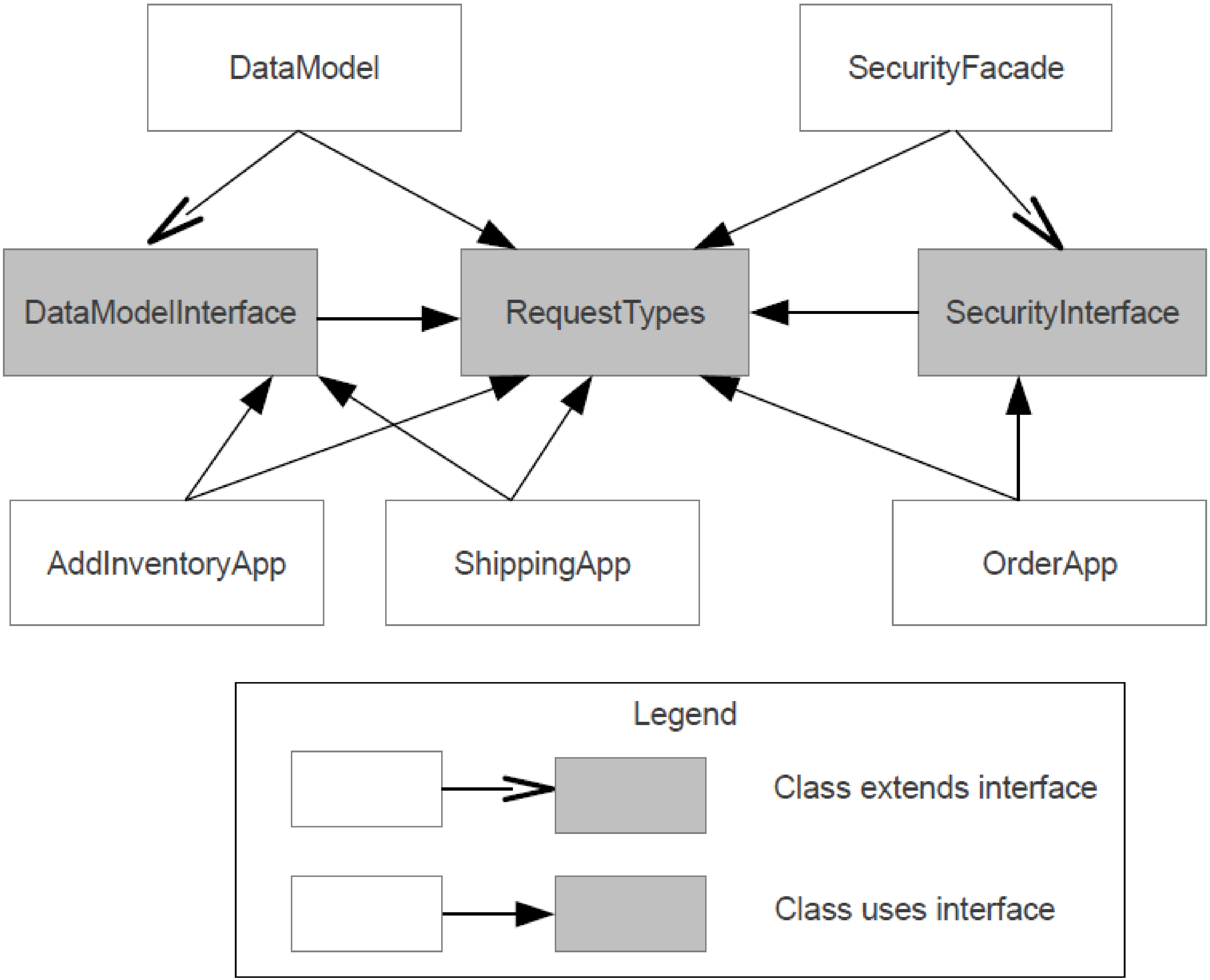
Allegheny River
Highland Park
Allegheny Cemetery
Butler St
Stanton Ave
N Negley Ave
N Euclid Ave
Washington Blvd
Lincoln Ave
Larimer Ave
Penn Ave
Liberty Ave
Smallman St
31st St
E Ohio St
Spring Garden Ave
Pittview Ave
East St
N Charles St
Brighton Rd
1000 ft
River Ave
Spring Way
Bigelow Blvd
Baum Blvd
E Busway
Centre Ave
Ellsworth Ave
5th Ave
Hamilton Ave
Bennett St
Mellon Park
West Park
Heinz Field
Clemente Park
Point State Park
Pittsburgh
University of Pittsburgh
Carnegie Mellon University
Wilkins Ave
Homewood Cemetery
Wilkinsburg
Penn Ave
Forbes Ave
Duquesne University
2nd Ave
Schenley Park
Frick Park
S Braddock Ave
E Busway
Ninemile Run
Josephine St
E Carson St
Murray Ave
Hazelwood Ave
Saw Mill Run Blvd
Becks Run Rd
E Waterfront Dr
The Waterfront
E 8th Ave
Amity St
Ann St
Johnston Ave
Munhall
Swissv
Glass Run
279
28
579
376
22
885
51
8
837

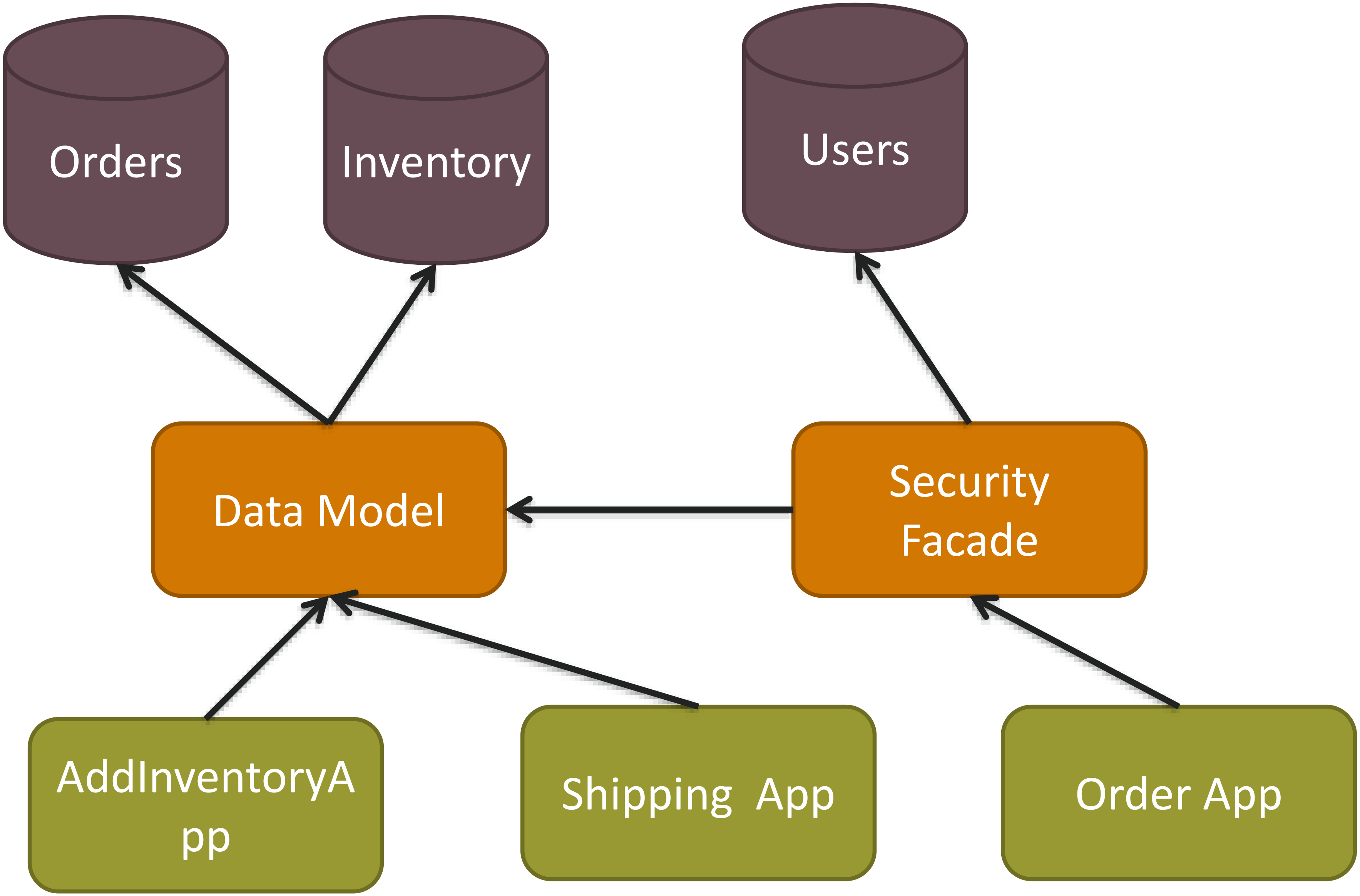
Common Views in Documenting Software Architecture

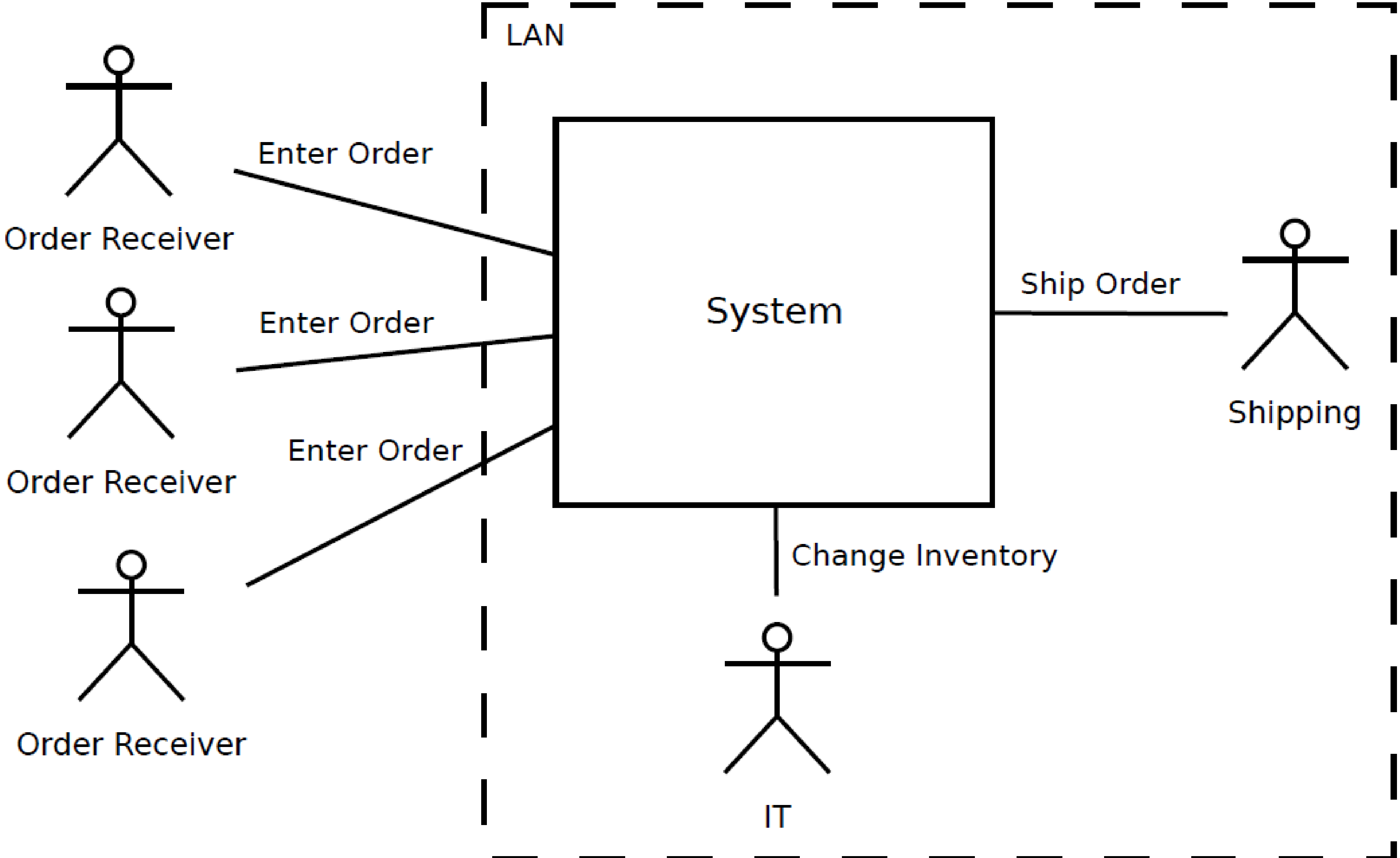
- Static View
  - Modules (subsystems, structures) and their relations (dependencies, …)
- Dynamic View
  - Components (processes, runnable entities) and connectors (messages, data flow, …)
- Physical View (Deployment)
  - Hardware structures and their connections

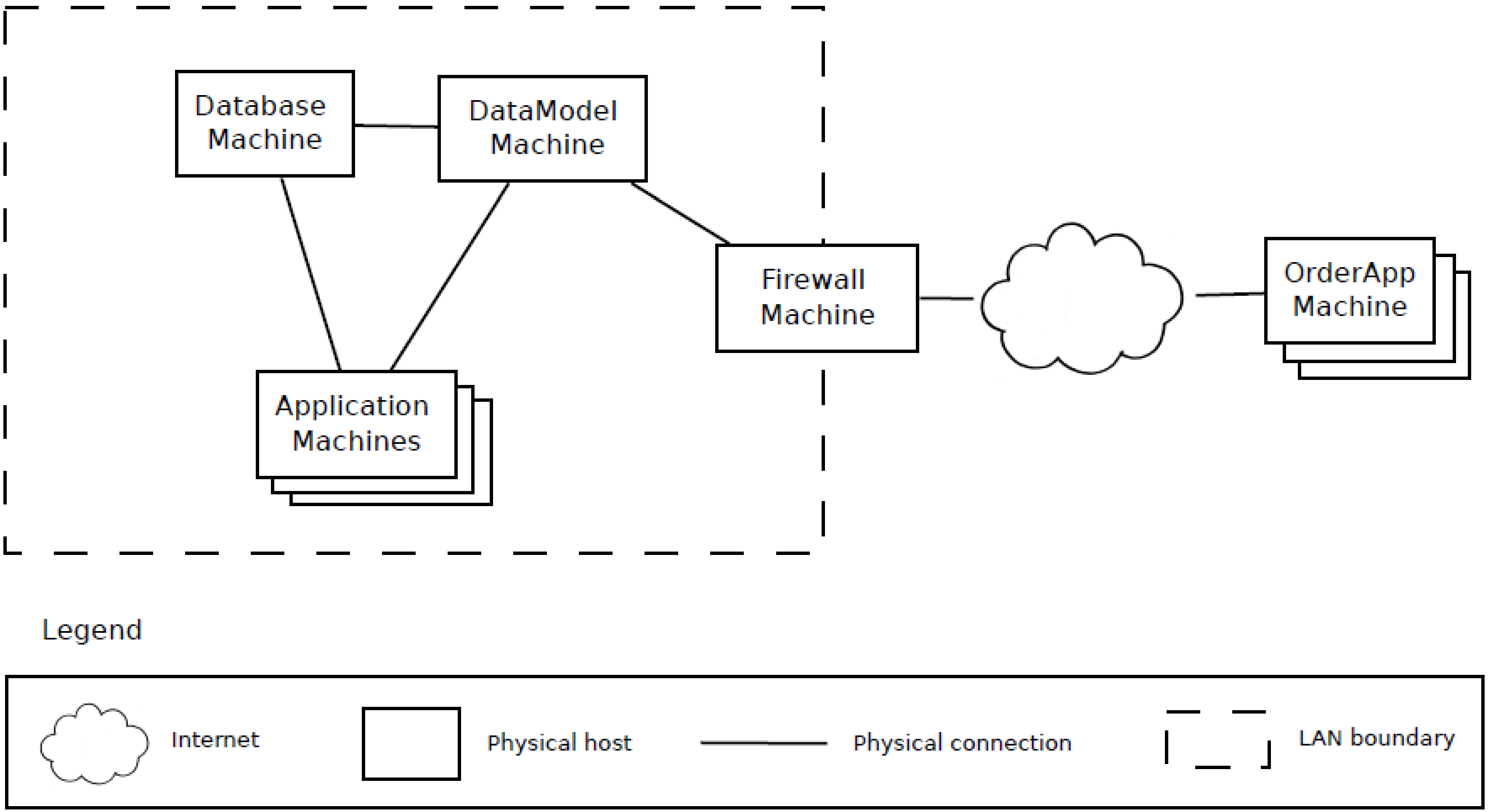
Views and Purposes

- Every view should align with a purpose
- Different views are suitable for different reasoning aspects (different quality goals), e.g.,
  - Performance
  - Extensibility
  - Security
  - Scalability
  - …


DataModel
SecurityFacade
DataModelInterface
RequestTypes
SecurityInterface
AddInventoryApp
ShippingApp
OrderApp
Legend
Class extends interface
Class uses interface


Orders
Inventory
Users
Data Model
Security
Facade
AddInventoryA
pp
Shipping App
Order App


LAN
Order Receiver
Enter Order
Order Receiver
Enter Order
Order Receiver
Enter Order
System
Ship Order
Shipping
Change Inventory
IT


Database Machine
DataModel Machine
Firewall Machine
Application Machines
OrderApp Machine
Legend
Internet
Physical host
Physical connection
LAN boundary

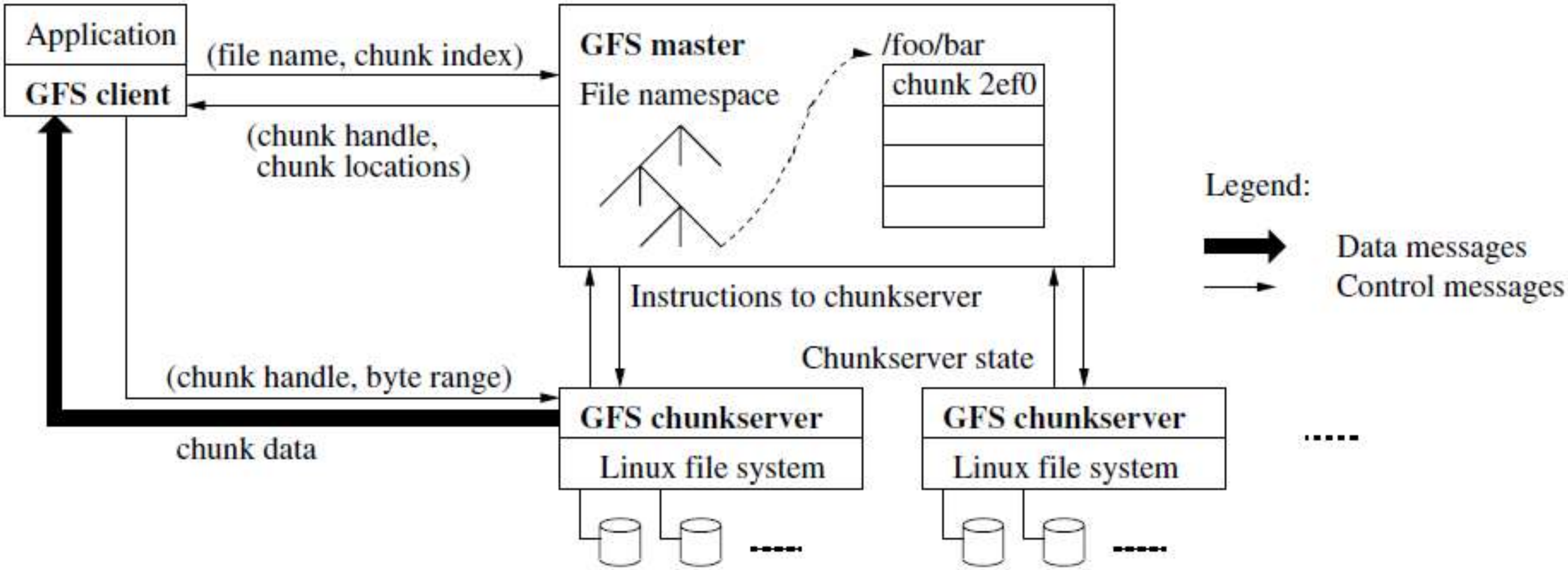


Figure 1: GFS Architecture

Ghemawat, Sanjay, Howard Gobioff, and Shun-Tak Leung. "The Google file system." *ACM SIGOPS operating systems review*. Vol. 37. No. 5. ACM, 2003.

