Applied Architectures, Part 2

Software Architecture
Lecture 18

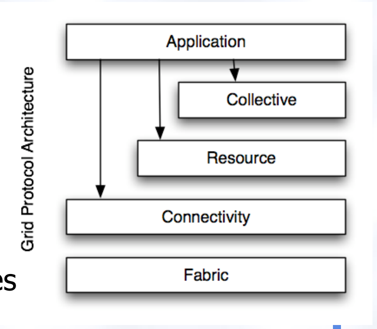
Decentralized Architectures

- Networked applications where there are multiple authorities
- In other words
  - Computation is distributed
  - Parts of the network may behave differently, and vary over time

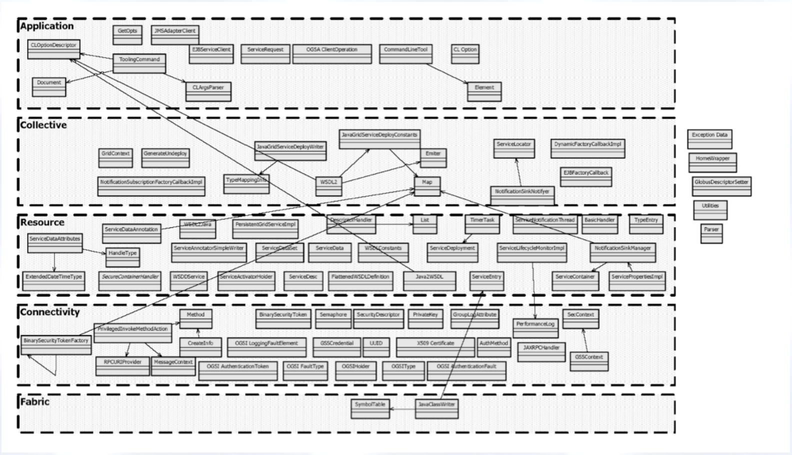
It's just like collaboration in the real world

Grid Protocol Architecture

- Coordinated resource sharing in a distributed environment
  - E.g., Folding-at-home
- GLOBUS
  - A commonly used infrastructure
- "Standard architecture"
  - Fabric manages low-level resources
  - Connectivity: communication and authentication
  - Resource: sharing of a single r.
  - Collective: coordinating r. usage



Grid GLOBUS (Recovered)



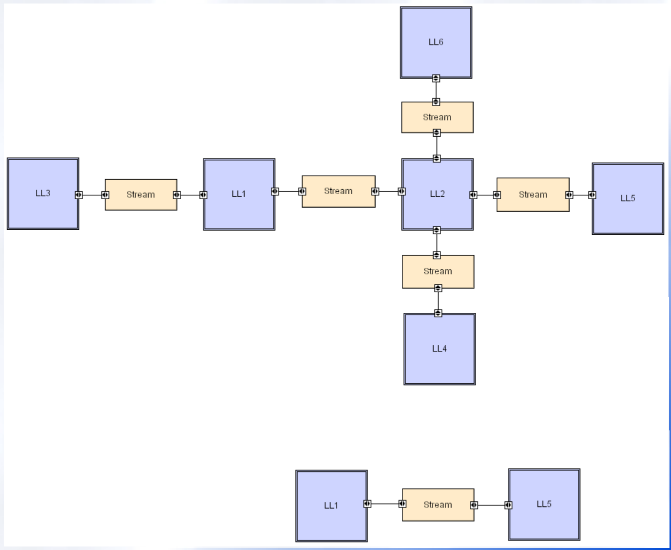
Peer-to-Peer Architectures

- Decentralized resource sharing and discovery
  - Napster
  - Gnutella
- P2P that works: Skype
  - And BitTorrent

Peer-to-Peer Style

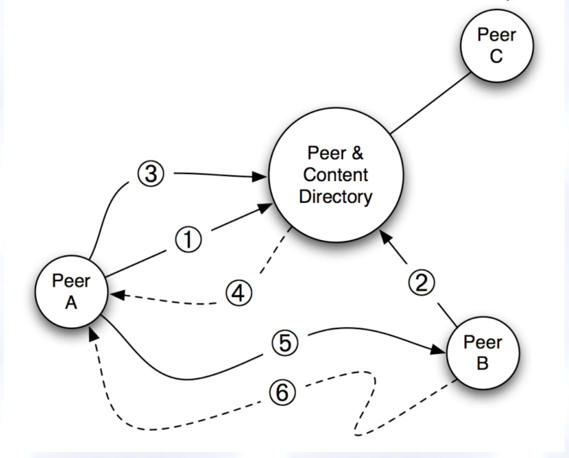
- State and behavior are distributed among peers which can act as either clients or servers.
- Peers: independent components, having their own state and control thread.
- Connectors: Network protocols, often custom.
- Data Elements: Network messages
- Topology: Network (may have redundant connections between peers); can vary arbitrarily and dynamically
- Supports decentralized computing with flow of control and resources distributed among peers. Highly robust in the face of failure of any given node. Scalable in terms of access to resources and computing power. But caution on the protocol!

