

Execution Architecture

John Reekie
University of Technology, Sydney

Contributors

Lian Loke, University of Technology, Sydney
Rohan McAdam, Honeywell Inc.

Terms of Use: Creative Commons Attribution-ShareAlike 2.5
<http://creativecommons.org/licenses/by-sa/2.5/>

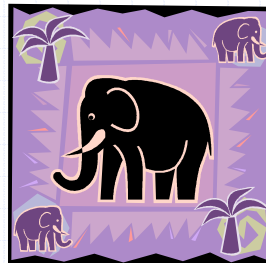
An elephant with a view

◆ Structure

- Compute nodes
- Processes
- Subsystems

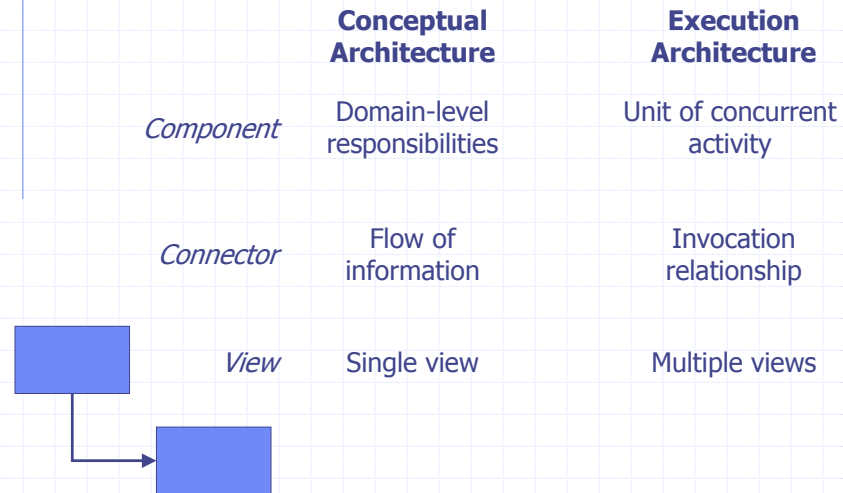
◆ Activity

- Behavior
- Communication

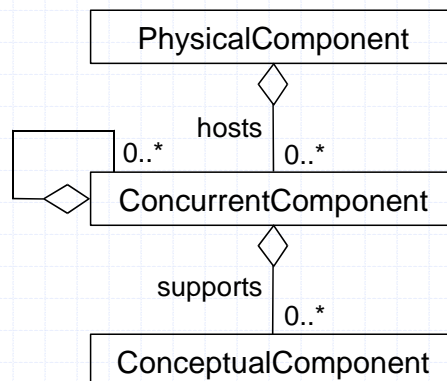


The execution architecture focuses on the “run-time” structures that exist in the system.

Conceptual vs execution

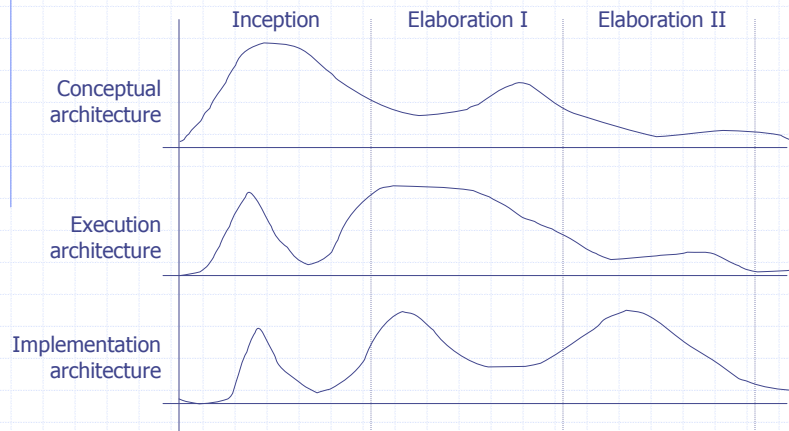


Component relationships



This is a simplified and idealized (but useful) diagram.

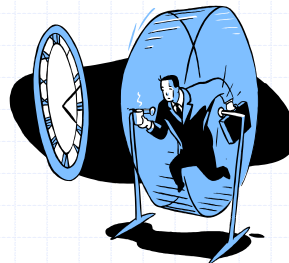
When you do it



A close-up view of a possible Unified Process instance.

