Multi-view Architecting

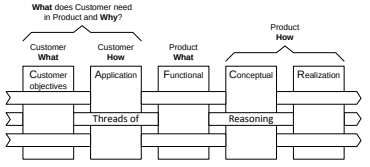
by *Gerrit Muller, JürgenMüller, Jan Gerben Wijnstra* Buskerud University College,
Philips Research
e-mail: gaudisite@gmail.com
www.gaudisite.nl

Abstract

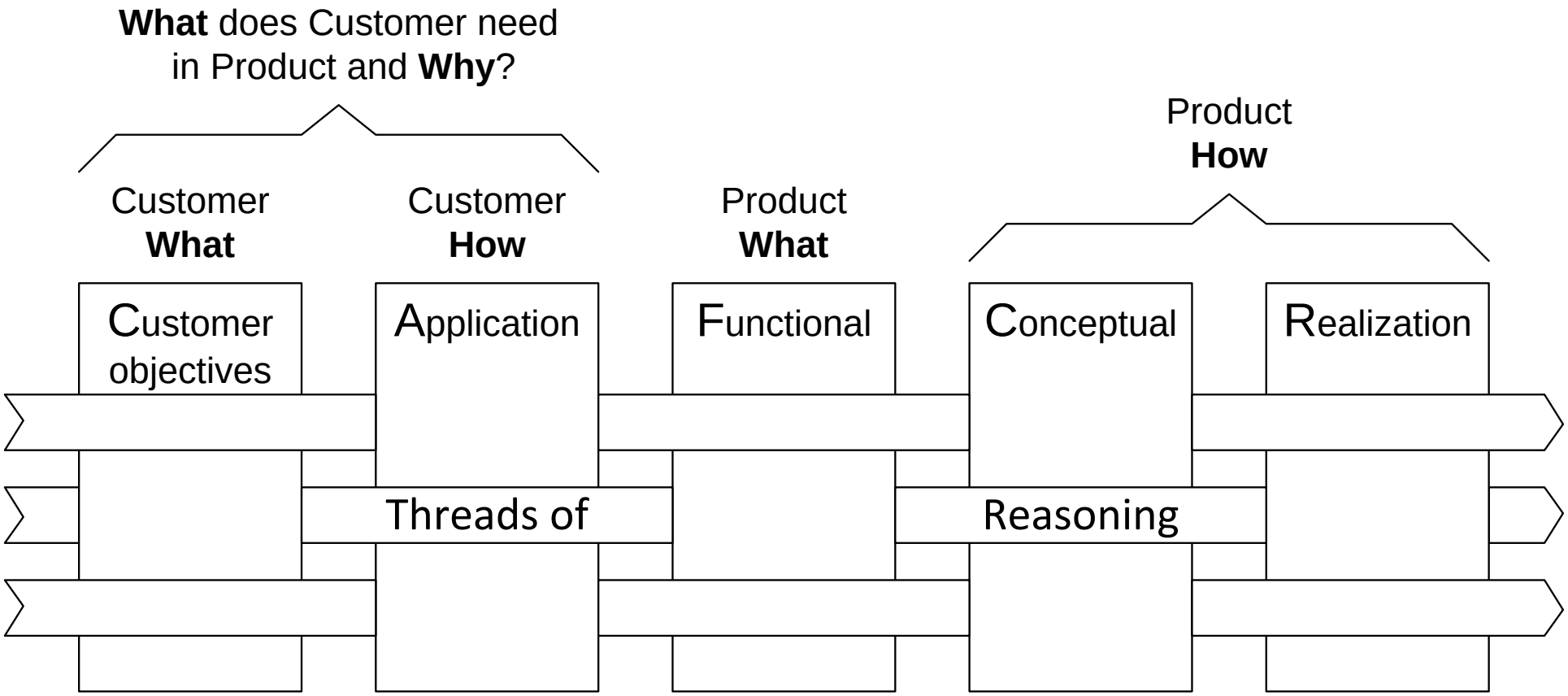
The development of large SW-intensive products needs to take requirements of multiple stakeholders into account. A design of such a system has to address functional and quality requirements adequately. However, for most of the required qualities no straight-forward design method exists even for a single quality.
A multi-view architecting model is described based upon a decomposition of an architecture in 5 architectural views, ranging from customer objectives to realization. It is the task of the architect to keep these views consistent and to balance design decisions in the perspective of the stakeholder needs.
We derived this model from our experience in developing software intensive industrial products, 2 cases are described from the medical domain.

This paper has been written as part of the "Composable" project in Philips and the Gaudí project.