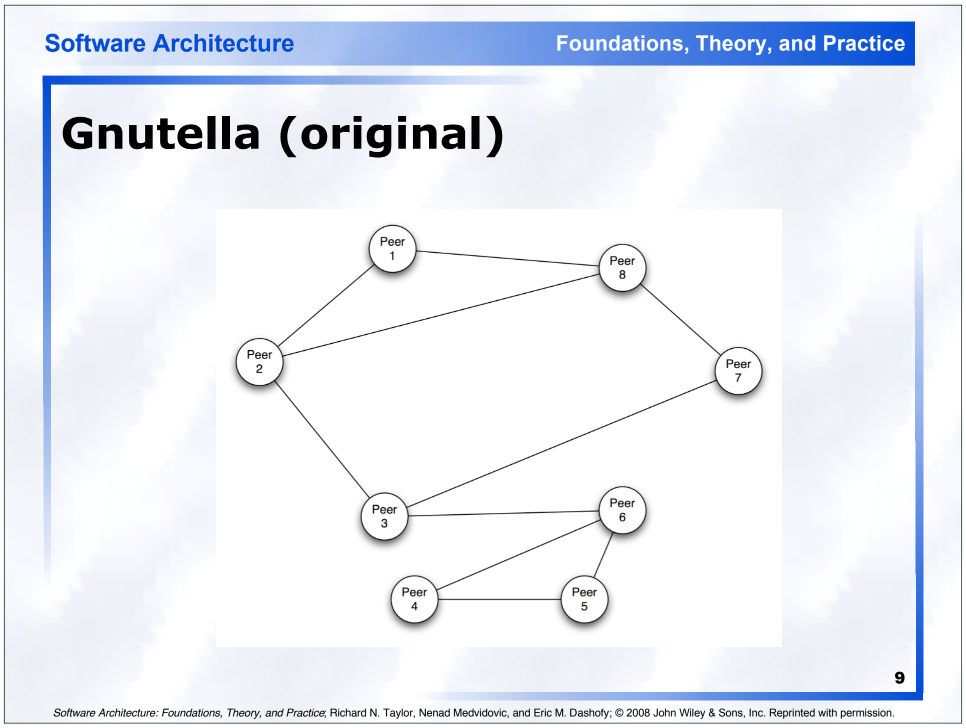
Peer-to-Peer LL



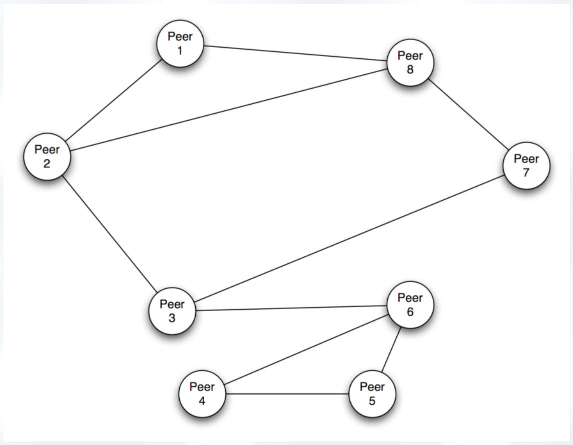
Napster (“I *am* the Napster!”)

