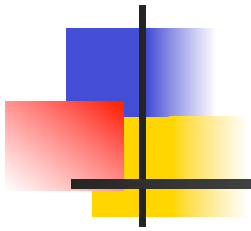
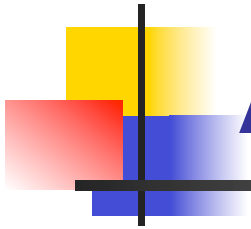


The Past, Present and Future of Software Architecture

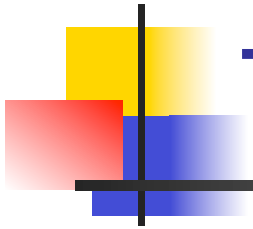
Eoin Woods
UBS Investment Bank

eoin@artechra.com
www.eoinwoods.info



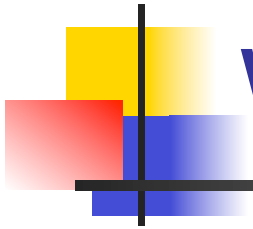
About me

- I’m a working software architect
- Always worked as a software engineer
  - 8 years of products, 7 of applications
- Recently moved to end-user company
  - Lead a central architecture group
- Co-author of software architecture book
  - With Nick Rozanski, Addison-Wesley, 2005
- Participant in IFIP 2.10 WG



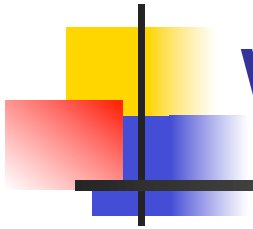
Topics

- **Introducing Software Architecture**
- The Past and Present
- Exploring the Fundamentals
- An Example of Architecture in Practice
- The Future



What Is Software Architecture

- *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*
  - Bass, Clements, Kazman (SEI)

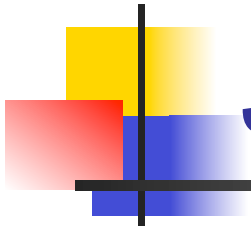


What is Software Architecture

- *The set of **design decisions** which, if **made incorrectly**, will cause your project to be **cancelled***
  - Eoin Woods *(heads the SEI definitions list)*

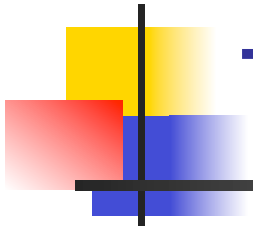
The SEI definitions list:

`www.sei.cmu.edu/architecture/definitions.html`



Just Design, Surely?

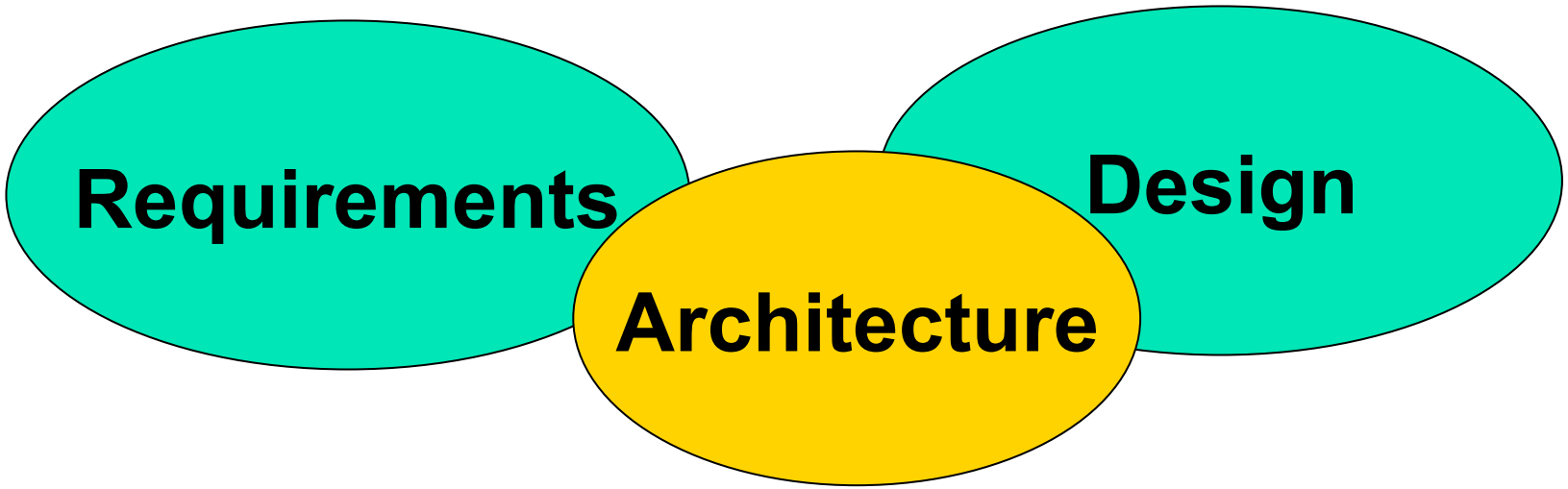
- All architecture is design, not all design is architecture [Paul Clements]
- Not all design decisions are equal
  - Some have "*architectural significance*"
- Architectural design is outward looking
  - Focus on stakeholders, not technology
- Architecture more fluid than design
  - Context, scope, success criteria all unclear



The Essence

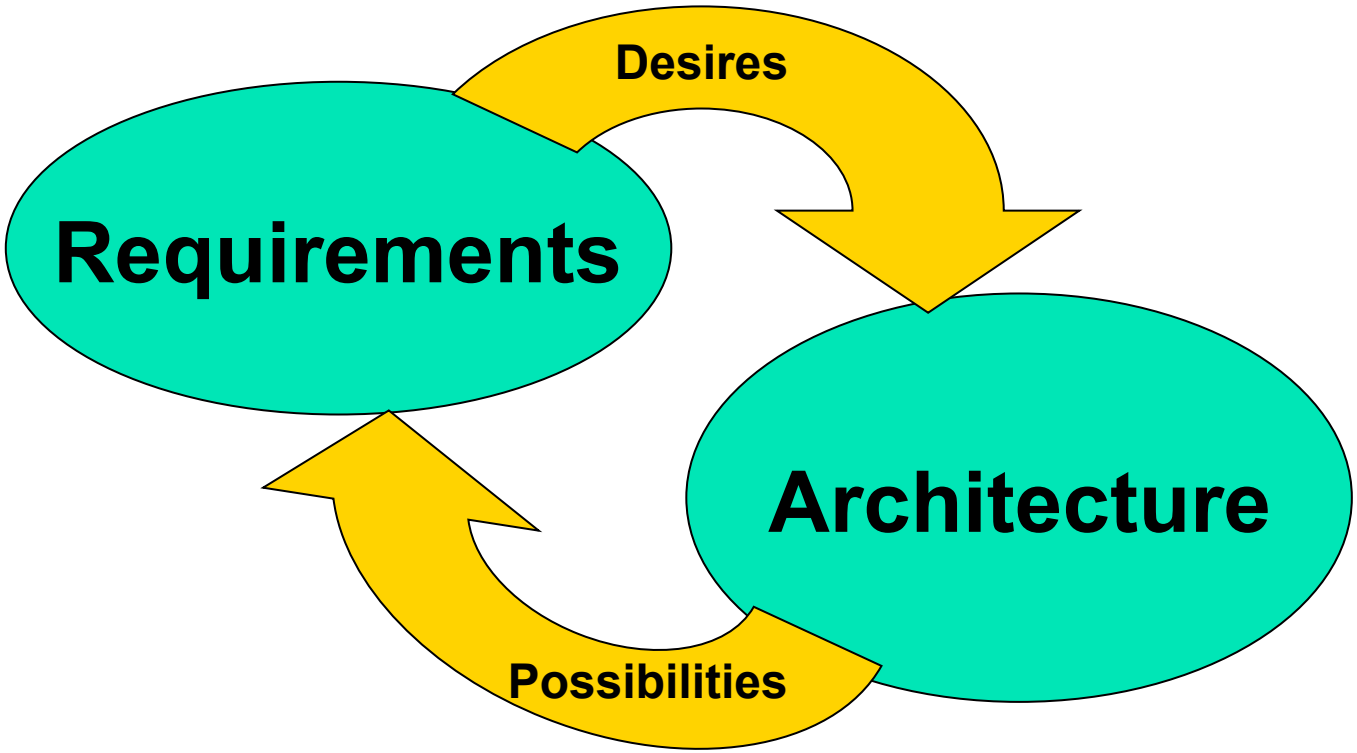
- Software architecture is concerned with:
  - ***Stakeholders***
  - System-Level ***Structures***
  - System ***qualities***
- Software architecture involves:
  - ***Understanding*** domains, problems and solutions
  - ***Making*** design ***decisions*** & tradeoffs
  - ***Delivering*** working systems

