Information Systems Architecture
Stakeholders, Viewpoints, Perspectives

Eoin Woods
Barclays Global Investors

Content

Defining Software Architecture

- A common definition …

  The software architecture of a program or computing system is the **structure or structures** of the system, which comprise software **elements** the externally visible **qualities** of those elements, and the **relationships** among them

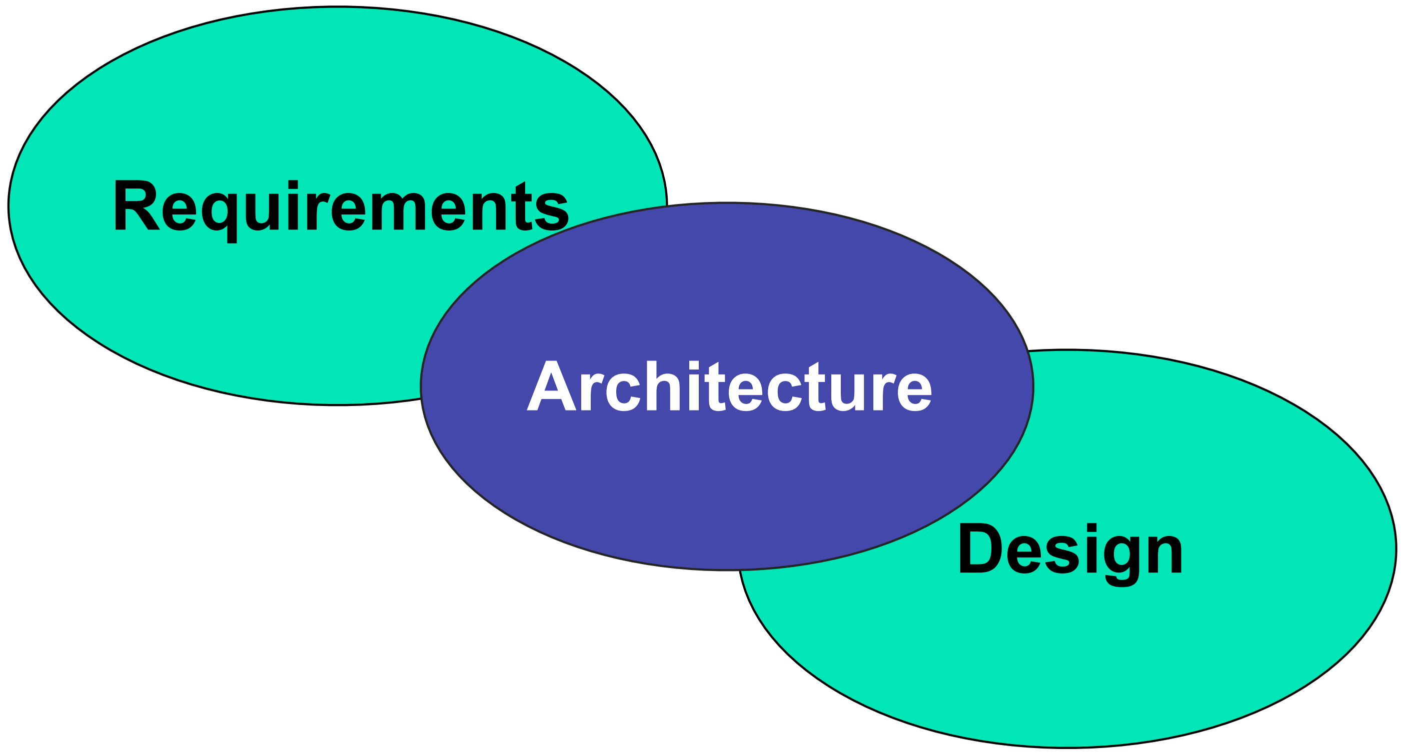
  Len Bass, Paul Clements and Rick Kazman (SEI)
  Software Architecture in Practice, 2nd Edition

Defining Software Architecture

- Alternative definitions …
  - The set of system design decisions that dictate the fundamental structure and properties of a system Thus, the set of decisions that will cause the system to fail if made incorrectly
  - The set of design decisions that, if made wrongly, cause your project to be cancelled!

Role of Software Architecture

The bridge between requirements and design



Architecture & Requirements

- Requirements are an input to architecture
  - Requirements frame the architectural problem
  - Stakeholder needs and desires
- Architecture must influence requirements
  - “The art of the possible”
  - Stakeholder understanding of risk/cost
  - Stakeholder understanding of possibilities

Architecture & Design

- Architecture frames design
  - architecture is part of the design process
- Captures the system-wide decisions
  - what has to be consistent or constant
- Importance of role increases with scale
- Perfectly compatible with agile
  - even XP with the right approach

Just Design, Surely?

Architecture	Global	Intentional
Design	Local	Intentional
Implementation	Local	Extensional

"*Architecture, Design, Implementation*",
Amnon Eden And Rick Kazman, ICSE 2003

Intentional: infinitely many possible ways of satisfying the statement (i.e. constraint rather than instruction)
Local: satisfied in design "d" => satisfied in all possible extensions of "d"

Quality Properties

- Non-functional characteristics (“-illities”)
  - Performance, Security, Availability, …
- Often crucial to stakeholders
  - Slow functions don’t get used
  - Unavailable systems stop the business
  - Security problems cause headlines
- Yet often an after-thought

Quality Properties

- Addressing quality properties is a key architectural task
  - Understanding real stakeholder needs
  - Understanding what is possible
  - Making the key trade-offs to allow delivery
  - Avoiding expensive “retro-fit”

Content

- Defining Software Architecture
- Stakeholders
- The Software Architecture Problem
- Viewpoints to Guide Structure
- Perspectives to Guide Qualities
- Example Application
- Uses for Viewpoints and Perspectives

Stakeholders

- Identifying Stakeholders
  - People, Groups, Entities
  - Those who have an interest in or concerns about the realisation of the architecture
- Importance of Stakeholders
  - Architectures are built for stakeholders
  - Decisions must reflect stakeholder needs
  - Involving a wide stakeholder community increases your chances of success

Talking Point: Stakeholders

- Who cares whether your systems get built?
- Why do they care?
- Are they positively or negatively impacted?

Create a list of the stakeholders for your system

Stakeholders

- Attributes of a good stakeholder
  - Informed, to allow them to make good decisions
  - Committed, to the process and willing to make themselves available in a constructive manner, even if decisions are hard
  - Authorised, to make decisions
  - Representative, of their stakeholder group so that they present its views validly

Stakeholder Groups

- Acquirers pay for the system
- Assessors check it for compliance
- Communicators create documents and training
- Developers create it
- Maintainers evolve and fix the system
- Suppliers provide system components
- Support Staff help people to use the system
- System Administrators keep it running
- Testers verify that it works
- Users have to use the system directly

Content

The Challenge

- Essential difficulties
  - Multi-dimensional problem
  - Highly complex mix of people and technology
  - Diverse stakeholder community to serve
  - Making trade-offs is essential but hard
  - Often no “right” answer

The Challenge

- Accidental difficulties
  - Little standardisation in description
  - Difficult to compare and discuss alternatives
  - Little standardisation in architectural activities
  - Little sharing of proven practice and known problems and their solutions
  - No framework for handling quality properties

Meeting the Challenge

- Organise the architectural design process
  - roles & activities, relationship to requirements & design
- Define the use of architecture artefacts
  - which models? when? why?
- Capture, classify and share knowledge
  - best practice, problems and pitfalls, proven solutions

Content

- Defining Software Architecture
- Stakeholders
- The Software Architecture Problem
- Viewpoints to Guide Structure
- Perspectives to Guide Qualities
- Example Application
- Uses for Viewpoints and Perspectives

Architectural Viewpoints

- Help to deal with architectural structure
- Decompose architectural description into views
  - each view addresses one aspect of the system
  - functional view, deployment view, …
- Guide development of views via viewpoints
  - viewpoint contains proven practice, pitfalls, …
  - each view defined by one viewpoint
- Organises the process and the artefacts

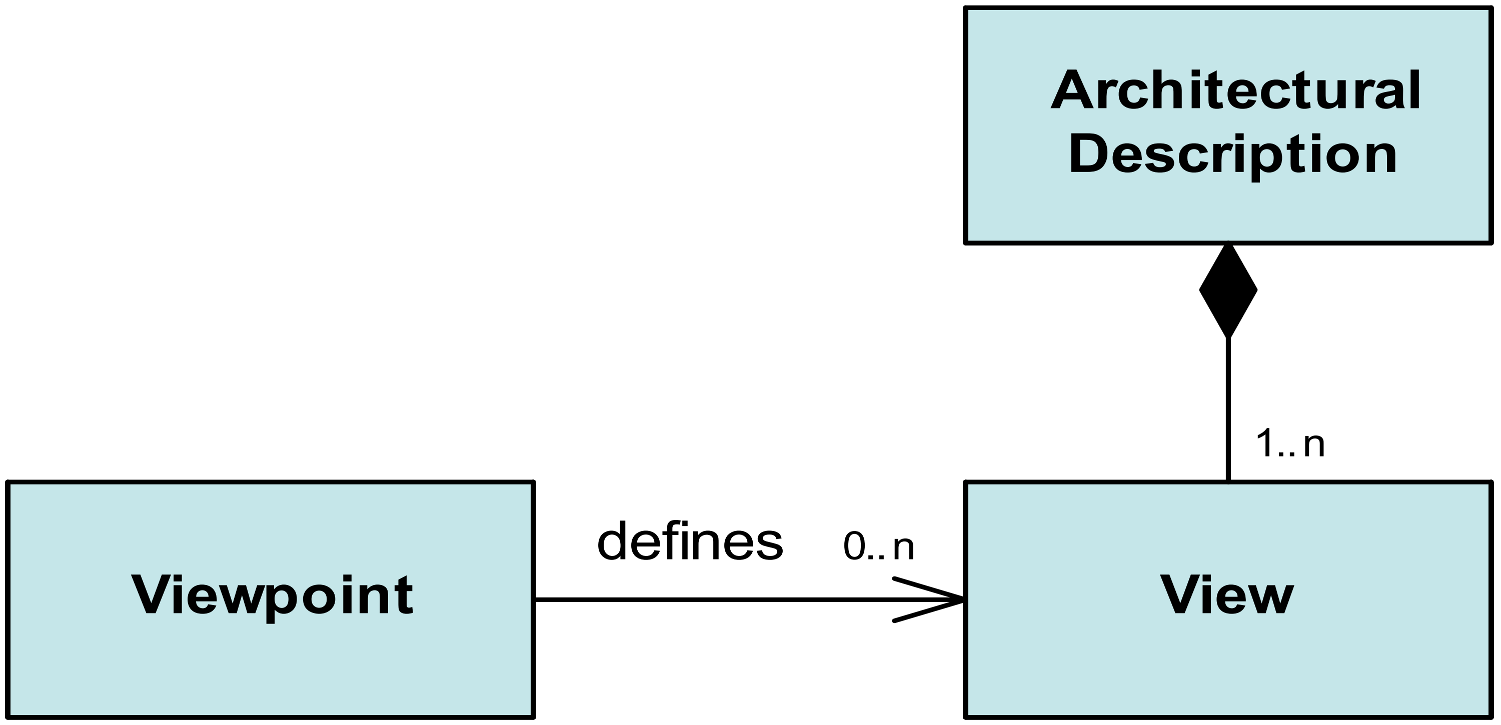
Architectural Viewpoints

- Well understood, widely applied, many exist
  - RUP/Kruchten “4+1” set (1995)
  - RM-ODP set (1995)
  - Siemens set (1999)
  - Garland and Anthony set (2003)
  - Rozanski & Woods set (2005)
  - Conceptual basis in IEEE 1471 (2000)

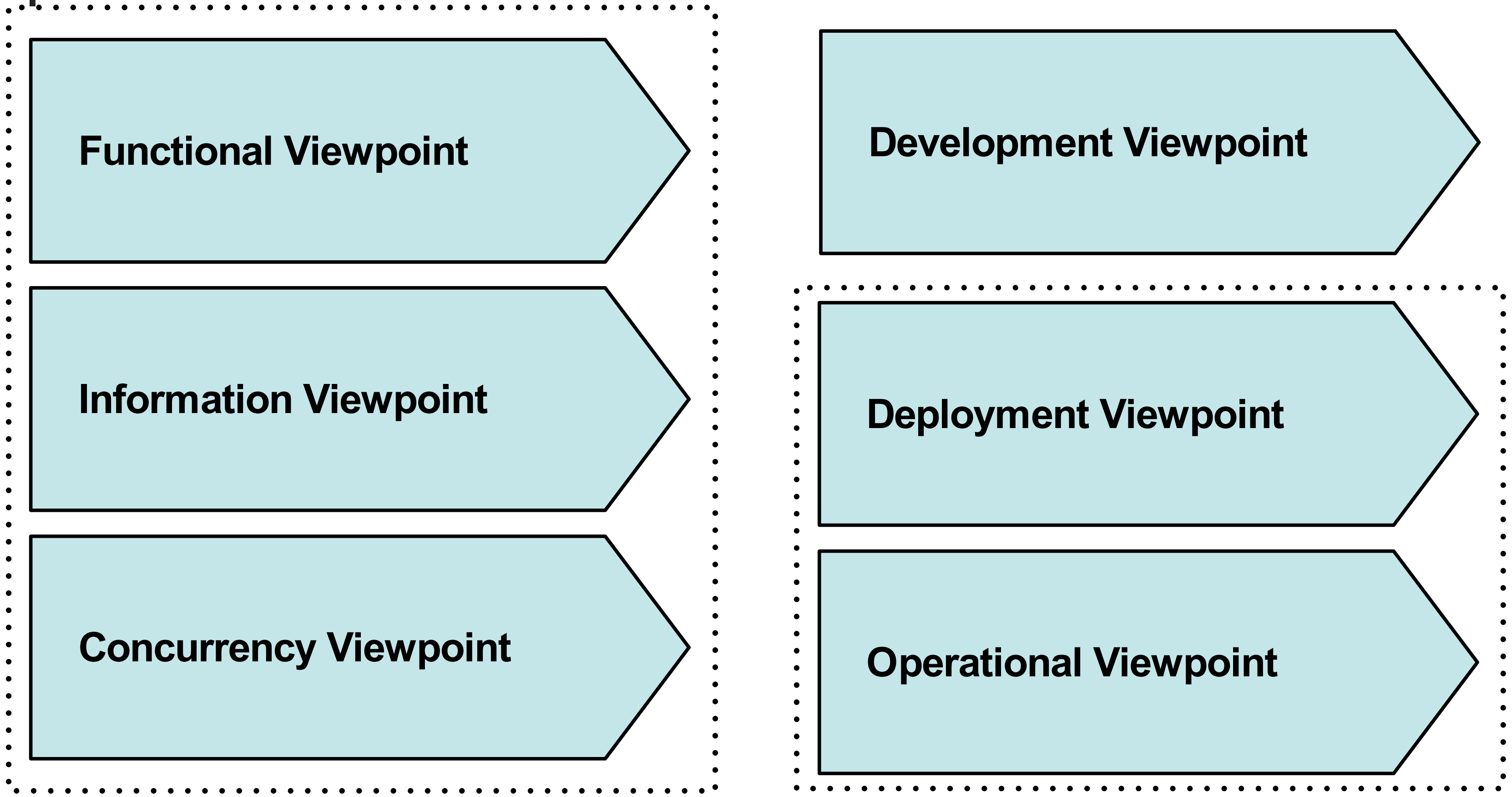
Viewpoints and Views

- IEEE 1471 provides standard definitions
  - A viewpoint is a collection of patterns, templates and conventions for constructing one type of view. It defines the stakeholders whose concerns are reflected in the viewpoint, and guidelines and principles and template models for constructing its views.
  - A view is a representation of all or part of an architecture, from the perspective of one or more concerns which are held by one or more of its stakeholders.
  - from IEEE Standard 1471 – Recommended Practice for Architectural Description (2000)