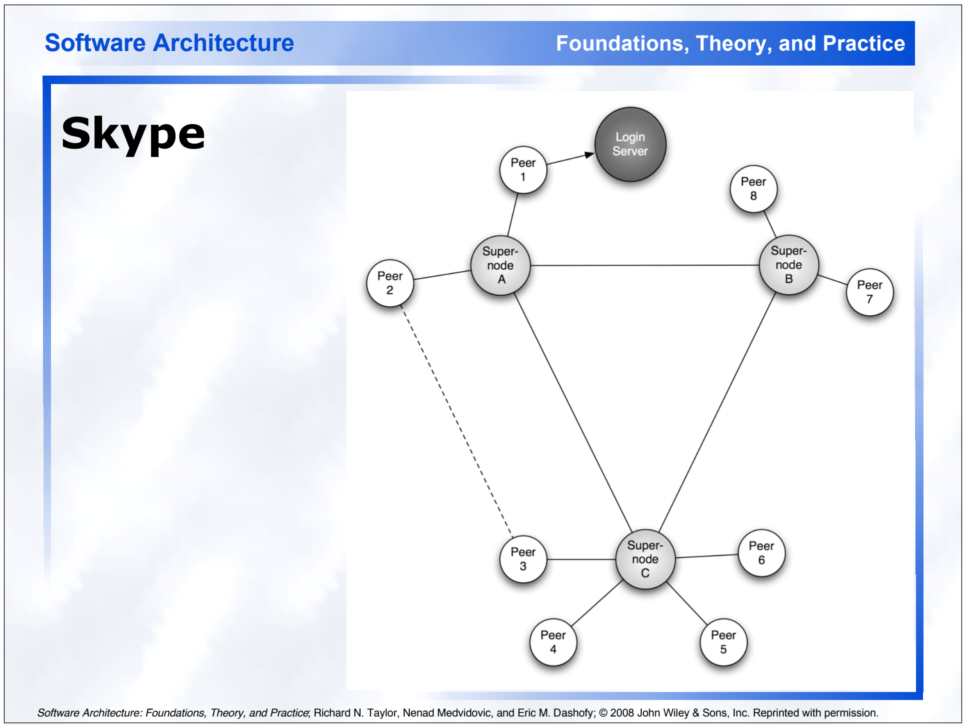
--Lyle, “The Italian Job” (2003)



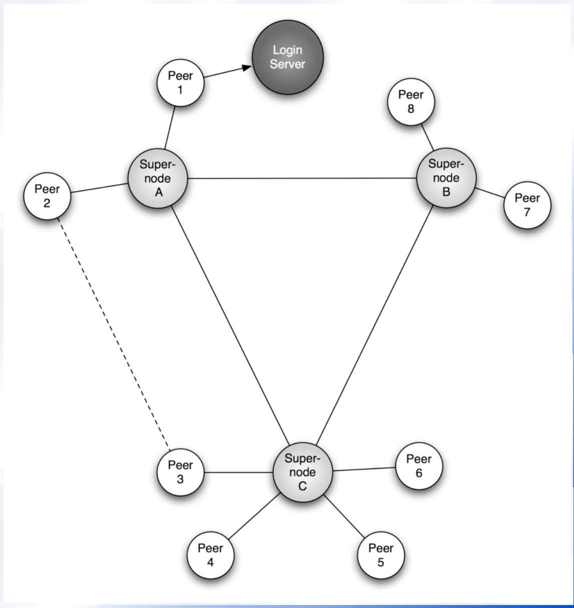


Gnutella (original)





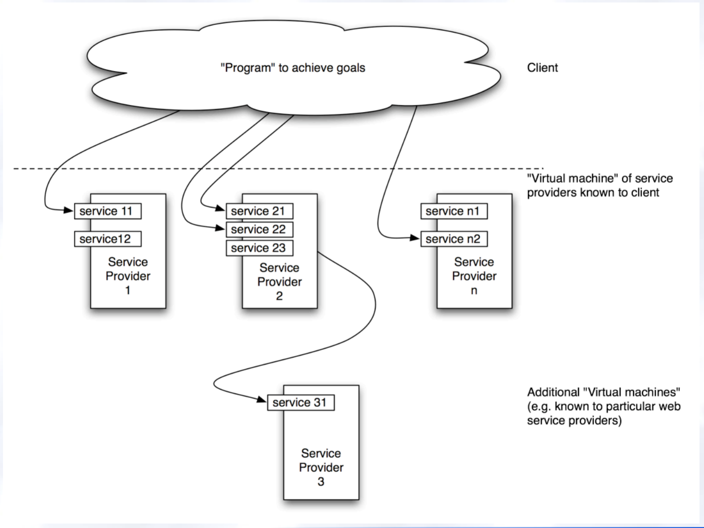
Skype



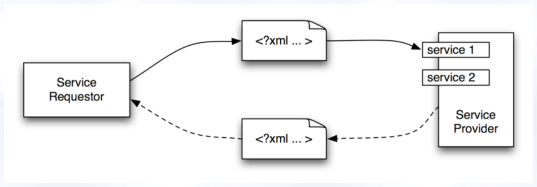
Insights from Skype

- A mixed client-server and peer-to-peer architecture addresses the discovery problem.
- Replication and distribution of the directories, in the form of supernodes, addresses the scalability problem and robustness problem encountered in Napster.
- Promotion of ordinary peers to supernodes based upon network and processing capabilities addresses another aspect of system performance: "not just any peer" is relied upon for important services.
- A proprietary protocol employing encryption provides privacy for calls that are relayed through supernode intermediaries.
- Restriction of participants to clients issued by Skype, and making those clients highly resistant to inspection or modification, prevents malicious clients from entering the network.