Architecture in Context

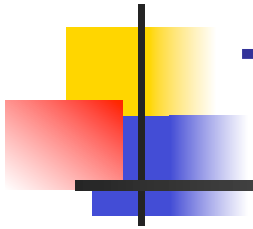


The crucial bridge between requirements and design

Architecture in Context



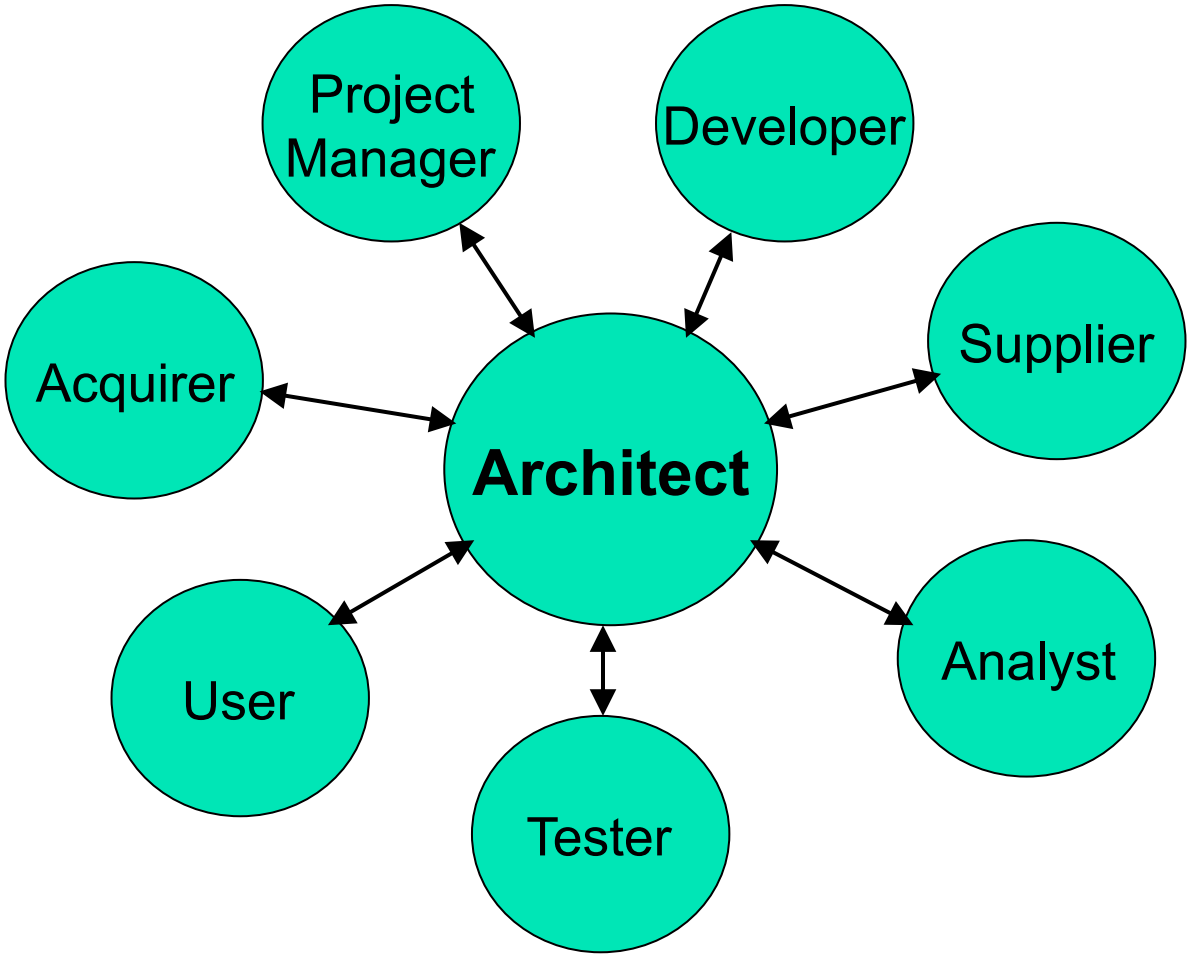
This interplay is core to the architectural process

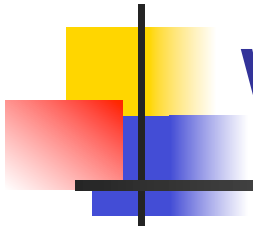


The Role of the Architect

- Central technical communicator
- Key decision maker
- Responsible for delivery

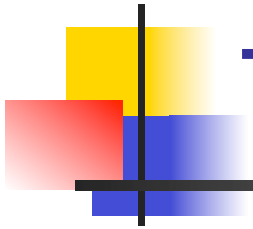
Architect as Communicator





Why Architecture is Important

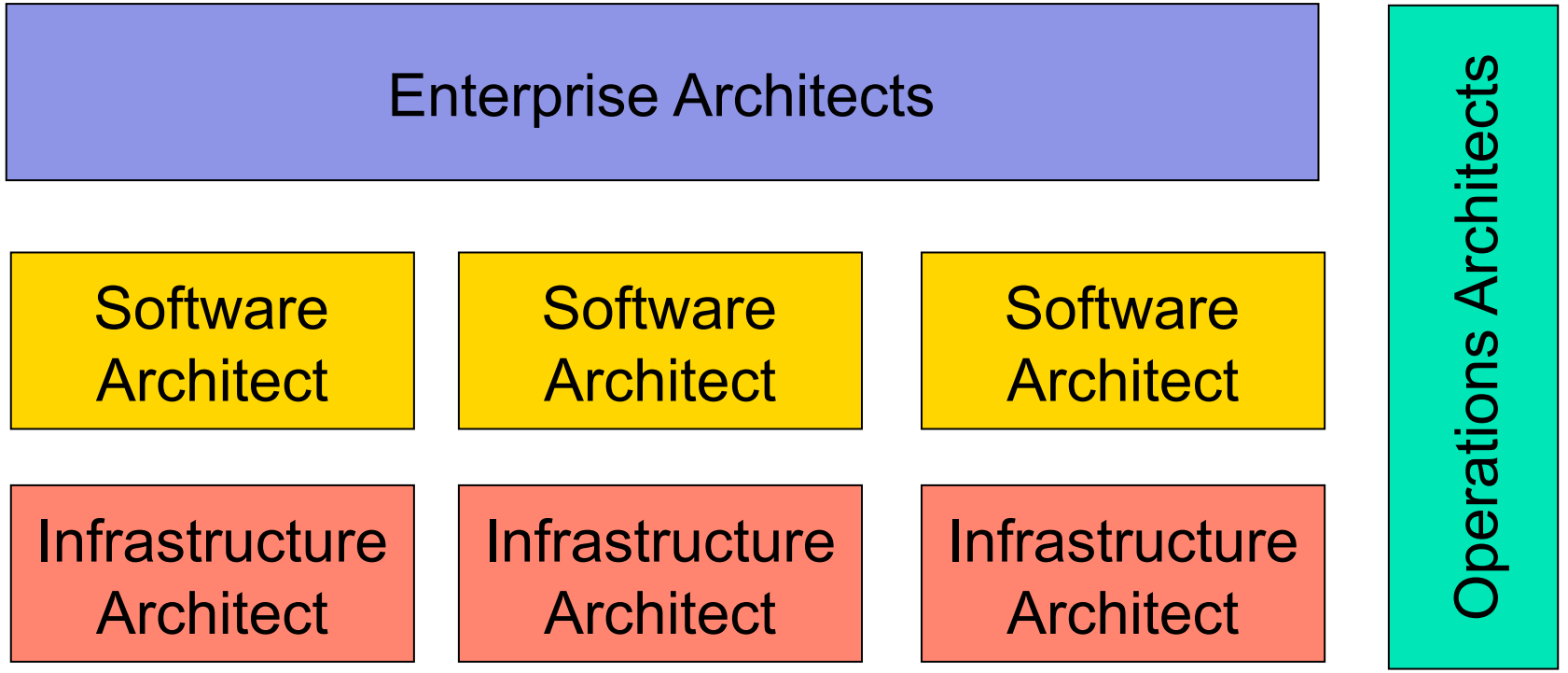
- Stakeholder focus
- Ensuring that the right system is built
- System-wide (or cross-system) consistency