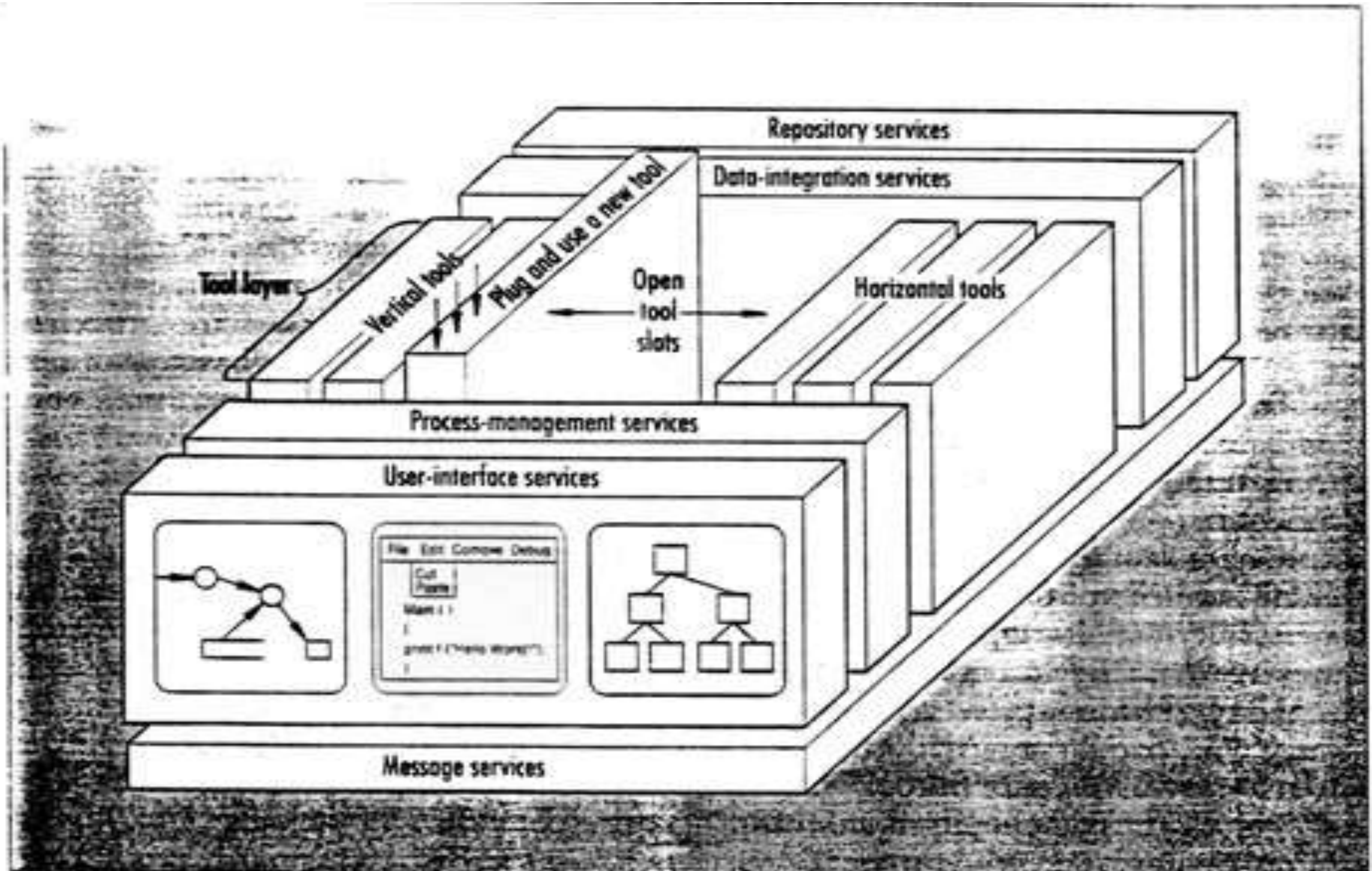


Figure 1. The NIST/ECMA reference model.

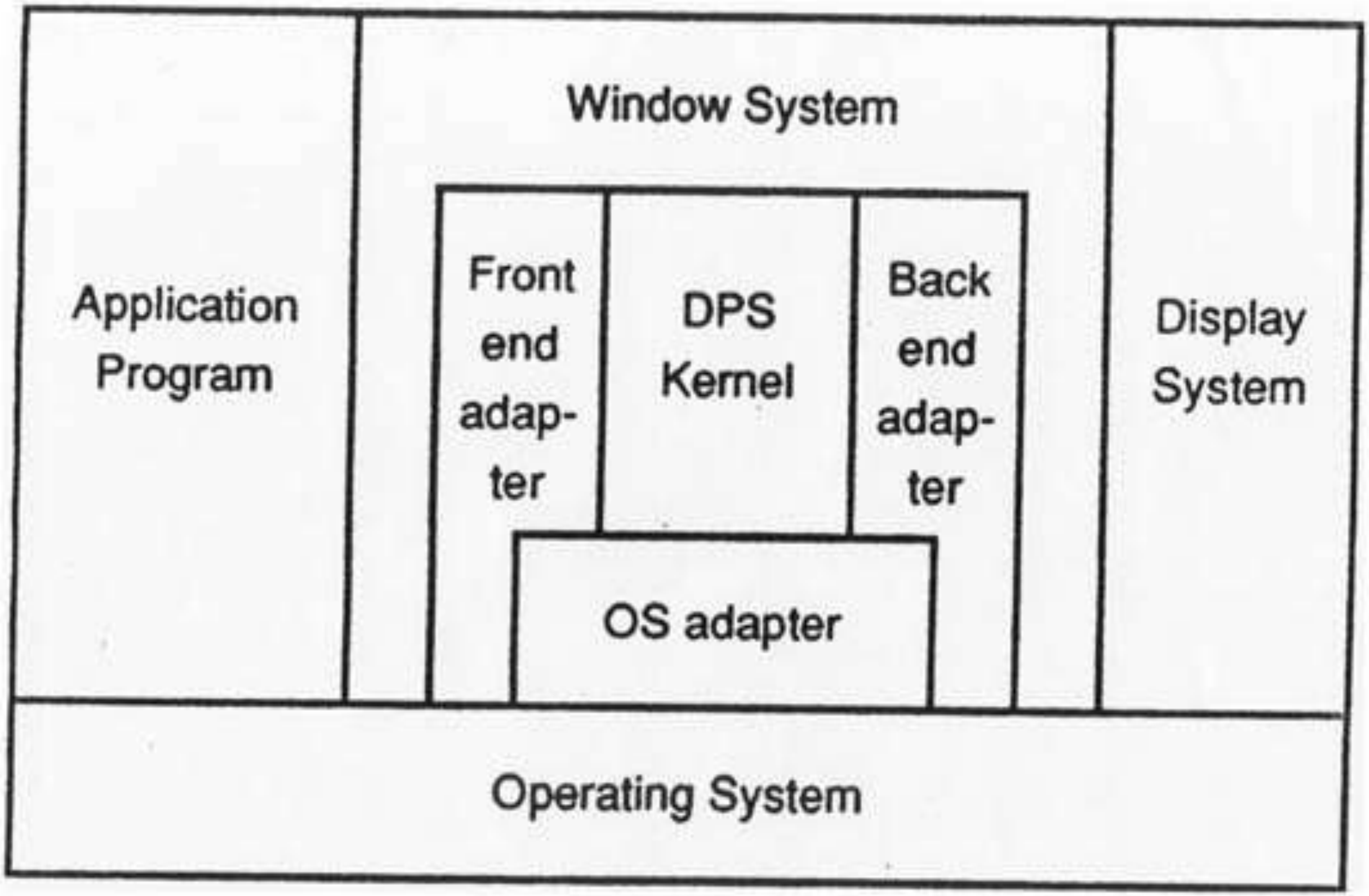


Figure 2. Display PostScript interpreter components.

An Overview of the DISPLAY POSTSCRIPT™ System, Adobe Systems Incorporated, March 16, 1988, P. 10

Client Layer*

- Access domain management
- Buffering and record-level I/O
- Transaction coordination

Agent Layer

- Implementation of standard server interface
- Logger, agent, and instance tasks

Helix Directories

- Path name to FID mapping
- Single-file (database) update by one task
- Procedural interface for queries

Object (FID directory)

- Identification and capability access (via FIDs)
- FID to tree-root mapping; table of (FID,root,ref_count)
- Existence and deletion (reference counts)
- Concurrency control (file interlocking)

Secure Tree

- Basic crash-resistant file structure
- Conditional commit
- Provision of secure array of blocks

System

- Commit and restart authority
- Disk space allocation
- Commit domains

Cache

- Caching and performance optimization
- Commit support (flush)
- Frame allocation (to domains)
- Optional disk shadowing

Canonical Disk

- Physical disk access

*Also called client Helix.

Figure 2. Abstraction layering.

IEEE Software, "Helix: The architecture of the XMS Distributed File System, Marek Fridrich and William Older, May 1985, Vol. 2, No. 3, P. 23

institute for SOFTWARE RESEARCH

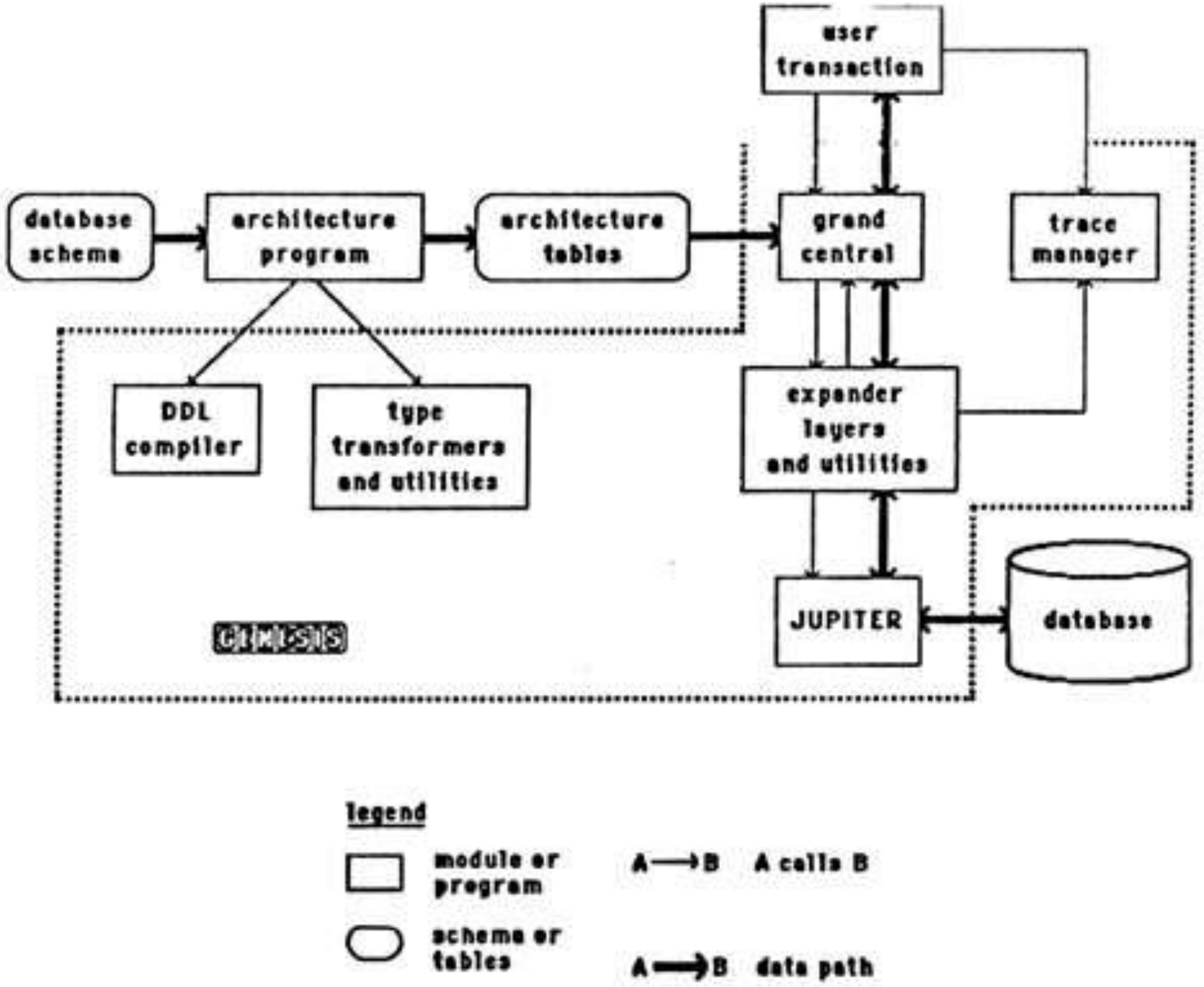
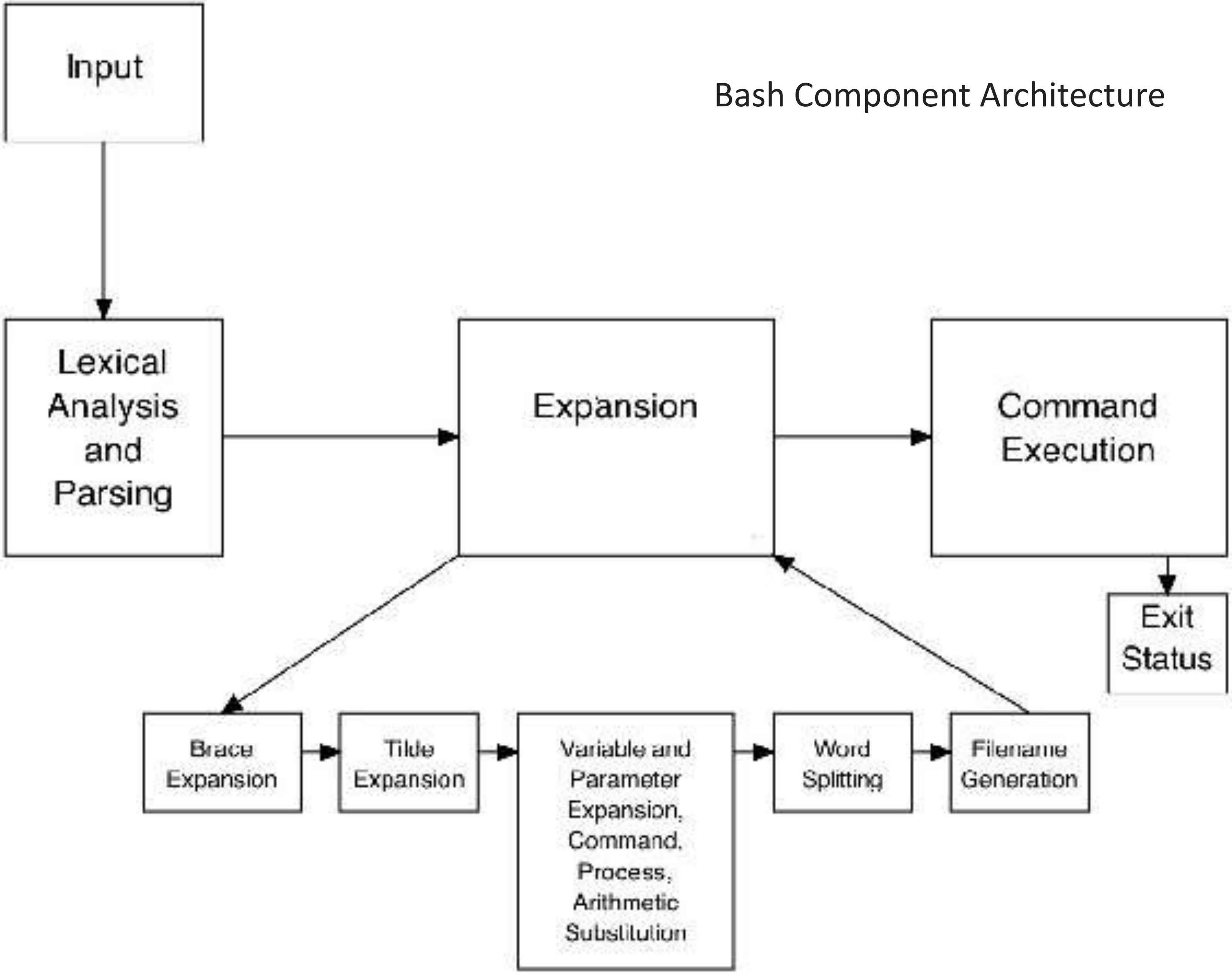


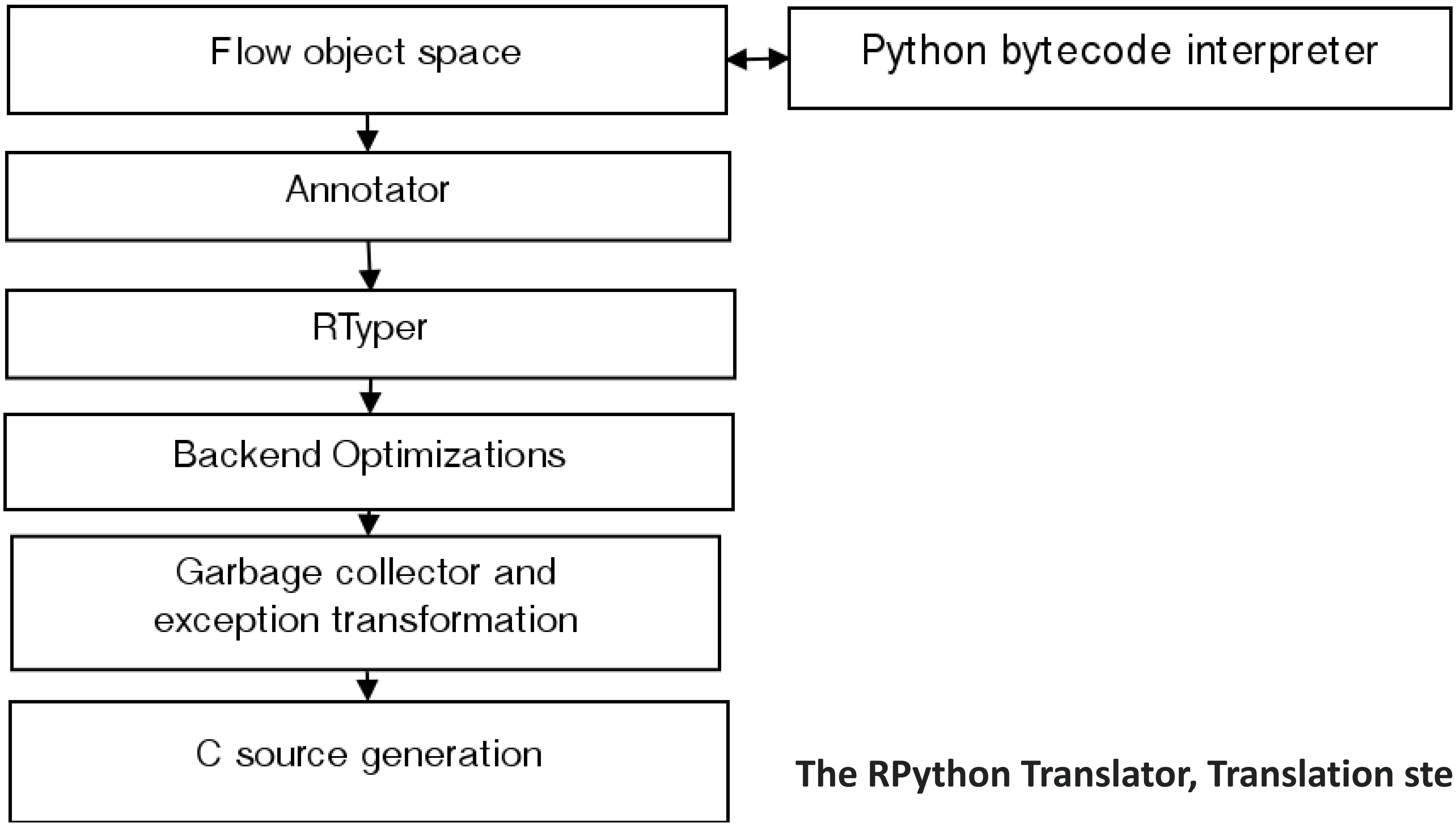
Figure 3.1 The Configuration of the GENESIS Prototype

Genesis: A Reconfiguration Database Management System, D. S. Batory, J.R. Barnett, J.F. Garza, K.P. Smith, K. Tsukuda, B.C. Twichell, T.E. Wise, Department of Computer Sciences, University of Texas at Austin.