Web Services


"Program" to achieve goals
Client
"Virtual machine" of service providers known to client
service 11
service12
Service Provider 1
service 21
service 22
service 23
Service Provider 2
service n1
service n2
Service Provider n
service 31
Service Provider 3
Additional "Virtual machines" (e.g. known to particular web service providers)

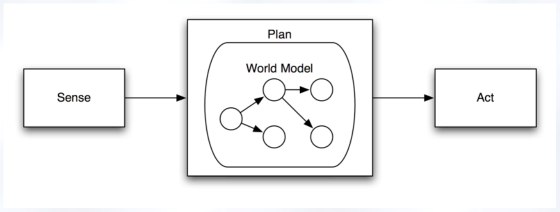
Web Services (cont'd)



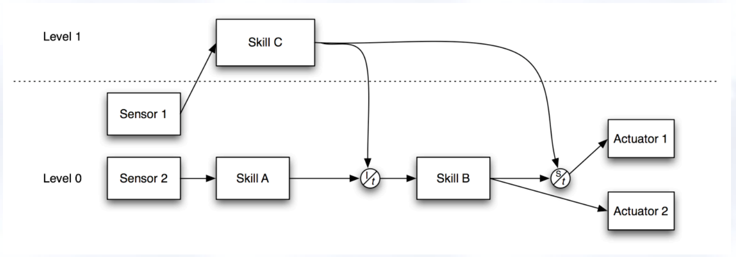
Mobile Robotics

- Manned or partially manned vehicles
- Uses
  - Space exploration
  - Hazardous waste disposal
  - Underwater exploration
- Issues
  - Interface with external sensors & actuators
  - Real-time response to stimuli
  - Response to obstacles
  - Sensor input fidelity
  - Power failures
  - Mechanical limitations
  - Unpredictable events