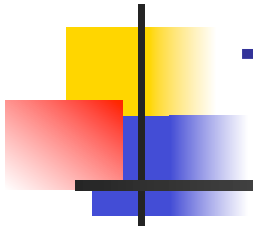


Types of IT Architect

- Software architect
  - Focus of this talk
  - Responsible for a system's structures
- Enterprise architect
  - Responsible for cross-system structures
- Infrastructure architect
  - Responsible for technology-specific structures
- Operations architect
  - Responsible for operational structures

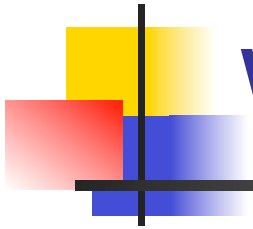
Types of IT Architect





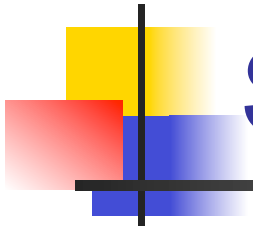
Topics

- Introducing Software Architecture
- **The Past and Present**
- Exploring the Fundamentals
- An Example of Architecture in Practice
- The Future



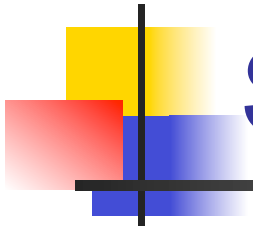
Where Did it Come From?

- Ideas from David Parnas (1985)
  - also Zachman Framework (1987)
- Perry and Wolf paper (1992)
- Began largely as an academic interest
- Enthusiastically transferred into industry
  - little interchange between the two since!
- Mirrors the rise in importance and status of the technical IT professional



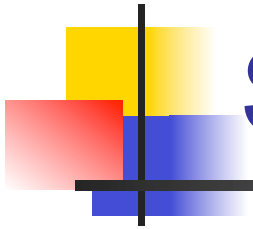
Some Milestones (i)

- 1995:
  - IEEE Software special issue on Software Architecture
  - Philippe Kruchten's "4+1" viewpoint set published
- 1996:
  - Shaw and Garlan's book "*Software Architecture: Perspectives on an Emerging Discipline*"
- 1998:
  - Bass, Clements and Kazman publish "*Software Architecture in Practice*", 1st edition



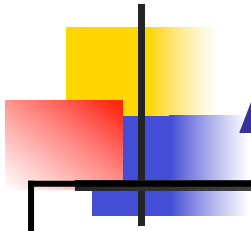
Some Milestones (ii)

- 2000:
  - Views and Viewpoints standardised via IEEE-1471
  - SEI's "ATAM" analysis method published
  - RUP becomes "architecture centric"
  - J2EE 1.0 specification released
- 2001:
  - WICSA conference series starts
- 2002:
  - .NET 1.0 released



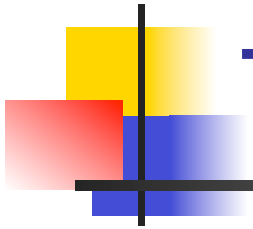
Some Milestones (iii)

- 2003:
  - Bass, Clements and Kazman, **2nd** Edition
  - Martin Fowler admits software architecture exists!
- 2005:
  - IASA is founded and starts growing
  - New “Perspectives” concept identified
    - Nick Rozanski and Eoin Woods
    - Part of a new practitioner-oriented book
  - WICSA 5 runs in Pittsburgh
    - 10th anniversary of the IEEE Software issue
    - Significant practitioner focus and attendance



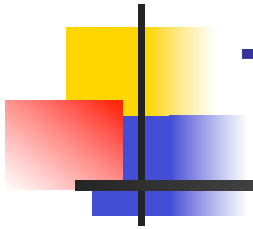
As an Aside – My Parallels

	Me	The Field
1990	▪ Trainee s/w engineer	▪ Little s/w architecture discussion
1993	▪ Trainee s/w architect	▪ Perry and Wolf paper (92)
1995	▪ First architecture role	▪ 1st book (Witt, Baker, Merritt, 1994) ▪ IEEE Software issue & “4+1” ▪ Shaw and Garlan book (96)
2000-02	▪ Systems architect role	▪ Book explosion ▪ IEEE 1471 ▪ J2EE and .NET
2005	▪ My book ▪ Current role	▪ WICSA 5 runs ▪ IASA founded ▪ Fowler’s P of EAA (94)



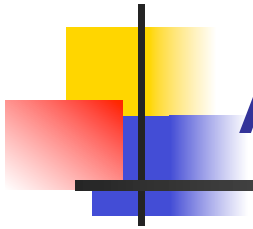
The Present

- The language of SWA is widely used
  - Even if rarely defined or understood
- Reasonably large set of (basic) books
- Organisations & certification emerging
  - IASA, Microsoft
- Organisations are seeing value
  - Supply: IBM, Microsoft, Evolution-Detica,…
  - Demand: Hartford Financial, UBS, BP, …



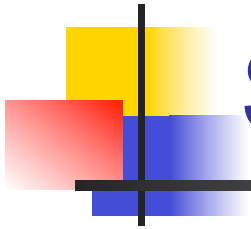
Topics

- Introducing Software Architecture
- The Past and Present
- **Exploring the Fundamentals**
- An Example of Architecture in Practice
- The Future



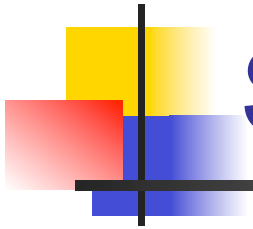
Architecture Fundamentals

- Stakeholders
  - Who cares what we build?
  - Why do they care?
- System structures
  - What do we build?
- System qualities
  - Why do we build it *that* way?