Viewpoints and Views



Example Viewpoint Set



[Rozanski & Woods, 2005]

Example Viewpoint Set

- Core architectural structures
  - Functional
    - elements, connectors, interfaces, responsibilities, interactions
  - Information
    - entities, constraints, relationships, timeliness, usage, ownership
  - Concurrency
    - processes, threads, coordination, element to process mapping

Example Viewpoint Set

- Building the system
  - Development
    - layers, module structure, standard design, codeline
- Moving towards deployment
  - Deployment
    - hardware, network, software dependencies, process to node mapping
  - Operational
    - installation, migration, administration, support

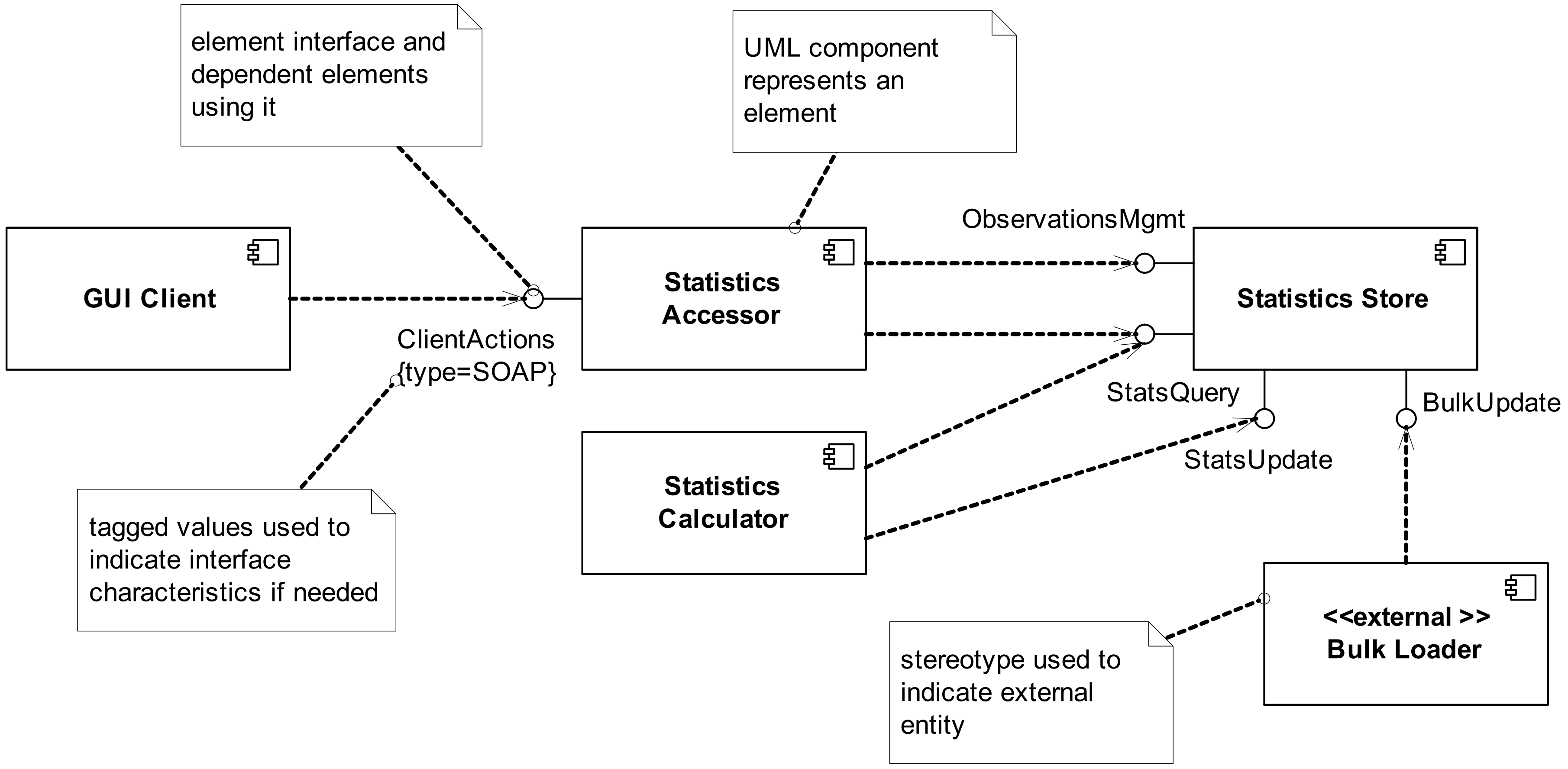
Example Viewpoint Set

- Rozanski/Woods Viewpoint Set
- Aimed at large scale information systems
- Extension and refinement of the “4+1” set
  - renamed “Logical”, “Process” and “Physical”
  - added “Information” and “Operational”
- Standard content for viewpoints
  - applicability, concerns, models, stakeholders, problems & pitfalls, solutions, checklists

Functional Viewpoint

Focus	functional structure of the system
Content	design of runtime functional elements and their responsibilities, interfaces, and primary interactions
Concerns	▪ functional capabilities ▪ external interfaces ▪ internal structure ▪ design qualities
Models	▪ functional structure model
Pitfalls	▪ poorly defined interfaces / responsibilities ▪ infrastructure modelled as functional elements ▪ unintentionally overloaded view ▪ diagrams without element definitions […]

Functional View Fragment


element interface and dependent elements using it
UML component represents an element
GUI Client
Statistics Accessor
ObservationsMgmt
Statistics Store
ClientActions {type=SOAP}
StatsQuery
BulkUpdate
Statistics Calculator
StatsUpdate
tagged values used to indicate interface characteristics if needed
<<external >> Bulk Loader
stereotype used to indicate external entity

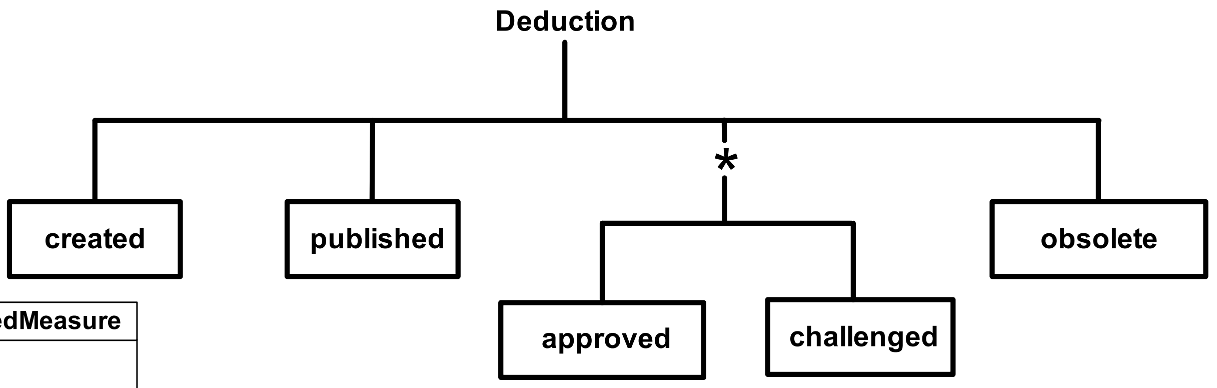
Information Viewpoint

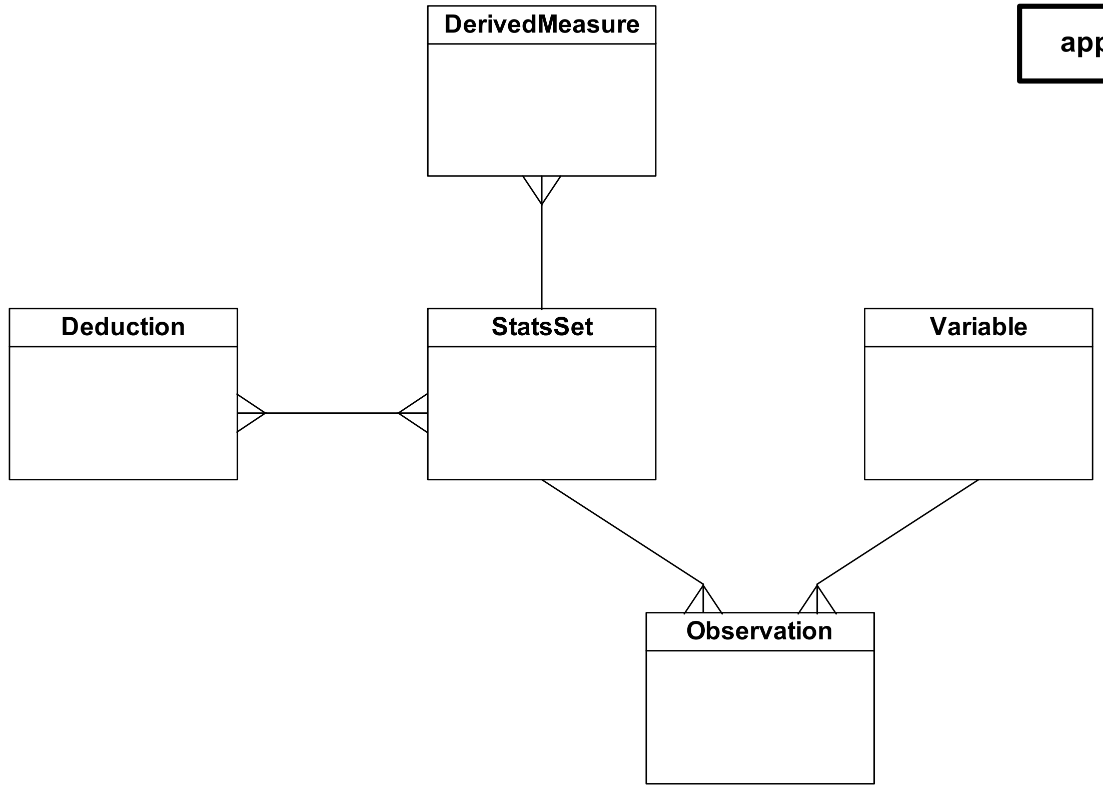
Focus	information structure, ownership and processing
Content	design of storage, manipulation, management, and distribution of information
Concerns	▪ information structure and content ▪ information flow ▪ data ownership and quality ▪ timeliness, latency, and age ▪ references and mappings ▪ transaction management and recovery ▪ data volumes ▪ archives and data retention ▪ regulation

Information Viewpoint (ii)

Models	▪ static data and metadata structure models ▪ information flow models ▪ information lifecycle models ▪ data ownership and access models ▪ volumetric models
Pitfalls	▪ data incompatibilities ▪ poor data quality ▪ unavoidable multiple updaters ▪ key matching deficiencies ▪ poor information latency ▪ interface complexity ▪ inadequate volumetrics

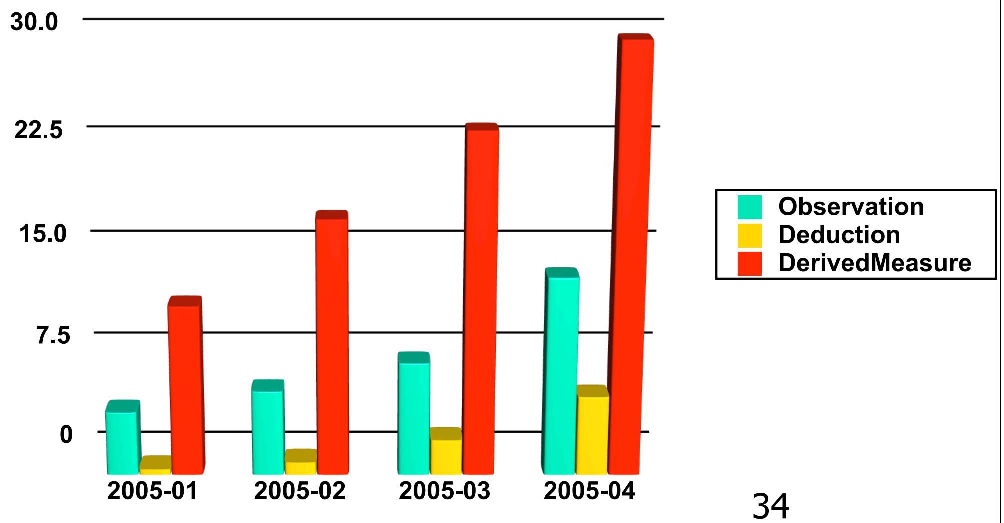
Information View Fragments





Information View Fragments (ii)

	Observation	Deduction	Derived Measure
Statistics Accessor	R	C,R,U,D	R
Statistics Calculator	-	-	C,U,D
Bulk Loader	C,U,D	-	-



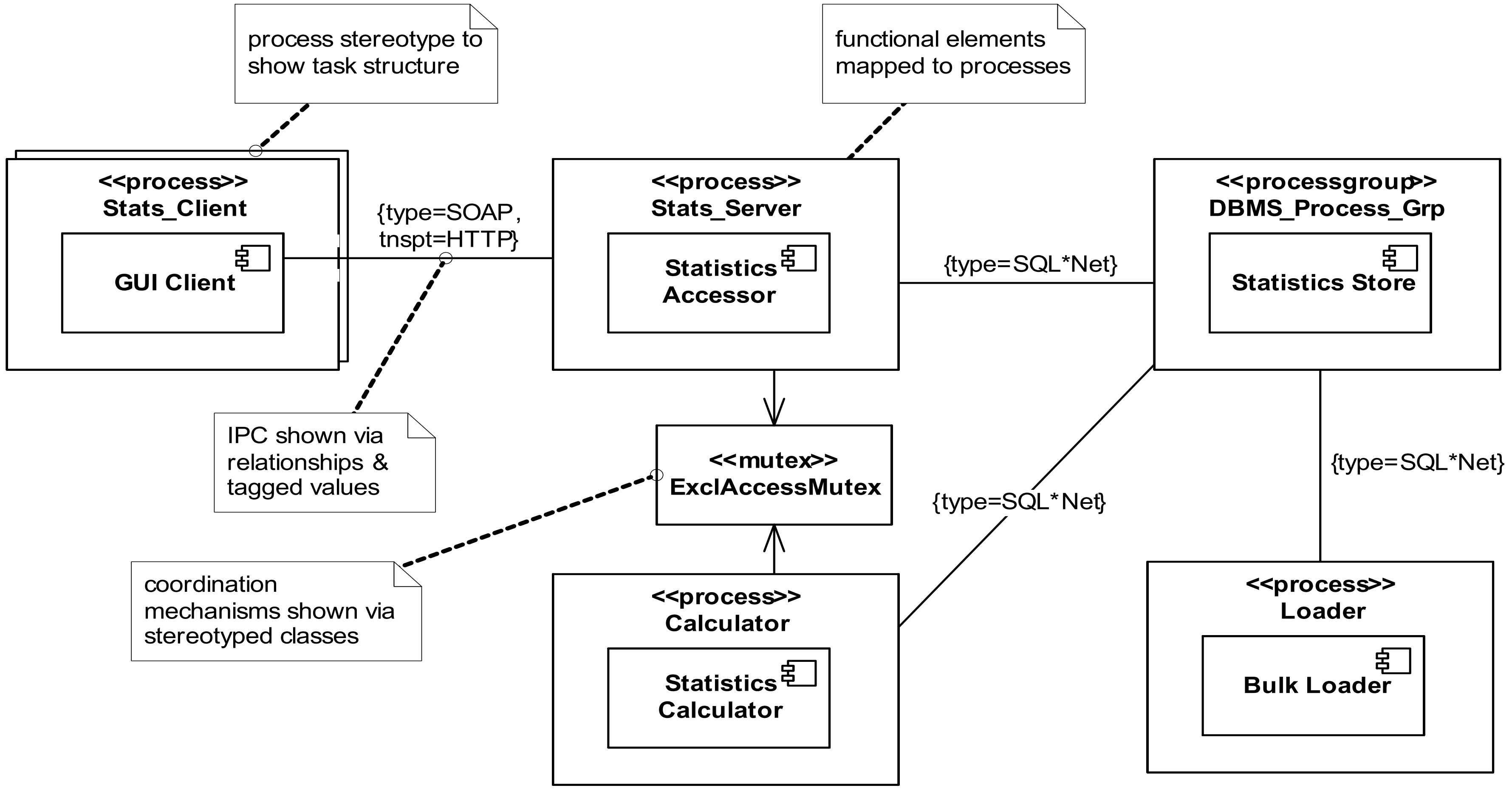
Concurrency Viewpoint

Focus	packaging elements into processes and threads
Content	the concurrency structure, mapping functional elements to concurrency units to clearly identify the parts of the system that can execute concurrently, and how this is coordinated and controlled
Concerns	▪ task structure ▪ mapping of functional elements to tasks ▪ inter-process communication & re-entrancy ▪ state management ▪ synchronization and integrity ▪ task startup, shutdown and recovery from failure