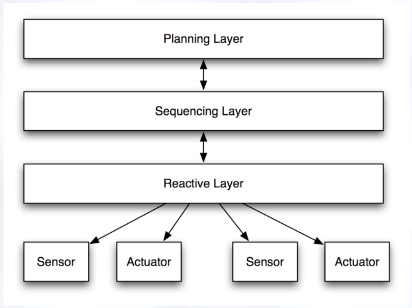
Robotics: Sense-Plan-Act



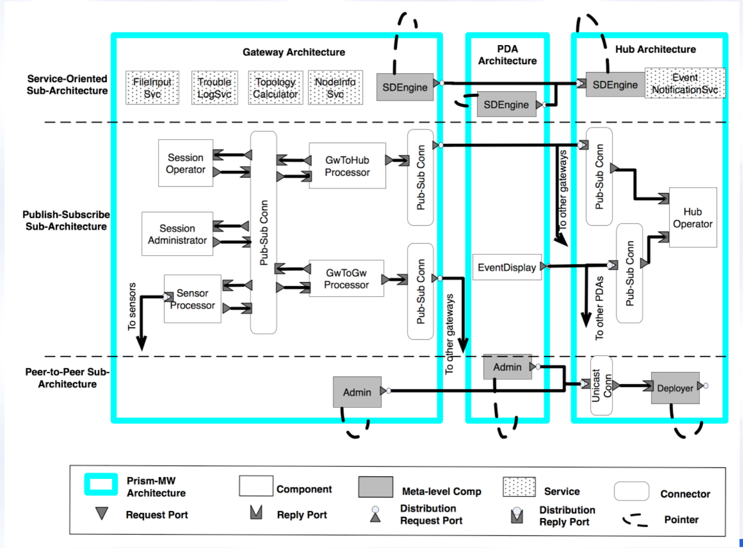
Robotics Subsumption Architecture

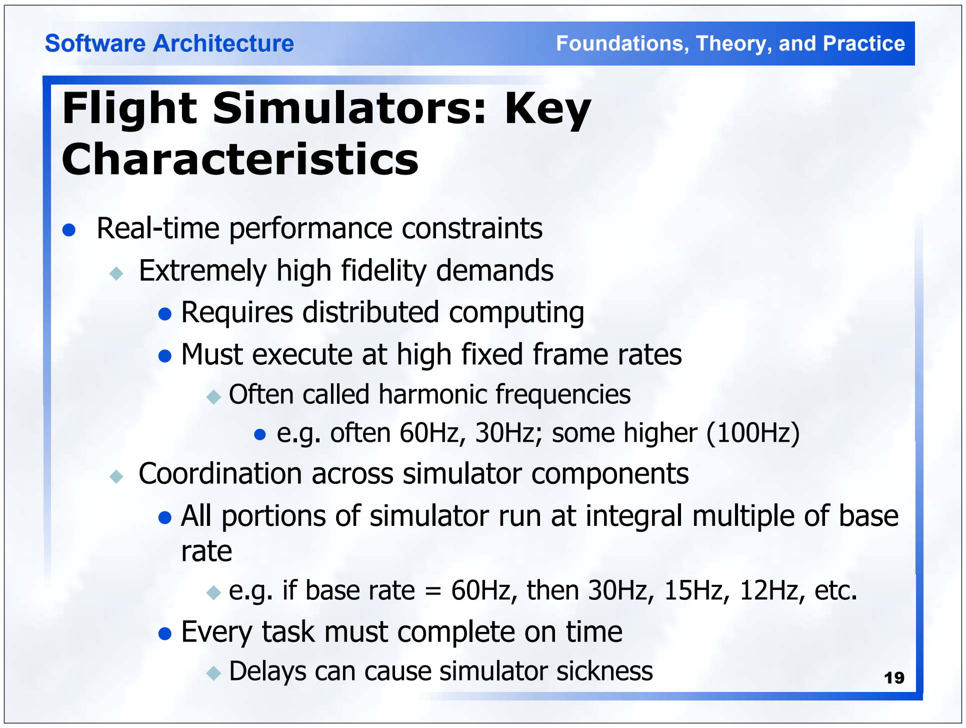


Robotics: Three-Layer



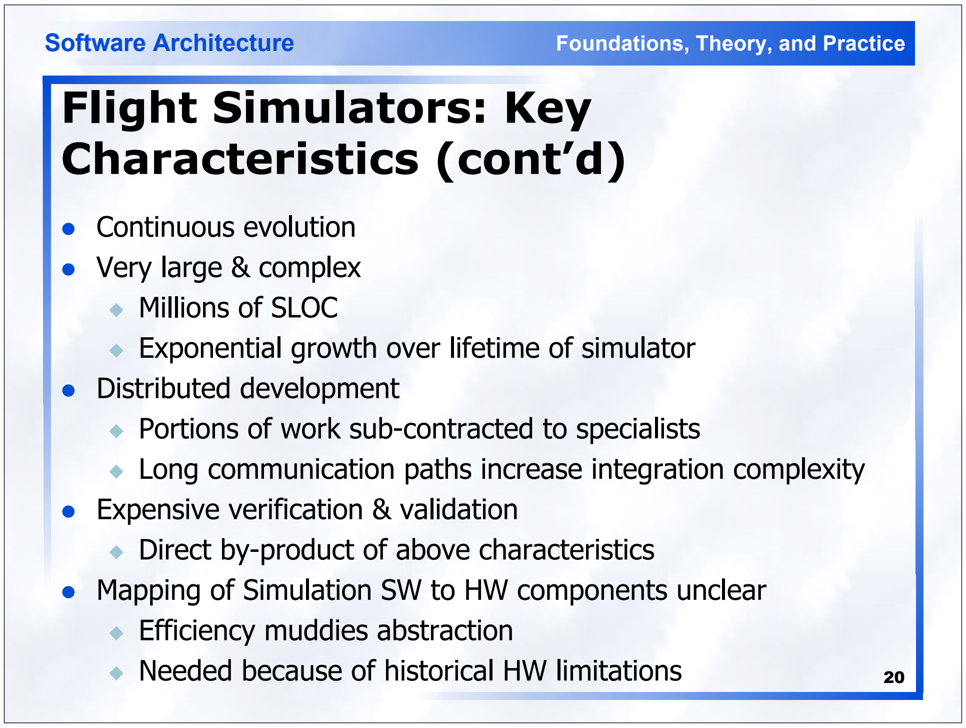
Wireless Sensor Networks





Flight Simulators: Key Characteristics

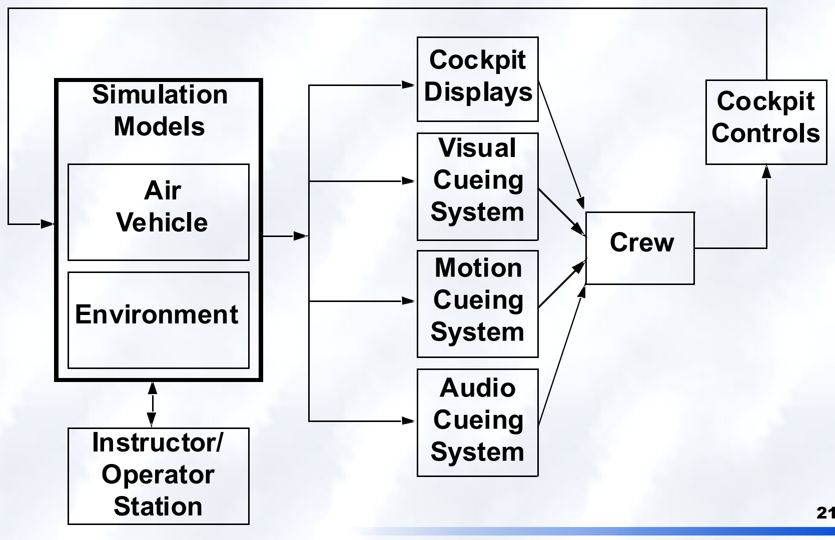
- Real-time performance constraints
  - Extremely high fidelity demands
    - Requires distributed computing
    - Must execute at high fixed frame rates
      - Often called harmonic frequencies
        - e.g. often 60Hz, 30Hz; some higher (100Hz)
  - Coordination across simulator components
    - All portions of simulator run at integral multiple of base rate