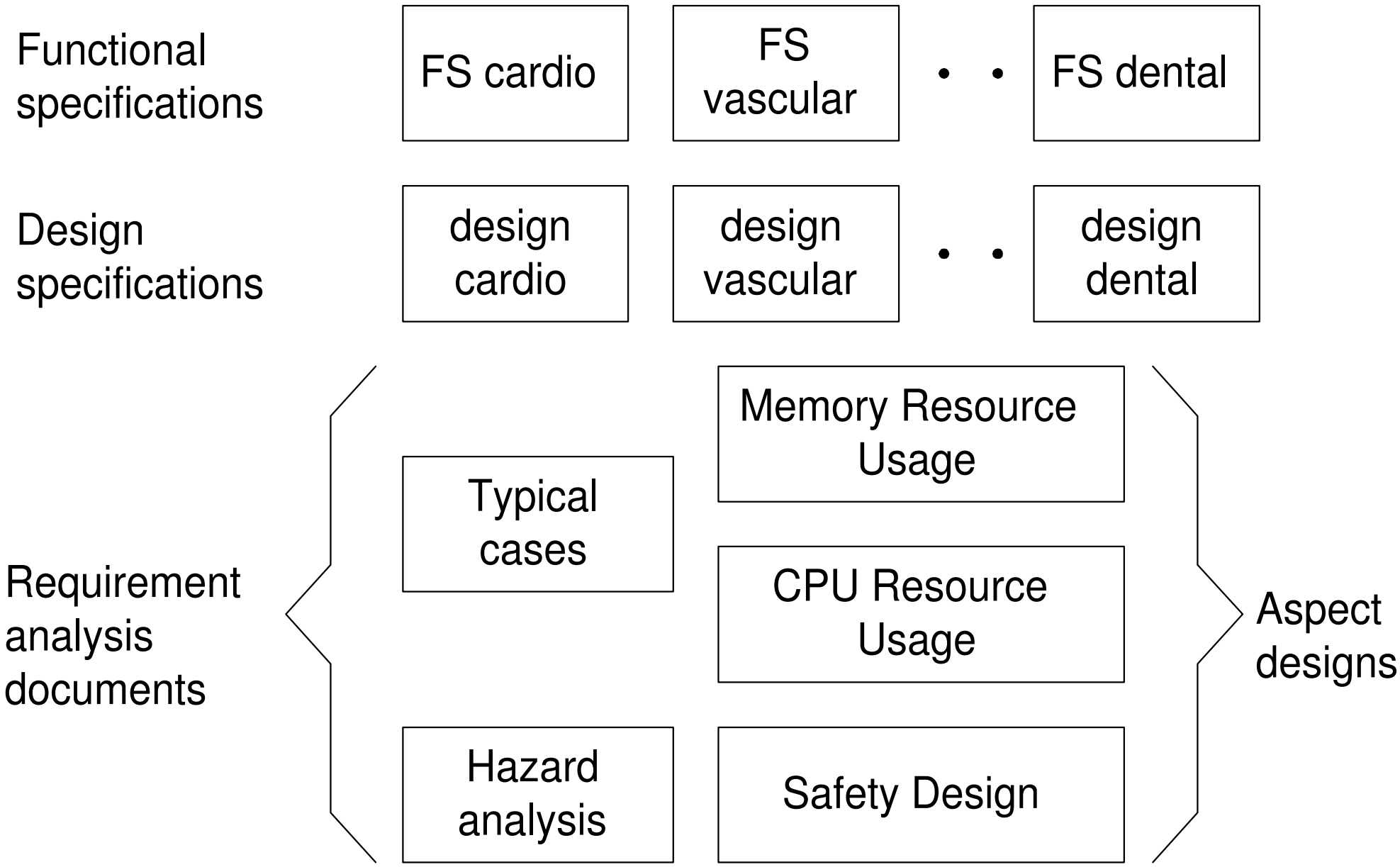
October 20, 2017
status: finished
version: 1.0