Concurrency Viewpoint (ii)

Models	▪ system-level concurrency model ▪ system-level state model
Pitfalls	▪ modelling of the wrong concurrency ▪ excessive complexity ▪ resource contention ▪ deadlock and race conditions

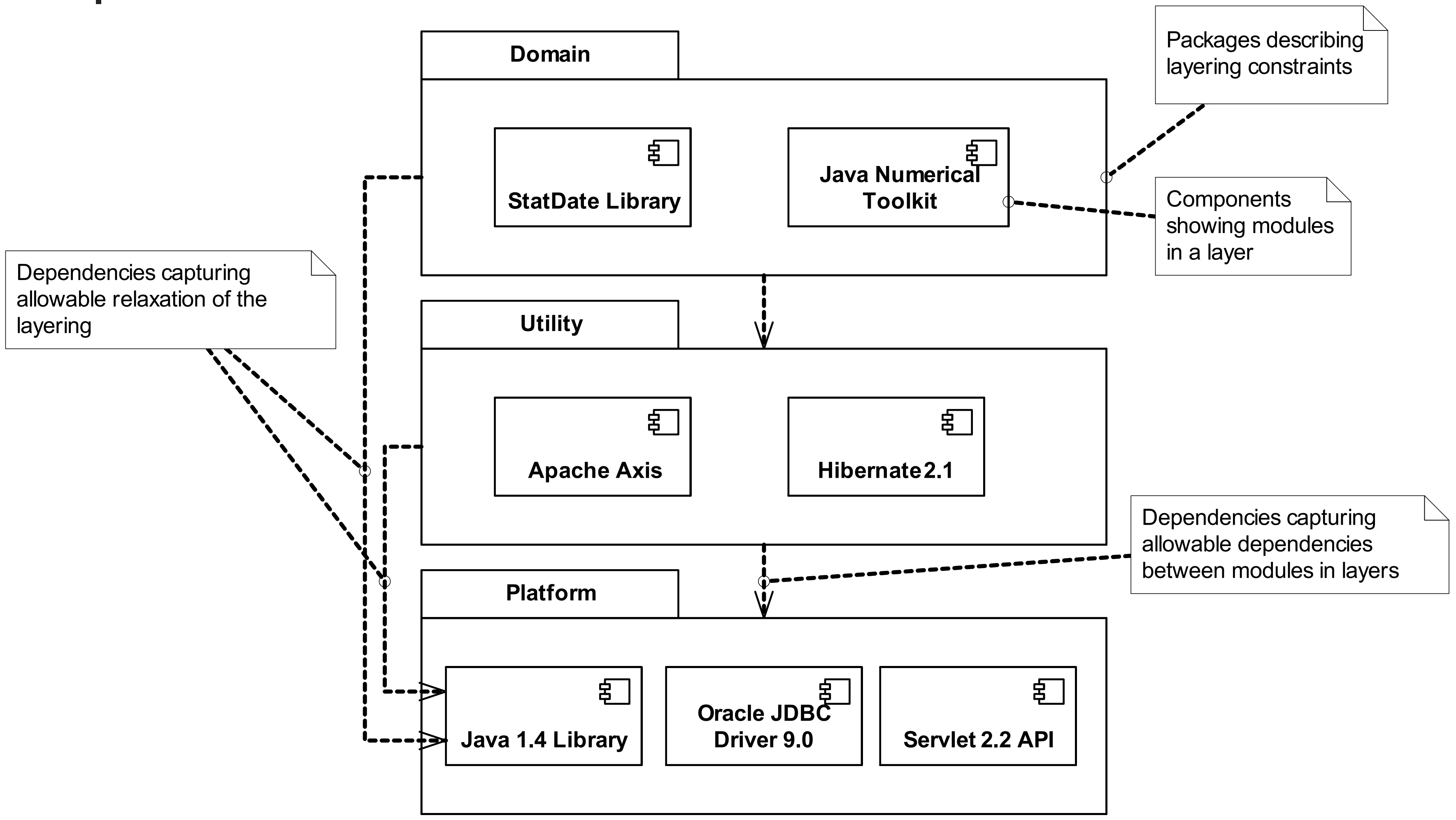
Concurrency View Fragment


process stereotype to show task structure
functional elements mapped to processes
<<process>> Stats_Client
GUI Client
{type=SOAP, tnspt=HTTP}
<<process>> Stats_Server
Statistics Accessor
{type=SQL*Net}
<<processgroup>> DBMS_Process_Grp
Statistics Store
IPC shown via relationships & tagged values
<<mutex>> ExclAccessMutex
{type=SQL*Net}
{type=SQL*Net}
coordination mechanisms shown via stereotyped classes
<<process>> Calculator
Statistics Calculator
<<process>> Loader
Bulk Loader

Development Viewpoint

Focus	architectural constraints on the development process
Content	architectural design that supports and constraints the software development process
Concerns	▪ module organization ▪ codeline organization ▪ common processing ▪ standardization of design and testing ▪ instrumentation
Models	▪ module structure models ▪ common design models ▪ codeline models
Pitfalls	▪ too much detail or lack of precision ▪ overburdening the architectural description ▪ uneven focus or lack of developer focus ▪ problems with the specified environment

Development View Fragment


Domain
StatDate Library
Java Numerical Toolkit
Utility
Apache Axis
Hibernate 2.1
Platform
Java 1.4 Library
Oracle JDBC Driver 9.0
Servlet 2.2 API
Packages describing layering constraints
Components showing modules in a layer
Dependencies capturing allowable relaxation of the layering
Dependencies capturing allowable dependencies between modules in layers

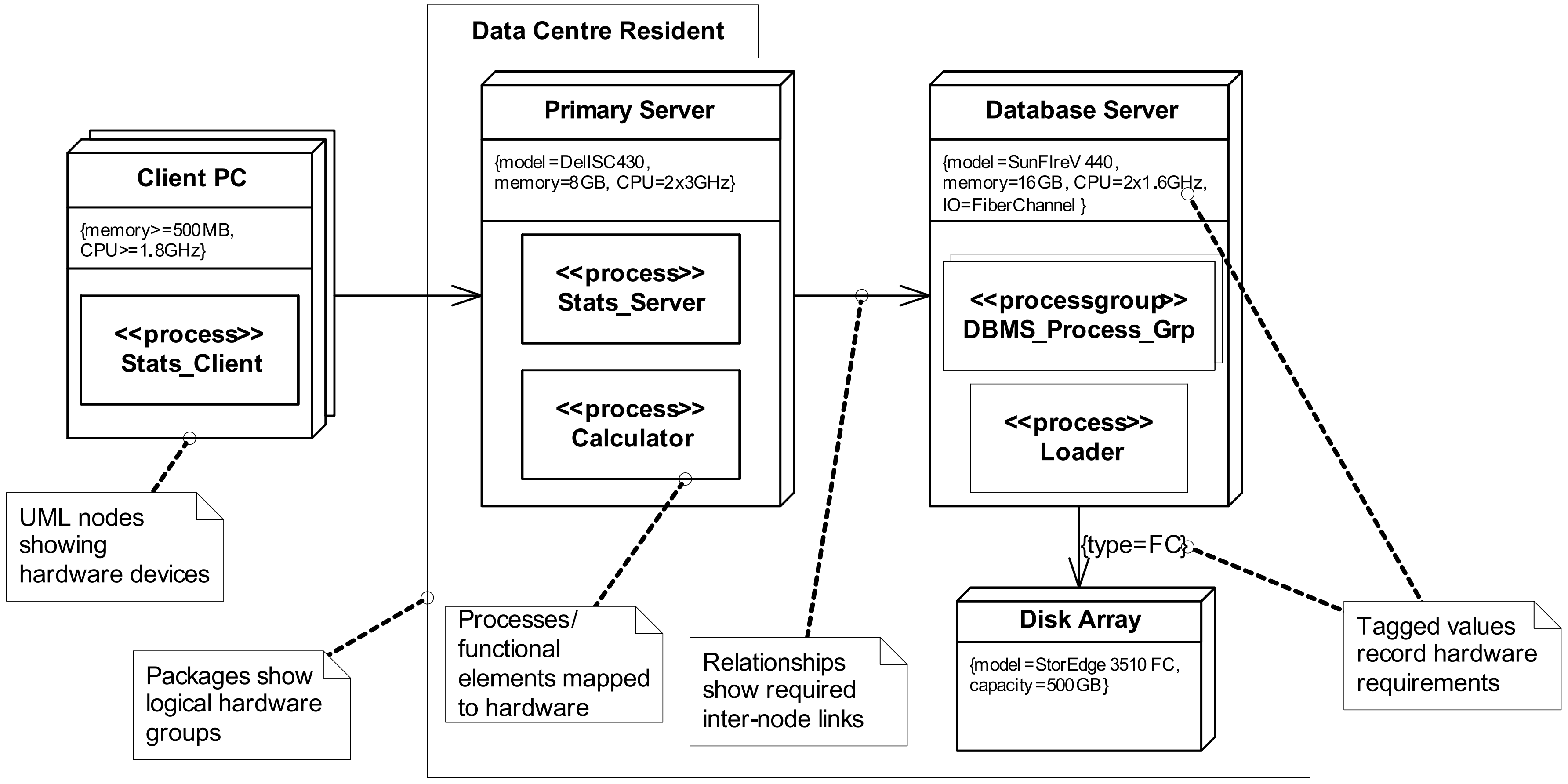
Deployment Viewpoint

Focus	runtime environment structure and the distribution of software across it
Content	design of the environment into which the system will be deployed, including the system's runtime dependencies
Concerns	▪ types of hardware required ▪ specification and quantity of hardware required ▪ third-party software requirements ▪ technology compatibility ▪ network requirements ▪ network capacity required ▪ physical constraints

Deployment Viewpoint (ii)

Models	▪ runtime platform models ▪ network models ▪ technology dependency models
Pitfalls	▪ unclear or inaccurate dependencies ▪ unproven technology ▪ lack of specialist technical knowledge ▪ late consideration of the deployment environment

Deployment View Fragment


Data Centre Resident
Client PC
{memory>=500MB, CPU>=1.8GHz}
<<process>> Stats_Client
Primary Server
{model=DellSC430, memory=8GB, CPU=2x3GHz}
<<process>> Stats_Server
<<process>> Calculator
Database Server
{model=SunFIreV 440, memory=16GB, CPU=2x1.6GHz, IO=FiberChannel }
<<processgroup>> DBMS_Process_Grp
<<process>> Loader
{type=FC}
Disk Array
{model=StorEdge 3510 FC, capacity=500GB}
UML nodes showing hardware devices
Packages show logical hardware groups
Processes/ functional elements mapped to hardware
Relationships show required inter-node links
Tagged values record hardware requirements

Deployment View Fragment (ii)

Client PC	▪ Windows XP SP1 ▪ Java JRE 1.4.2_06 or later ▪ Internet Explorer 6.0 SP1
Primary Server	▪ Windows 2003 server, w/sec patches ▪ Java SDK 1.4.2_06 or later ▪ Apache Tomcat 5.5.9 or later
Database Server	▪ Solaris 9.0 w/Aug05 patch cluster ▪ Oracle 9.2.0.2 Std Edition ▪ 10GB buffer cache, auto sized SGA ▪ auto storage management, 2 table spaces ▪ OEM 9.2.0.2 installed and working

Operational Viewpoint

Focus	system installation, migration, operation & support
Content	defines strategies for how the system will be operated, administered, and supported when it is running in its production environment
Concerns	▪ installation and upgrade ▪ functional and data migration ▪ operational monitoring and control ▪ operational configuration management ▪ performance monitoring ▪ support responsibilities and procedures ▪ backup and restore

Operational Viewpoint (ii)

Models	▪ installation models ▪ migration models ▪ configuration management models ▪ administration models ▪ support and escalation models
Pitfalls	▪ lack of engagement with the operational staff ▪ lack of migration & backout planning ▪ insufficient migration window ▪ missing management tools ▪ lack of integration into the production environment ▪ inadequate backup and recovery modelling

Operational View Content

- Installation Model
  - Installation groups
  - Dependencies and constraints
  - Backout strategy
- Operational CM Model
  - Configuration groups and dependencies
  - Configuration parameter sets
  - Operational control (switching between sets)

Operational View Content (ii)

- Administration Model
  - Monitoring and control facilities required and provided
  - Required routine operational procedures
  - Required operational action in case of error conditions

Talking Point: Views

- Which views would be useful for your systems?
- Who would be interested in each view?
- Which views wouldn’t be useful – why?

Draw up a list of which views you would use

Who would be interested in each ?

Can you sketch fragments of one or two?

Viewpoints and Views Recap

- Viewpoints
  - A store of knowledge and experience
  - A guide to the architect
  - Templates to guide the process
- Views
  - A structure for description
  - A separation of concerns
  - Aid to stakeholder communication

Limitations of Viewpoints

- Quality properties are critical
  - most viewpoint sets don’t explicitly consider qualities
- Quality properties usually need cross-view consideration
  - viewpoints are relatively independent
- Viewpoint focus may lead to late consideration of quality properties
  - architects focus on desired structures not qualities
  - qualities are often expensive to add later

Content

Talking Point: Qualities

- Which quality properties are crucial to your systems?
- Which views are going to be impacted by considering each such quality property?