Bash Component Architecture

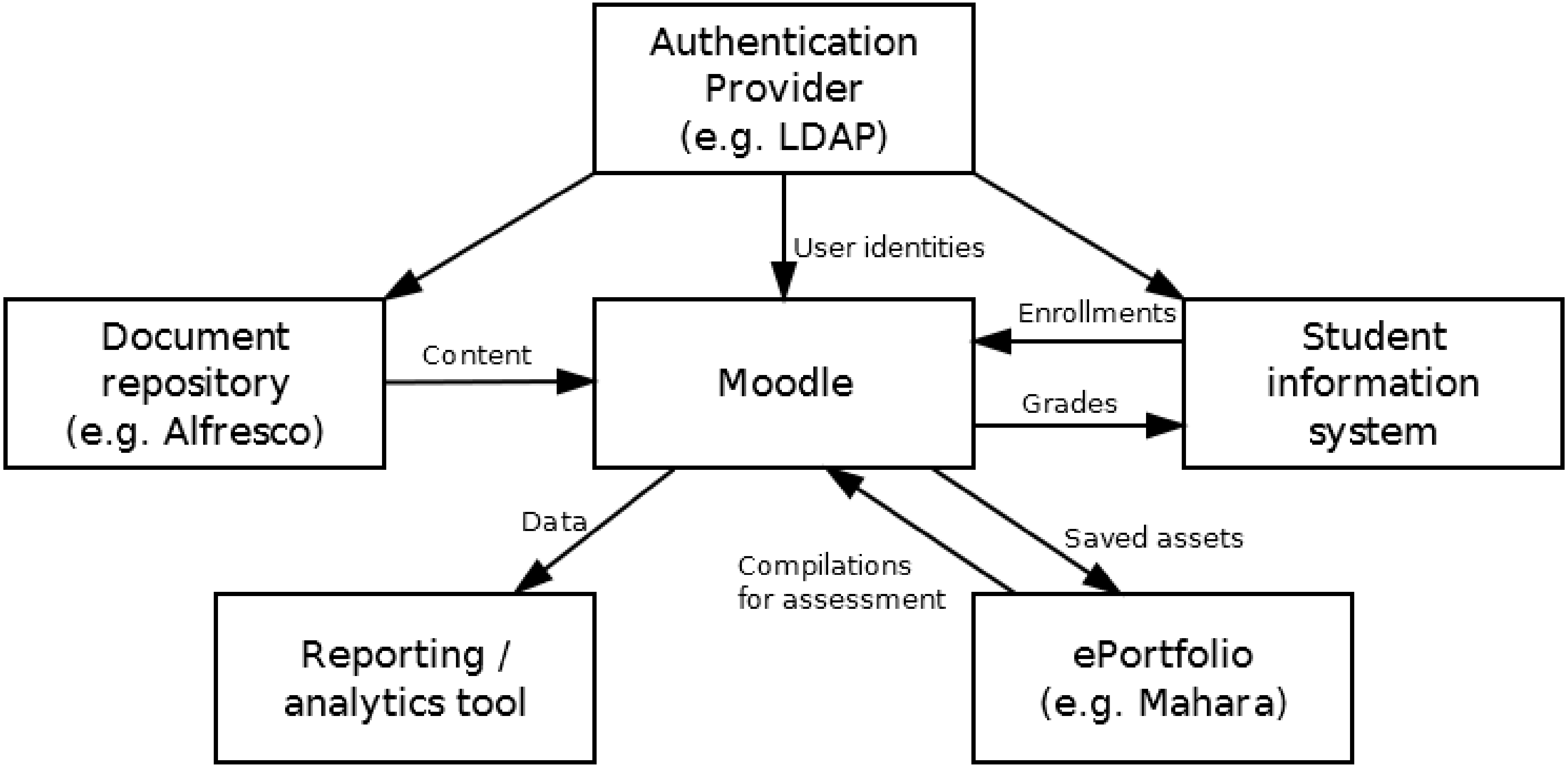


Example source:
http://www.aosabook.org



The RPython Translator, Translation steps

Example source:
http://www.aosabook.org



Moodle: Typical university systems architecture – Key subsystems

Example source: http://www.aosabook.org

Selecting a Notation

- Suitable for purpose
- Often visual for compact representation
- Usually boxes and arrows
- UML possible (semi-formal), but possibly constraining
  - Note the different abstraction level – Subsystems or processes, not classes or objects
- Formal notations available
- Decompose diagrams hierarchically and in views

What is Wrong Today?

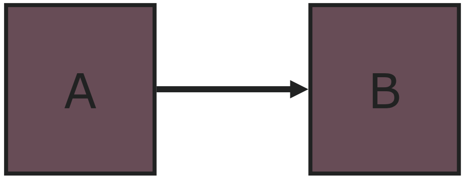
- In practice today's documentation consists of
  - Ambiguous box-and-line diagrams
  - Inconsistent use of notations
  - Confusing combinations of viewtypes
- Many things are left unspecified:
  - What kind of elements?
  - What kind of relations?
  - What do the boxes and arrows mean?
  - What is the significance of the layout?

Guidelines: Avoiding Ambiguity

- Always include a legend
- Define precisely what the boxes mean
- Define precisely what the lines mean
- Don't mix viewtypes unintentionally
  - Recall: Module (classes), C&C (components)
- Supplement graphics with explanation
  - Very important: rationale (architectural intent)
- Do not try to do too much in one diagram
  - Each view of architecture should fit on a page
  - Use hierarchy

What could the arrow mean?

- Many possibilities
  - A passes control to B
  - A passes data to B
  - A gets a value from B
  - A streams data to B
  - A sends a message to B
  - A creates B
  - A occurs before B
  - B gets its electricity from A
  - …



Recommendations for Recitation and Homework

- Use UML or UML-like notations:
  - Class diagrams for static and physical views
  - Communication diagrams for dynamic view
  - Use correct abstraction level (usually not classes/objects)
- Extend notation as needed
  - Provide a legend explaining the extensions or deviations from standard UML notation

Case Study: The Google File System

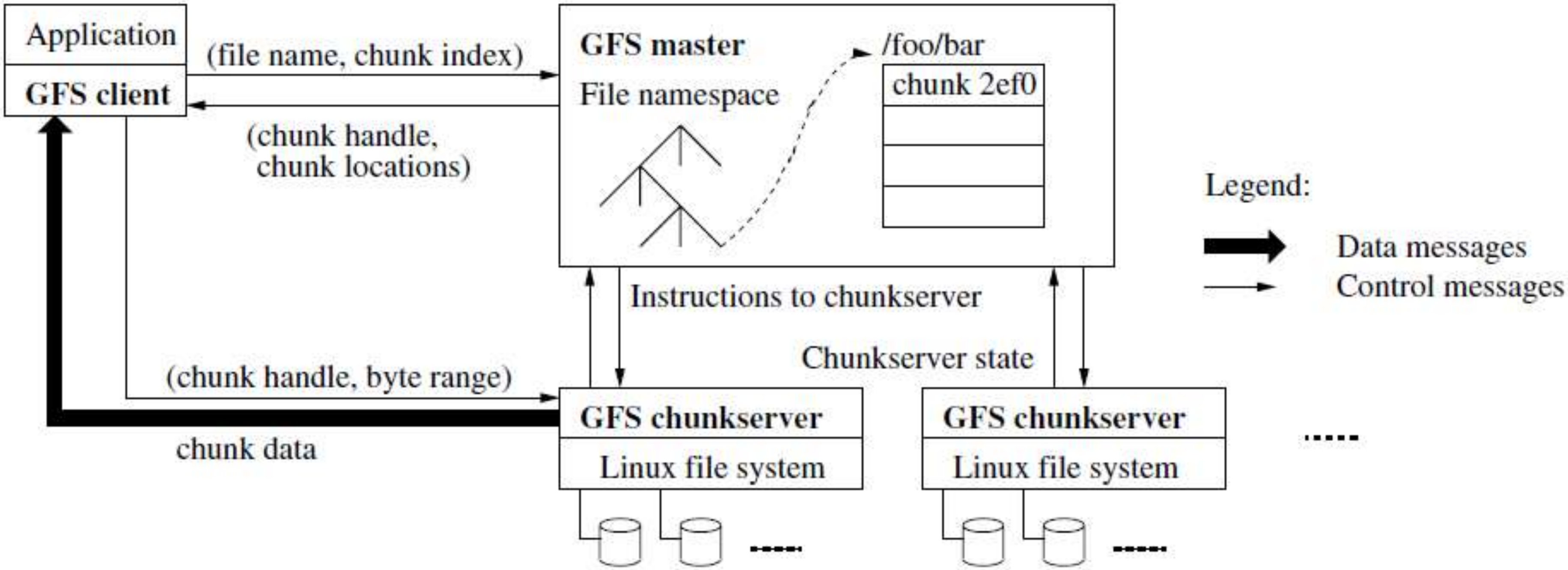


Figure 1: GFS Architecture

Ghemawat, Sanjay, Howard Gobioff, and Shun-Tak Leung. "The Google file system." *ACM SIGOPS operating systems review*. Vol. 37. No. 5. ACM, 2003.

Assumptions

- The system is built from many inexpensive commodity components that often fail.
- The system stores a modest number of large files.
- The workloads primarily consist of two kinds of reads: large streaming reads and small random reads.
- The workloads also have many large, sequential writes that append data to files.
- The system must efficiently implement well-defined semantics for multiple clients that concurrently append to the same file.
- High sustained bandwidth is more important than low latency.

Ghemawat, Sanjay, Howard Gobioff, and Shun-Tak Leung. "The Google file system." *ACM SIGOPS operating systems review*. Vol. 37. No. 5. ACM, 2003.

Qualities:
Scalability
Reliability
Performance
Cost

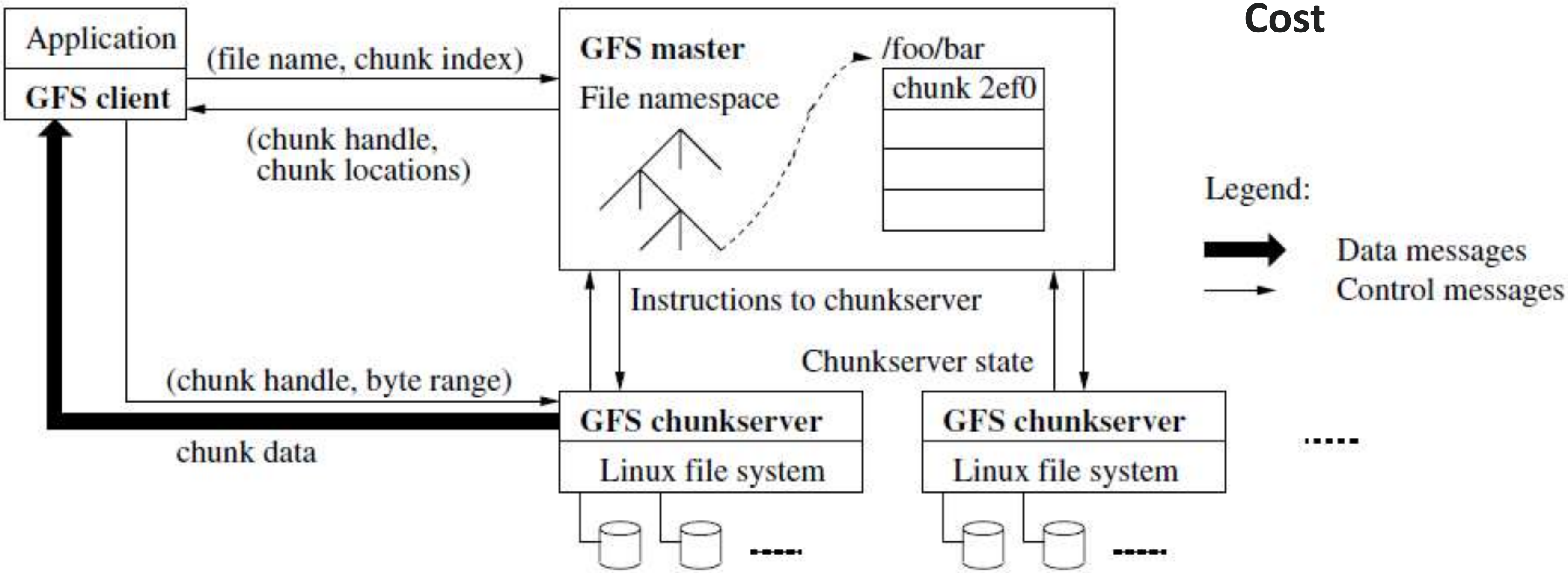


Figure 1: GFS Architecture

Ghemawat, Sanjay, Howard Gobioff, and Shun-Tak Leung. "The Google file system." *ACM SIGOPS operating systems review*. Vol. 37. No. 5. ACM, 2003.

Questions

1. What are the most important quality attributes in the design?
2. How are those quality attributes realized in the design?

Qualities:
Scalability
Reliability
Performance
Cost

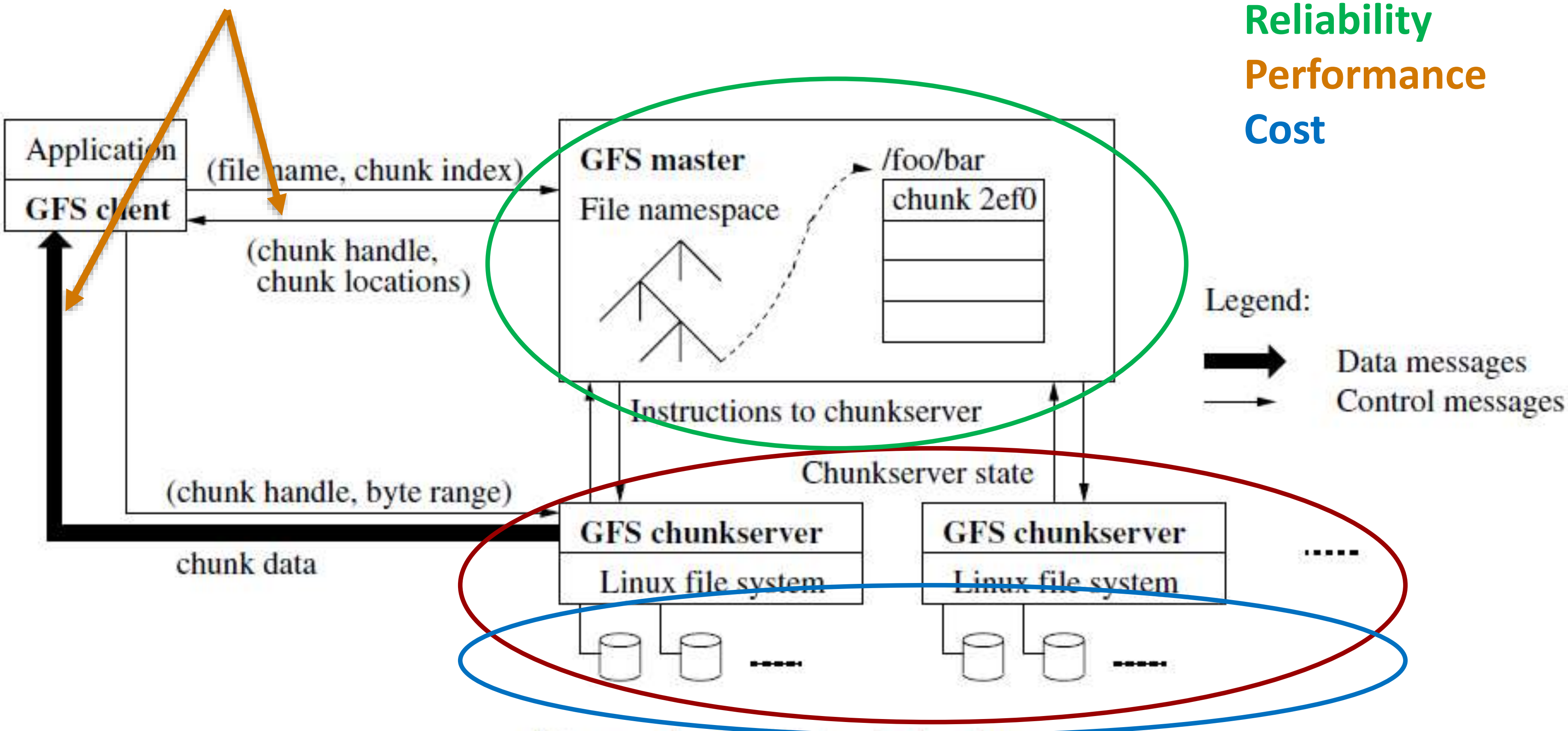


Figure 1: GFS Architecture

Ghemawat, Sanjay, Howard Gobioff, and Shun-Tak Leung. "The Google file system." *ACM SIGOPS operating systems review*. Vol. 37. No. 5. ACM, 2003.