What is concurrent activity?

- ◆ When two actions occur "simultaneously".
- ◆ In actuality, the activities are time-sliced into a single processor.
- ◆ Concurrency needs are often related to *time*:
 - *Waiting* for something
 - Working at the same *time*
 - *When* does the event occur?



When to use concurrency



Waiting for a slow external device

++ Performance



Locate components on different processors

++ Scalability



Independent development and compilation of components

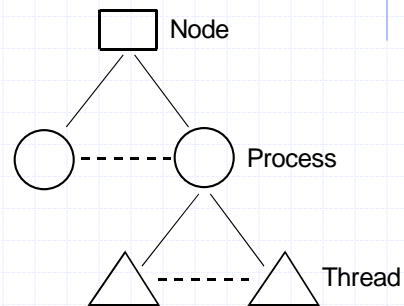
++ Maintainability

Processes and threads

- ◆ A more "concrete" execution unit than concurrent subsystems
- ◆ Map directly to operating system processes and threads

Process
Own memory space
Communicate via IPC, pipes, sockets
Visible to most O/S utilities

Thread
Shared memory space
Communicate by shared data structures
Not so visible



Activity stereotypes

◆ How is concurrent activity initiated?

- By user action
- By an active thread or threads of control
- In response to a request



User-initiated



Active



Service

Communication

◆ Arrows indicate an *invocation* relationship

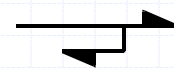
- Synchronous
- Asynchronous
- Callback



Synchronous call



Asynchronous call



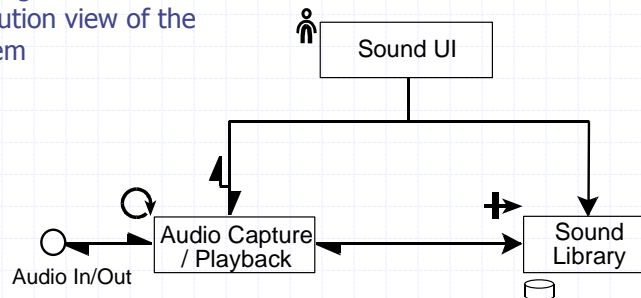
Callback

Concurrent subsystems

- ◆ A large-grained unit of execution
 - Database
 - Web/application server
 - Real-time disk recorder
- ◆ A concurrent subsystem may:
 - Exhibit significant (and complex) internal concurrency
 - Consist of a number of processes, which may be both static and dynamic

Concurrent subsystems view

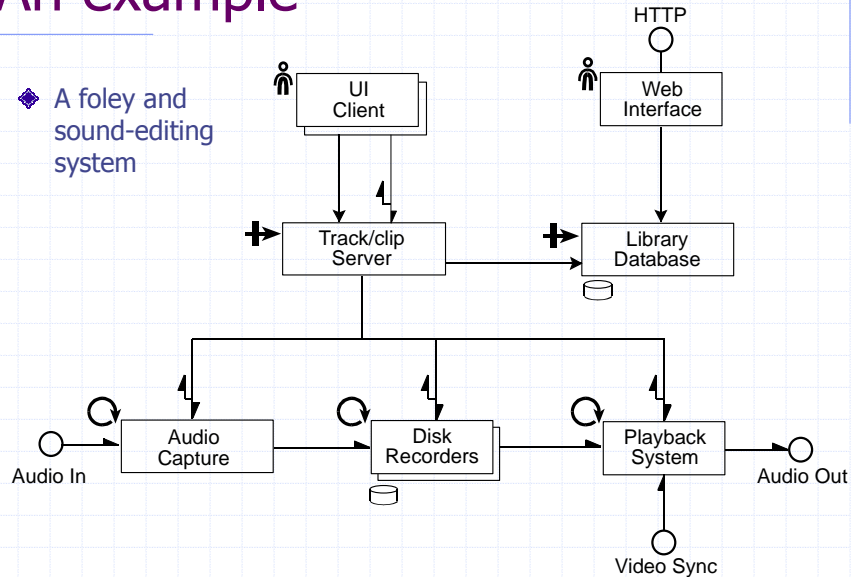
- ◆ Contains only concurrent subsystems
- ◆ The highest-level execution view of the system