Integrating 5 System Architecture Views

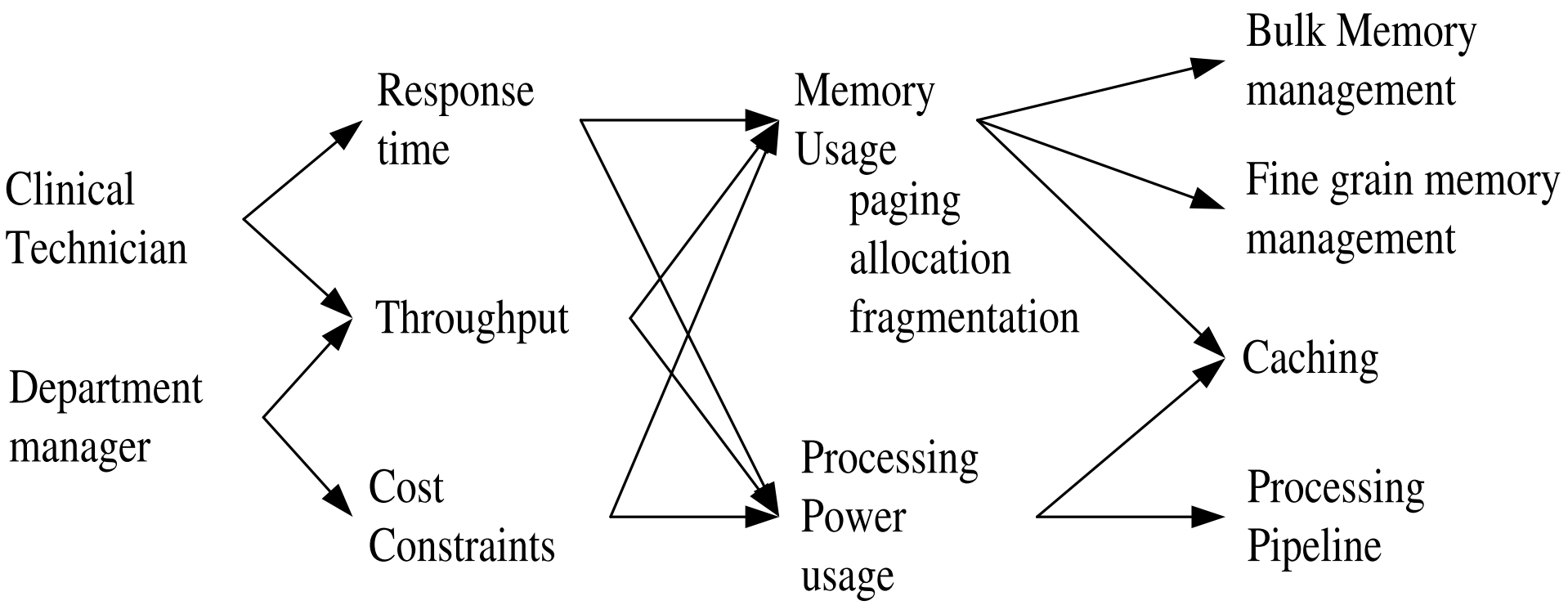


Toplevel documentation structure Medical Imaging Workstation

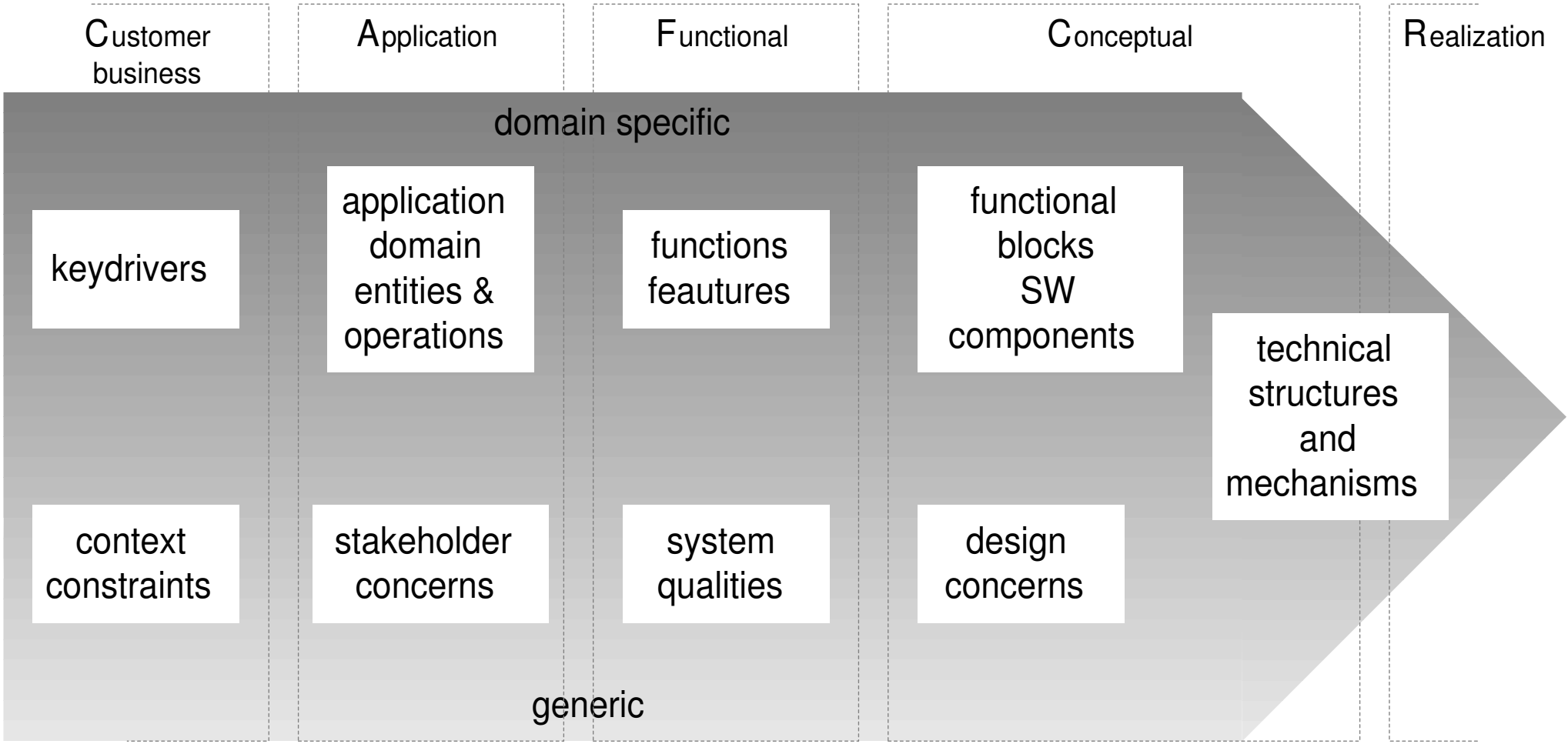


Memory usage aspect Medical Imaging Workstation

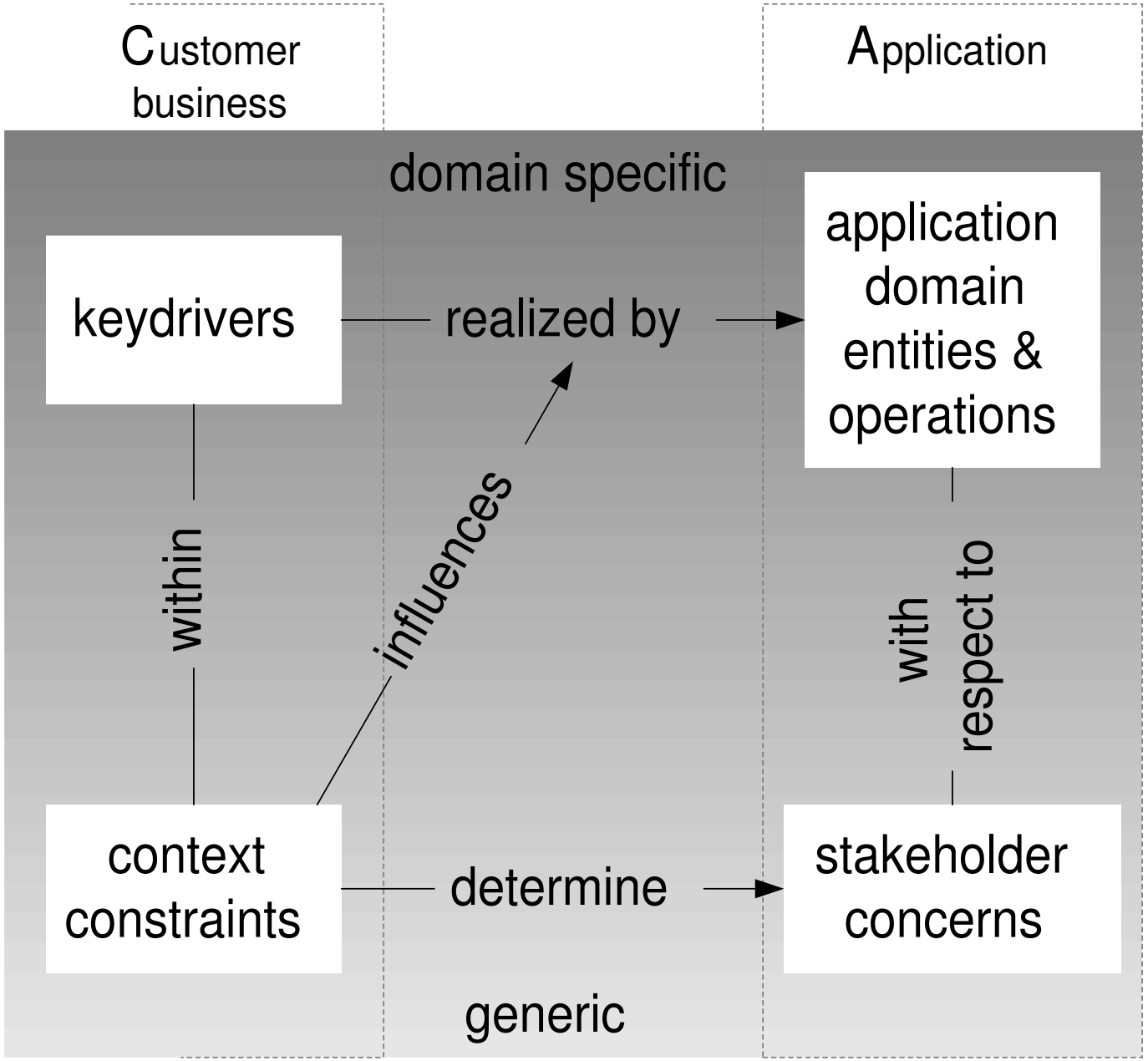
Application	Functional	Conceptual	
Stakeholders	Quality Attributes	Design Concerns	Technical Structures and Mechanisms