Create a matrix showing which qualities are likely to impact which of your views

Possible Qualities to Consider

- Performance
- Scalability
- Security
- Availability
- Resilience
- Evolution
- Maintainability
- Supportability
- Geographical distribution
- Localisation
- Efficiency
- Development constraints
- Time to deliver
- Reliability
- Cost
- Safety

Dealing with Quality Properties

- A new concept could help
  - Allowing cross-view focus
  - Being quality rather than structure oriented
  - Providing similar organisation and guidance to a viewpoint but for a quality property
  - That can be used in tandem with viewpoints
- We call this new concept a "perspective"
  - or "architectural perspective" in full

Architectural Perspectives

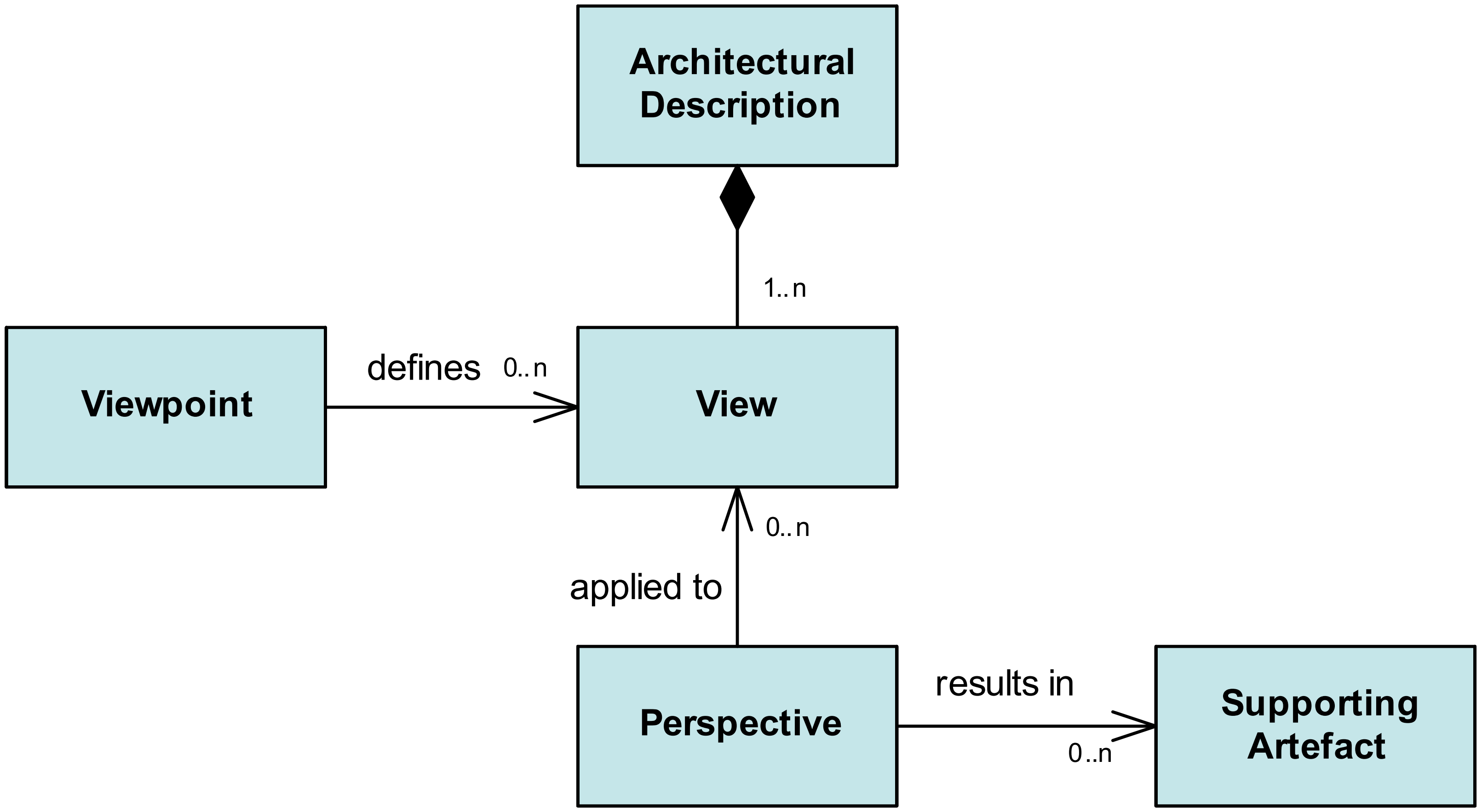
An architectural **perspective** is a collection of **activities, checklists, tactics and guidelines** to guide the process of ensuring that a **system exhibits** a particular set of closely related **quality properties** that require consideration across a number of the system's architectural views.

Rozanski and Woods, 2005

Architectural Perspectives

- A guide for dealing with quality properties
  - Guide the architect in achieving the required quality properties
  - Suggest changes to the existing views
  - Avoid possible redundancy between quality and structural views
- A new concept to use with viewpoints
  - Related to and extends SEI tactics work
  - Adds more context and advice to tactics

Adding Perspectives



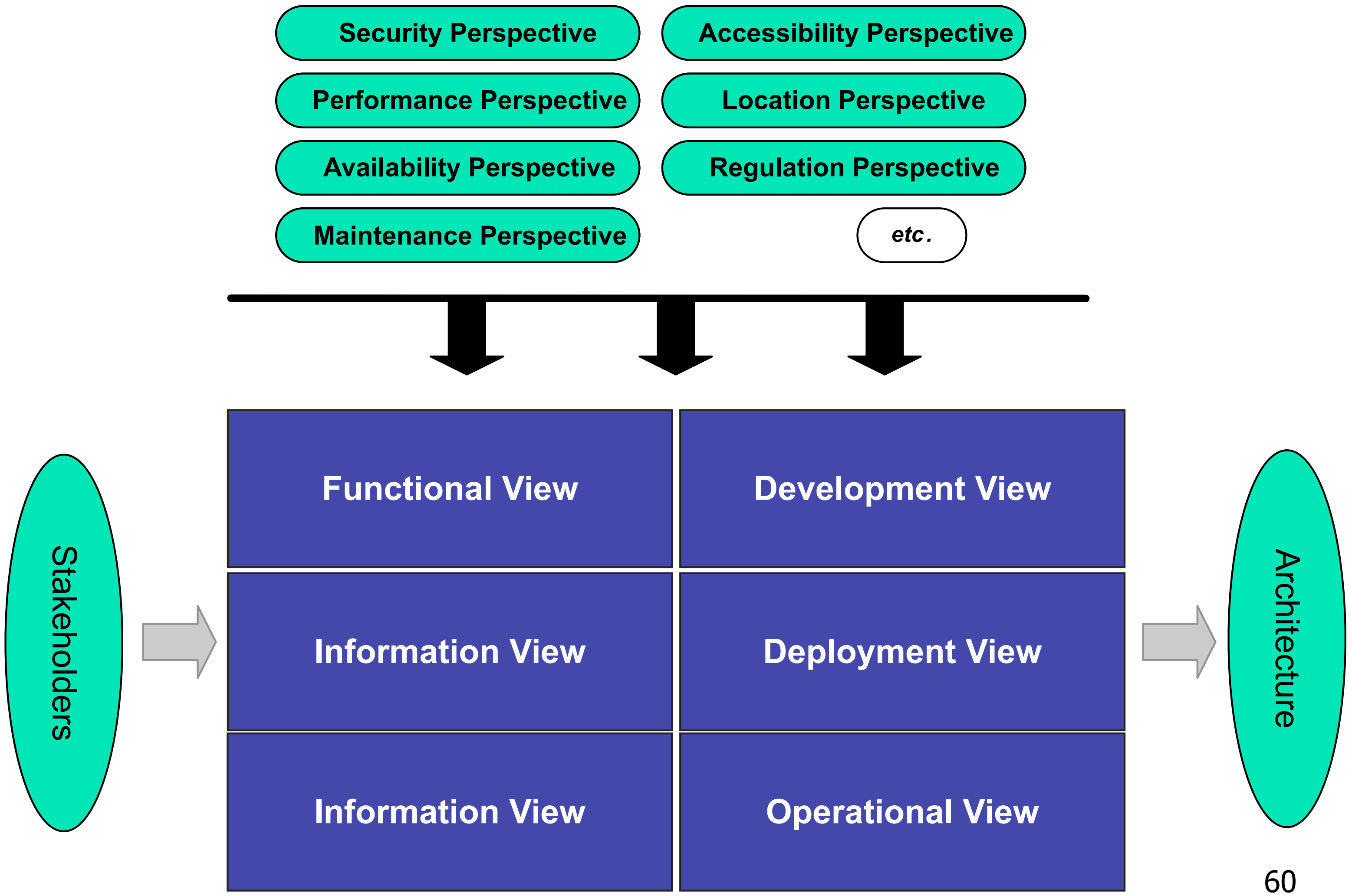
Architectural Perspectives

- A simple but effective idea
  - A store of knowledge and experience
  - A guide to the architect based on proven practice
  - Templates to guide the process
- Analogous to viewpoints but for quality properties, rather than structures
- Perspectives “applied” to views to assess qualities and guide changes needed

Perspectives vs. Viewpoints

	Perspective	Viewpoint
Focus	▪ A quality property	▪ A type of structure
Result	▪ Changes to views ▪ Supporting artefacts	▪ A view – model(s) ▪ A primary arch structure
Guidance	▪ A process for application ▪ Advice based on practice	▪ Models to create ▪ Advice based on practice
Versus Views	▪ 1 perspective : n views	▪ 1 viewpoint : 1 view

Perspectives and Views



Architectural Perspectives

- Our initial core set for information systems
  - Performance and Scalability
  - Security
  - Availability and Resilience
  - Evolution
  - Also: Location, I18N, Usability, Regulation, …

- Different sets in different domains

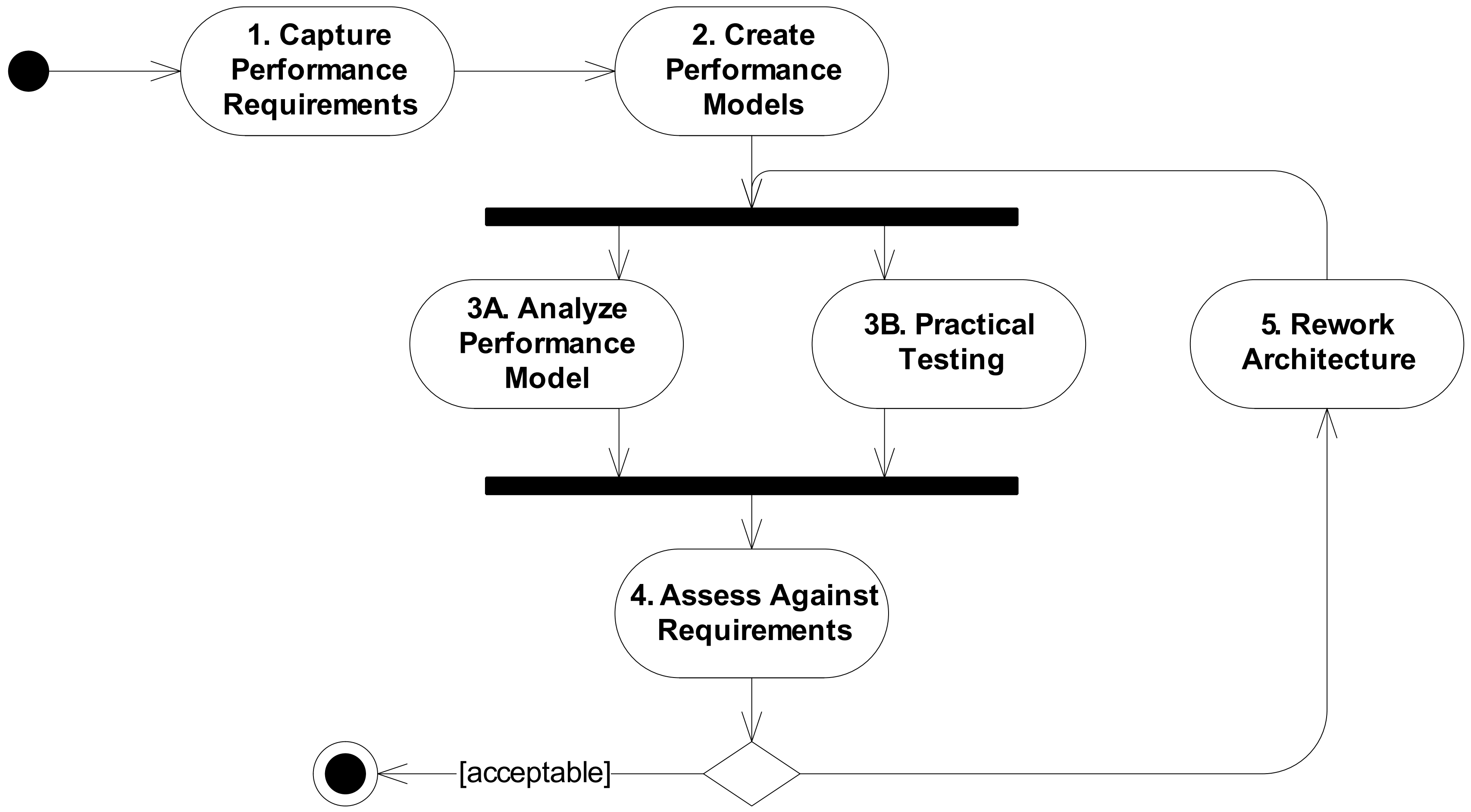
Performance and Scalability

Quality	ability of the system to predictably execute within its mandated performance profile and to handle increased processing volumes
Concerns	▪ processing volume ▪ response time ▪ responsiveness ▪ throughput ▪ predictability
Tactics	▪ optimize repeated processing ▪ reduce contention via replication ▪ prioritize processing ▪ consolidate related workloads ▪ distribute processing over time ▪ […]

Performance and Scalability (ii)

Tactics (cont.)	▪ distribute processing over time ▪ minimize the use of shared resources ▪ partition and parallelize ▪ use asynchronous processing ▪ make design compromises
Pitfalls	▪ imprecise performance and scalability goals ▪ unrealistic models ▪ use of simple measures for complex cases ▪ inappropriate partitioning ▪ invalid environment and platform assumptions ▪ too much indirection ▪ concurrency-related contention ▪ careless allocation of resources ▪ disregard for network and in-process differences

P & S Perspective Activities


1. Capture Performance Requirements
2. Create Performance Models
3A. Analyze Performance Model
3B. Practical Testing
5. Rework Architecture
4. Assess Against Requirements
[acceptable]

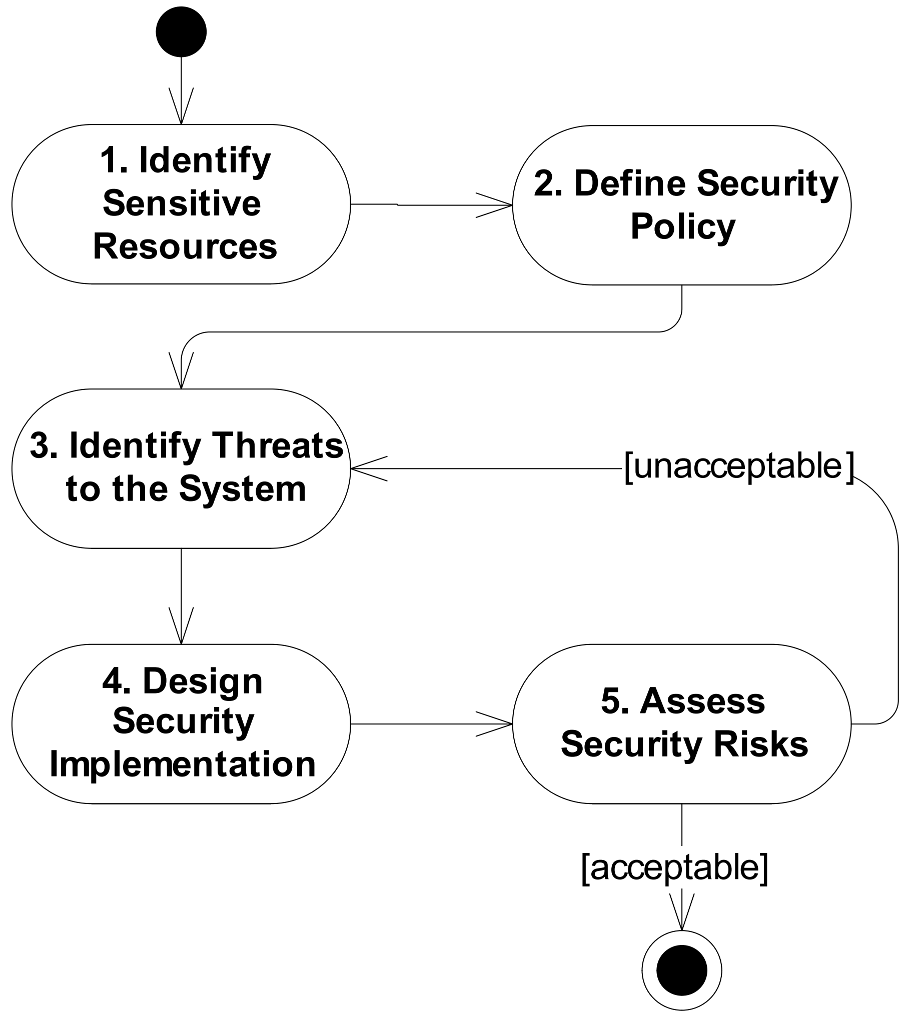
Security

Quality	ability of the system to reliably control, monitor, and audit who can perform actions on resources and to detect and recover from security failures
Concerns	▪ policies ▪ threats ▪ mechanisms ▪ accountability ▪ availability ▪ detection and recovery
Tactics	▪ apply recognized security principles ▪ authenticate the principals ▪ authorize access ▪ ensure information secrecy ▪ ensure information integrity ▪ [...]

