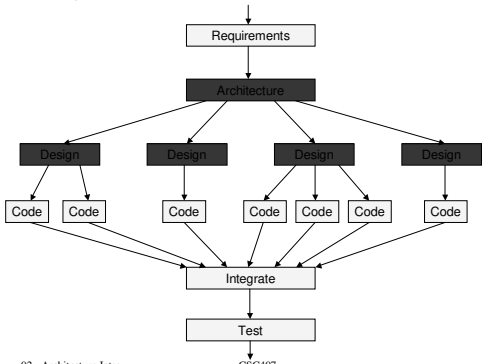
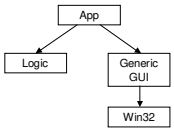
Development Process



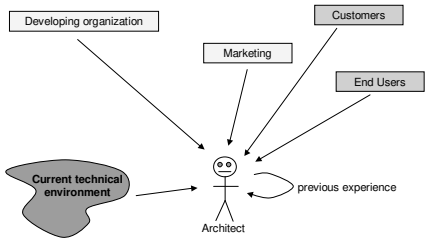
Definition

– A "software architecture" is the structure (or structures) of a system, which comprise
  - software components,
  - the externally visible properties of those components,
  - and the relationships among them.



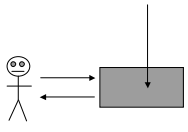
- Architecture defines "components"
  – an abstraction
  – suppresses details not pertinent to its interactions with other components
- An architecture can comprise more than one structure
  - modular structure (calls/uses)
  - process structure (invokes, communicates with, synchronises with)
  - physical structure (libraries, DLL's, processors)
  - inheritance structures (inherits)
  - …

Where does architecture come from?



What does architecture affect?

– The structure of the developing organisation
– The enterprise goals of the developing organisation
– customer requirements for the next system
– influence later architectural decisions



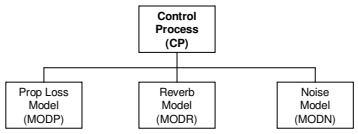
Why is architecture important?

– Manifests early design decision
  - most difficult to get correct and hardest to change
  - defines constraints on the implementation
  - inhibits or enables quality attributes
– Defines a work-breakdown structure
  - organization (especially important for long-distance development)
  - estimation
– A vehicle for stakeholder communication
  - an architecture is the earliest artefact that enables the priorities among competing concerns to be analysed
– Reviewable
  - architectural errors are vastly more expensive to fix once a system has been coded
– Can serve as a basis for training new developers

Architecture process steps

- create the business case
- understand the requirements
- create the architecture
- represent and communicate the architecture
- evaluate the architecture
- implement based on the architecture
  - ensuring conformance
- enhance/maintain based on the architecture
  - ensuring conformance

How do we describe an architecture?



- What is the nature of the components?
- What is the nature of the links?
- Does the layout have any significance?
- How does it operate at runtime
  - Dataflow
  - Control flow
- Can we evaluate this architecture?

Functionality & Quality Attributes

- Functionality usually takes 1st place during development.
- Systems are more frequently re-designed not because they are functionally deficient, but rather because
  - They are difficult to maintain
  - Difficult to port
  - Won't scale
  - Too slow
  - Too insecure
  - Not fault tolerant

Quality Attributes

- System qualities
  - Observable via execution
  - Not observable via execution
- Business qualities
- Architectural qualities

System Qualities

- Observable via execution
  - Performance
  - Security
  - Availability
    - Reliability = mttf = mean time to failure
    - Availability = mttf/(mttf + time to repair)
  - Functionality
  - Usability
- Not observable via execution
  - Modifiability
  - Portability
  - Reusability
  - Integrability
  - Testability

Business Qualities

- Time-to-market
- Cost
- Projected lifetime
- Target market
- Rollout schedule
- Use of legacy systems

Architectural Qualities

- Conceptual integrity
- Correctness
- Completeness
- Buildability
  - Completed by available team in a timely manner

Architectural Means of Achieving Quality

- Two questions
  - What structure shall I employ to
    - Assign workers
    - Derive a work breakdown
    - Exploit pre-packaged components
    - Plan for modification
  - What structure shall I employ so that the system, at runtime, fulfills its behavioral and quality attributes.

Two Structures

- Modular structure
  - Purely static
  - Disappears at run-time
- Structures that survive through execution
  - E.g., pipes, processes, networks, …
- Both views need to be considered (not the same)