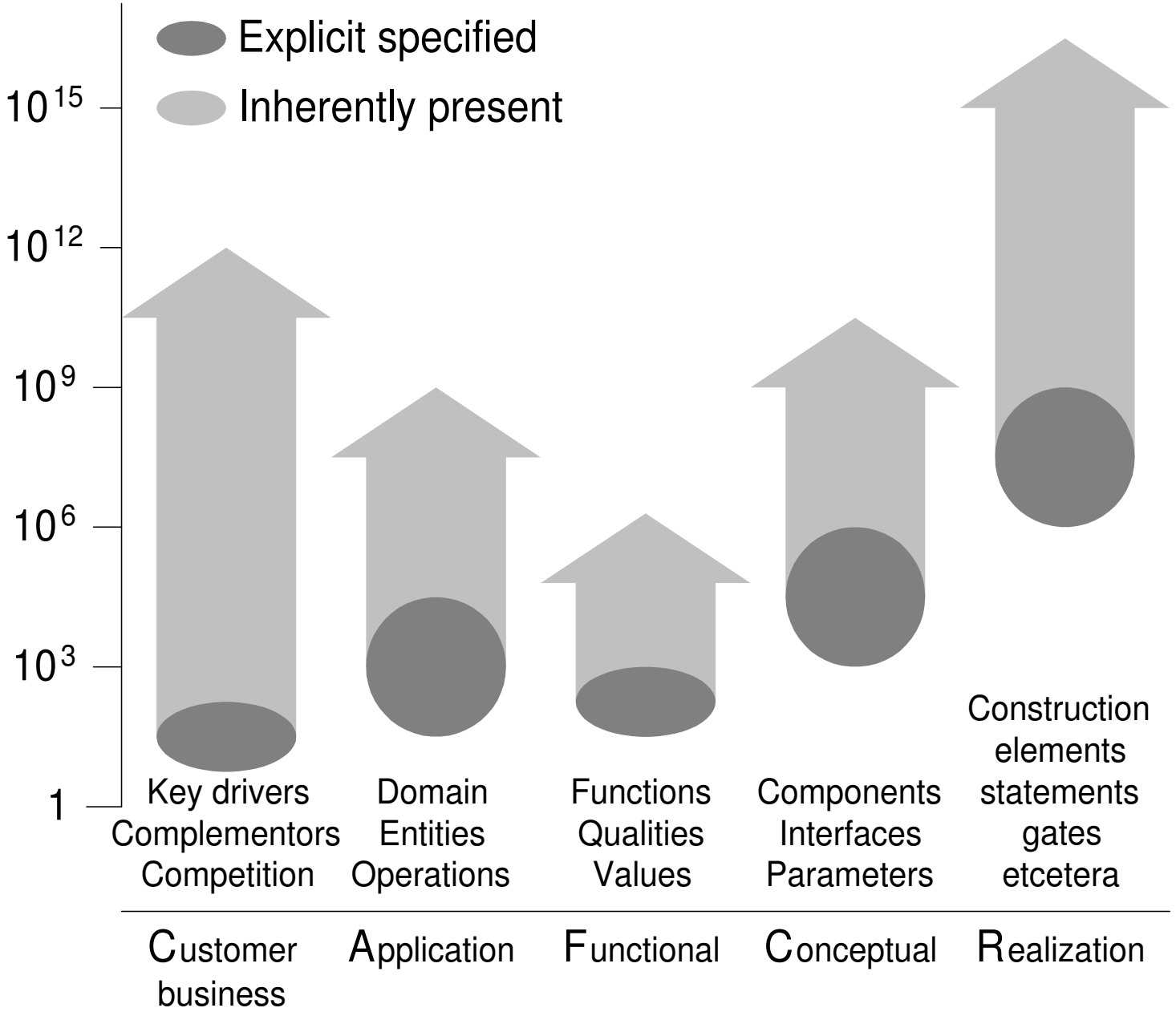
Issues per view



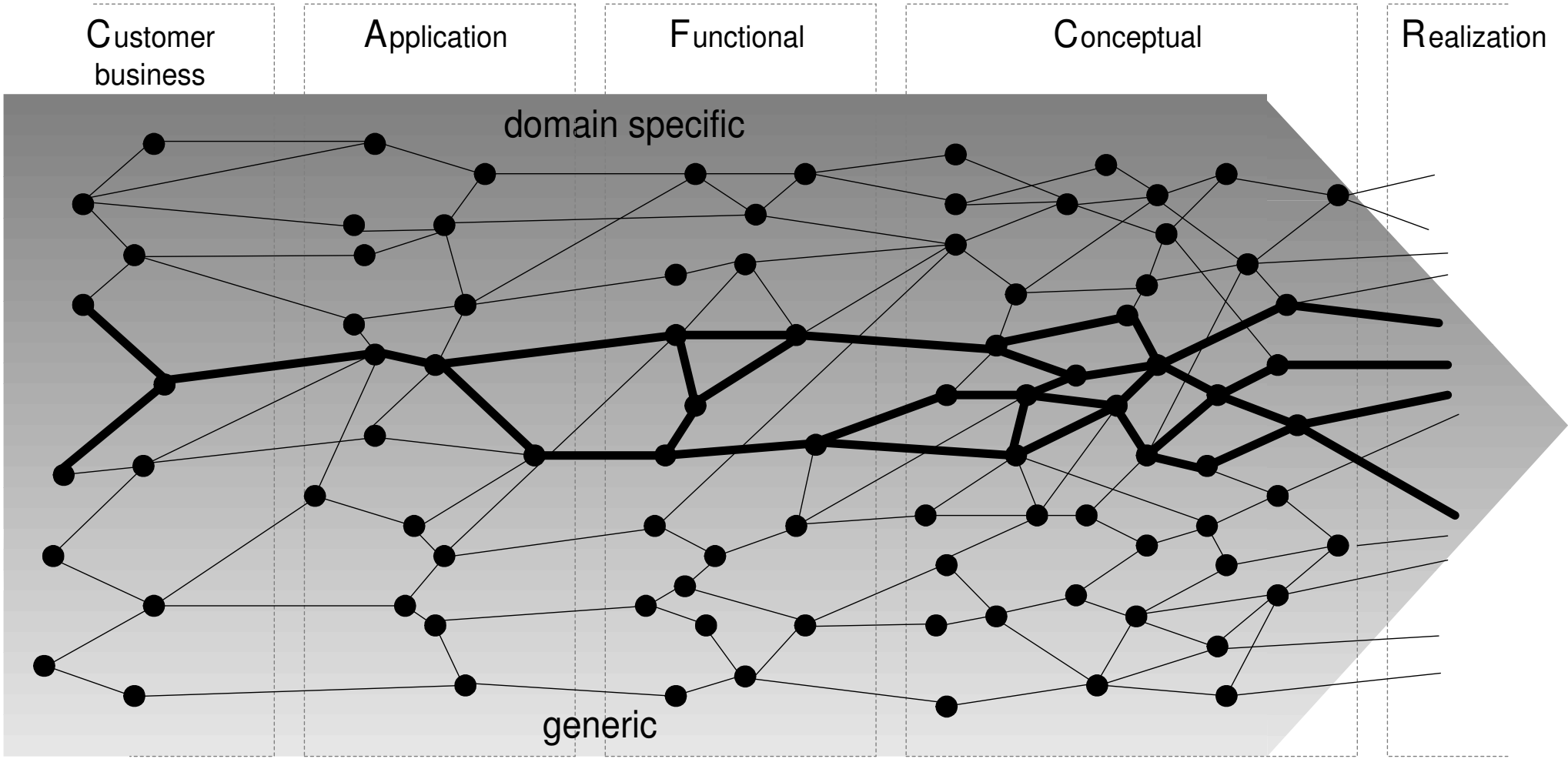