Security (ii)

Tactics (cont.)	▪ vulnerability analysis ▪ application of security technology
Pitfalls	▪ no clear requirements or models ▪ complex security policies ▪ unproven or ad-hoc security technologies ▪ not designing for failure ▪ lack of security administration facilities ▪ technology-driven approach (or over-reliance) ▪ failure to consider time sources ▪ security as an afterthought ▪ security embedded in the application code ▪ piecemeal security

Security Perspective Activities


1. Identify Sensitive Resources
2. Define Security Policy
3. Identify Threats to the System
4. Design Security Implementation
5. Assess Security Risks
[unacceptable]
[acceptable]

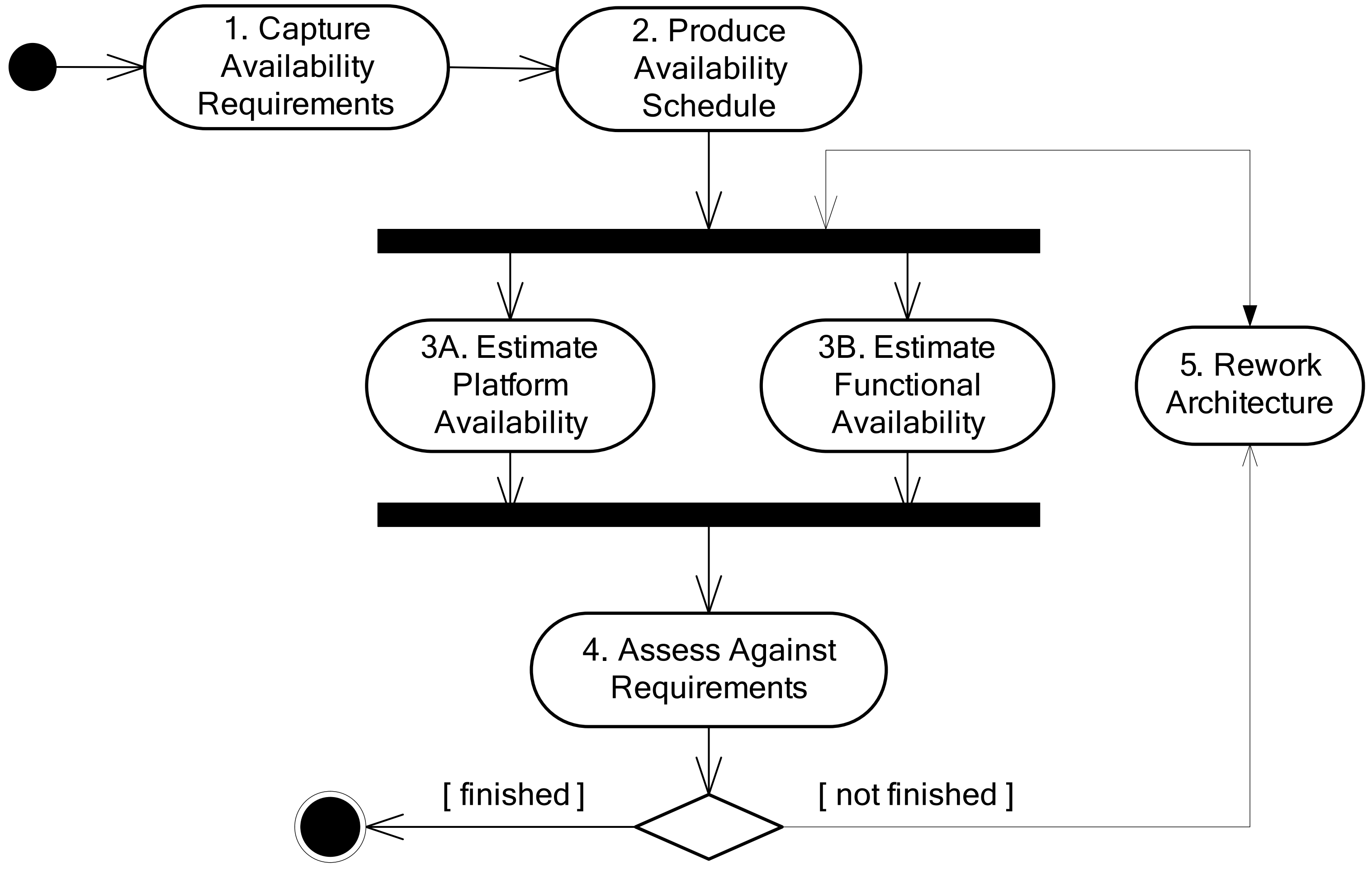
Availability and Resilience

Quality	ability of the system to be fully or partly operational as and when required and to effectively handle failures that could affect system availability
Concerns	▪ classes of service ▪ planned / unplanned downtime ▪ mean time between failures & mean time to repair ▪ disaster recovery ▪ redundancy, clustering, failover
Tactics	▪ select fault-tolerant hardware ▪ use hardware clustering and load balancing ▪ log transactions ▪ apply software availability solutions ▪ select or create fault-tolerant software ▪ identify backup and disaster recovery solutions

Availability and Resilience (ii)

Pitfalls	▪ single point of failure ▪ overambitious availability requirements ▪ ineffective error detection ▪ overlooked global availability requirements ▪ incompatible technologies

A & R Perspective Activities


1. Capture Availability Requirements
2. Produce Availability Schedule
3A. Estimate Platform Availability
3B. Estimate Functional Availability
5. Rework Architecture
4. Assess Against Requirements
[finished]
[not finished]

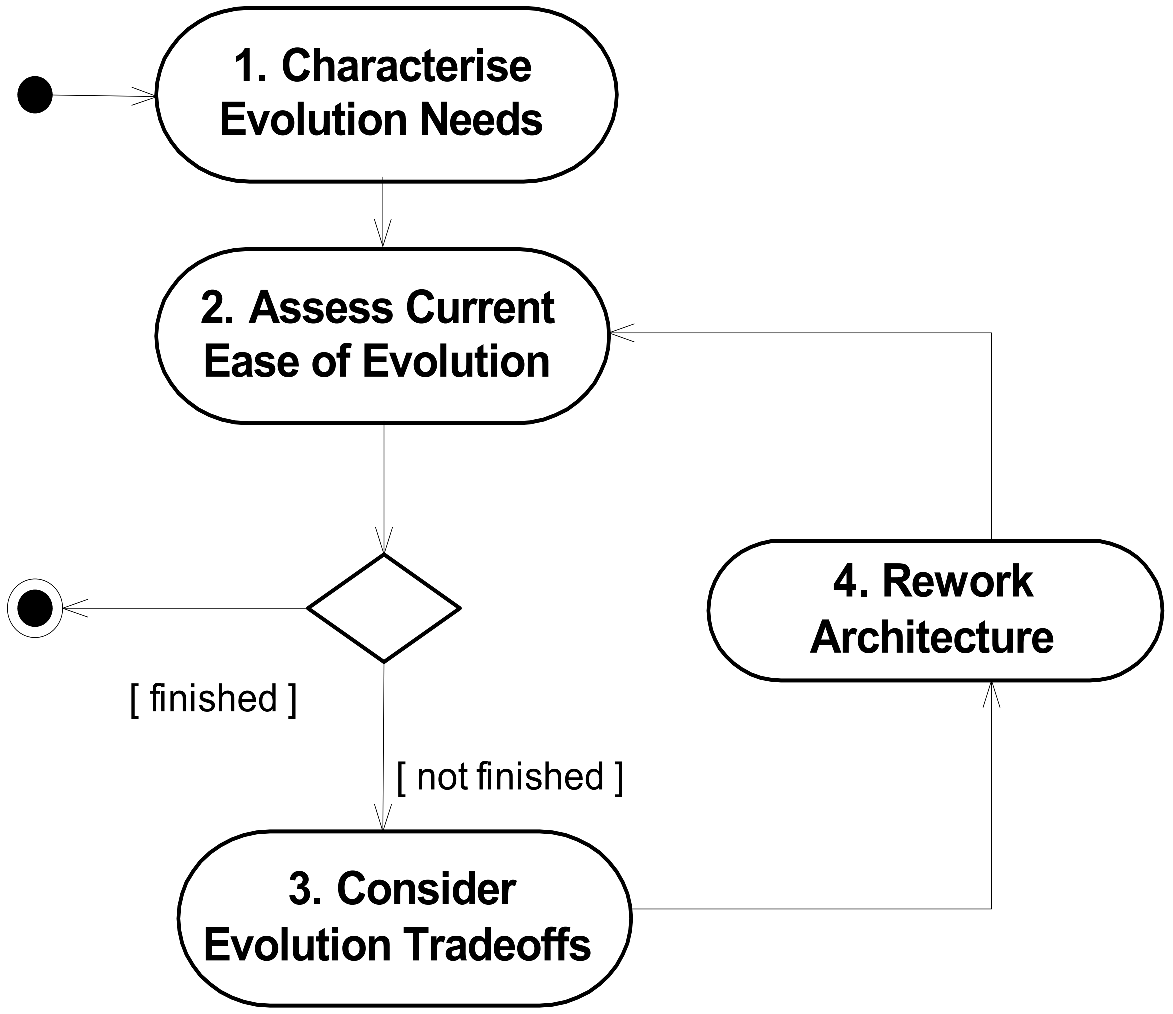
Evolution

Quality	ability of the system to be flexible in the face of the change that all systems experience, balanced against the costs of providing such flexibility
Concerns	▪ flexibility ▪ extensibility ▪ functional evolution ▪ deployment evolution ▪ integration evolution
Tactics	▪contain change ▪create flexible interfaces ▪apply change-oriented architectural styles ▪build variation points into the software ▪use standard extension points ▪achieve reliable change ▪preserve development environments [...]

Evolution (ii)

Tactics (cont.)	▪ configuration management ▪ automated testing ▪ build and release management
Pitfalls	▪ prioritization of the wrong dimensions ▪ changes that never happen ▪ impact of evolution on critical quality properties ▪ lost development environments ▪ ad hoc release management

Evolution Perspective Activities


1. Characterise Evolution Needs
2. Assess Current Ease of Evolution
[finished]
[not finished]
3. Consider Evolution Tradeoffs
4. Rework Architecture

Other Perspectives

Accessibility	Can the system be used by people with disabilities?
Development Resource	Can the system be built within people, time and budget constraints?
Internationalisation	Is the system independent of language, country and culture?
Location	Will the system work, given its required geographical constraints?
Regulation	Does the system meet any required regulatory constraints?
Usability	Can people use the system effectively?

Talking Point: Perspectives

- Going back to your qualities/views matrix
- Which perspectives would help you to achieve your quality properties?
  - Are they in the primary or secondary sets?
  - Useful feedback for us if in secondary
- Where may you have conflicts between advice in different relevant perspectives?

Talking Point: Perspectives

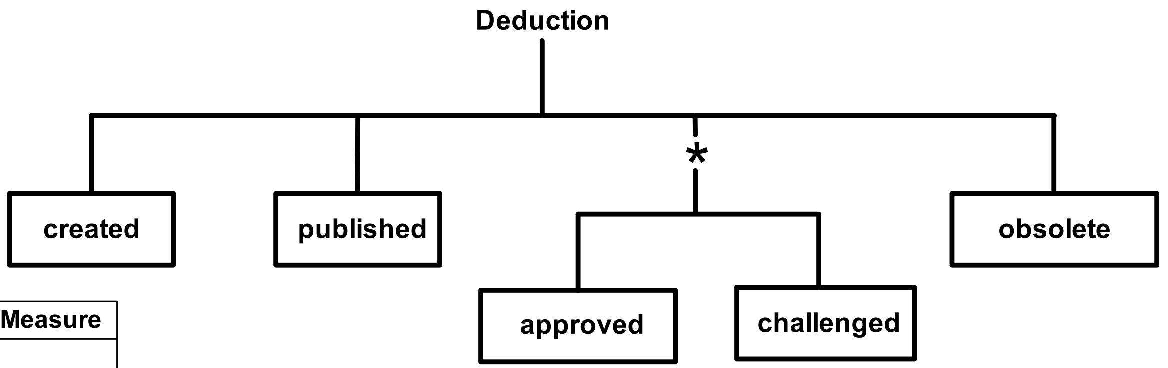
- Decide on the most important qualities for the system you’ve been considering
  - Top 3
- For the most important (or interesting) property identify the likely impact of applying its perspective
  - How does achieving that quality affect the architecture?