      - e.g. if base rate = 60Hz, then 30Hz, 15Hz, 12Hz, etc.
    - Every task must complete on time
      - Delays can cause simulator sickness



Flight Simulators: Key Characteristics (cont’d)

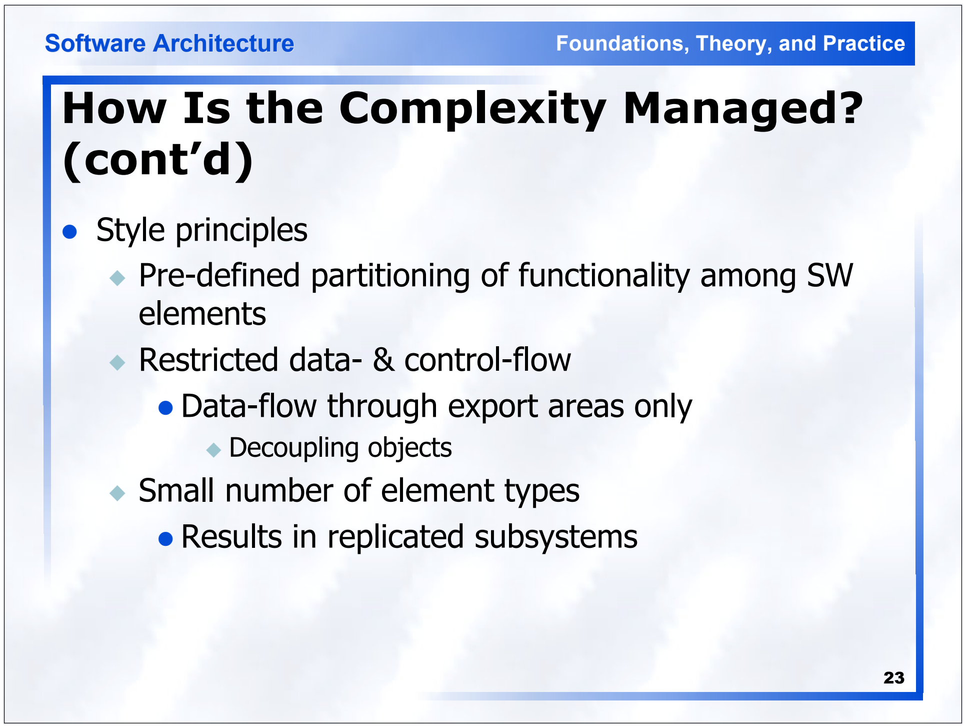
- Continuous evolution
- Very large & complex
  - Millions of SLOC
  - Exponential growth over lifetime of simulator
- Distributed development
  - Portions of work sub-contracted to specialists
  - Long communication paths increase integration complexity
- Expensive verification & validation
  - Direct by-product of above characteristics
- Mapping of Simulation SW to HW components unclear
  - Efficiency muddies abstraction
  - Needed because of historical HW limitations

Functional Model



How Is the Complexity Managed?