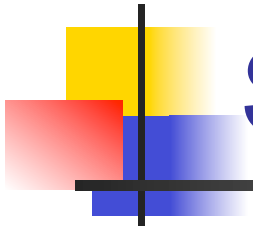
System-Level Structures

- The traditional deliverable of the software architect
- Note that there are many structures
  - Functional, Deployment, Information, …
- Represented as a set of views
  - Separate concerns
  - Communicate effectively
  - Organise deliverables and activities



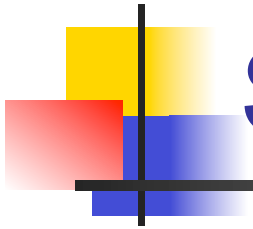
Stakeholders

- Those who care if the system gets built
  - Can be a positive or negative interest
  - Includes people, groups and entities
- The reason we build systems
  - Systems are built for stakeholders
  - Design decisions must reflect their needs
  - A wide community often increases the chances of success



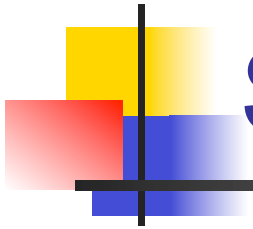
Stakeholders

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



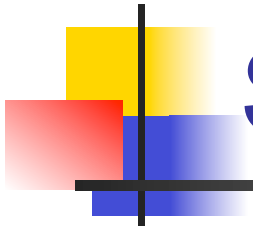
Stakeholders

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



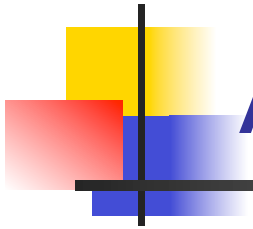
System Qualities

- 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



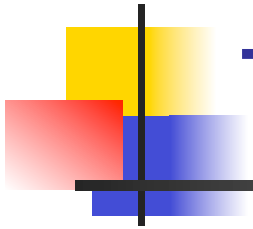
System Qualities

- Achieving system qualities is a key task
  - Understanding real stakeholder needs
  - Understanding what is possible
  - Making the key trade-offs to allow delivery
  - Avoiding expensive “retro-fit”



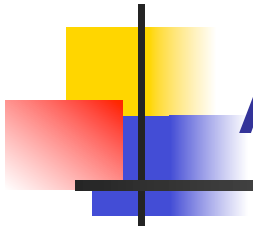
Architect's Responsibilities

- Understand requirements
  - Identify architectural impact
- Lead development
  - Help, don't hinder
- Analyse architectures
- Make design decisions and tradeoffs
- Communicate
- Ensure delivery!



Typical Architect's Activities

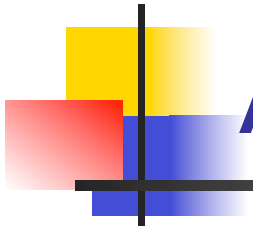
- Create models (UML, ADL, B+L, …)
- Build skeleton systems
- Write PoCs and prototypes
- Write documents
- Give presentations
- Undertake "rescue missions"



Attributes of a Good Architect

- Technology knowledge
- Design ability
- Communication skills
- Pragmatism
- Political sensitivity
- Tact

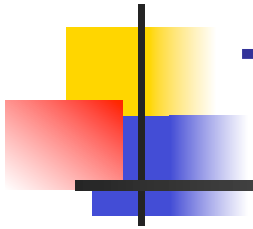
… and a good sense of humour doesn't hurt!



Architectural Significance

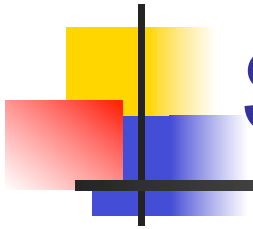
- *A concern, problem, or system element is architecturally significant if it has a wide impact on the structure of the system, or on its important quality properties such as performance, scalability, security, reliability or evolvability*
  - Philippe Kruchten

- Externally visible
- Has a real affect on stakeholder utility
- Difficult to change later
- Affects many parts of the system



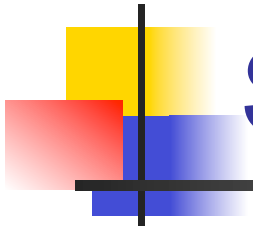
Topics

- Introducing Software Architecture
- The Past and Present
- Exploring the Fundamentals
- **An Example of Architecture in Practice**
- The Future



Simple Example

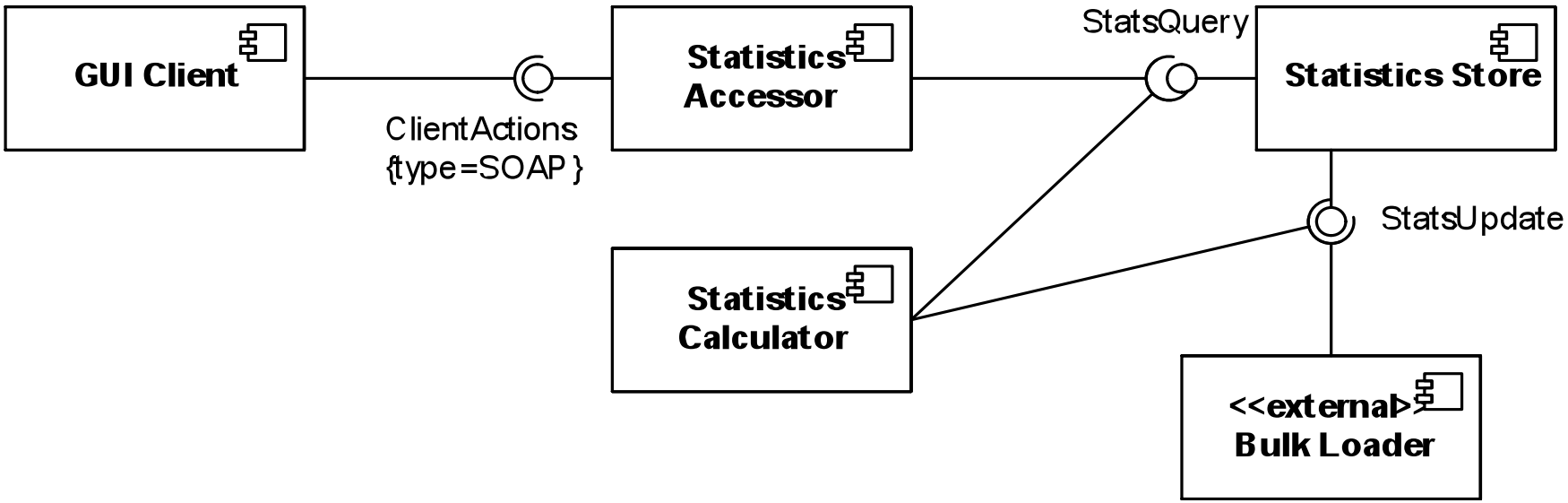
- A statistics management system
  - Data bulk-loaded into the database
  - Derived measures calculated automatically
  - Statisticians view and report on the data
  - Deductions recorded and reviewed manually



