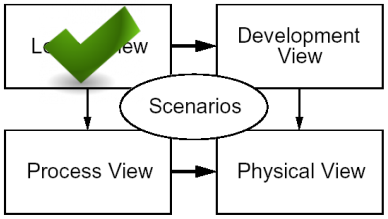
Documenting Software Architecture

4+1 view model continue…

4+1 View model

End-user
Functionality

Programmers
Software management



Integrators
Performance
Scalability

System engineers
Topology
Communications

The Process View: Integrators

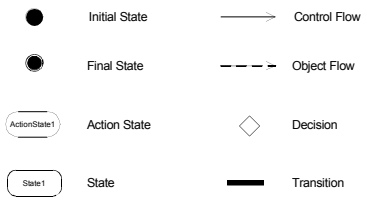
- √ Focus on non-functional requirements:
  - Performance (scalability)
  - Availability
- √ Addresses issues like:
  - Concurrency and distribution
  - System integrity
  - Fault tolerance
  - Execution threads
- √ From logical view:
  - How main abstractions fit with processing architecture

The Process View: Abstractions

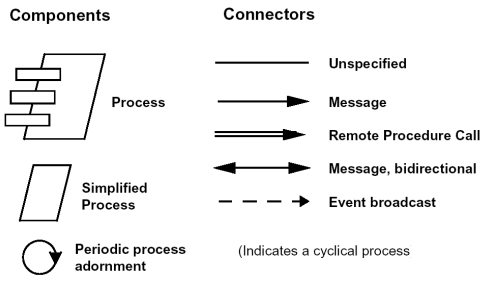
√ Abstraction levels of the process view:

- Logical networks of independently executing ***Processes***:
  - Distributed across hardware resources connected by LAN/WAN
  - *Process* is grouping of *tasks* that form an executable unit.
  - *Processes* can be tactically controlled.
  - *Processes* can be replicated for improved performance and availability.
- Software is partitioned into a set of independent ***Tasks***:
  - Separate thread of control.
  - Can be individually scheduled on one processing node.
  - *Major tasks:* Architectural elements that can be uniquely addressed
  - *Minor tasks:* Introduced for implementation reasons.

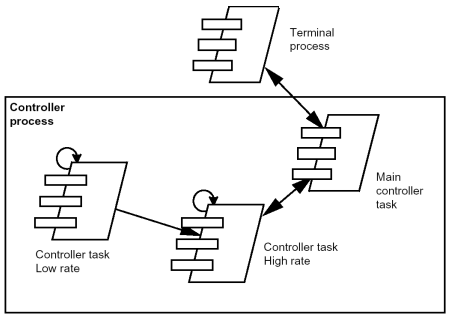
Notation for Processing View (UML)



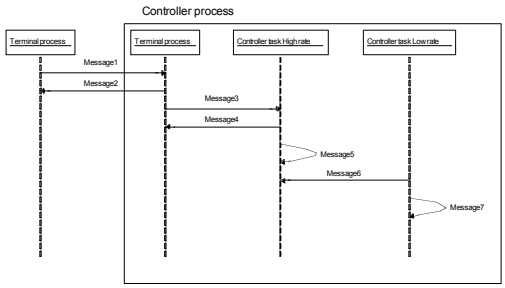
Notation for Processing View (Booch)



Process view: Example of Phone control



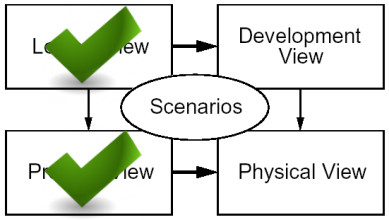
Process view (UML): Example of Phone control



4+1 View model

End-user
Functionality

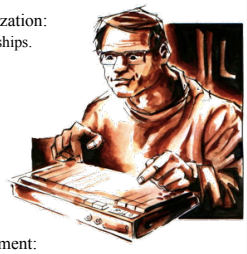
Programmers
Software management



Integrators
Performance
Scalability

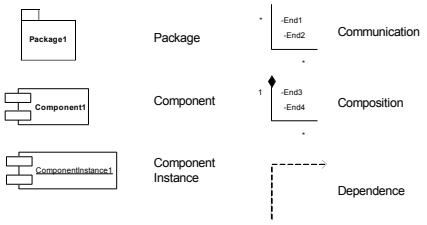
System engineers
Topology
Communications

The Development View: Programmers

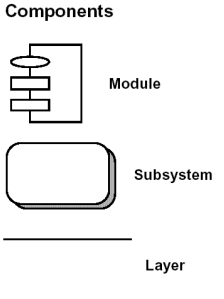


- √ Focus on actual software module organization:
  - Subsystems with export/import relationships.
- √ Software packaged in small chunks:
  - Subsystems can be developed by developers.
  - Subsystems are organized in hierarchy of layers.
  - Each layer provides well-defined interface to layers above.
- √ Rules for Development view:
  - Partitioning, grouping, visibility.
- √ Development view should ease development:
  - Software management, reuse, commonality, constraints imposed by toolset or programming language.
  - Foundation for organization, cost planning, monitoring etc.

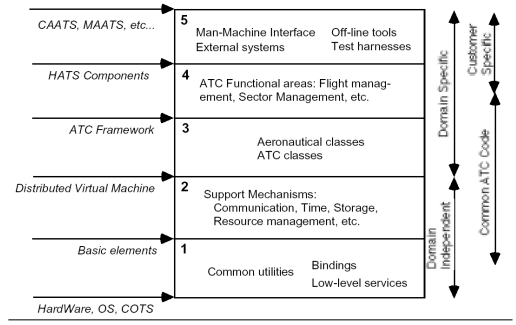
Notation for Development View (UML)



Notation for Development View (Booch)



Development View Example (Layered): Air Traffic System



Development View Example (UML): Air Traffic System