Zooming in on relations



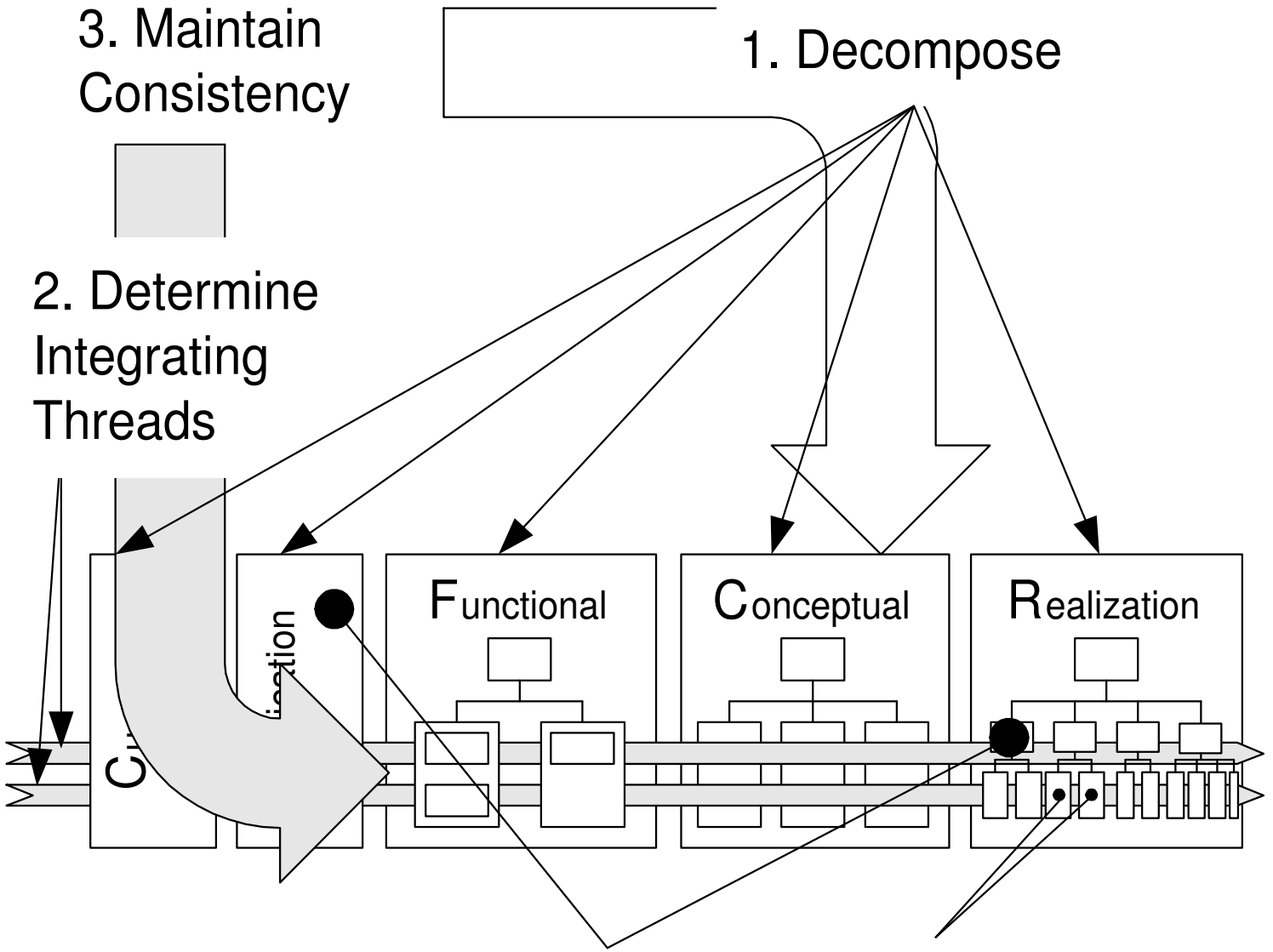
Explicit facts and inherent details per view