Exercise

For the Google File System, create a physical architecture view that addresses a relevant quality attribute

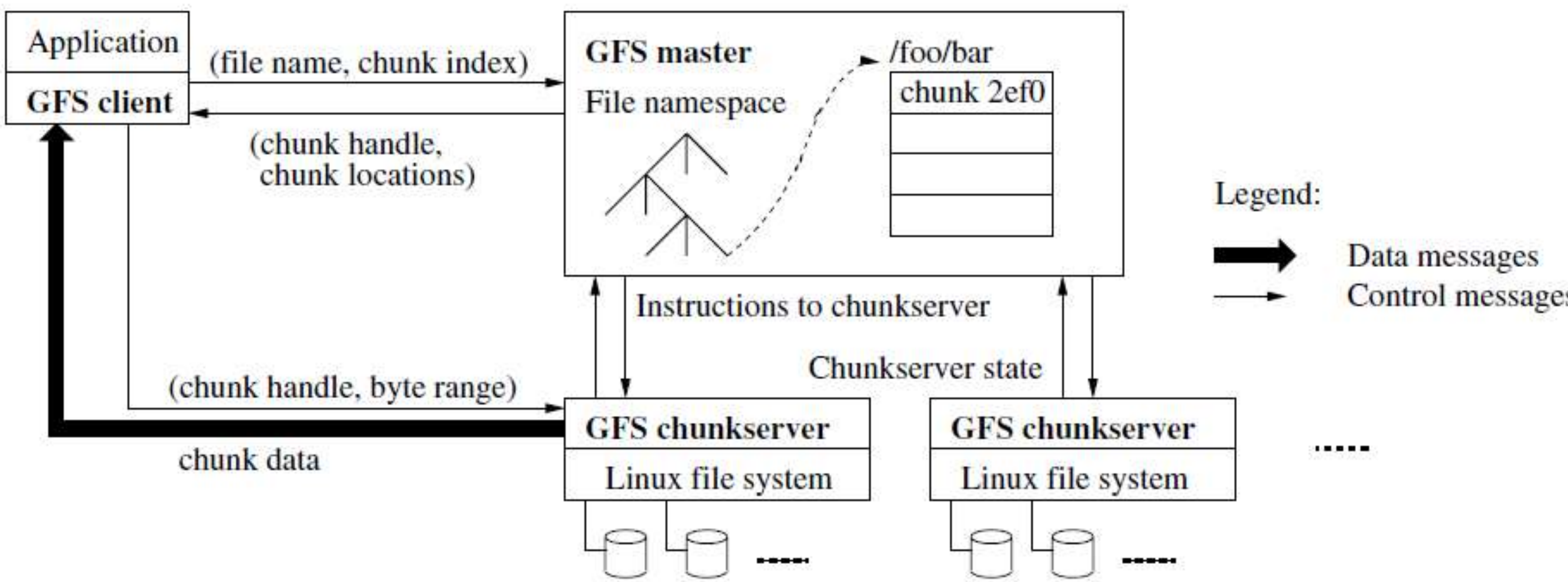


Figure 1: GFS Architecture

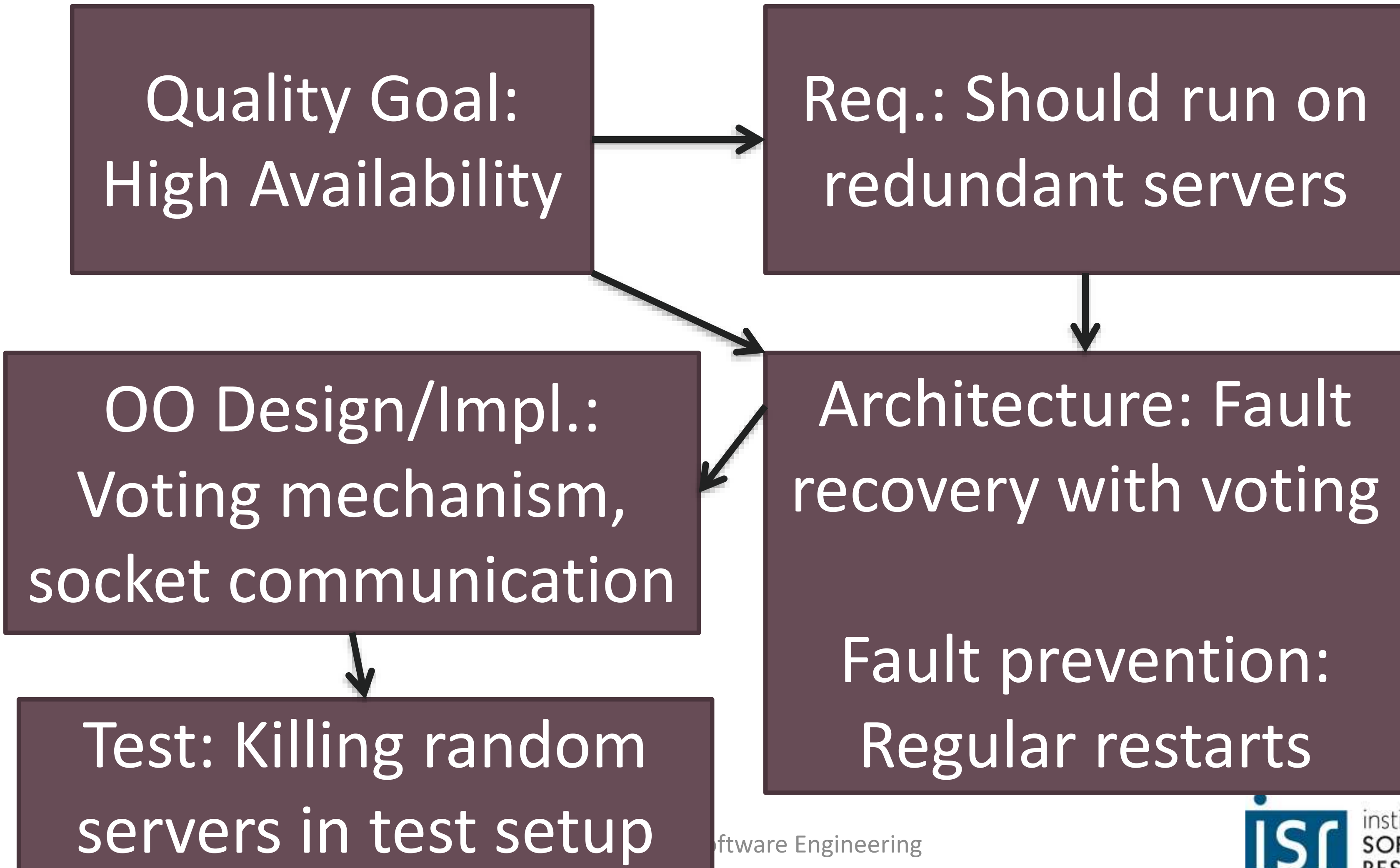
Traceability

Traceability - Definition

"The ability to interrelate any uniquely identifiable software engineering artifact to any other, maintain required links over time, and use the resulting network to answer questions of both the software product and it's development process" – CoEST

Traceability in Requirements?

Traceability



Traceability Compliance

- Traceability required in some domains (avionics)
  - Why does X piece of code exist?
- "Enable verification of the absence of undocumented source code and verification of the complete implementation of the low-level requirements"
- Link to specifications and test procedures

Traceability and Architecture

- Architecture links quality attributes to the high-level and low-level system design
- Ensures quality attributes often not even visible in code
- Cost, effort, discipline needed to create and maintain.
  - Often incomplete, incorrect, outdated
- Developers hate it, and often do not understand the need.
  - "Unnecessary evil"

institute for SOFTWARE RESEARCH

So far in course

Business Requirements Document

Feature	Definition	Requirement Shopping
Standard based and interoperable messaging protocol	Messaging protocol must be based on industry standards to enable interoperability	
Send Only	Also called Push MEP is simple one-way messaging where a message is sent with no expectation response	
Receive only	Also called Pull MEP is a message pattern where a non-addressable sender supports the ability to explicitly obtain messages from another application. This can be used for exchanges	
Request/Response exchange	Message pattern consists of one or more request/response pairs. The correlation between request and a response is well defined. In this response maybe deferred and the requesting application may or may not block application processing until a response is received	
Diagnostics	Authentication, diagnostic, logging & routing information should be included in the message and not the payload	
Reliability	Protocol capability to support assured and single delivery to the receiving application with no loss	

Requirements

Implementation

Architecture

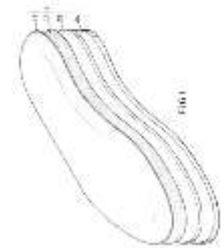
Levels of abstraction

- Requirements
  – high-level “what” needs to be done
- Architecture (High-level design)
  – high-level “how”, mid-level “what”
- OO-Design (Low-level design, e.g. design patterns)
  – mid-level “how”, low-level “what”
- Code
  – low-level “how”

What is architecture?

Architecture as structures and relations
(the actual system)

Architecture as documentation
(representations of the system)



Architecture as process
(activities around the other two)

institute for SOFTWARE RESEARCH

Architectural Styles and Tactics

Architectural style (pattern)

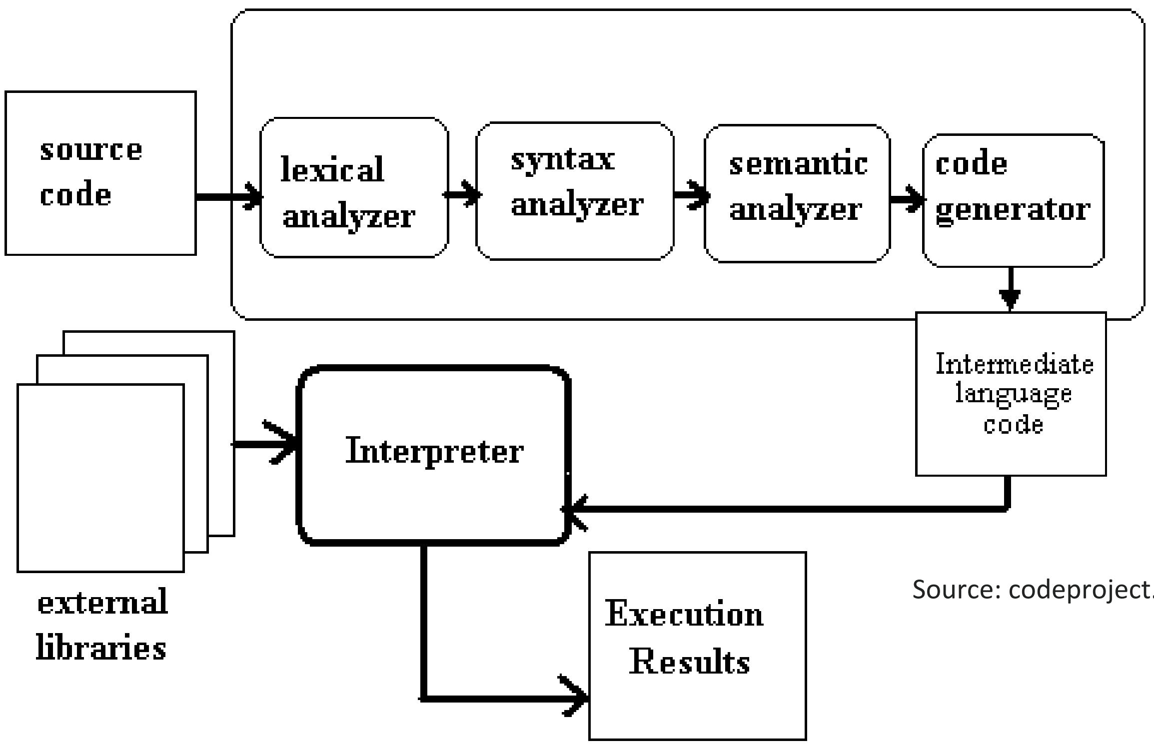
- Broad principle of system organization
- Describes computational model
  - E.g., pipe and filter, call-return, publish-subscribe, layered, services
- Related to one of common view types
  - Static, dynamic, physical

Example Architectural Patterns

- System organization
  - Repository model
  - Client-server model
  - Layered model
- Modular decomposition
  - Object oriented
  - Function-oriented pipelining
- Control styles
  - Centralized control
  - Event-driven systems

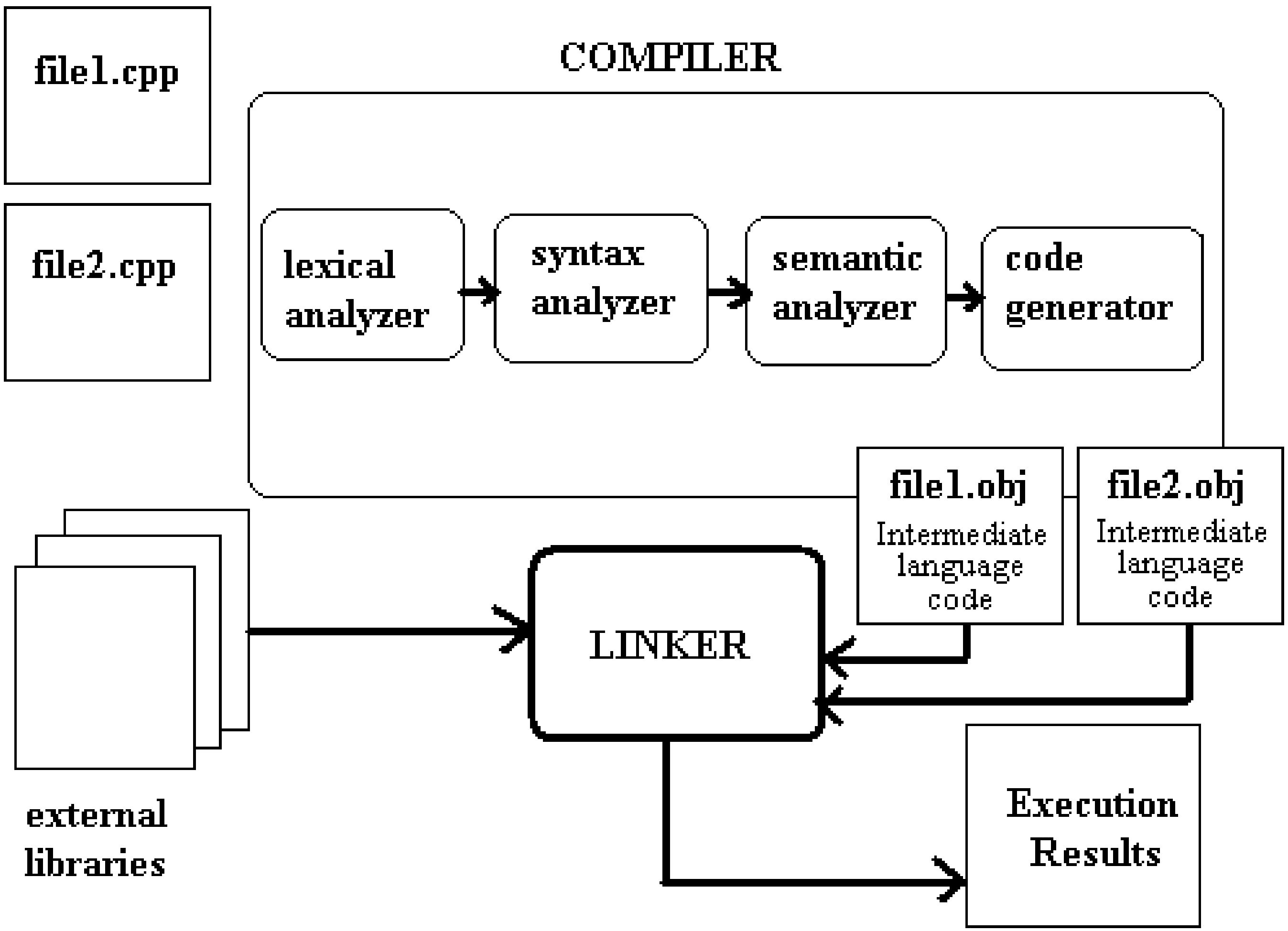
Architectural style (pattern)

- Broad principle of system organization
- See reading



Source: codeproject.org

Architectural style (pattern)



Source: codeproject.org

Client-server pattern

- Separation of clients and servers
  - Servers provide services; known and "stable"
  - Clients request services; come and go
- Varieties: synchronous/asynchronous
- Impact on security, performance, scalability
- Examples: TCP, HTTP, X11

Client

Server

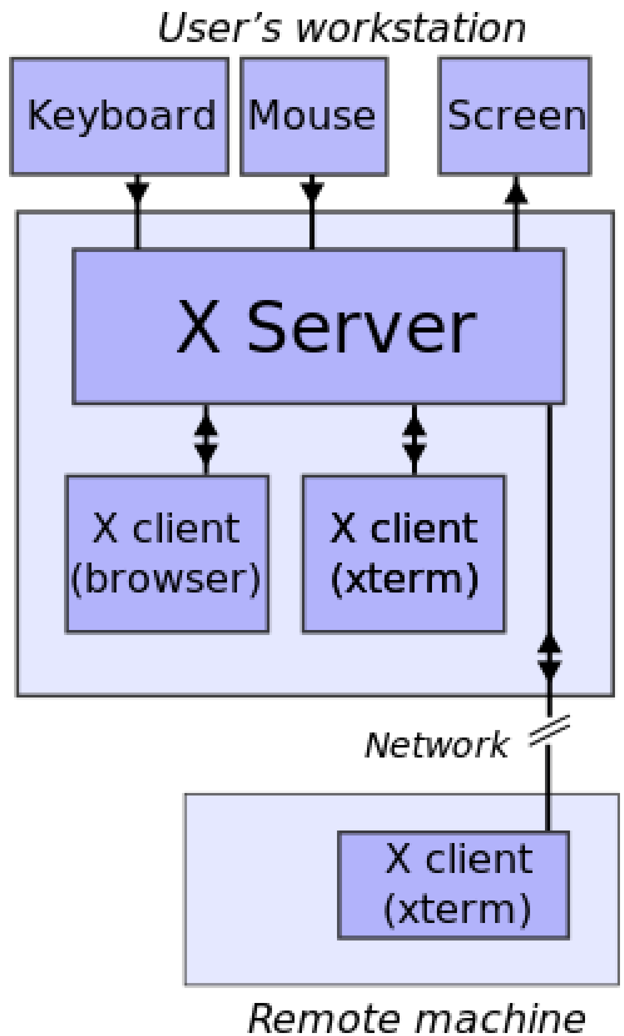
Database

Where to validate user input?