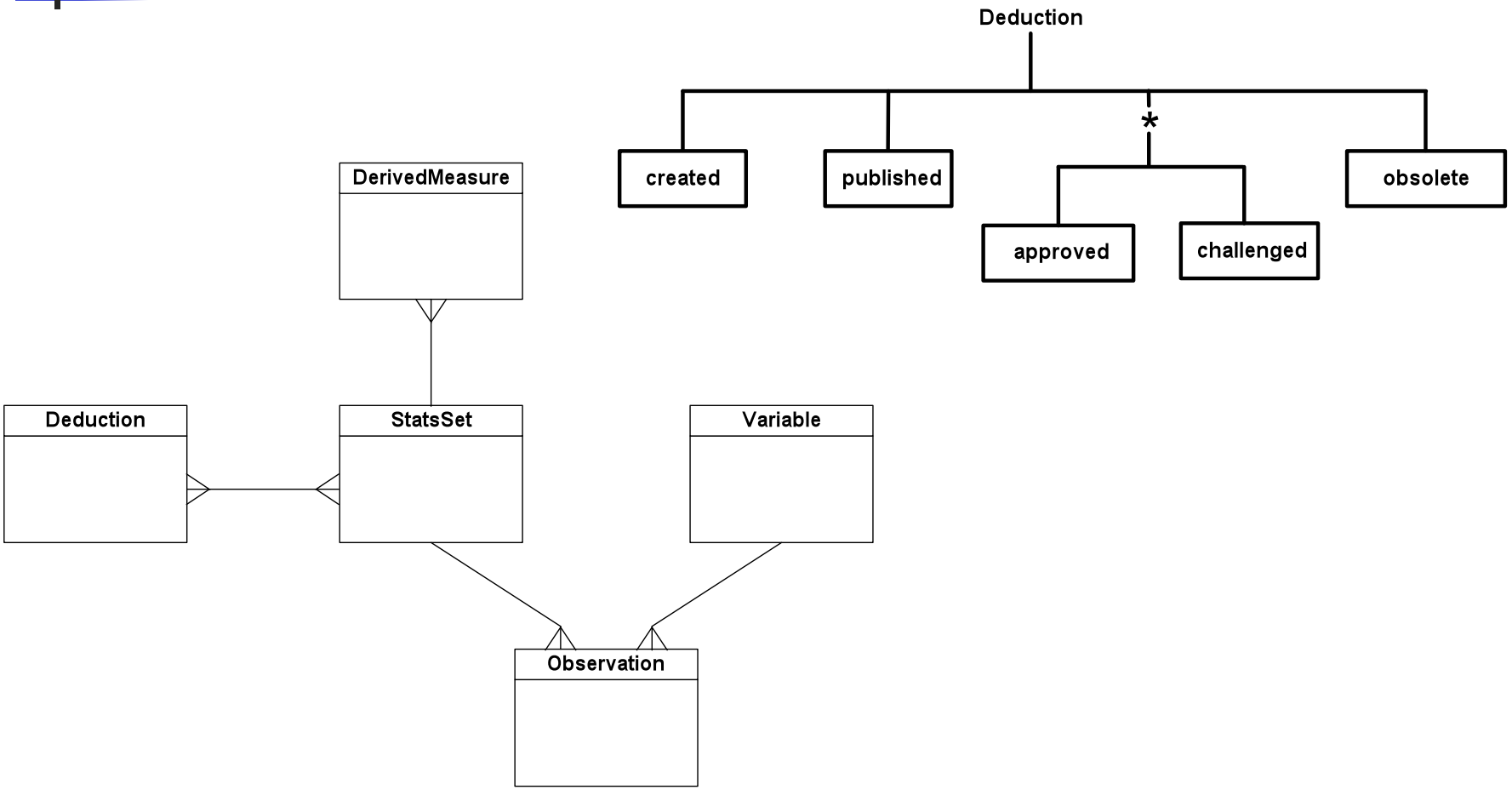
Simple Architecture

- Described through 5 views
  - Functional
  - Information
  - Concurrency
  - Development
  - Deployment
  - *(Operational view omitted)*