One thread of reasoning



Activities in multi-view architecting



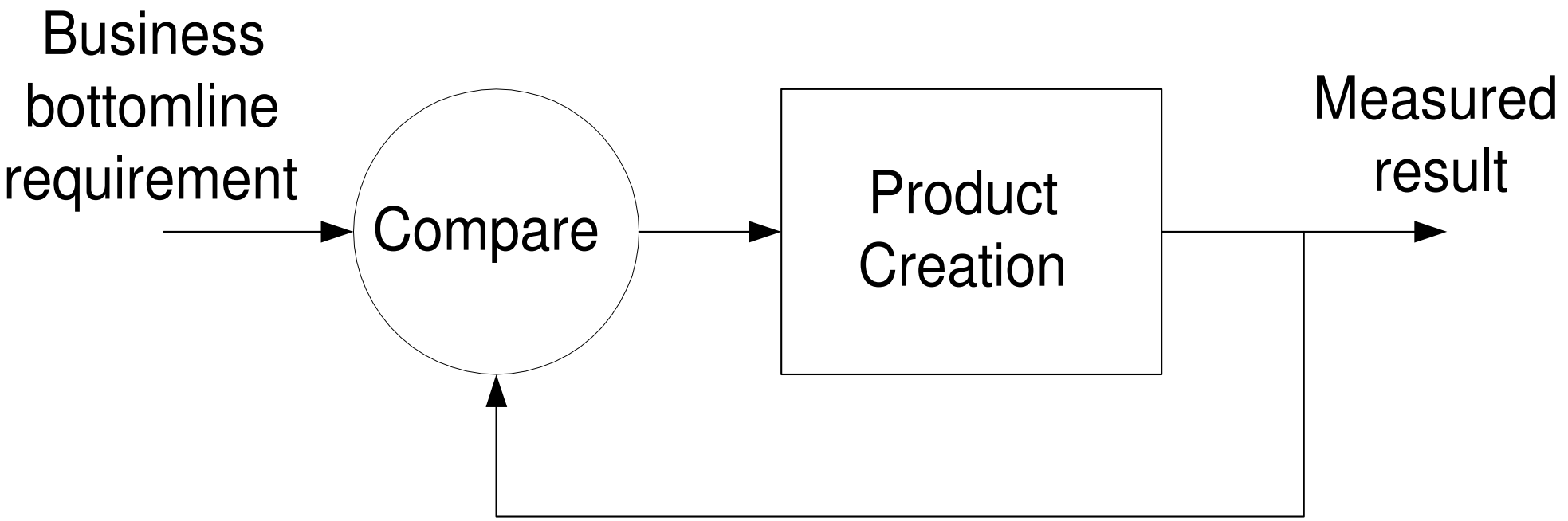
Criteria for thread selection

- Important for customer and the business
- Critical with respect to technical realization