Example: Yelp App

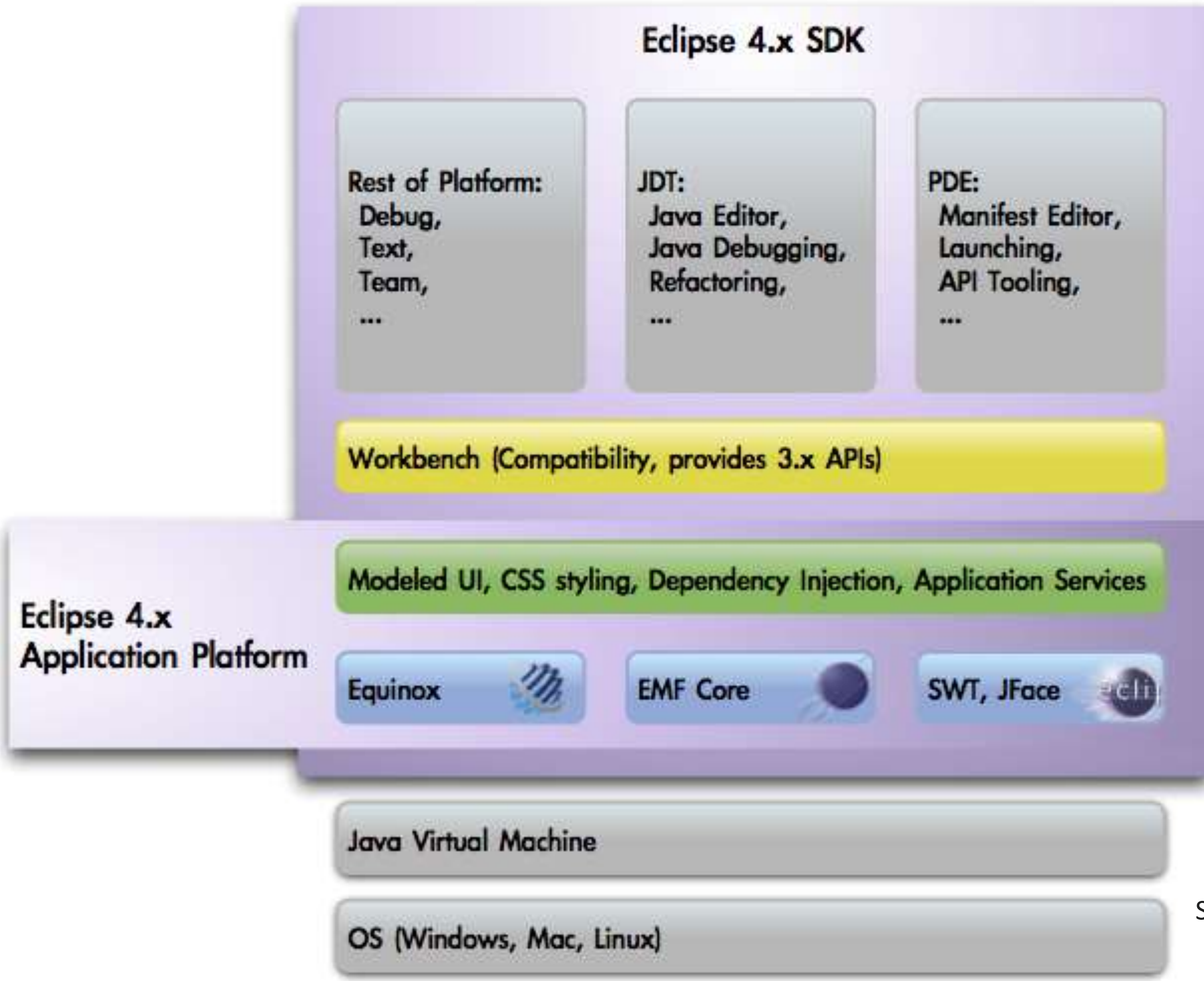
institute for SOFTWARE RESEARCH

Client-server style



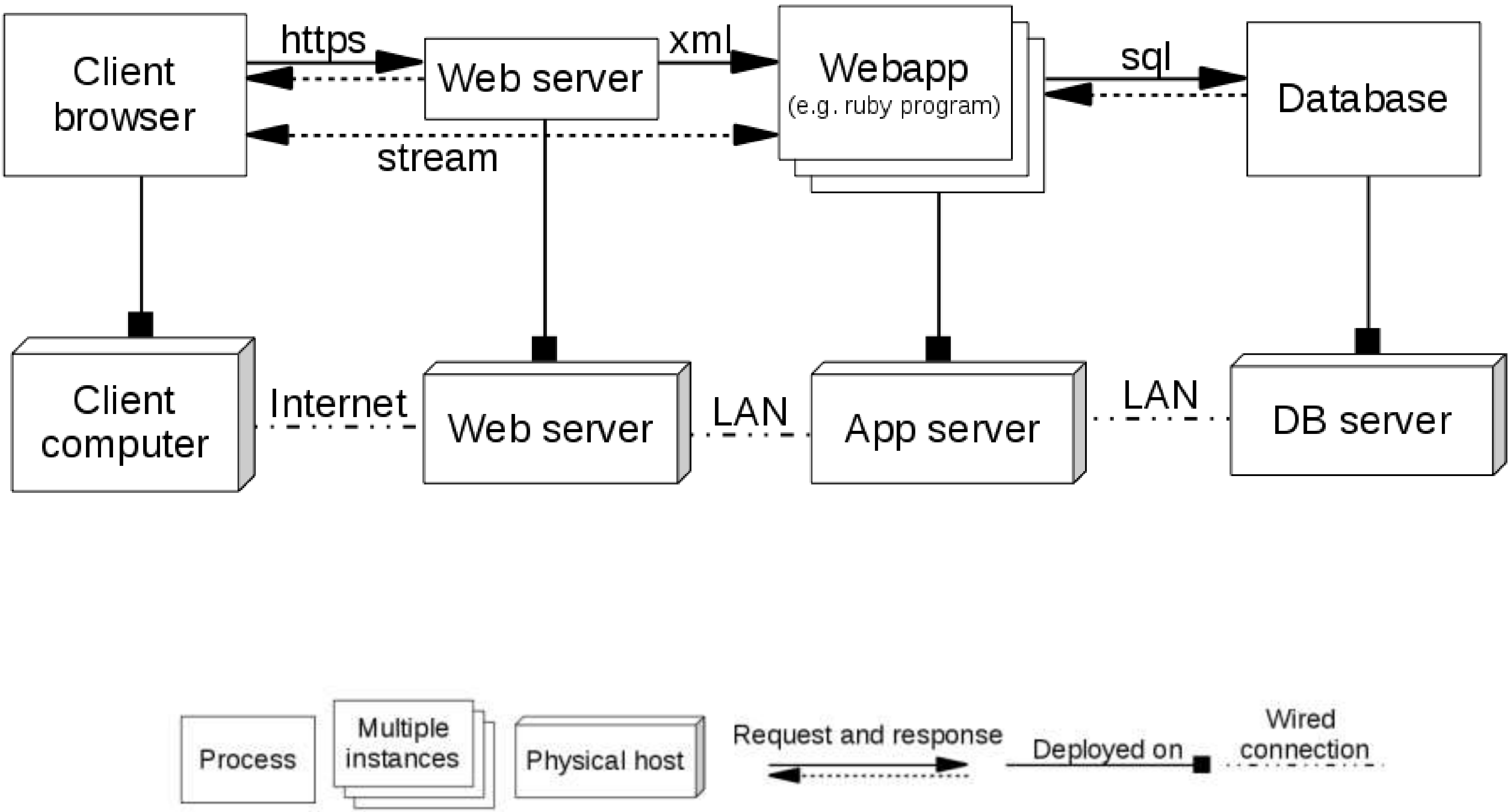
Source: wikimedia commons

Layered system



Source: eclipse.org

Tiered architecture



Architectural Style?

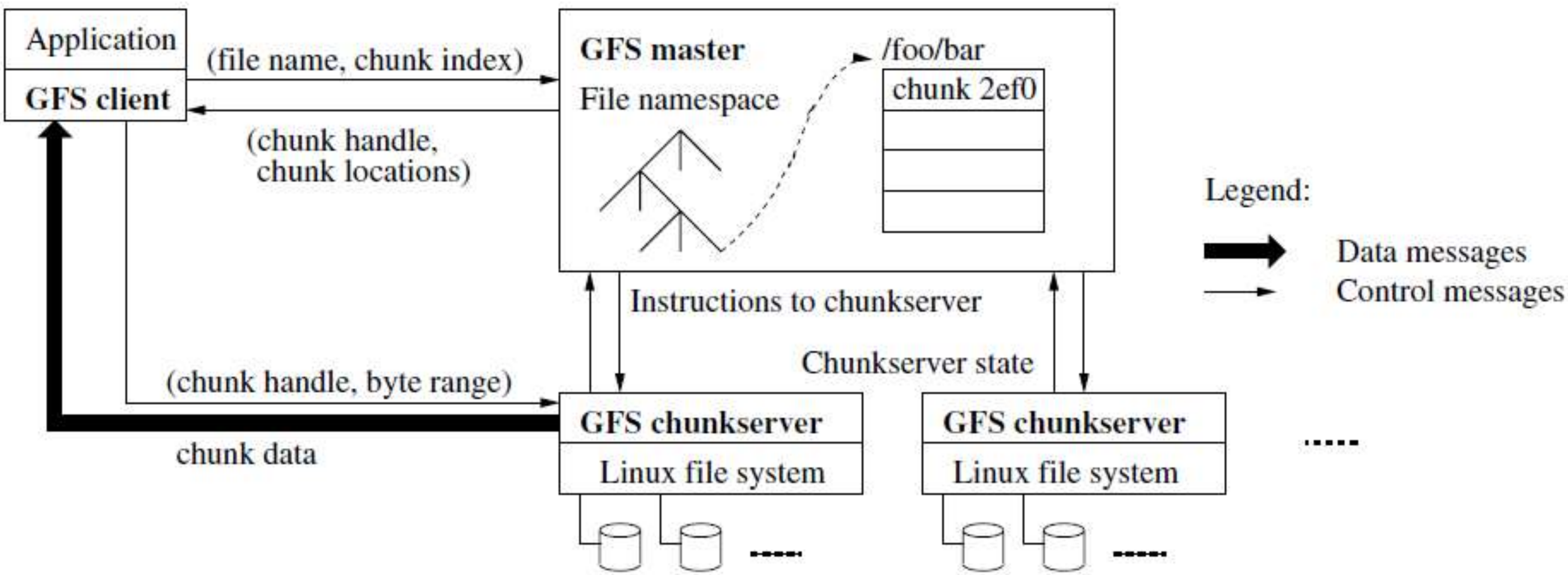
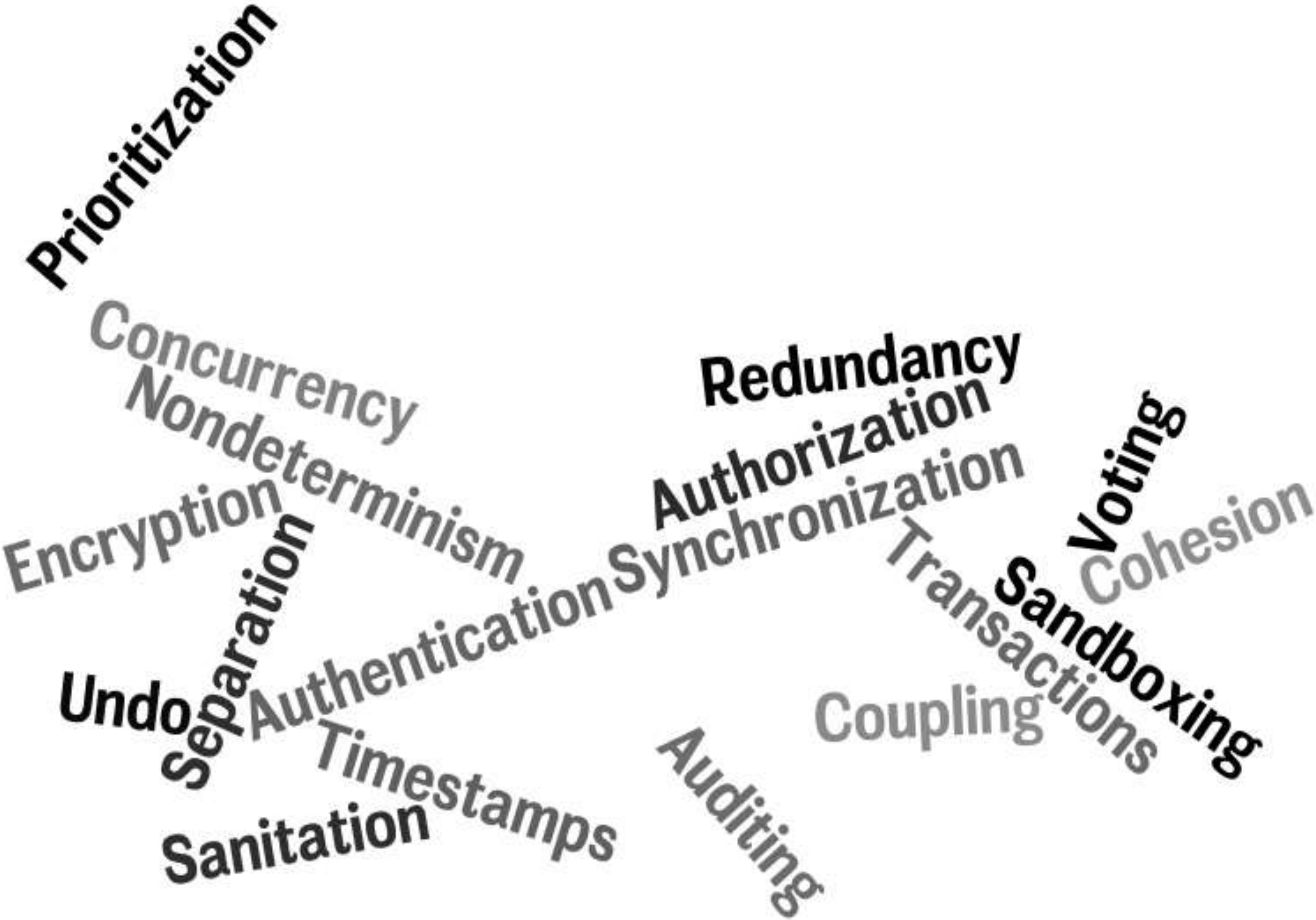


Figure 1: GFS Architecture

Ghemawat, Sanjay, Howard Gobioff, and Shun-Tak Leung. "The Google file system." *ACM SIGOPS operating systems review*. Vol. 37. No. 5. ACM, 2003.

Tactics

- Architectural techniques to achieve qualities
  - More tied to specific context and quality
- Smaller scope than architectural patterns
  - Problem solved by patterns: “How do I structure my (sub)system?”
  - Problem solved by tactics: “How do I get better at quality X?”
- Collection of common strategies and known solutions
  - Resemble OO design patterns

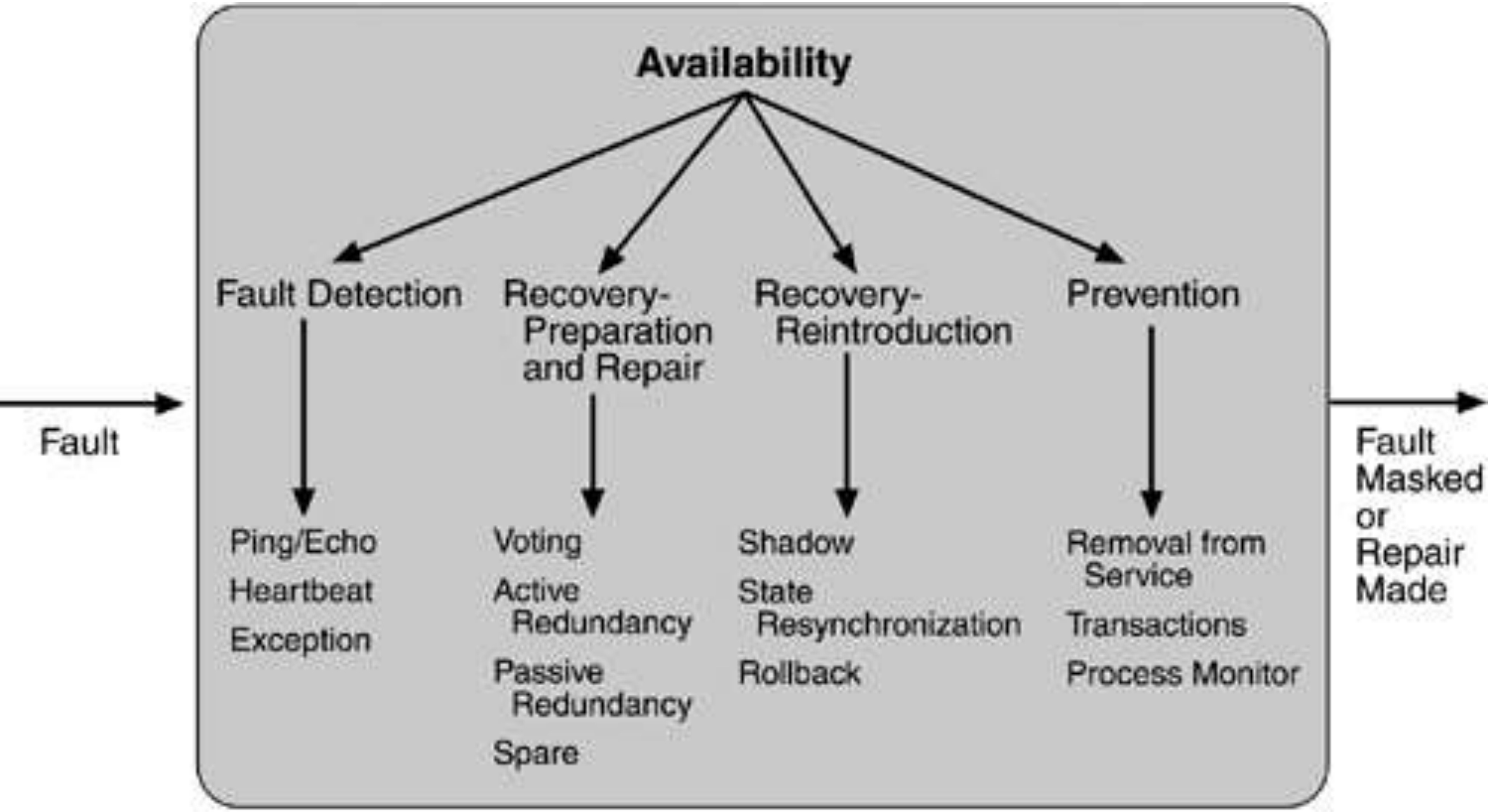

Prioritization
Concurrency
Nondeterminism
Encryption
Separation
Undo
Authentication
Timestamps
Sanitation
Redundancy
Authorization
Synchronization
Voting
Cohesion
Transactions
Sandboxing
Coupling
Auditing

Example Tactic Description: Record/playback

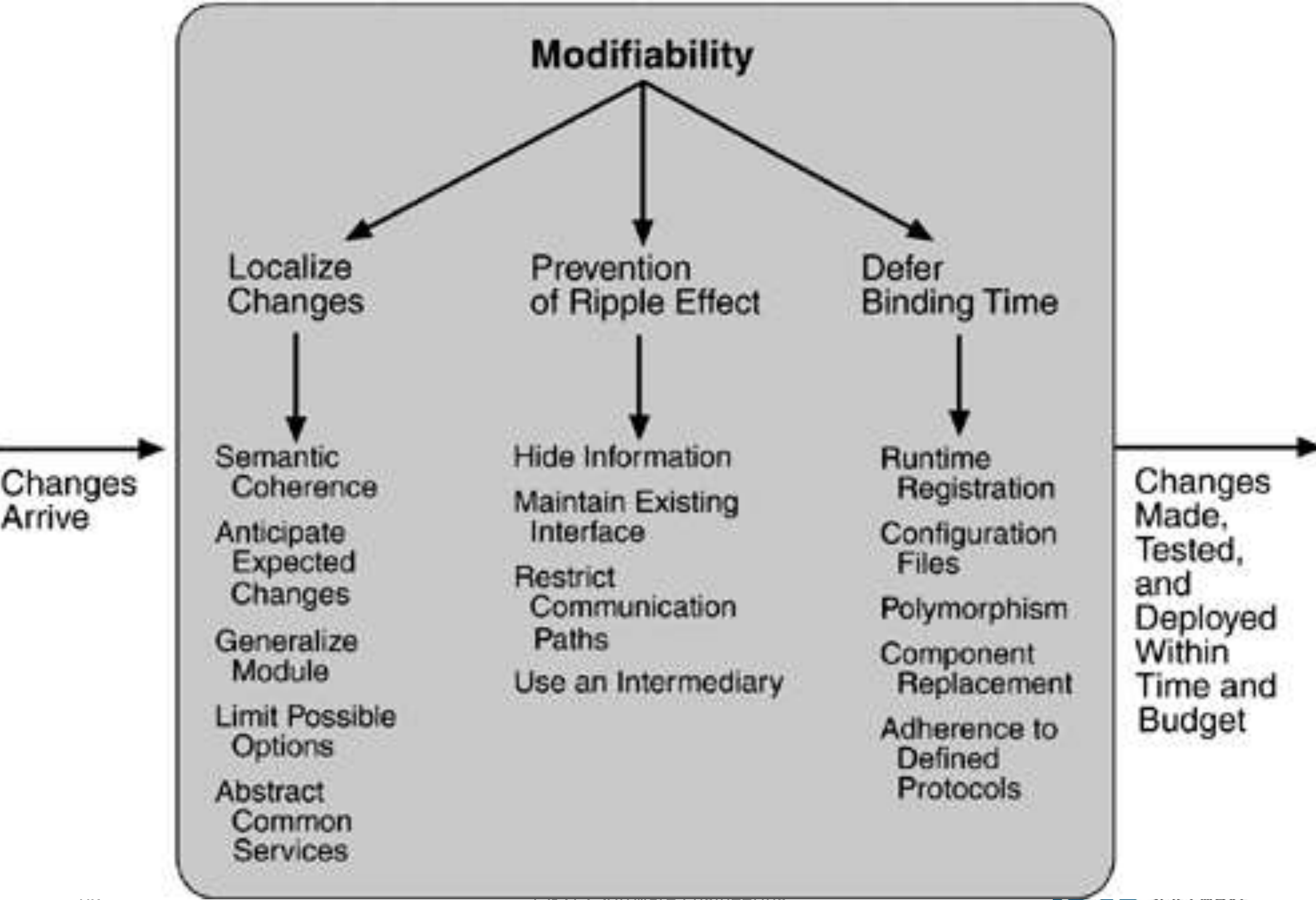
- Record/playback refers to both capturing information crossing an interface and using it as input into the test harness. The information crossing an interface during normal operation is saved in some repository and represents output from one component and input to another. Recording this information allows test input for one of the components to be generated and test output for later comparison to be saved.