- New style
  - "Structural Modeling"
  - Based on
    - Object–oriented design to model air vehicle's
      - Sub-systems
      - Components
    - Real-time scheduling to control execution order
- Goals of Style
  - Maintainability
  - Integrability
  - Scalability

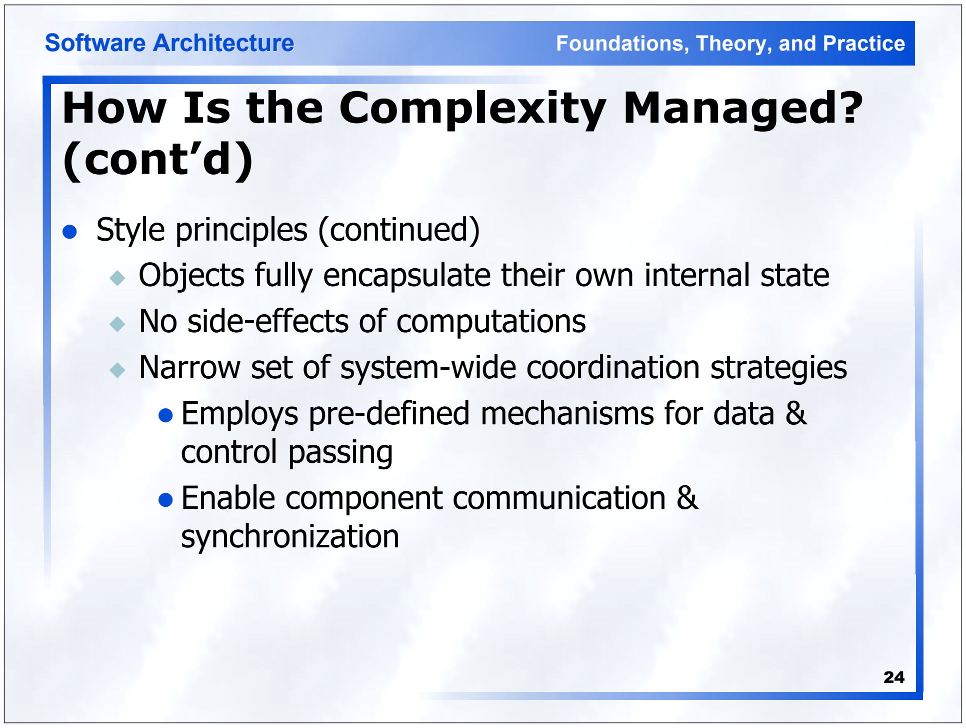


Software Architecture — Foundations, Theory, and Practice

How Is the Complexity Managed? (cont'd)

- Style principles
  - Pre-defined partitioning of functionality among SW elements
  - Restricted data- & control-flow
    - Data-flow through export areas only
      - Decoupling objects
  - Small number of element types
    - Results in replicated subsystems

23

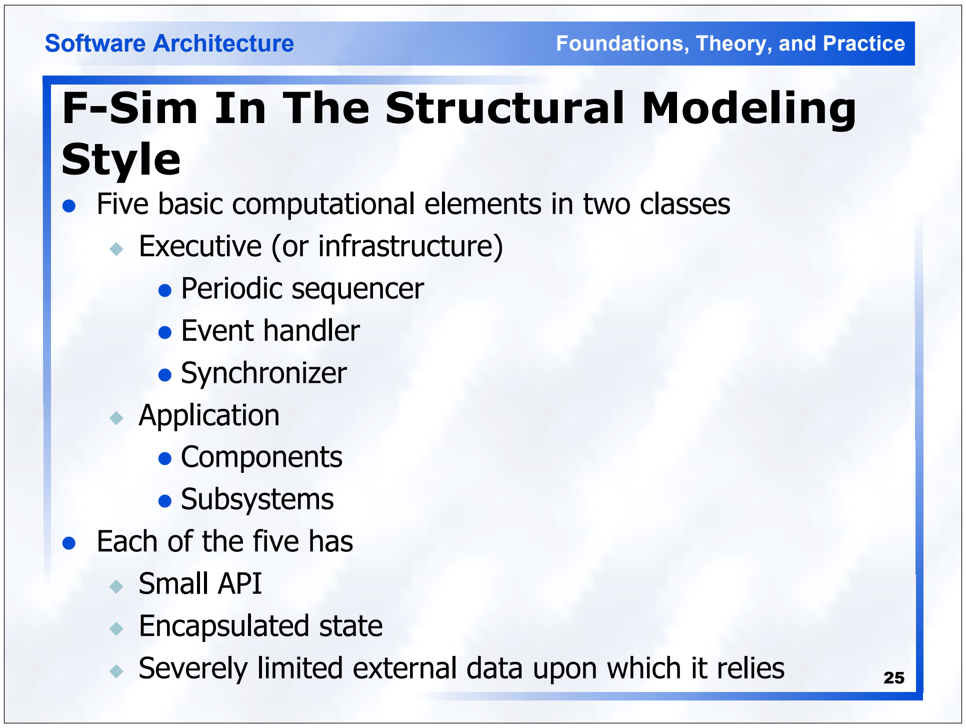


Software Architecture — Foundations, Theory, and Practice

How Is the Complexity Managed? (cont'd)