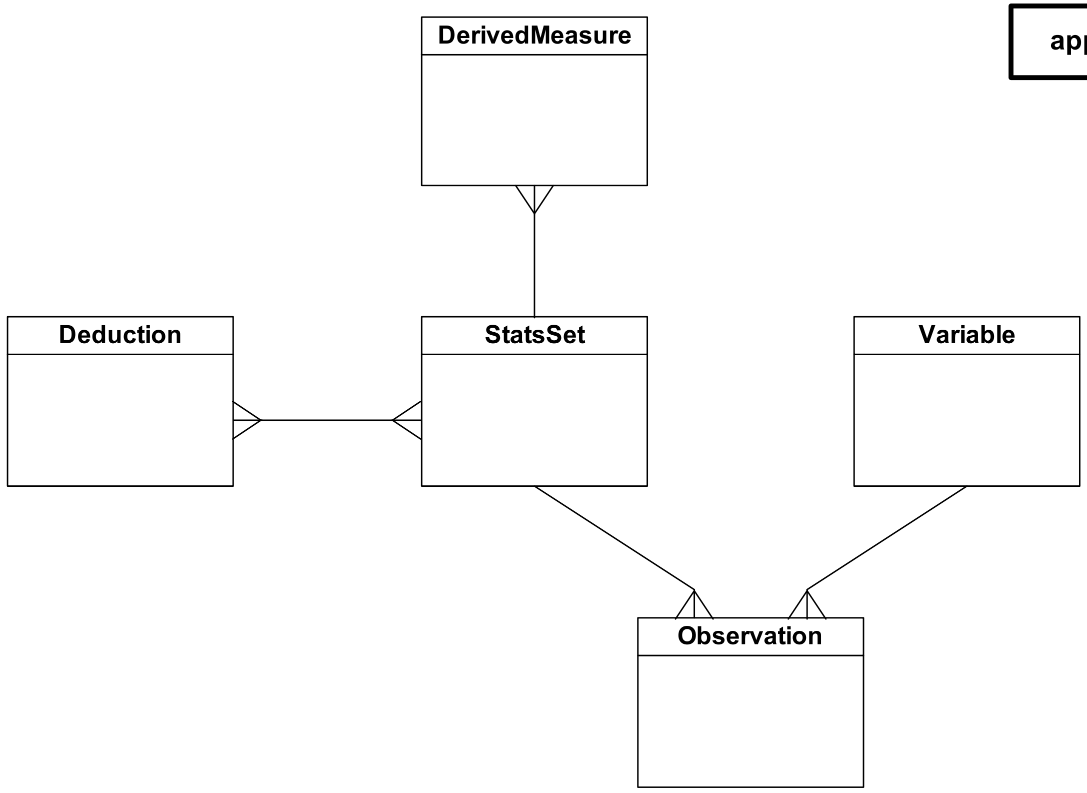
Content

Example Application

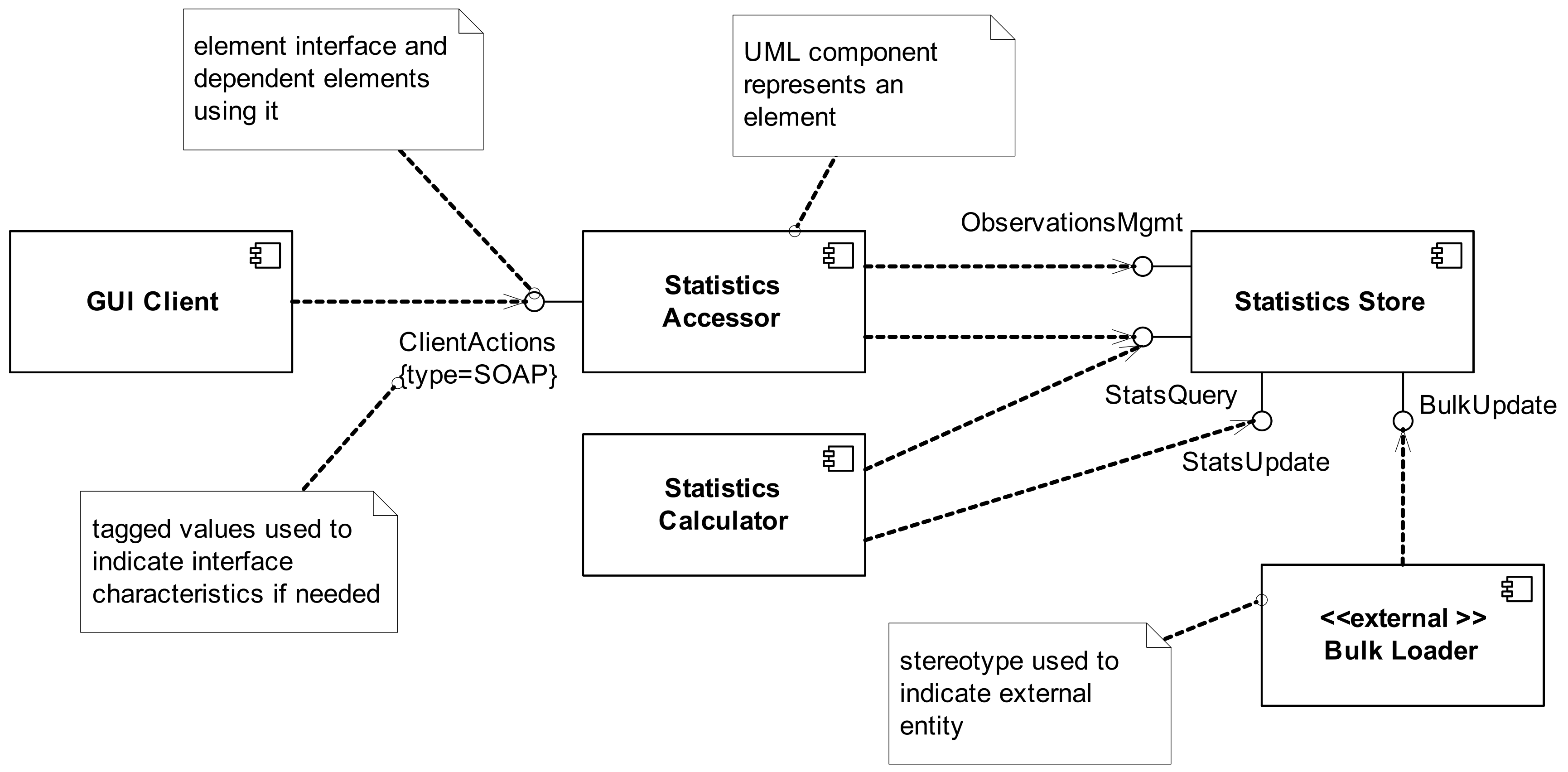
- Simple example of viewpoints and perspectives
- Used throughout the tutorial materials
- Statistics storage and processing system
  - Data loaded into the database
  - Derived measures calculated automatically
  - Statisticians view and report on the data
  - Deductions recorded and reviewed manually

Recap: Information View

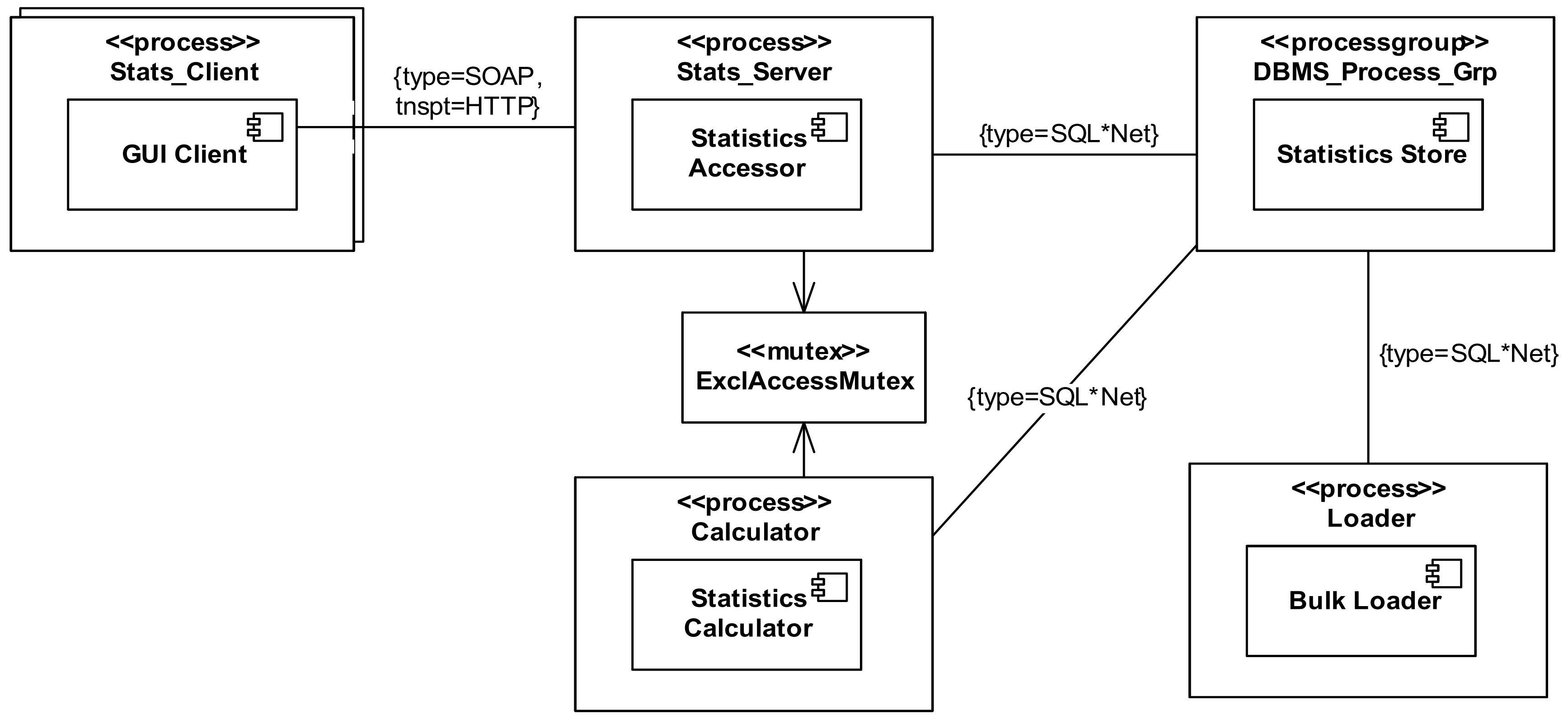




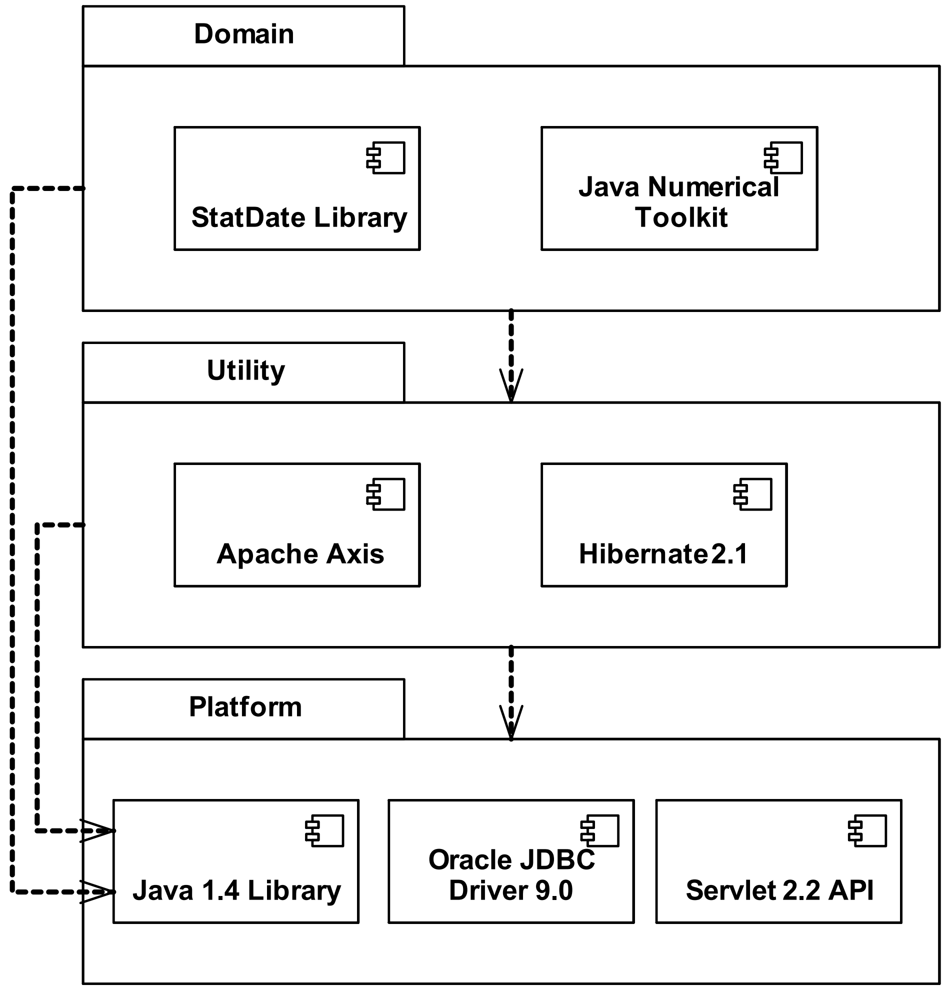
Recap: Functional View


element interface and dependent elements using it
UML component represents an element
GUI Client
Statistics Accessor
ObservationsMgmt
Statistics Store
ClientActions {type=SOAP}
StatsQuery
BulkUpdate
Statistics Calculator
StatsUpdate
tagged values used to indicate interface characteristics if needed
<<external >> Bulk Loader
stereotype used to indicate external entity

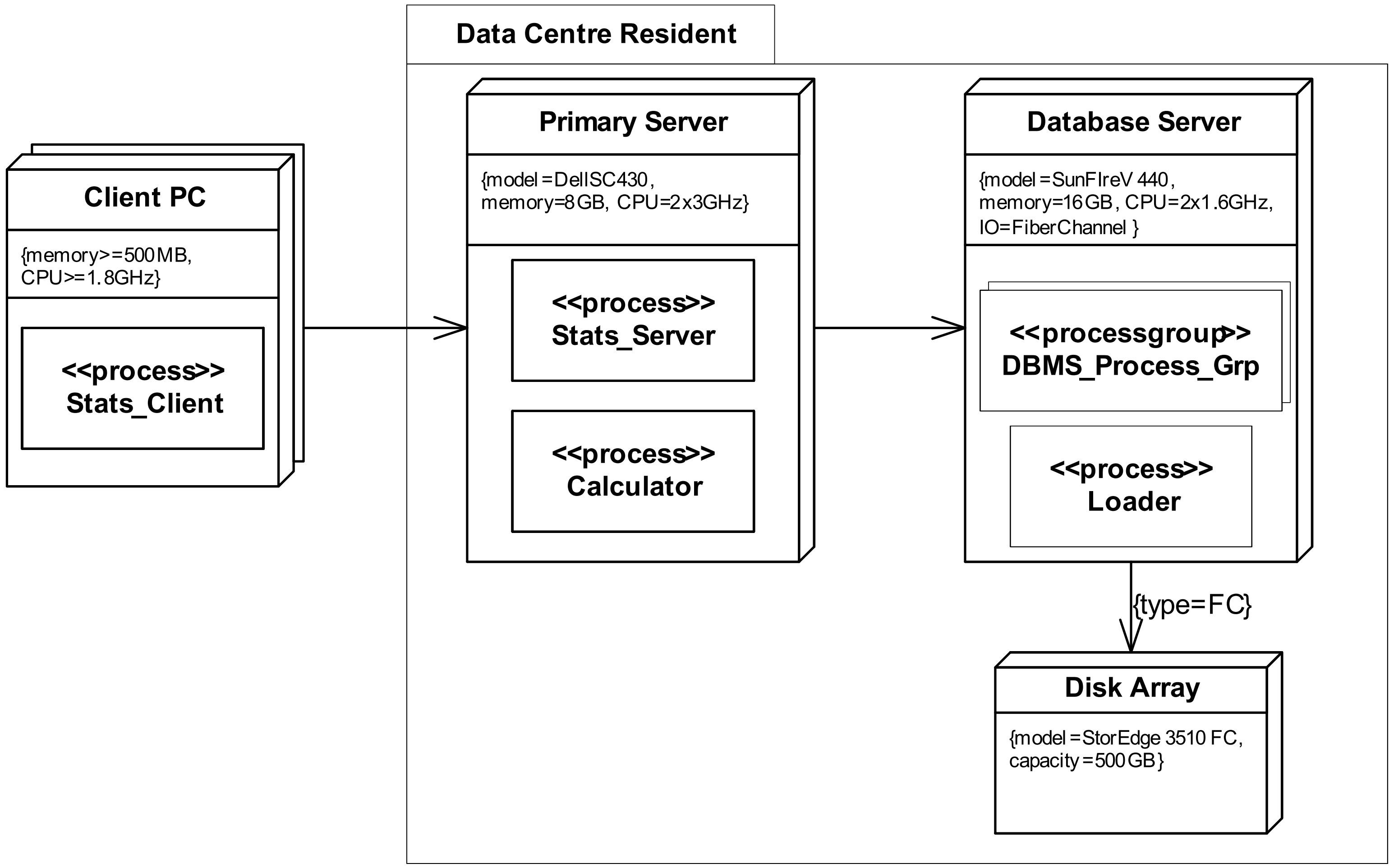
Recap: Concurrency View



Recap: Development View



Recap: Deployment View


Data Centre Resident
Client PC
{memory>=500MB, CPU>=1.8GHz}
<<process>> Stats_Client
Primary Server
{model=DellSC430, memory=8GB, CPU=2x3GHz}
<<process>> Stats_Server
<<process>> Calculator
Database Server
{model=SunFIreV 440, memory=16GB, CPU=2x1.6GHz, IO=FiberChannel }
<<processgroup>> DBMS_Process_Grp
<<process>> Loader
{type=FC}
Disk Array
{model=StorEdge 3510 FC, capacity=500GB}