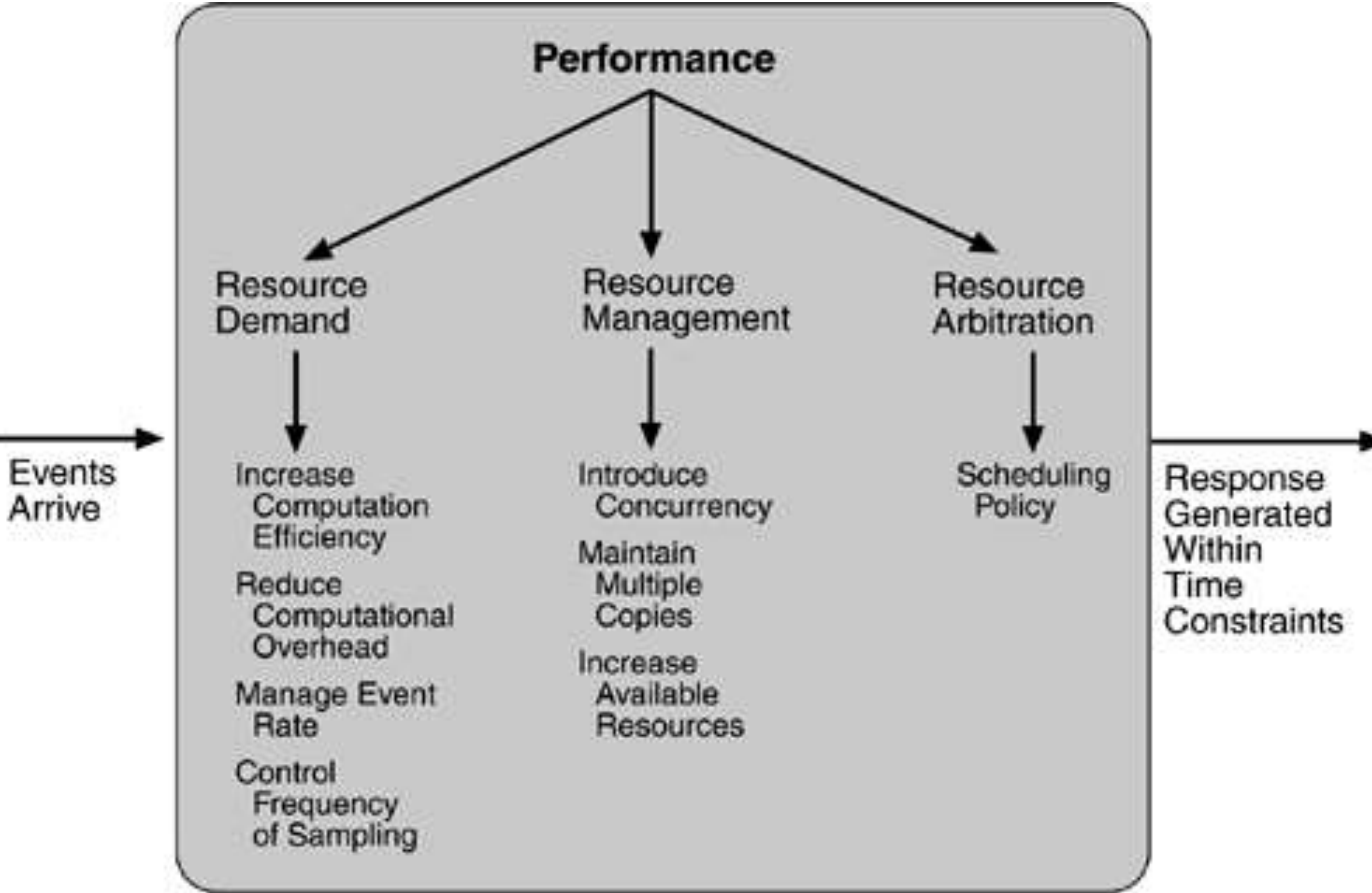
Example Tactic Description: Built-in monitors

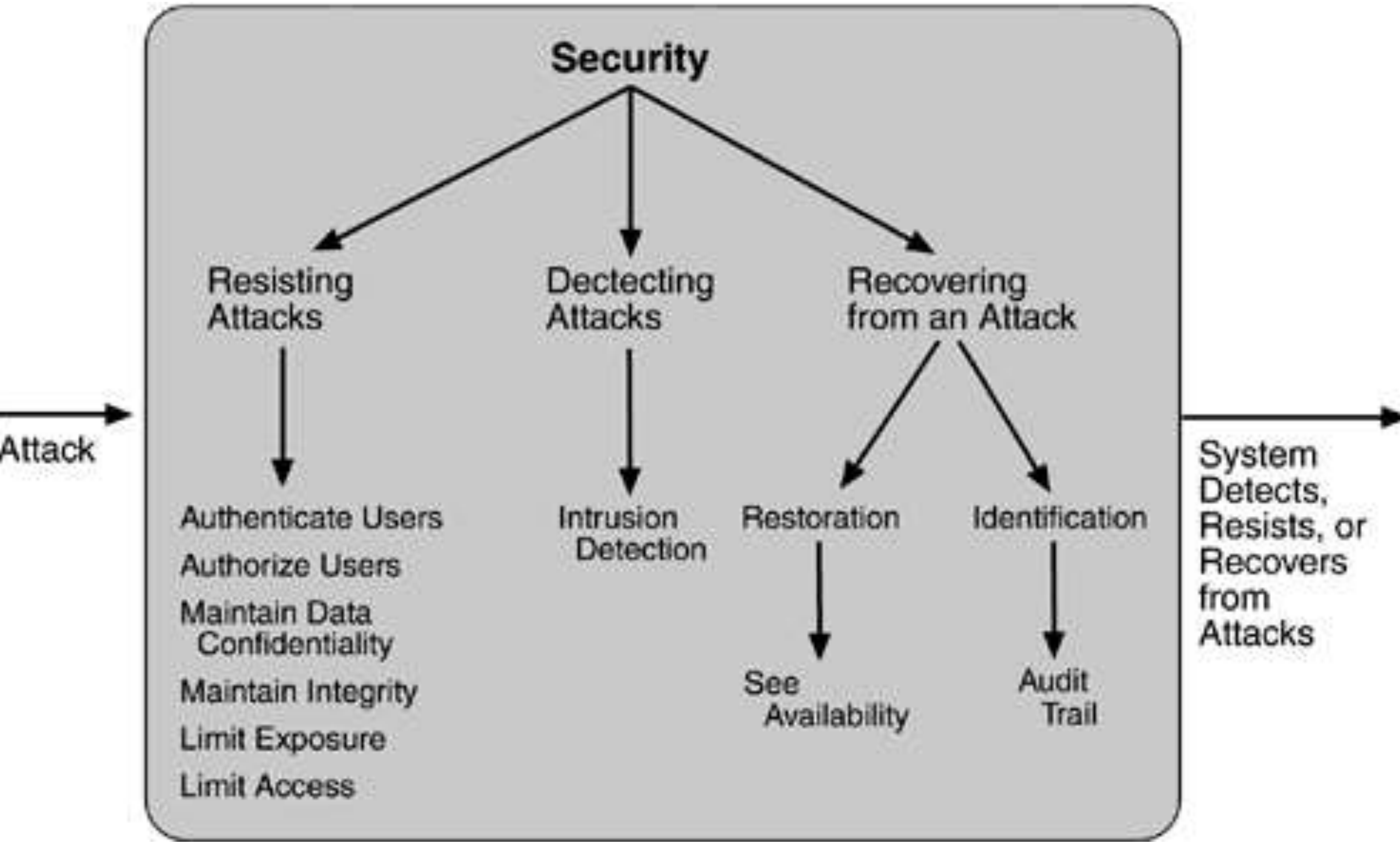
- The component can maintain state, performance load, capacity, security, or other information accessible through an interface. This interface can be a permanent interface of the component or it can be introduced temporarily via an instrumentation technique such as aspect-oriented programming or preprocessor macros. A common technique is to record events when monitoring states have been activated. Monitoring states can actually increase the testing effort since tests may have to be repeated with the monitoring turned off. Increased visibility into the activities of the component usually more than outweigh the cost of the additional testing.

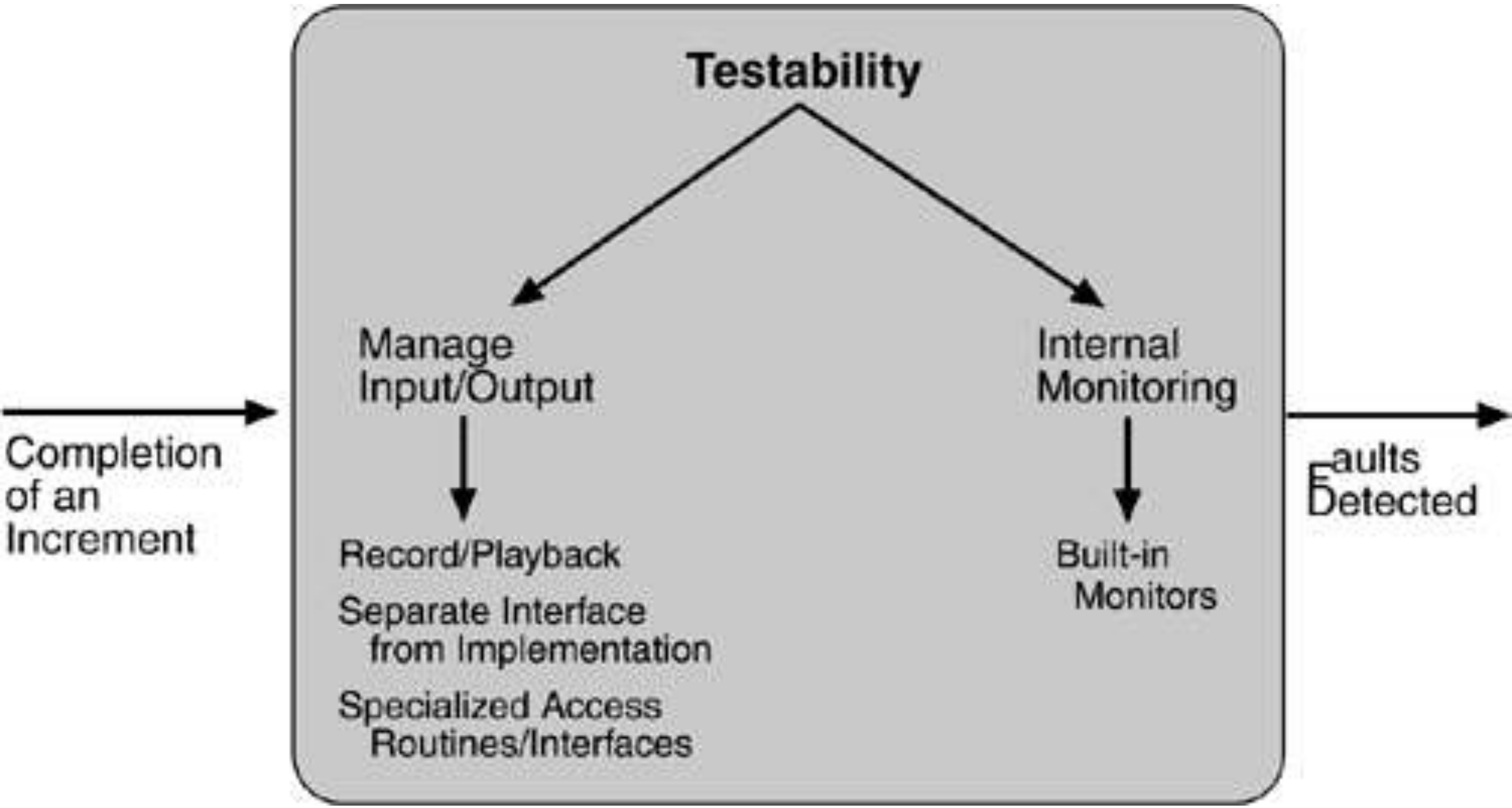


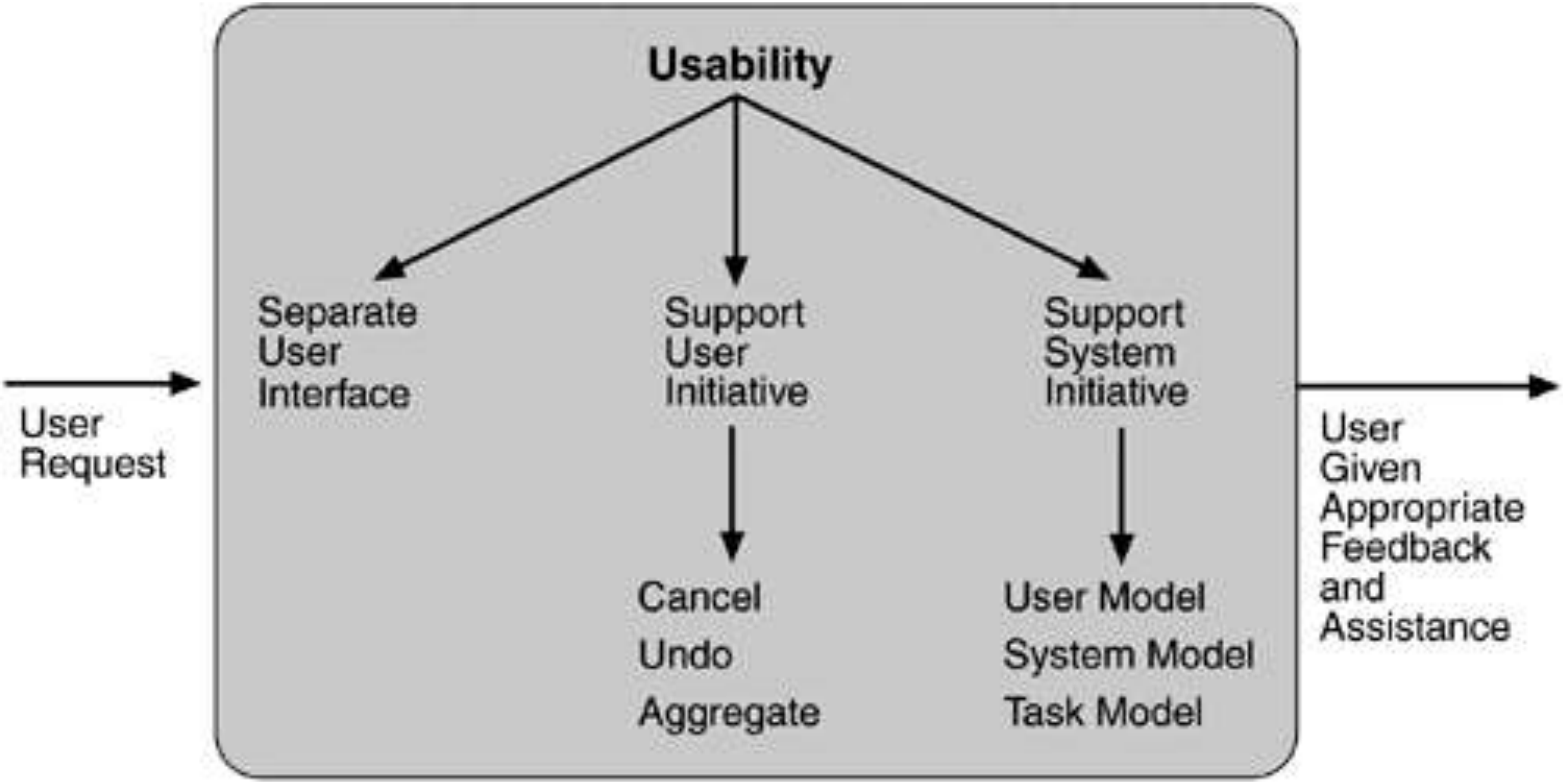
Availability
Fault Detection
Recovery-Preparation and Repair
Recovery-Reintroduction
Prevention
Fault
Ping/Echo
Heartbeat
Exception
Voting
Active Redundancy
Passive Redundancy
Spare
Shadow
State Resynchronization
Rollback
Removal from Service
Transactions
Process Monitor
Fault Masked or Repair Made

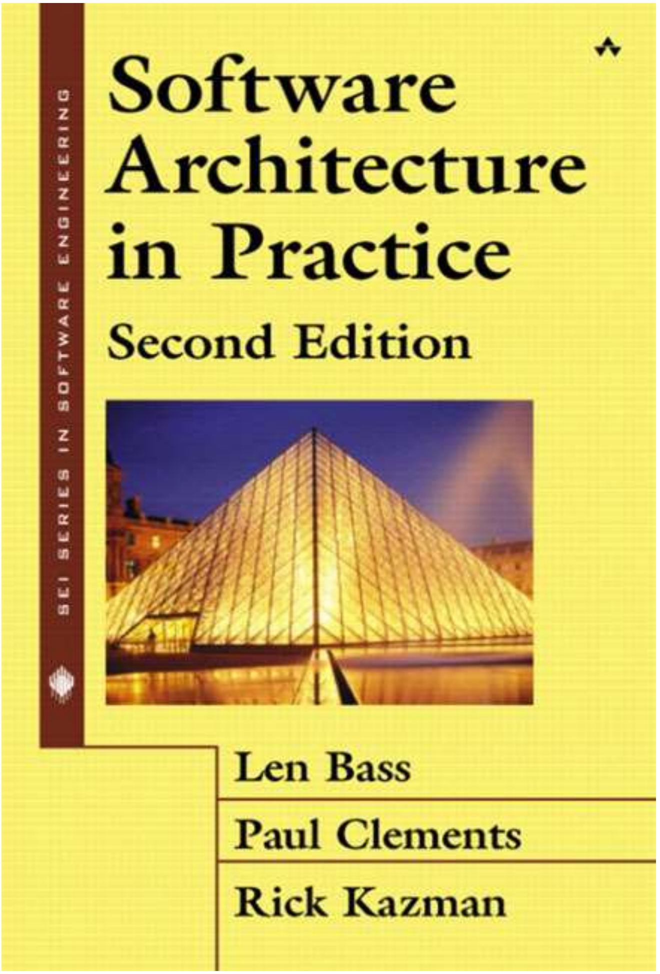

Modifiability
Localize Changes
Prevention of Ripple Effect
Defer Binding Time
Changes Arrive
Semantic Coherence
Anticipate Expected Changes
Generalize Module
Limit Possible Options
Abstract Common Services
Hide Information
Maintain Existing Interface
Restrict Communication Paths
Use an Intermediary
Runtime Registration
Configuration Files
Polymorphism
Component Replacement
Adherence to Defined Protocols
Changes Made, Tested, and Deployed Within Time and Budget


Performance
Resource Demand
Resource Management
Resource Arbitration
Events Arrive
Increase Computation Efficiency
Reduce Computational Overhead
Manage Event Rate
Control Frequency of Sampling
Introduce Concurrency
Maintain Multiple Copies
Increase Available Resources
Scheduling Policy
Response Generated Within Time Constraints


Security
Resisting Attacks
Dectecting Attacks
Recovering from an Attack
Attack
Authenticate Users
Authorize Users
Maintain Data Confidentiality
Maintain Integrity
Limit Exposure
Limit Access
Intrusion Detection
Restoration
Identification
See Availability
Audit Trail
System Detects, Resists, or Recovers from Attacks


Testability
Manage Input/Output
Internal Monitoring
Completion of an Increment
Faults Detected
Record/Playback
Separate Interface from Implementation
Specialized Access Routines/Interfaces
Built-in Monitors


Usability
Separate User Interface
Support User Initiative
Support System Initiative
Cancel
Undo
Aggregate
User Model
System Model
Task Model
User Request
User Given Appropriate Feedback and Assistance



Many tactics described in Chapter 5

Brief high-level descriptions (about 1 paragraph per tactic)

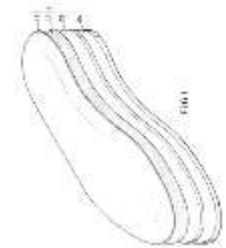
Second and more detailed third edition available as ebook through CMU library.