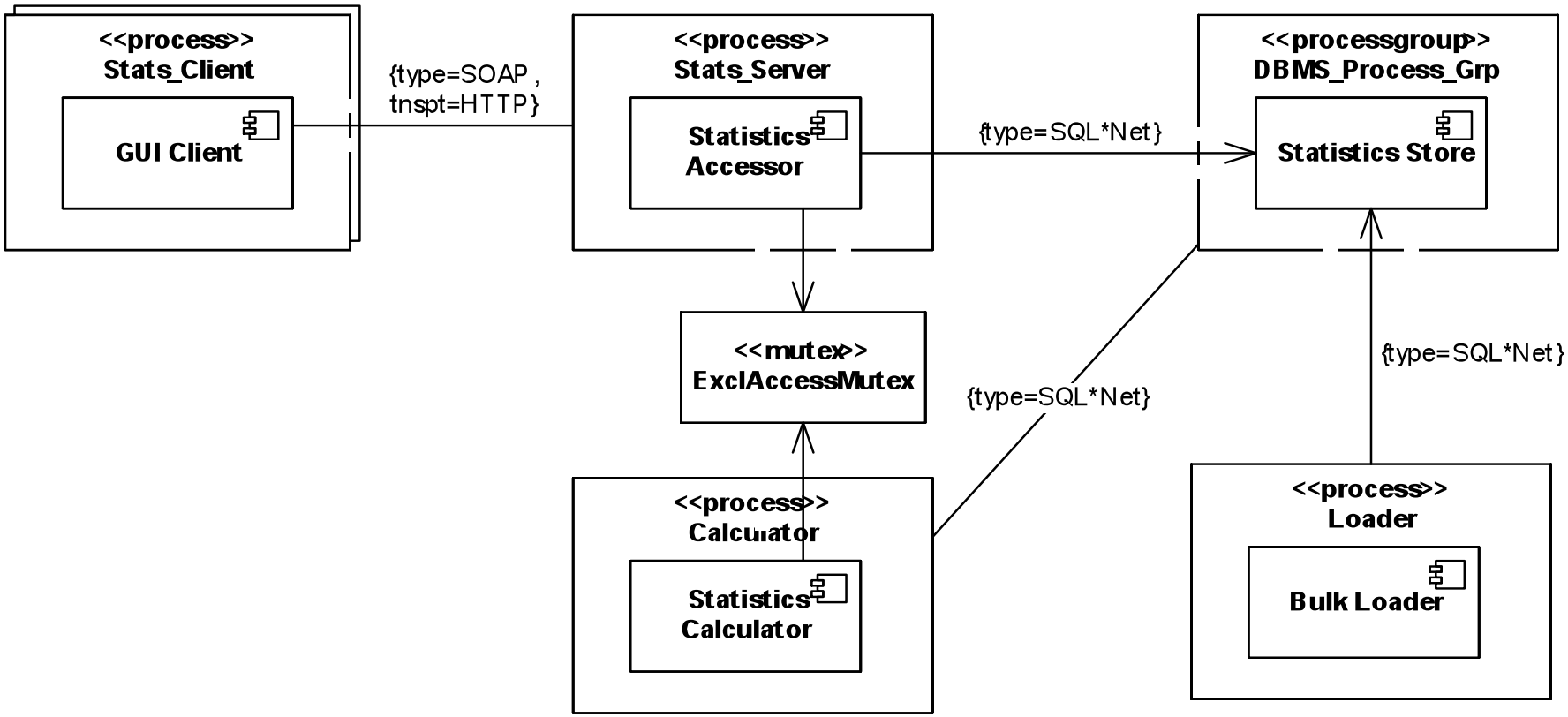
Functional View



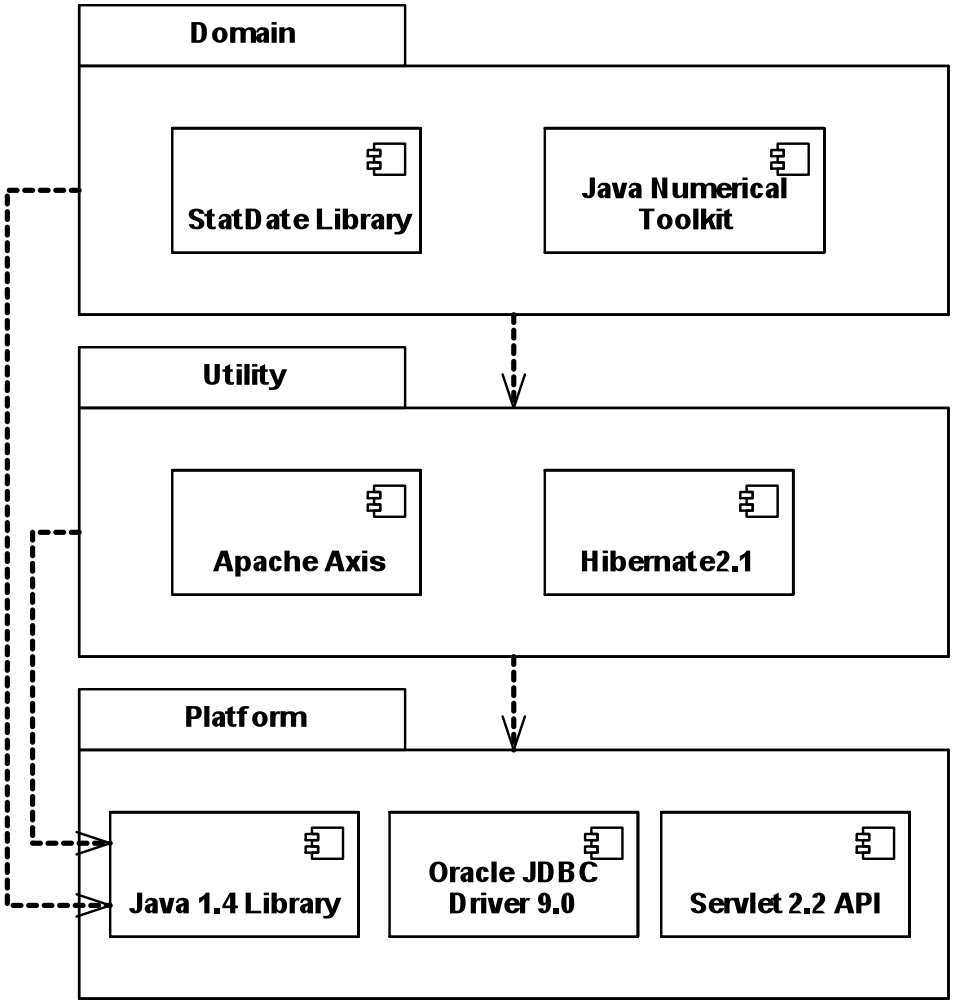
Information View



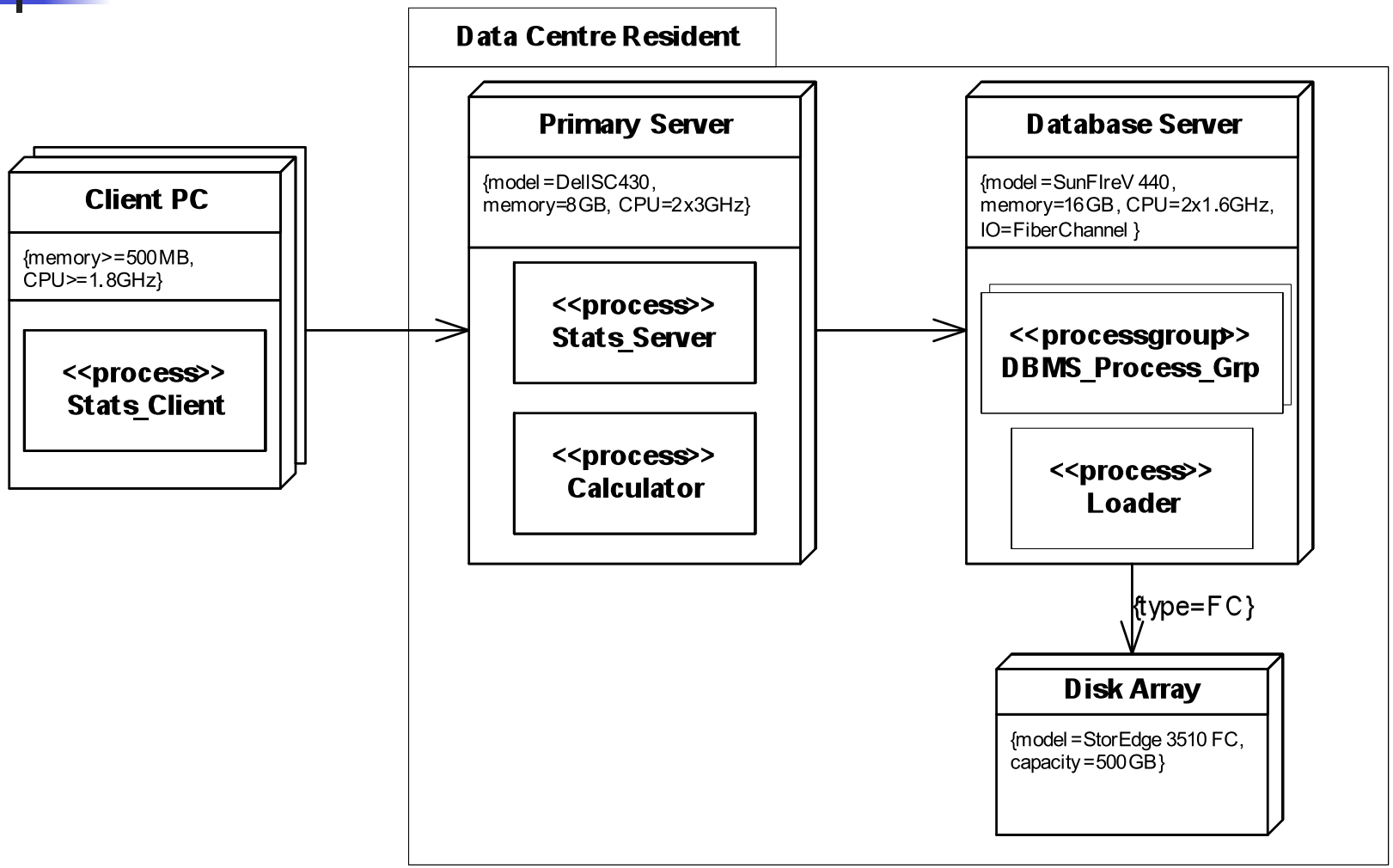
Concurrency View

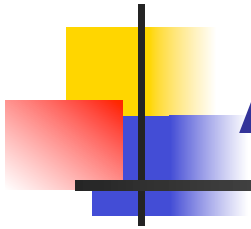


Development View



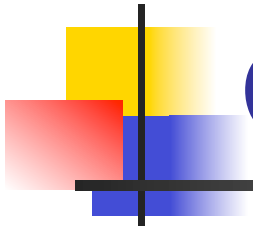
Deployment View





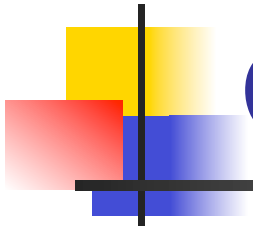
Architectural Impact

- The architecture we've described is credible
- What would happen if the system needed to protect the system's information?
  - Justice community system for criminal intelligence