Recap: Deployment View (ii)

Client PC	▪ Windows XP SP1 ▪ Java JRE 1.4.2_06 or later ▪ Internet Explorer 6.0 SP1
Primary Server	▪ Windows 2003 server, w/sec patches ▪ Java SDK 1.4.2_06 or later ▪ Apache Tomcat 5.5.9 or later
Database Server	▪ Solaris 9.0 w/Aug05 patch cluster ▪ Oracle 9.2.0.2 Std Edition ▪ 10GB buffer cache, auto sized SGA ▪ auto storage management, 2 table spaces ▪ OEM 9.2.0.2 installed and working

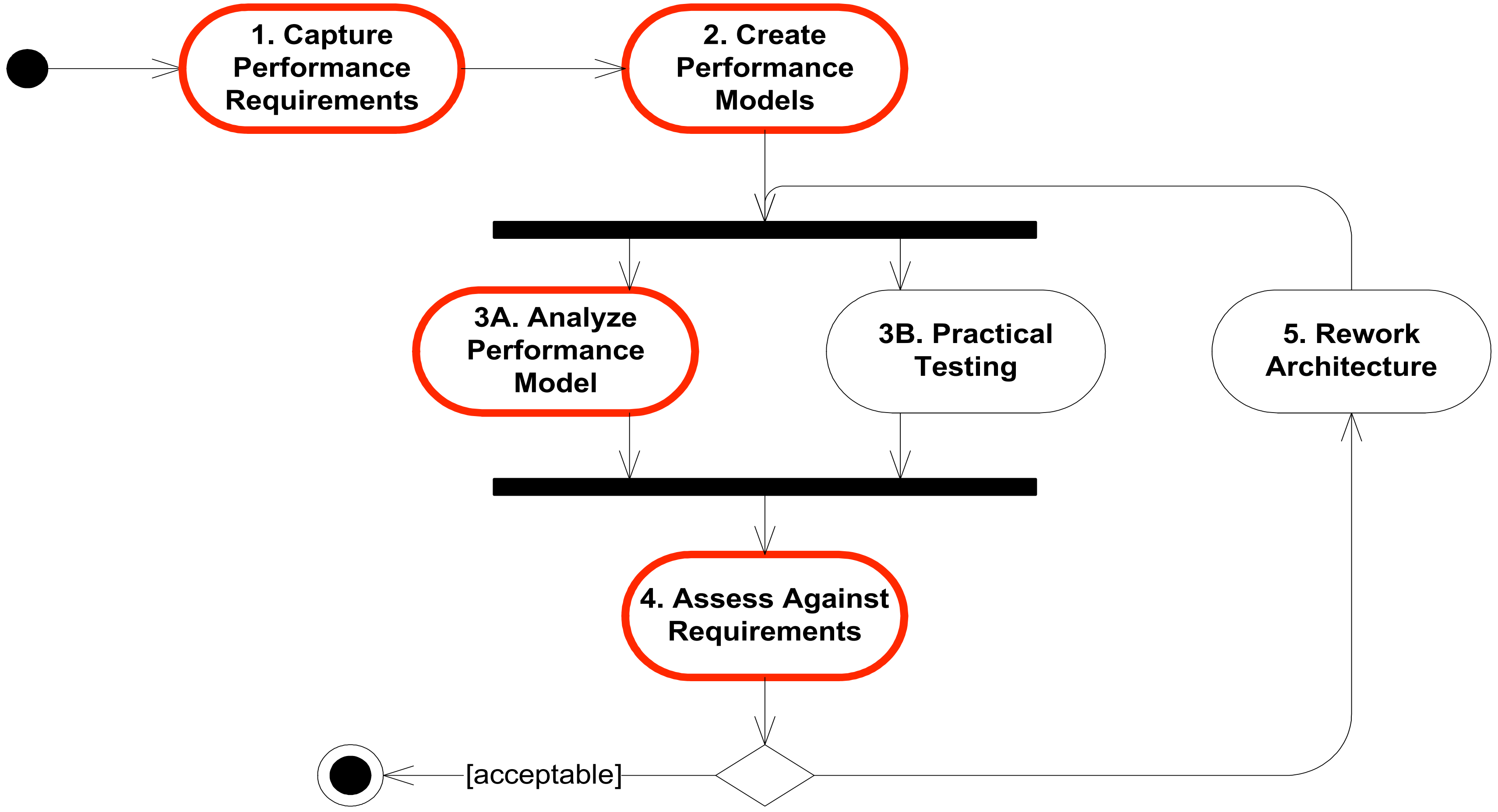
Recap: Operational View

- Omitted from slides for space reasons.
- Would include:
  - Operational CM approach
  - Monitoring and Control
  - Operational Needs
  - Installation / migration / backout strategies

Example: Applying Perspectives

- Performance and Scalability
  - Capture P & S Requirements
  - Create Performance Models
  - Analyse Models
  - Perform Practical Testing
  - Assess Against Requirements
  - Rework Architecture (apply tactics)
- What affect will this have on our system?

Example: Applying Perspectives



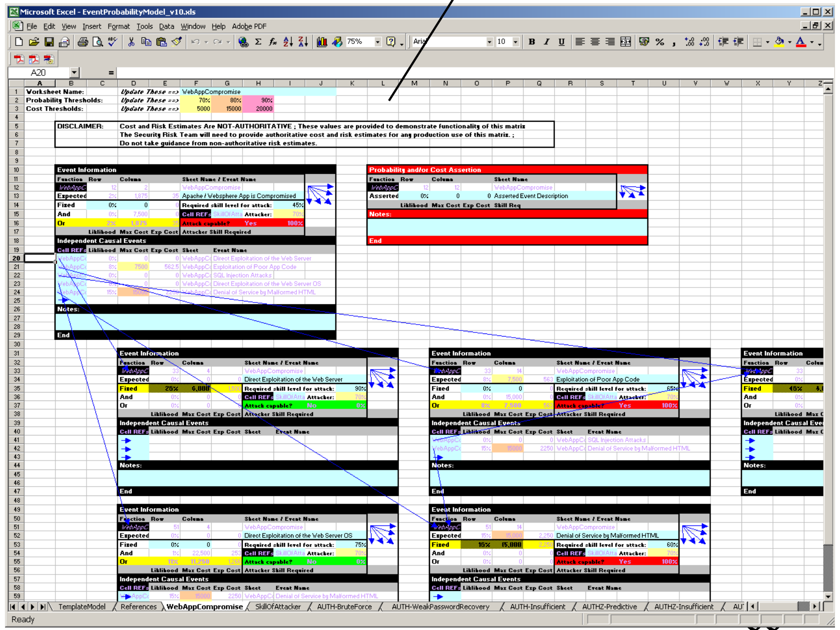
1. Capture Performance Requirements
2. Create Performance Models
3A. Analyze Performance Model
3B. Practical Testing
5. Rework Architecture
4. Assess Against Requirements
[acceptable]

Example: Applying Perspectives

Measure	Value
Network A	100Mb
DB access (ro)	20ms
DB access (rw)	50ms
Single derived calc	1400ms
Client network	10Mb
Bulk load 100K	2500ms
Online load users	100
Memory per user	1MB

Calibration measures

Performance model


Microsoft Excel - EventProbabilityModel_v10.xls

Worksheet Name:	*Update These ==>*	WebAppCompromise
Probability Thresholds:	*Update These ==>*	70% 80% 90%
Cost Thresholds:	*Update These ==>*	5000 15000 20000

DISCLAIMER: Cost and Risk Estimates Are NOT-AUTHORITATIVE ; These values are provided to demonstrate functionality of this matrix
The Security Risk Team will need to provide authoritative cost and risk estimates for any production use of this matrix. ;
Do not take guidance from non-authoritative risk estimates.

Event Information

Apache / Websphere App is Compromised

Probability and/or Cost Assertion

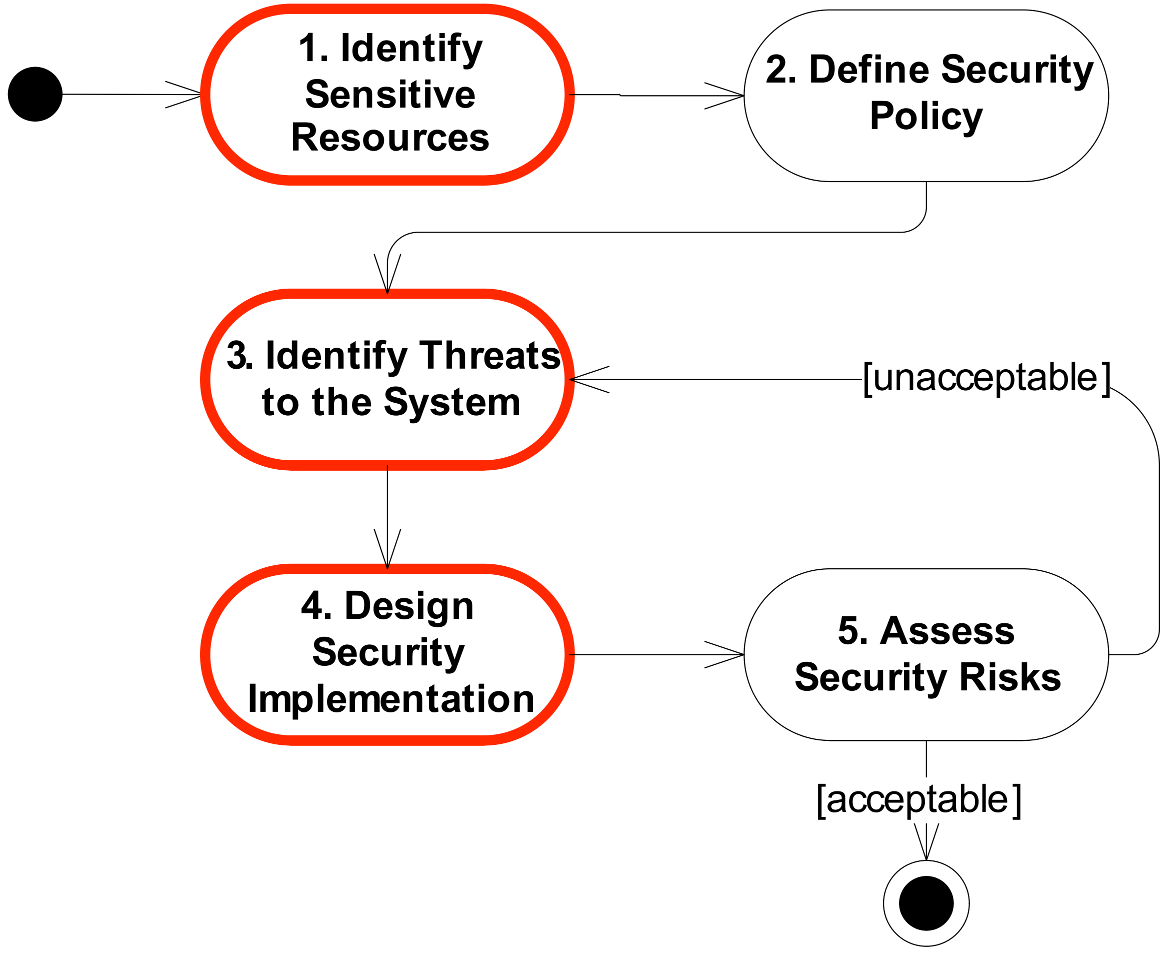
Asserted Event Description

[illegible]

Example: Applying Perspectives

- Security
  - Identify Sensitive Resources
  - Define Security Policy
  - Identify Threats to the System
  - Design Security Implementation (apply tactics)
  - Assess Security Risks
- What affect will this have on our system?