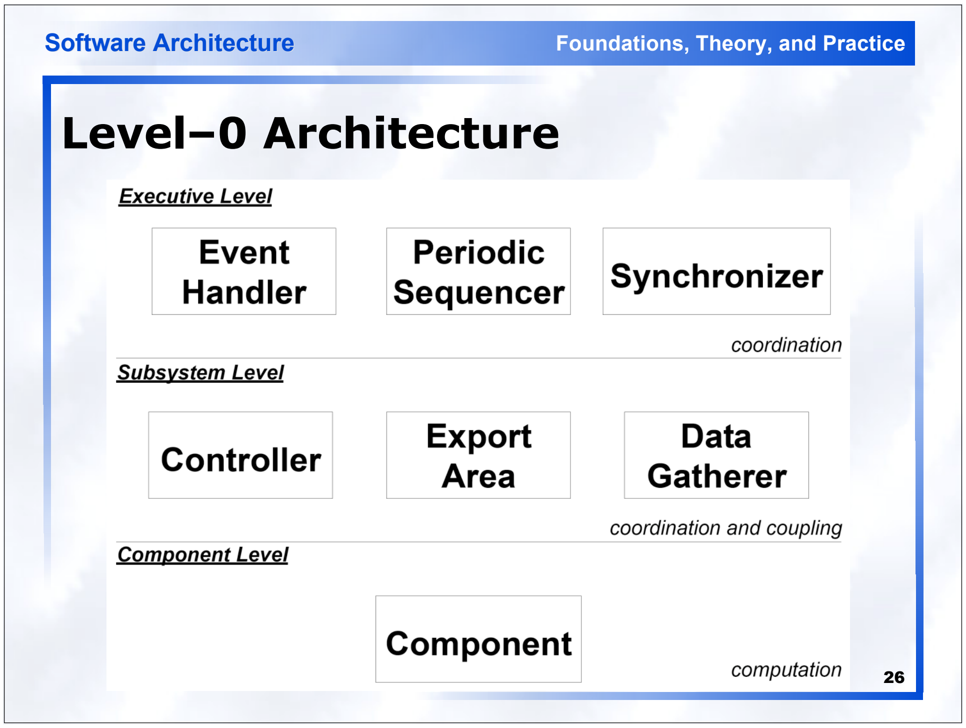
- Style principles (continued)
  - Objects fully encapsulate their own internal state
  - No side-effects of computations
  - Narrow set of system-wide coordination strategies
    - Employs pre-defined mechanisms for data & control passing
    - Enable component communication & synchronization

24



F-Sim In The Structural Modeling Style

- Five basic computational elements in two classes
  - Executive (or infrastructure)
    - Periodic sequencer
    - Event handler
    - Synchronizer
  - Application
    - Components
    - Subsystems
- Each of the five has
  - Small API
  - Encapsulated state
  - Severely limited external data upon which it relies



Level–0 Architecture

Executive Level

Event Handler | Periodic Sequencer | Synchronizer

coordination

Subsystem Level

Controller | Export Area | Data Gatherer

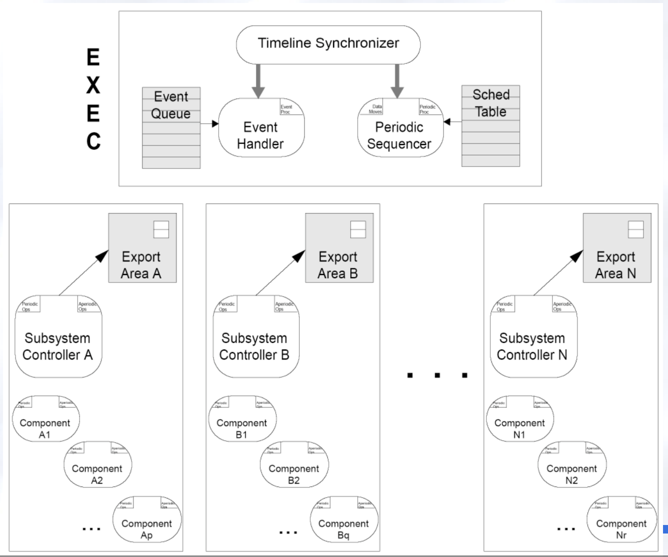
coordination and coupling

Component Level

Component

computation

Level–1 Architecture



Takeaways

- A great architecture is the ticket to runaway success
- A great architecture reflects deep understanding of the problem domain
- A great architecture probably combines aspects of several simpler architectures
- Develop a new architectural style with great care and caution. Most likely you don’t need a new style.