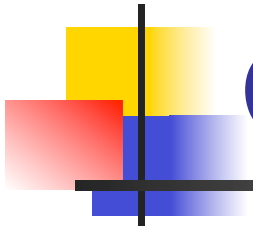
Considering Security

- 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



Considering Security

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

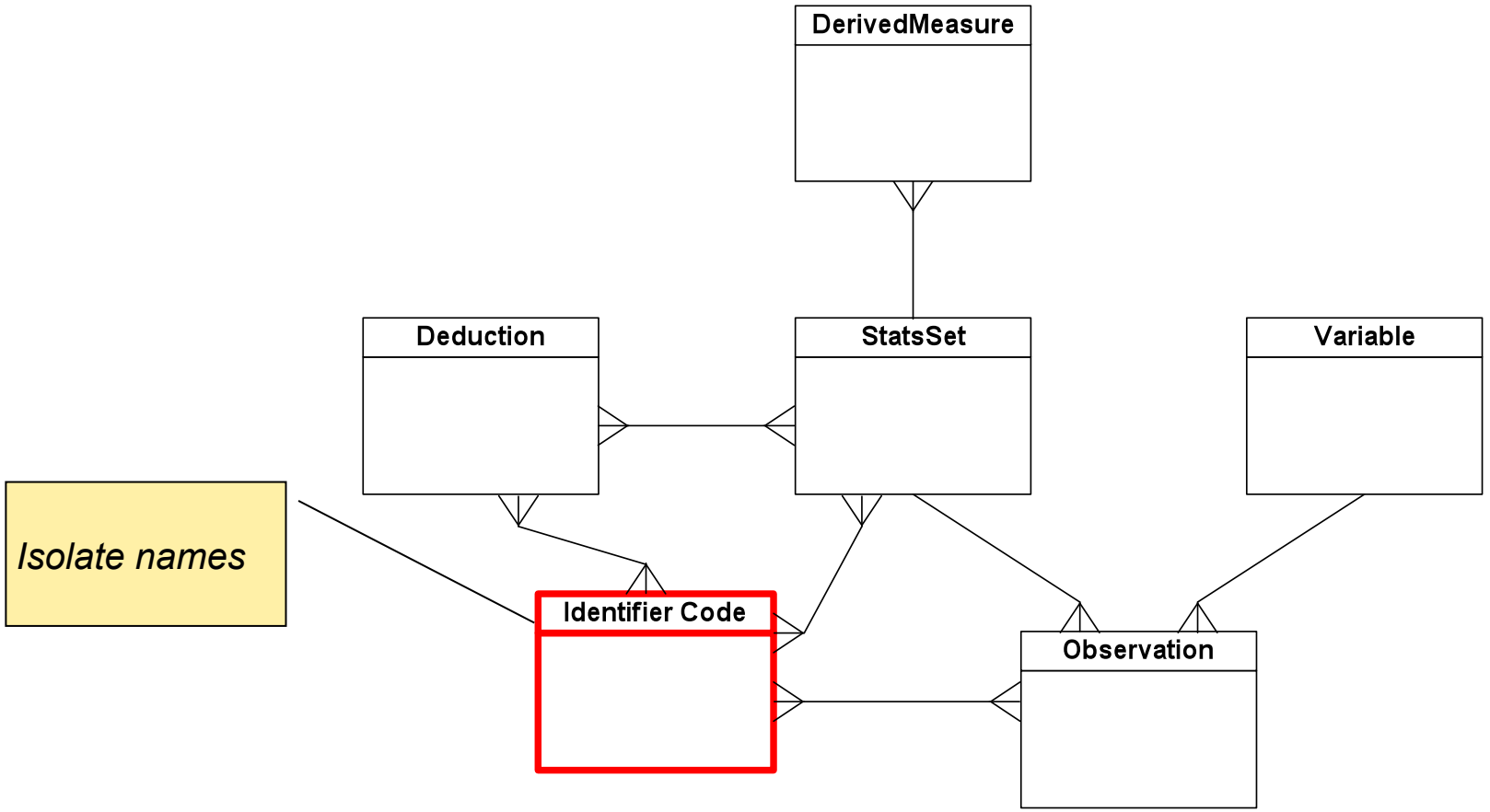


Considering Security

- Security Countermeasures
  - Network Attacks: firewalls, IDS
    - More cost
    - More deployment / administration complexity
    - Operational impact if IDS trips
  - Bribery: add audit trail for data access
    - Possible performance impact
    - More complexity
    - Protecting / using the audit trail