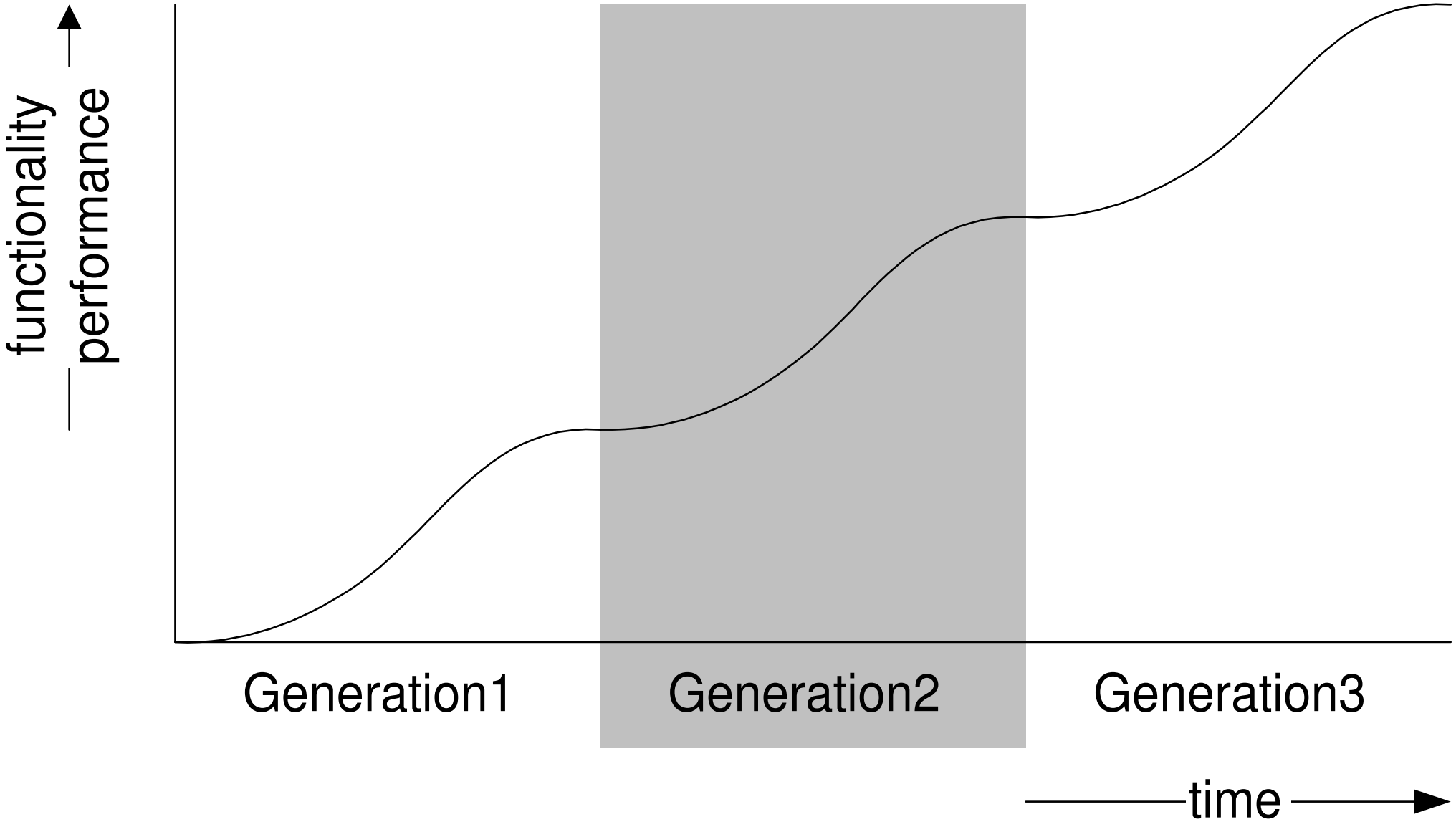
Pitfalls in multi-view architecting

- too few views
- completeness
- general formalization

Continuous feedback during Product Creation



Stepwise evolution



Qualities checklist

Safety
Security
Reliability
Robustness
Useability
Appeal, Appearance
Throughput or Productivity
Response Time
Image Quality
Reproduceability
Predicatability
Accuracy
Transportability
Wearability
Storability

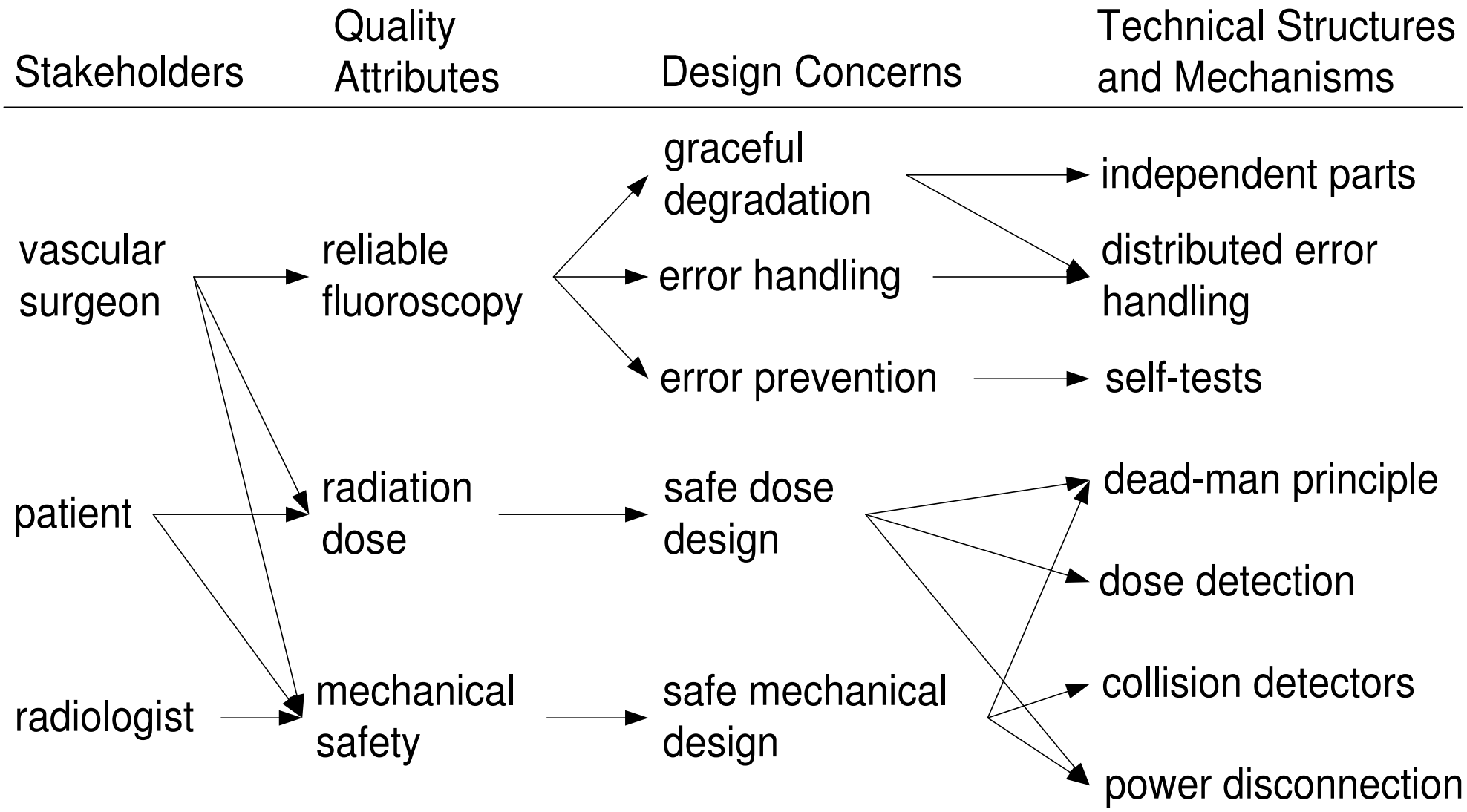
Manufacturability
Testability
Serviceability
Configurability
Installability
Evolvability
Portability
Upgradeability
Extendability
Maintainability
Logistics flexibility
Lead time
Standards Compliance

Cost price
Cost of operation
Interaction with environment
Power consumption
Consumption rate (water, air, chemicals, etcetera)
Disposability
Size, weight
Resource utilization