Architecture Design Process

What is architecture?

Architecture as
structures and relations
(the actual system)

Architecture as
documentation
(representations of the system)



Architecture as process
(activities around the other two)

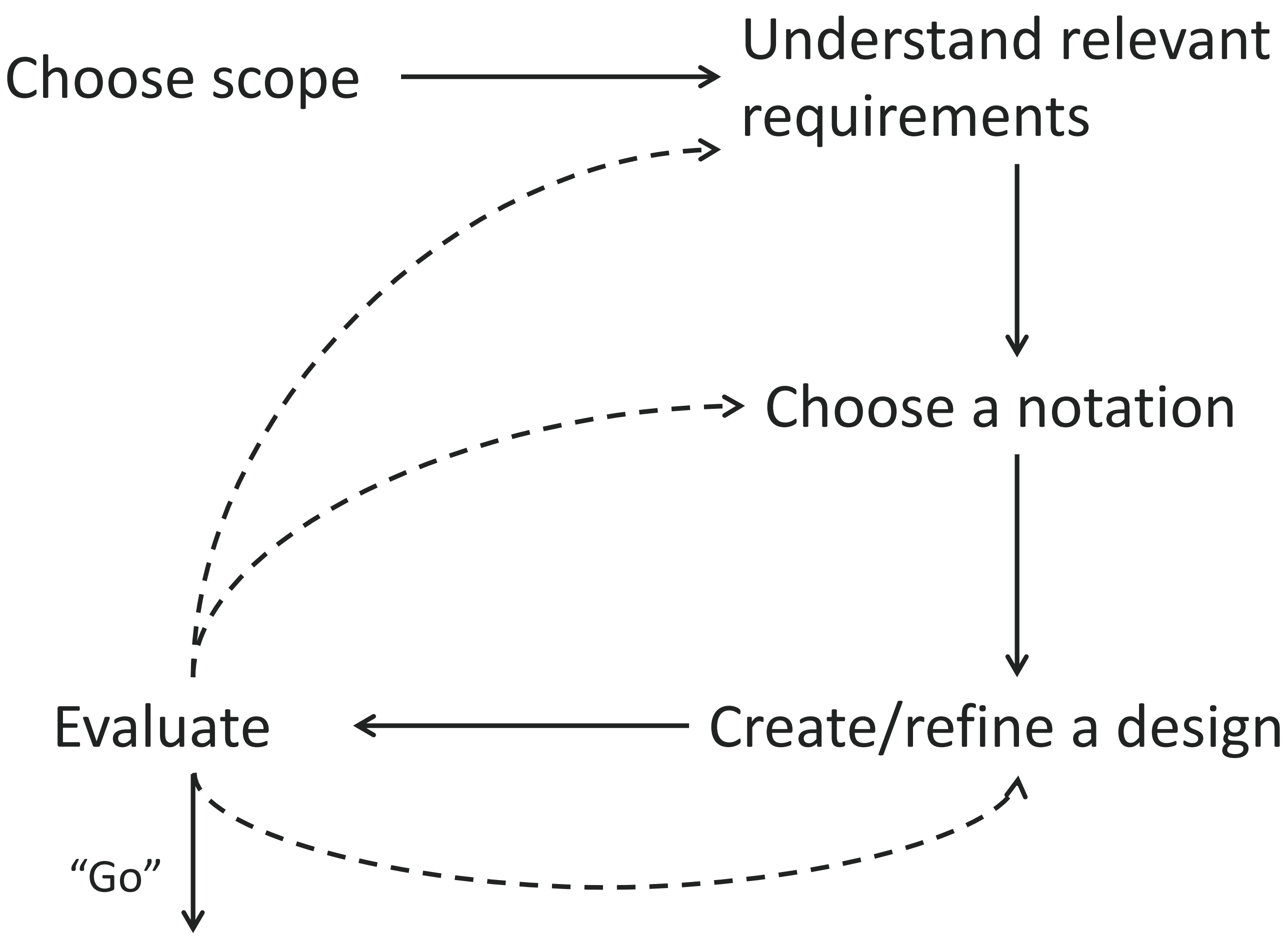
institute for SOFTWARE RESEARCH

Architecture design process

- Choose part or whole system to focus on
- Understand relevant requirements
- Choose a notation
  - Type of view, vocabulary of elements
- Create a design
  - Patterns, tactics
- Evaluate
- Go vs no-go
  - Issues feed back into process

institute for SOFTWARE RESEARCH

Architecture design process



Source: ACDM, ADD

Architectural decisions

- Heart of architecture – deciding which path to go
- Involve tradeoff analysis
- Representing the alternatives clearly – half of work

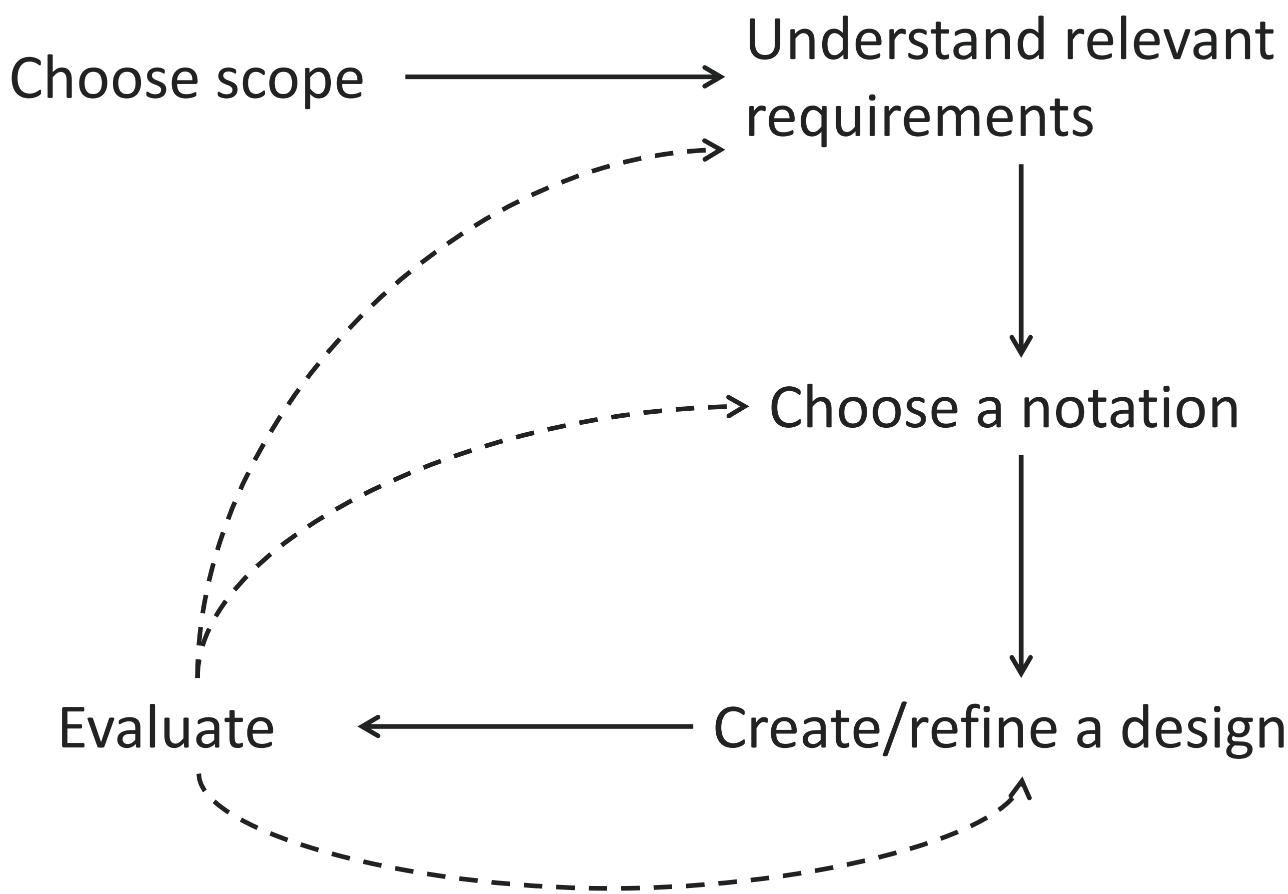
Architectural decisions

- Software architecture is *design*

  "Engineering design is [...] a decision-making process (often iterative), in which the basic sciences, mathematics, and engineering sciences are applied to convert resources optimally to meet a stated objective" – ABET

- A decision is a step in the process
  - **Record rationale!** (not just diagrams)
  - Tradeoffs
  - Backtracking

Architecture design process



Source: ACDM, ADD

Architectural decisions

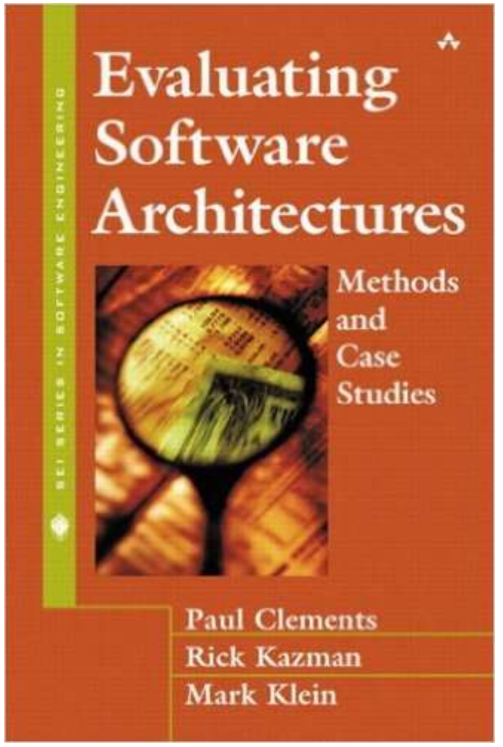
- Software architecture is *design*

  "Engineering design is […] a decision-making process (often iterative), in which the basic sciences, mathematics, and engineering sciences are applied to convert resources optimally to meet a stated objective" – ABET

- A decision is a step in the process
  - **Record rationale!** (not just diagrams)
  - Tradeoffs
  - Backtracking

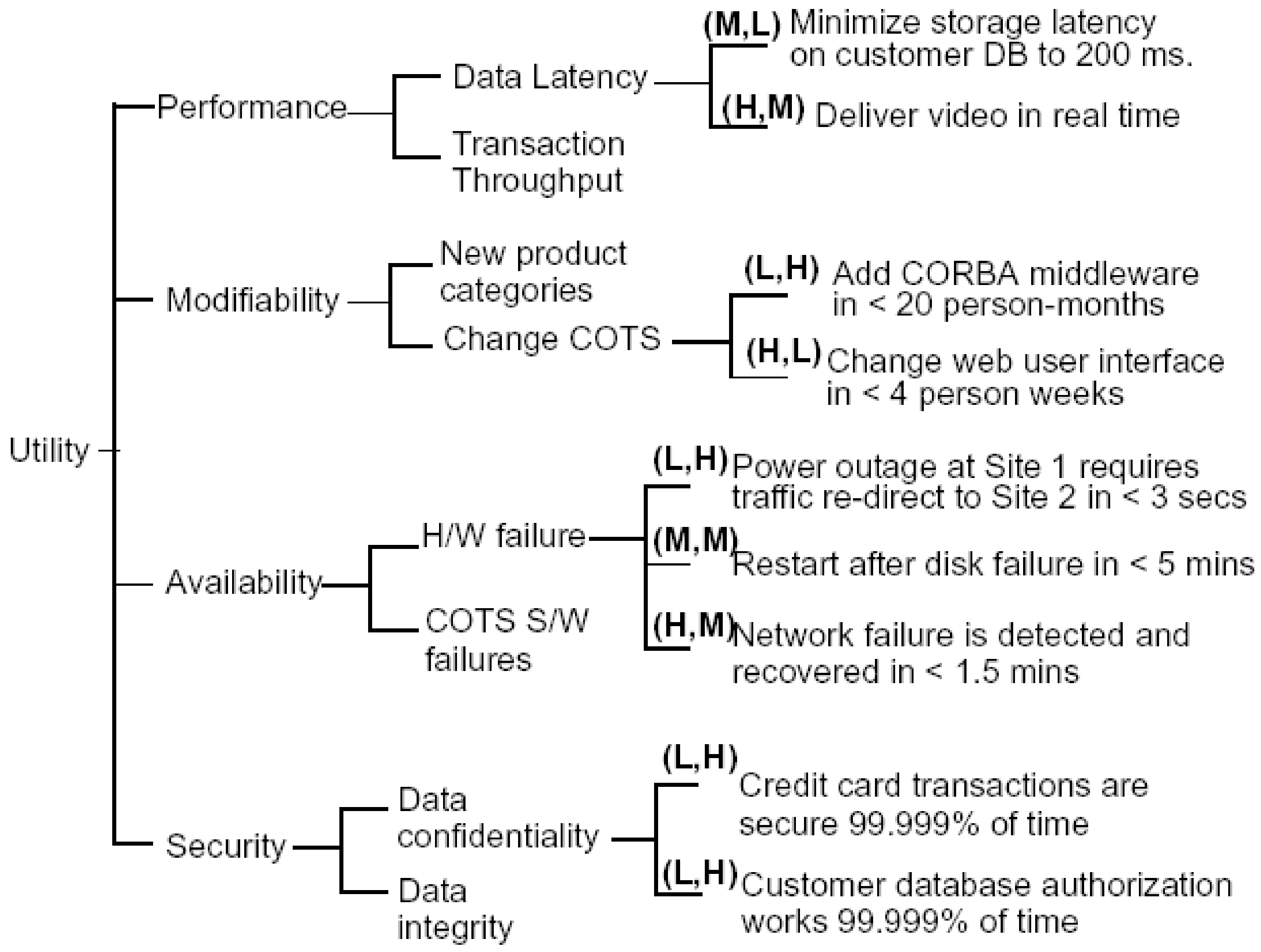
Architecture evaluation

- Goal: does the architecture satisfy requirements?
- ATAM – Architecture Tradeoff Analysis Method
  - Present requirements
  - Present architecture
  - Analyze architecture
  - Present results – risks and non-risks