Example: Applying Perspectives



1. Identify Sensitive Resources
2. Define Security Policy
3. Identify Threats to the System
[unacceptable]
4. Design Security Implementation
5. Assess Security Risks
[acceptable]

Example: Applying Perspectives

- Sensitive Resources
  - The data in the database
- Security Threats
  - Operators stealing backups
  - Administrators querying data, seeing names
  - Bribing investigating officers
  - Internal attack on the database via network

Example: Applying Perspectives

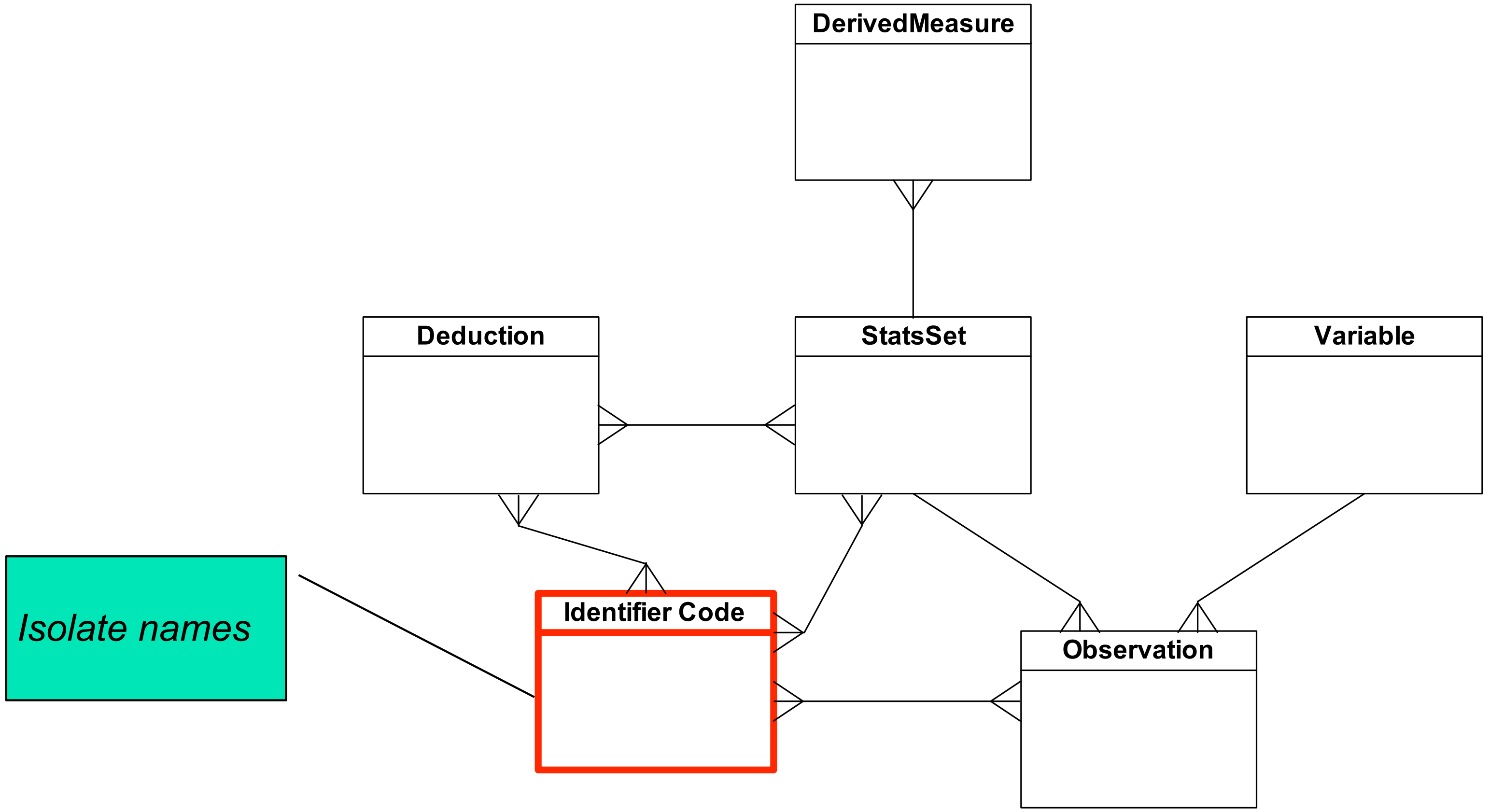
- Security Countermeasures
  - Backups: encrypt data in the database
    - How about performance?
    - Does this make availability (DR) harder?
  - Hiding names: use codes instead of names, protect the underlying names at a higher security level
    - More development complexity
    - Possible performance impact

Example: Applying Perspectives

- Security Countermeasures
  - Bribery: add audit trail for data access
    - Possible performance impact
    - More complexity
    - Protecting / using the audit trail
  - Network Attacks: harden database, firewalls, IDS
    - More deployment & administrative complexity
    - More hardware and operational cost
    - Operational impact if IDS trips

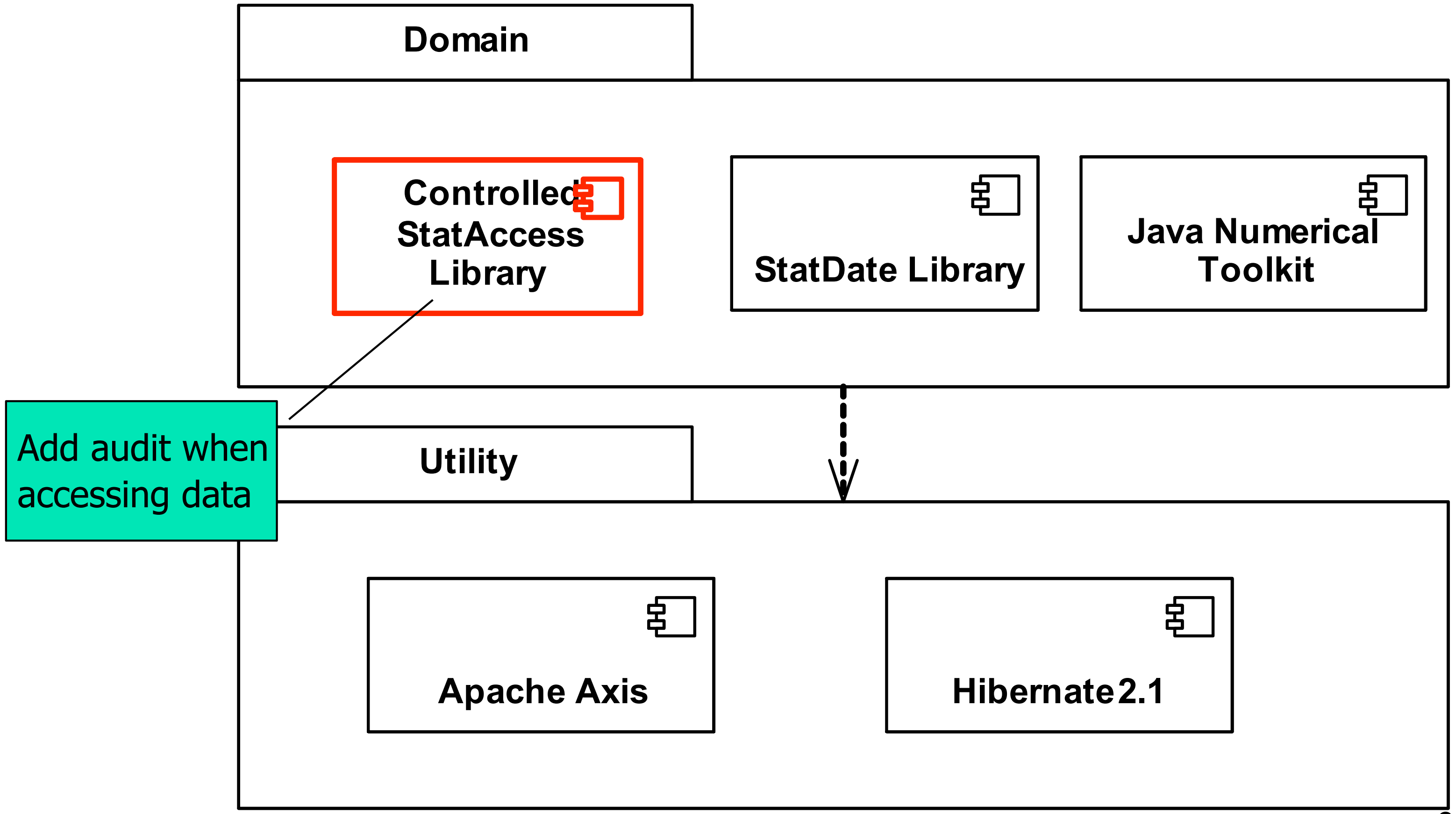
Example: Applying Perspectives

Impact on information view …



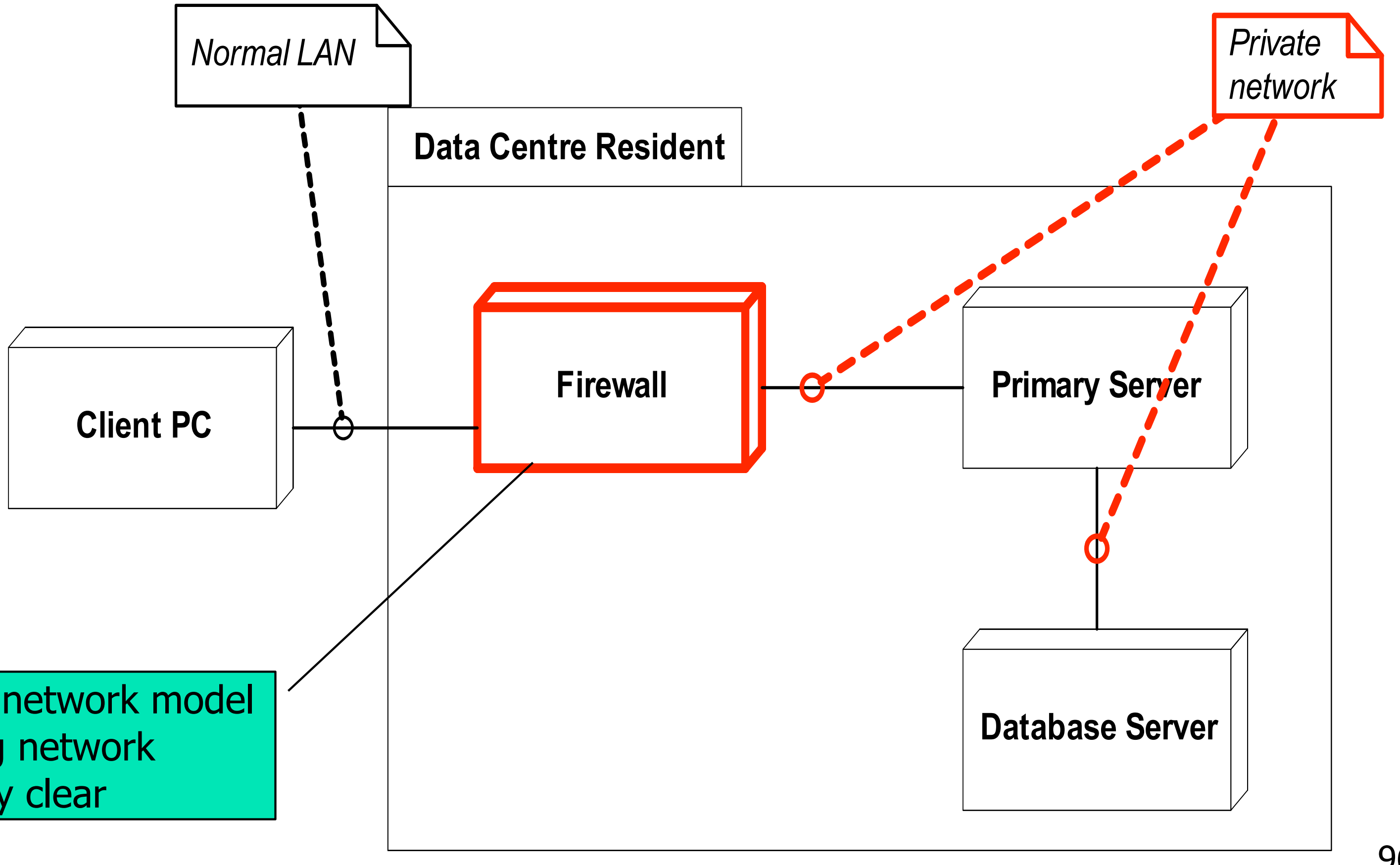
Example: Applying Perspectives

Impact on development view …



Example: Applying Perspectives

Impact on deployment view …



Added network model making network security clear

Example: Applying Perspectives

- Other Impact
  - Need IDS added to Development view
  - Database security
  - Need to capture impact on Operational view
  - Need to consider impact on availability
  - Need to re-work performance models to allow for database encryption, audit, …

- Note the need to change many views to address security needs

Content

- Defining Software Architecture
- Stakeholders
- The Software Architecture Problem
- Viewpoints to Guide Structure
- Perspectives to Guide Qualities
- Example Application
- Uses for Viewpoints and Perspectives

Using Viewpoints & Perspectives

- A framework for organising work
- A store of knowledge
  - Document proven practice
  - Help standardise language and approach
  - Help to standardise languages and approaches
- Applicable at different career stages
  - Mentor novice architects
  - Guide working architects
  - Support expert architects

Using Viewpoints & Perspectives

- For Novice Architects
  - An introduction to each area of knowledge
  - A guide to what is important
  - A structure for the process
  - Definitions of standards and norms
  - Repository of proven practice and tactics
  - Pitfalls and solutions to avoid common errors
  - Checklist to ensure nothing is forgotten

Using Viewpoints & Perspectives

- For Working Architects
  - A reminder of what is important
  - A guide to new or rarely used areas of practice
  - Repository of proven practice and tactics
  - Pitfalls and solutions to avoid common errors
  - Checklist to ensure nothing is forgotten

Using Viewpoints & Perspectives

- For Expert Architects
  - A framework to allow knowledge sharing
  - An aid to tutoring and mentoring
  - Checklists to ensure nothing is forgotten

Summary

- Architecture Essential Difficulties
  - Multi-dimensional problem, no right answer
  - Stakeholder needs conflict
  - Complex mix of people and technology
  - Tradeoffs are inevitable
- Architecture Accidental Difficulties
  - Lack of standardisation (approach, artefacts)
  - Little sharing of knowledge and experience

Summary (ii)

- Viewpoints and Views
  - Views provide a convenient approach for effective architectural description
  - Viewpoints standardise views by defining their content
  - Viewpoints contain proven architectural knowledge for a particular domain
  - Viewpoints and views can address many accidental difficulties of software architecture

Summary (iii)

- Viewpoints for Information Systems
  - Functional
  - Information
  - Concurrency
  - Development
  - Deployment
  - Operational

Summary (iv)

- Perspectives
  - Viewpoints handle structure well, less convinced about quality properties
  - Perspectives provide similar guidance and knowledge sharing for quality properties
  - Perspectives suggest the design activities required to achieve a property
  - Perspectives provide proven practice, pitfalls, solutions and checklists to share experience
  - **Applying perspectives modifies views**

Summary (v)

- Perspectives for Information Systems
  - Availability and Resilience
  - Evolution
  - Performance and Scalability
  - Security
  - Others
    - Accessibility, Development Resource, Geographical Location, I18N, Regulation, Usability

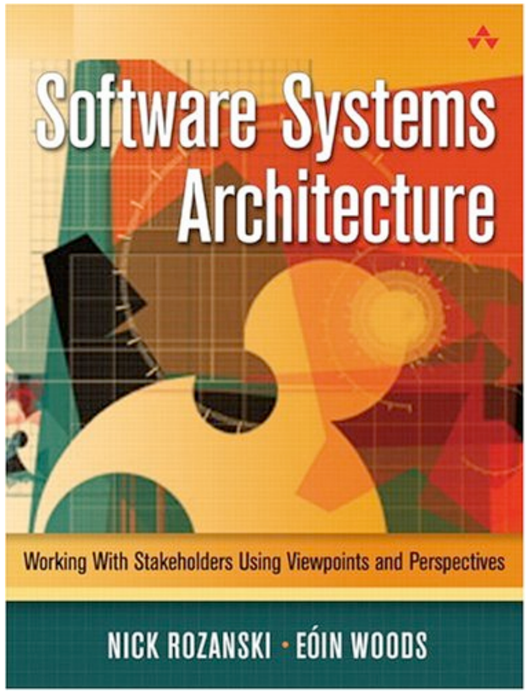
Summary (iv)

- Using Viewpoints and Perspectives
  - Novices
    - Overview, guides, focuses attention
    - Proven practice, pitfalls, solutions and checklists
  - Working Architects
    - Reminder of existing knowledge
    - Aid in new areas
  - Experts
    - Mentoring and communication vehicle
    - Reminders of hard won lessons

To Learn More

Software Systems Architecture
Working With Stakeholders Using Viewpoints and Perspectives

Nick Rozanski & Eoin Woods
Addison Wesley, 2005


Software Systems Architecture
Working With Stakeholders Using Viewpoints and Perspectives
NICK ROZANSKI · EÓIN WOODS

`http://www.viewpoints-and-perspectives.info`

Eoin Woods
contact@eoinwoods.info
www.eoinwoods.info

Comments and Questions?

Nick Rozanski
nick@rozanski.com
www.nick.rozanski.com