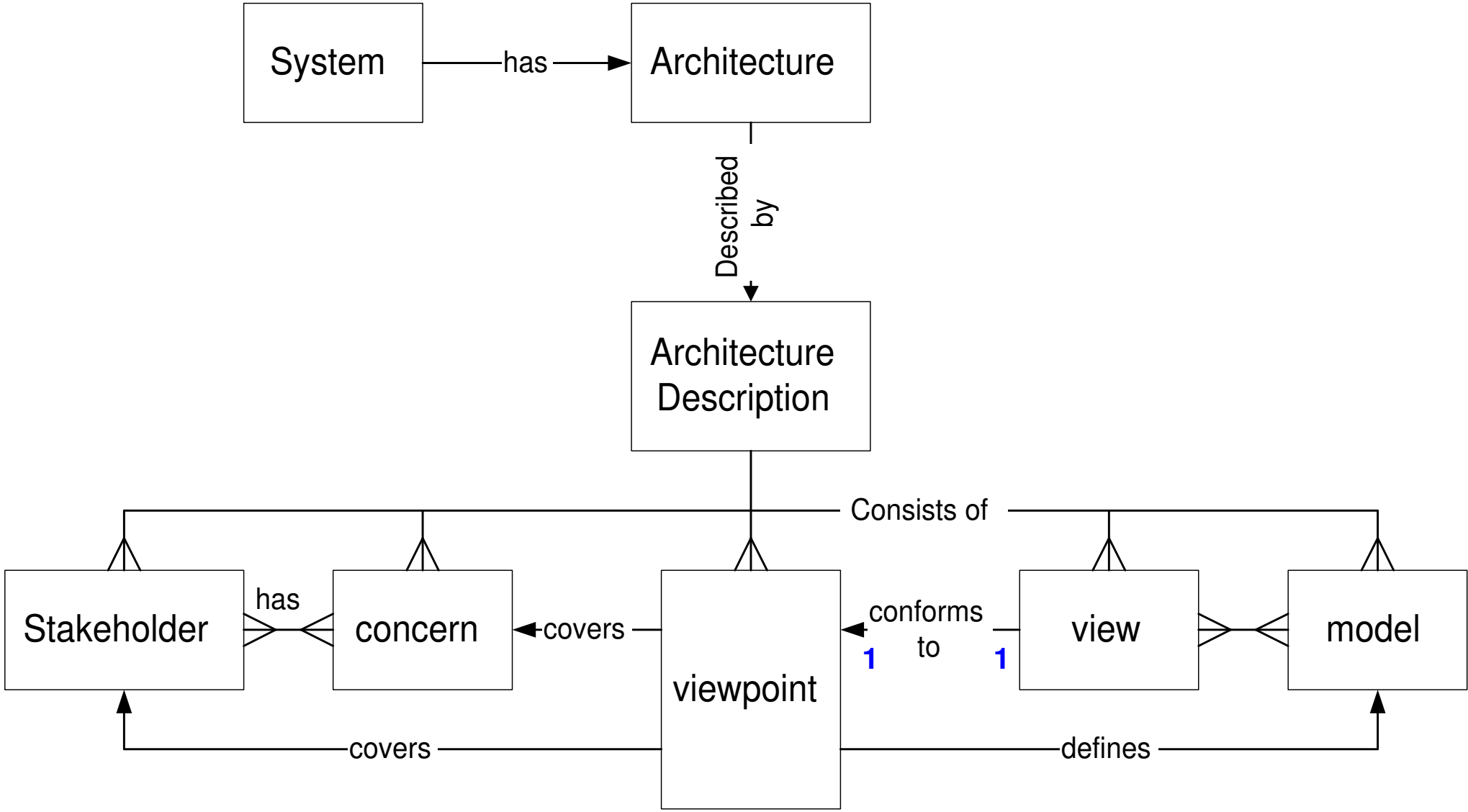
SW aspects checklist

- granularity
  - scoping
  - containment
  - cohesion
  - coupling
- interfaces
- allocation
  - budgets
- information model
  - entities
  - relations
  - operations
- characteristics
  - static
  - dynamic
- configuration man.
  - packages
  - components
  - files
  - objects
  - modules
  - interfaces
- meta-functional
  - operational
    - image processing
    - handling calls
    - ..
  - initialization
    - start-up
    - shutdown
    - bootstrap
    - discovery
    - negotiation
  - fault handling
    - exceptions
    - logs
    - traces
  - diagnostics
  - configuration handling
  - data replication
  - performance observation
  - capability query
  - testing
    - automation
    - special methods
    - harness
    - suites
  - off-line guidance
- supply chain
  - outsource
  - co-design
  - buy
  - interoperate
  - source vs binary
- technology choices
  - lifecycle
  - obsolesence
  - core, key, base
- SW development
  - environment
  - repository
  - tools
- feedback tools
  - monitoring
  - statistics
  - analysis
  - call graphs
  - message tracing
  - object tracing
- licensing
  - SW keys
- synchronization
  - signalling
  - messaging
  - call-back scheduling
  - notification
  - active data
  - watchdogs
  - time-outs
  - locking
  - semaphores
  - transactions
  - checkpoints
  - deadlock detection
  - roll-back
  - priorities
  - pre-emption
- concurrency
  - processes
  - tasks
  - threads
- persistence
  - caching
  - versioning,
  - prefetching
  - lazy evaluation
- identification
  - uniqueness
  - naming
  - data model,
  - registry
  - scoping
  - configuration
  - database
  - inheritance
- resource management
  - allocation
  - anti-fragmentation
  - garbage collection
- distribution
  - allocation,
  - transparency
  - component,
  - client/Server
  - multi-tier model
- infrastructure

Medical Safety



IEEE 1471 model



ISO 9126 quality framework

- **Functionality** suitability, accuracy, interoperability, compliance, security, *traceability*
- **Reliability** maturity, fault tolerance, recoverability, *availability, degradability*
- **Usability** understandability, learnability, operability, *explicitness, customisability, attractivity, clarity, helpfullness, user-friendliness*
- **Efficiency** time behaviour, resource behaviour
- **Maintainability**
- **Portability**