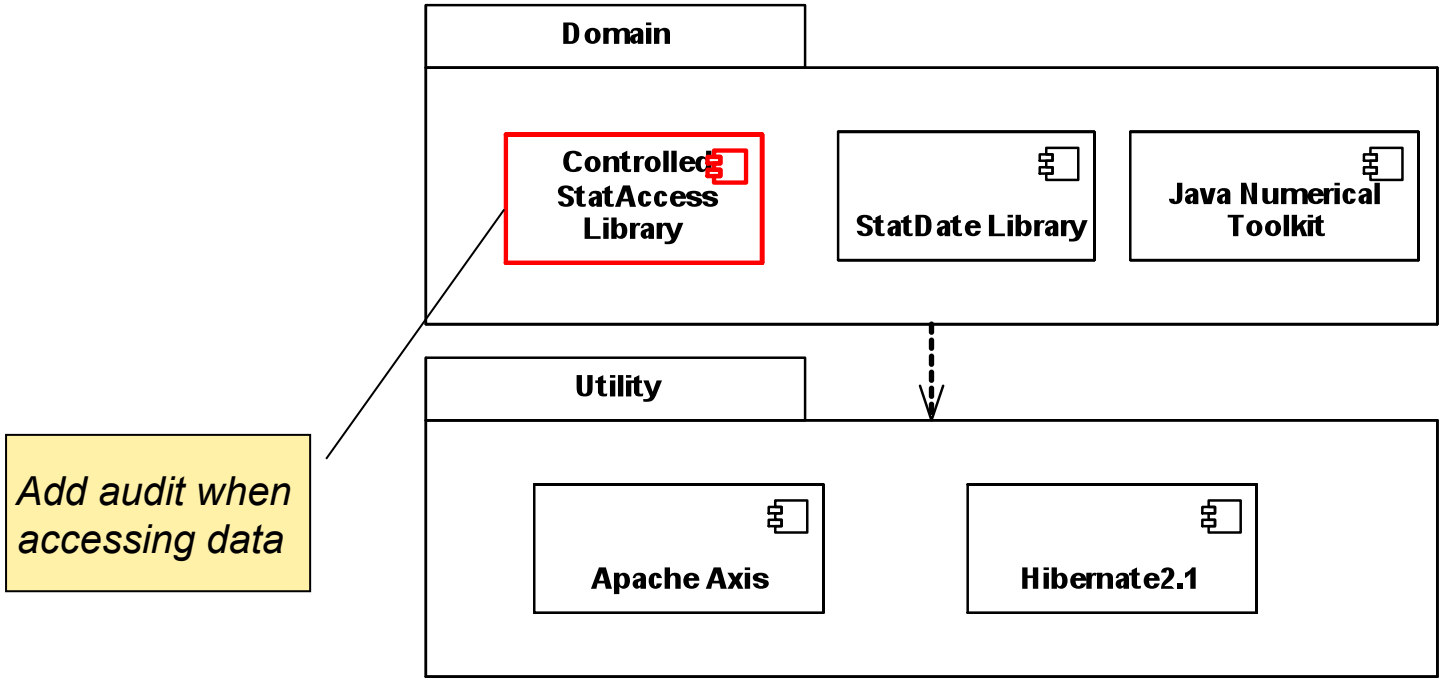
Considering Security

- Information View Impact



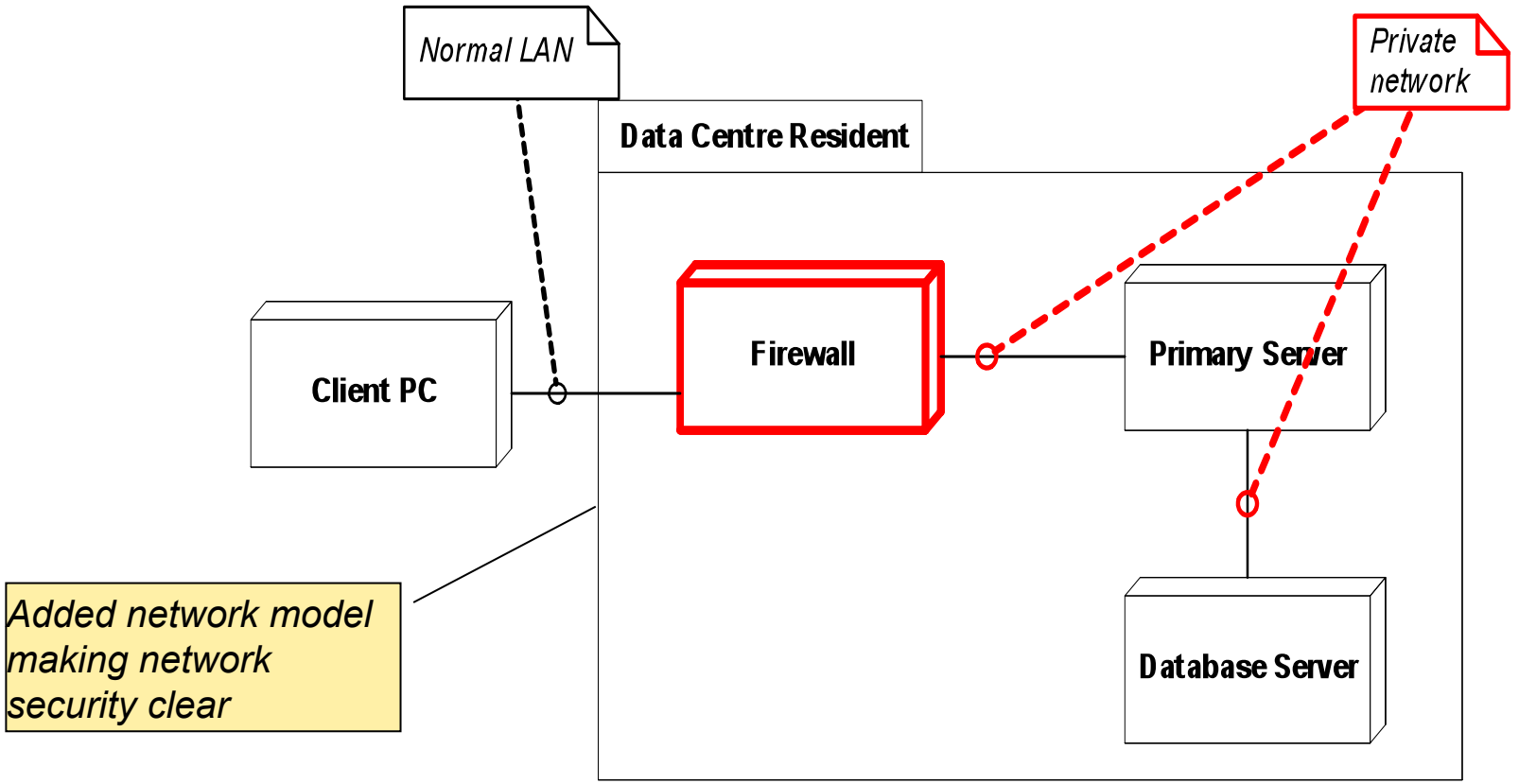
Considering Security

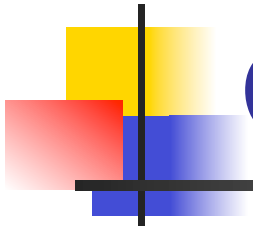
- Development View Impact



Considering Security

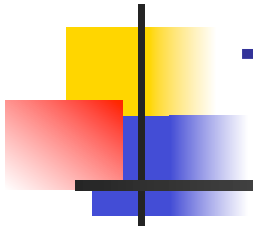
- Deployment View Impact





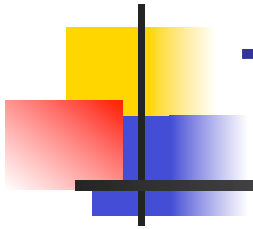
Considering Security

- Other Impact
  - Need IDS added to Development view
  - 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
- This is “architecturally significant”



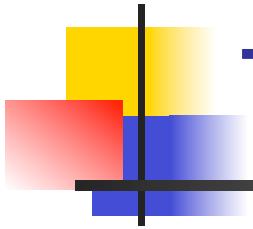
Topics

- Introducing Software Architecture
- The Past and Present
- Exploring the Fundamentals
- An Example of Architecture in Practice
- **The Future**



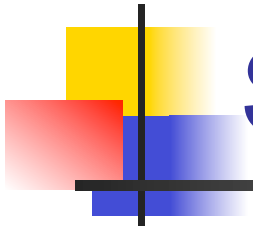
The Future (i)

- Career track recognition
  - Architect vs. developer, tester or PM
  - Possibly certification (e.g. IASA)
- Better description languages & tools
  - Executable and queryable architectures
  - Architecture in the running system
- Architect-specific tool support
  - Lattix and Sotograph are early examples



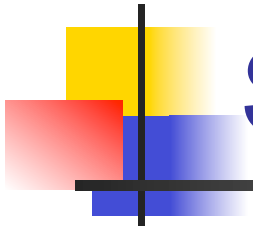
The Future (ii)

- Fundamental agreed definitions
  - Necessary or even desirable?
- Teaching
  - MSc in software architecture perhaps?
- Further styles codified as technologies
  - Grid, tuple-space, P2P, …



Summary (i)

- Software architecture is still young
  - Really a product of 1995 – 2005
- Mainstream since about 2002
- Good core of knowledge emerging
  - Approaches and techniques
- Some agreement on fundamentals
  - Stakeholders, structures, qualities
  - Understanding, designing, trading-off, delivering



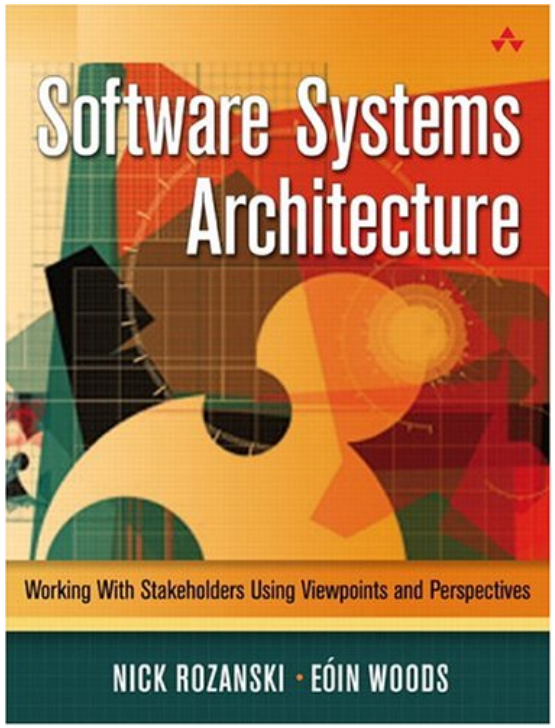
Summary (ii)

- Finally research & practice meeting
  - WICSA 5, IASA, Microsoft, …
- The future is architecture in the systems
  - Too much is lost today

To Learn More

Software Systems Architecture: Working With Stakeholders Using Viewpoints and Perspectives

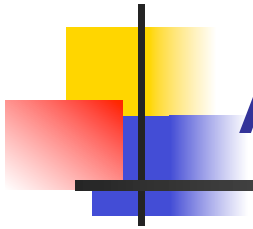
Nick Rozanski & Eoin Woods
Addison Wesley, 2005



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

Eoin Woods
eoin@artechra.com
www.eoinwoods.info

Comments and Questions?



Abstract

- The activity of "software architecture" has quite recently emerged, alongside the venerable quartet of traditional software development activities, specification, design, code and test. Why has this new activity emerged and what is it? Is it not just high level design? In this talk, Eoin Woods, a participant in the IFIP International Working Group 2.10 on Software Architecture, will trace the emergence of this new activity, explain what makes software architecture unique, and provide some thoughts on where this new specialisation is headed.