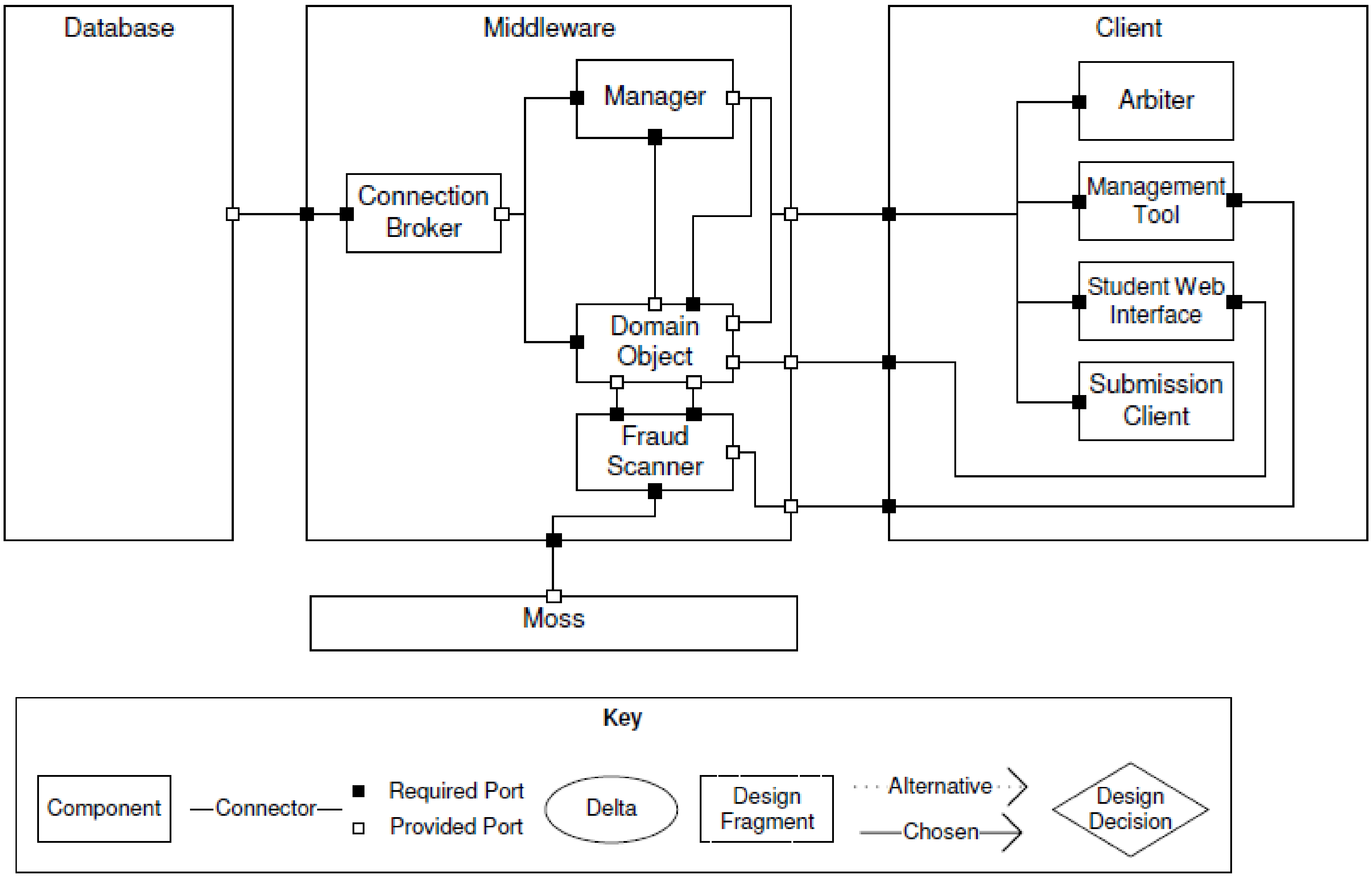
institute for SOFTWARE RESEARCH

Source:sei.cmu.edu

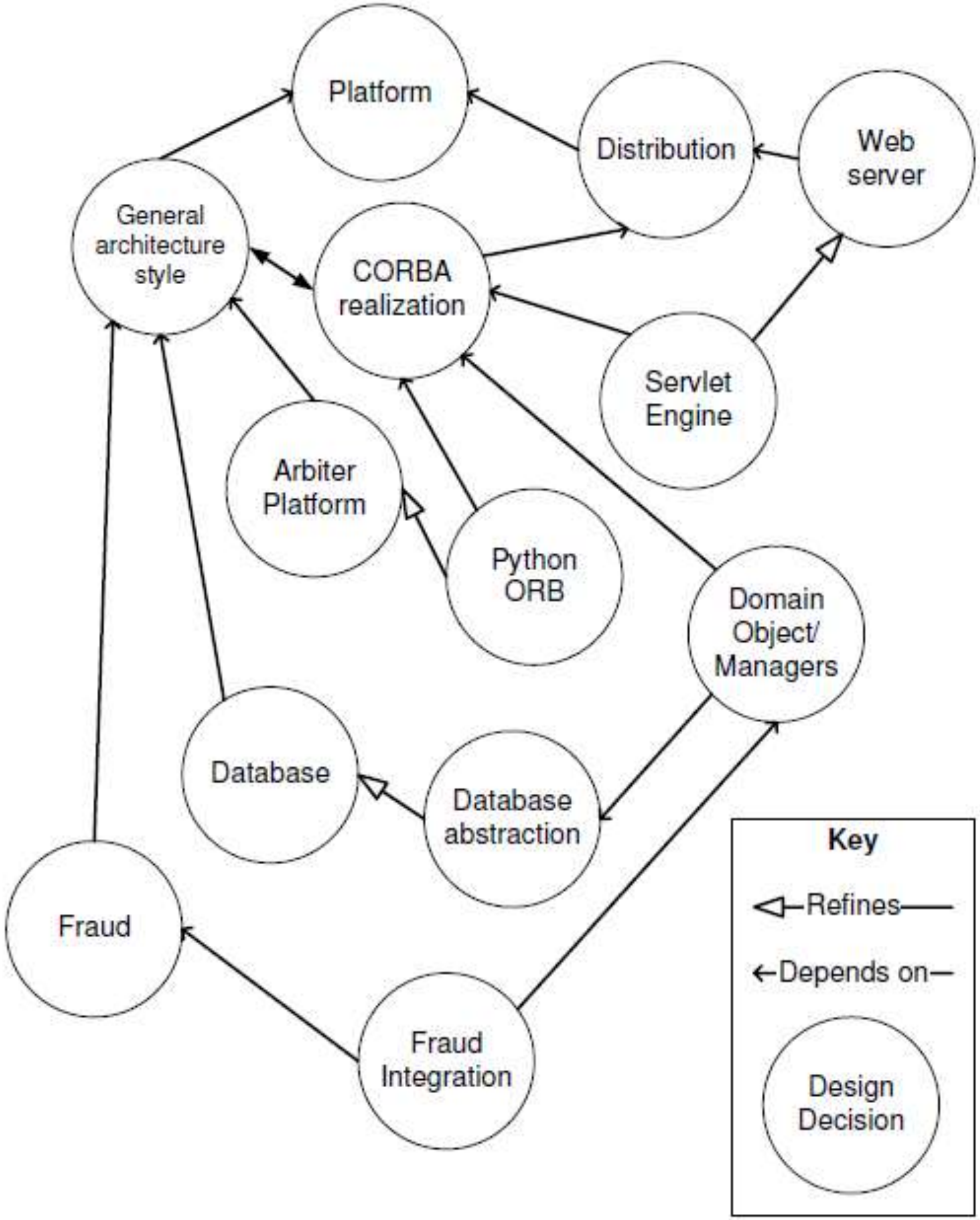


Source:arnon.me

Athena – code review system

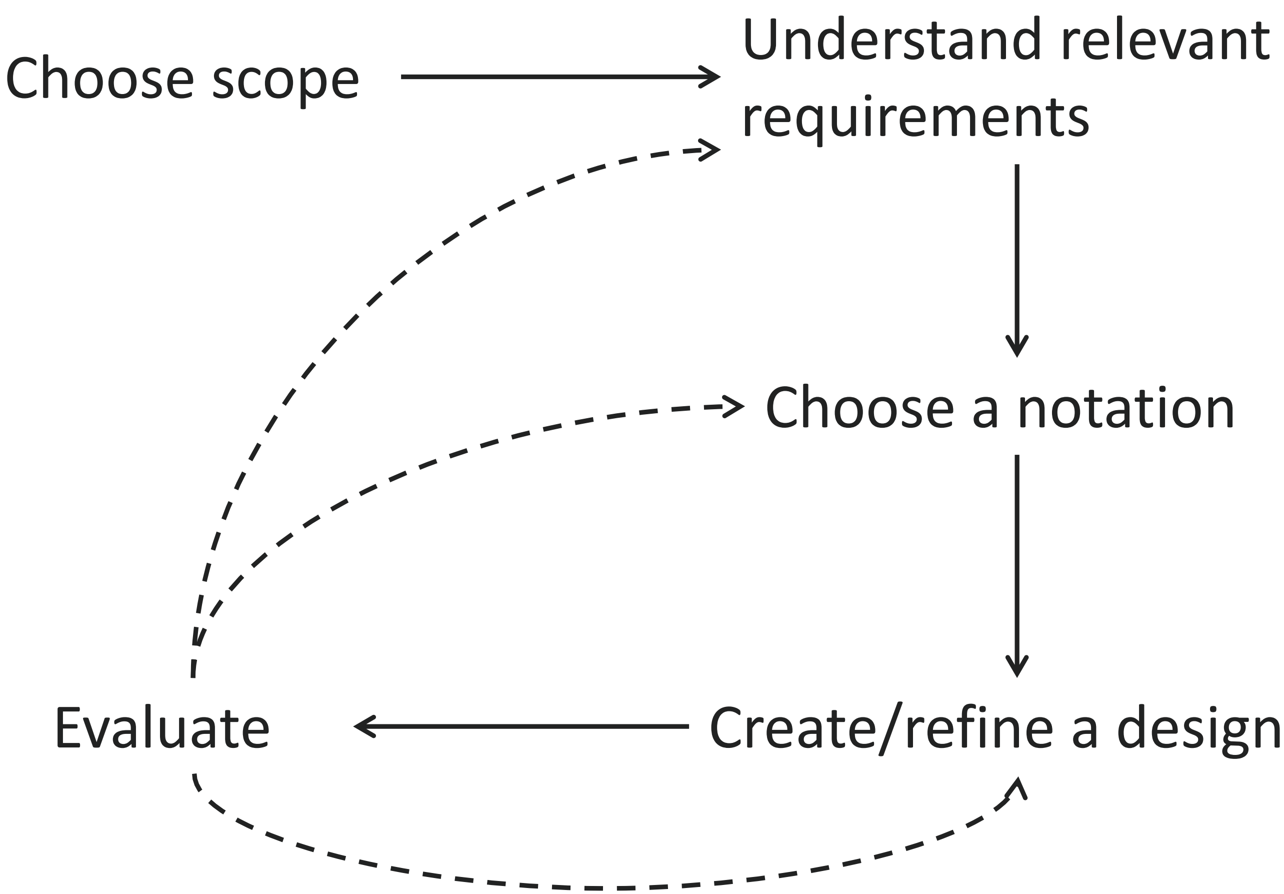


Source: Jansen and Bosch 2005



Source: Jansen and Bosch 2005

Architecture design process



Source: ACDM, ADD

Challenges of architecting

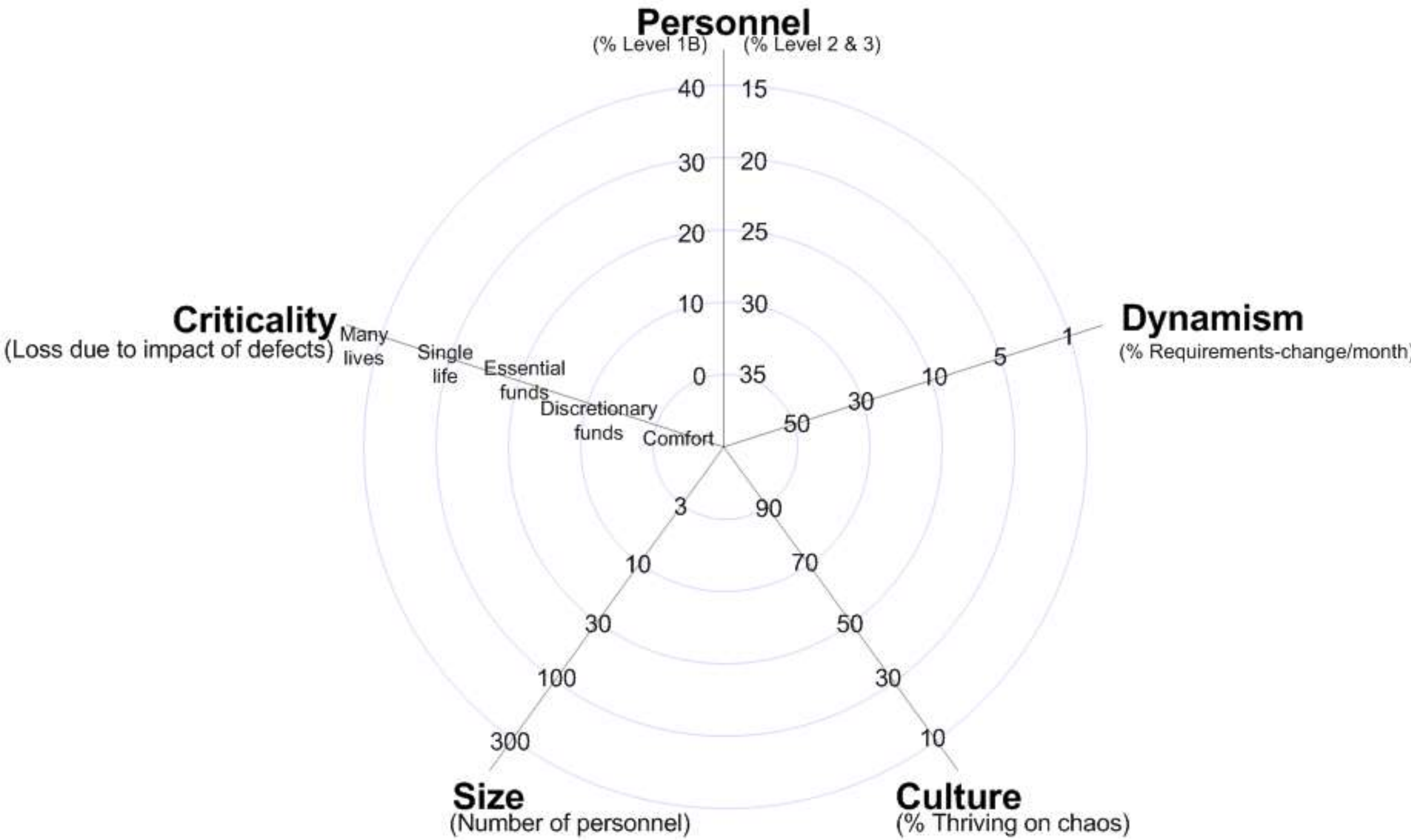
- Describe the system that is not built yet
- Domain knowledge is essential
- Huge space of options
- Heavily reliant on judgment

How much architecture?

- Design and document when needed, based on risk
- When:
  - Beginning
  - Whenever circumstances change
- Agile

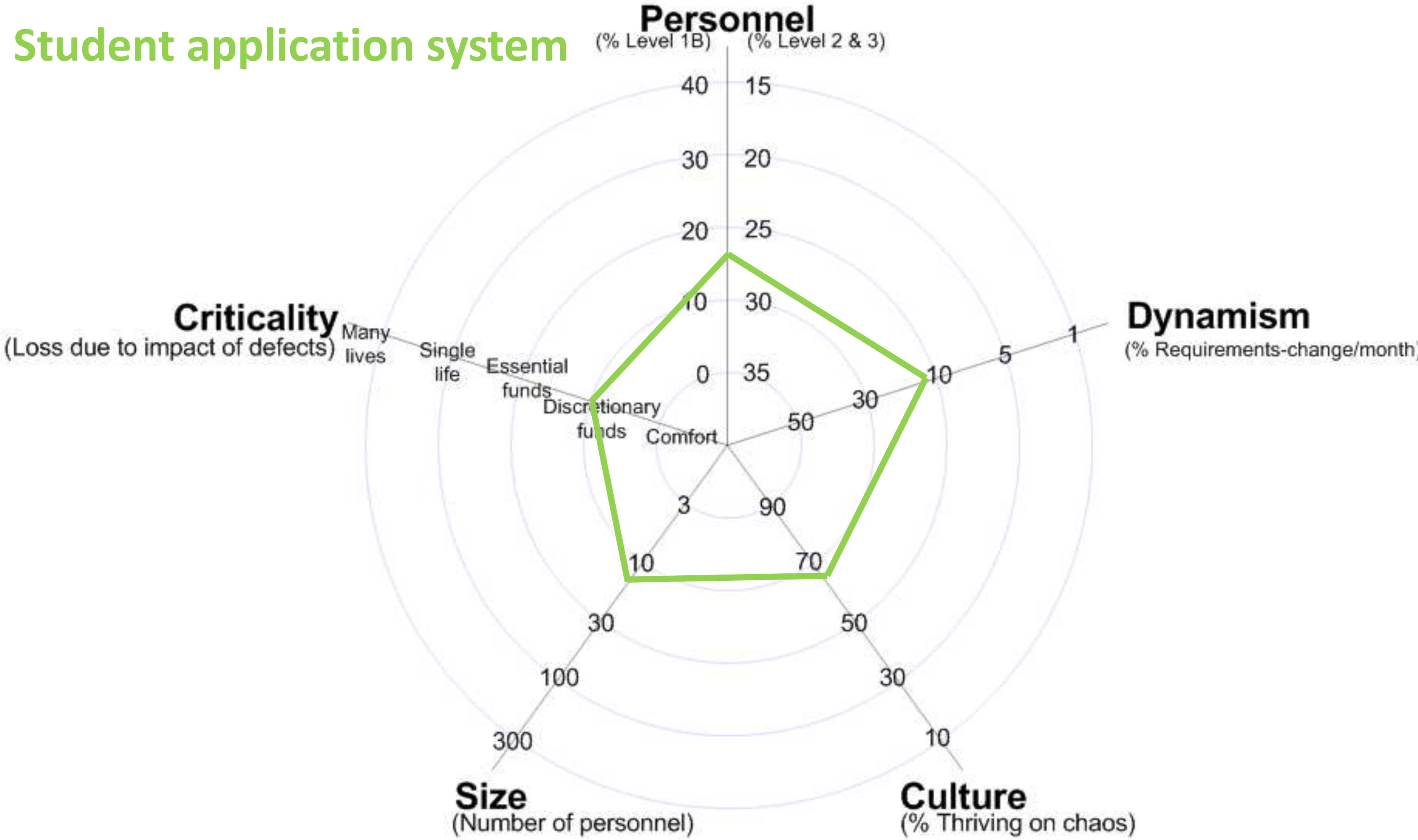
How much architecture?

- YAGNI
- Risk
- When to start:
  - Before implementation
  - Circumstances change
- When to stop:
  - Well-defined, requirements addressed, passes evaluation



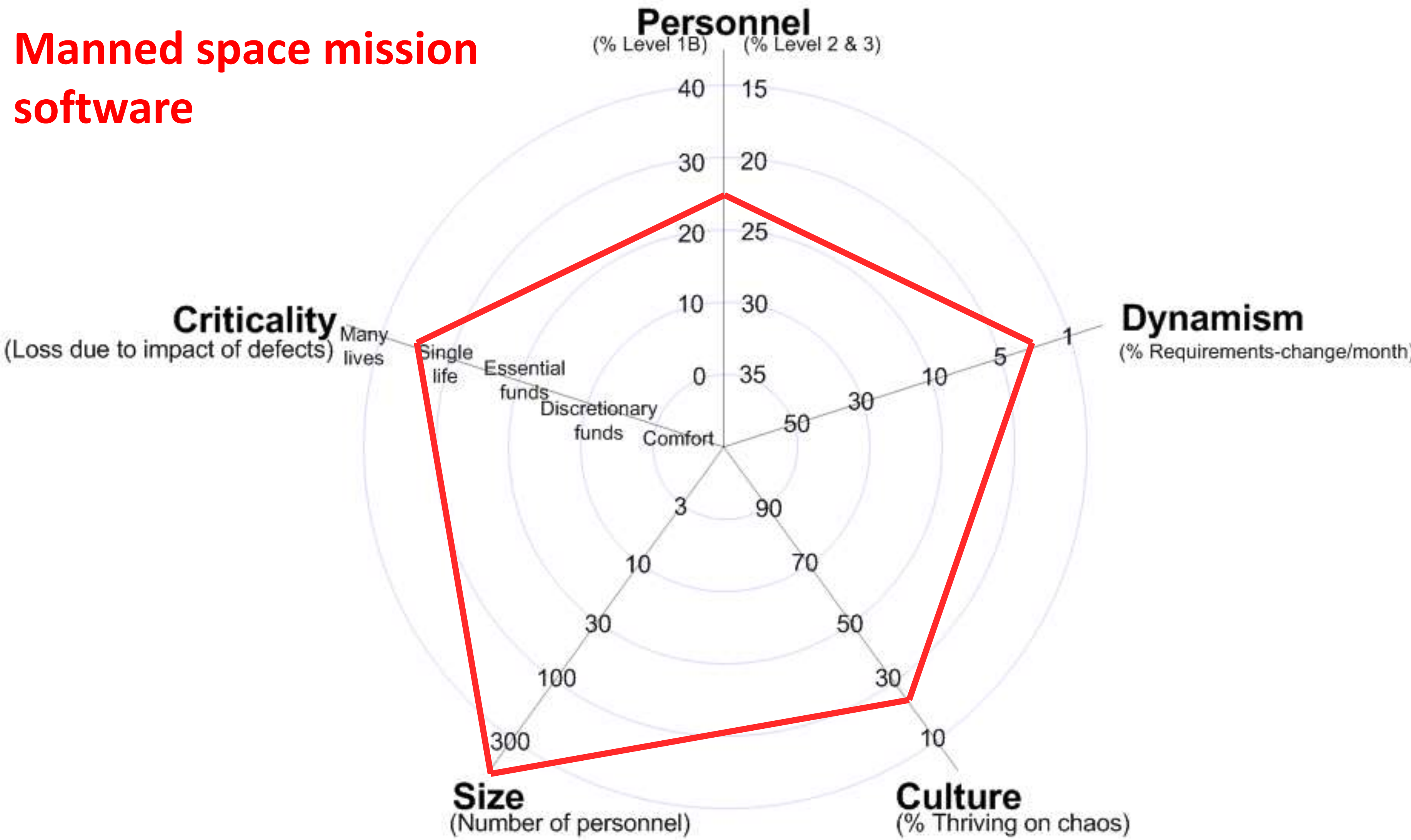
Source: Boehm and Turner 2003

Student application system



Source: Boehm and Turner 2003

Manned space mission software



Source: Boehm and Turner 2003

Challenges of architecting

- Describe the system that is not built yet
- Domain knowledge is essential
- Huge space of options
- Heavily reliant on judgment

Summary

Architecture as
structures and relations
- Patterns
- Tactics

Architecture as
documentation
- Views
- Rationale

Architecture as process
- Decisions
- Evaluation
- Reconstruction
- Agile

References

- Bass, Clements, Kazman. Software Architecture in Practice, 2013.
- Lattanze. Architecting Software Intensive Systems: a Practitioner’s Guide, 2009.
- Clements, Bachmann, Bass, Garlan, Ivers, Little, Merson, Nord, Stafford. Documenting Software Architectures: Views and Beyond, 2010.
- Boehm and Turner. Balancing Agility and Discipline: A Guide for the Perplexed, 2003.
- Jansen and Bosch. Software Architecture as a Set of Architectural Design Decisions, WICSA 2005.

Further Readings

- Bass, Clements, Kazman. Software Architecture in Practice, 2013.
- Lattanze. Architecting Software Intensive Systems: a Practitioner's Guide, 2009.
- Clements, Bachmann, Bass, Garlan, Ivers, Little, Merson, Nord, Stafford. Documenting Software Architectures: Views and Beyond, 2010.
- Boehm and Turner. Balancing Agility and Discipline: A Guide for the Perplexed, 2003.
- Jansen and Bosch. Software Architecture as a Set of Architectural Design Decisions, WICSA 2005.