A simple audio processing system

An example

- ◆ A Foley and sound-editing system

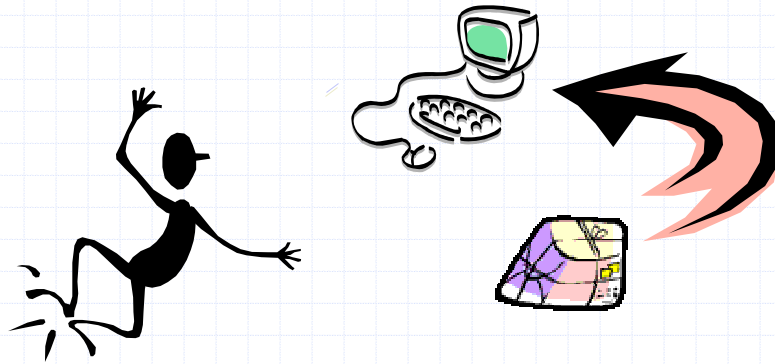


Process view

- ◆ Shows only processes
- ◆ A process is the *real* unit of deployment
 - The process view shows exactly how the software system is allocated to hardware units
 - Operating system utilities "understand" processes
- ◆ May be more palatable to traditional systems designers - ?



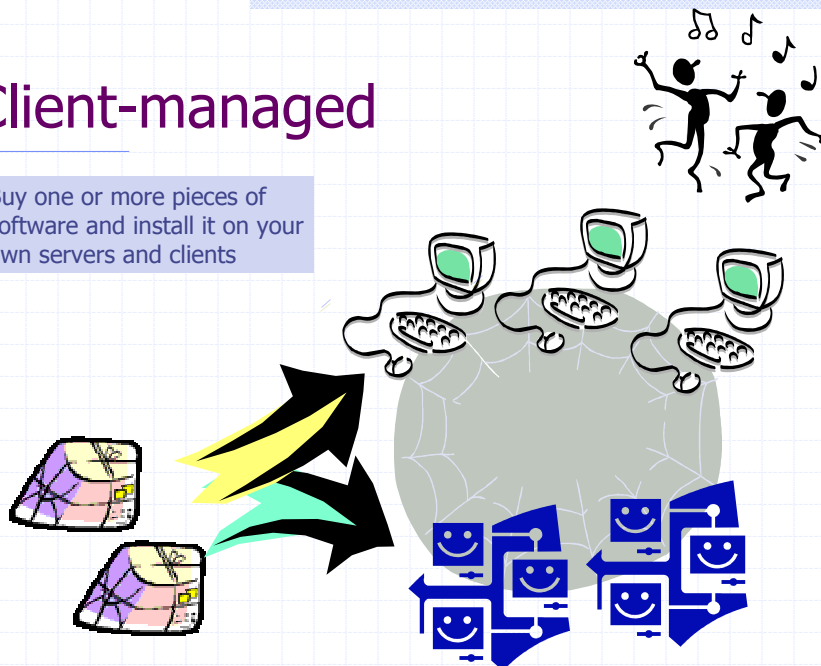
Shrink-wrap



Buy a piece of software and install it on your own computer

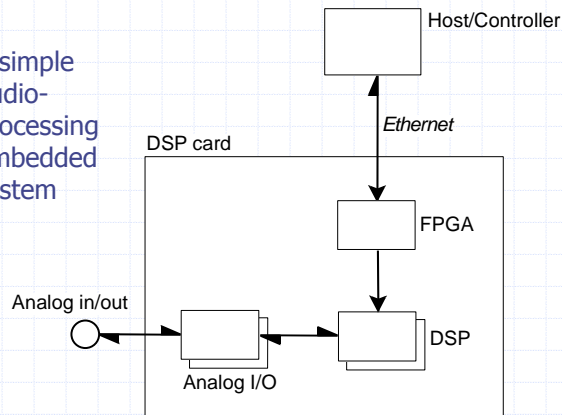
Client-managed

Buy one or more pieces of software and install it on your own servers and clients



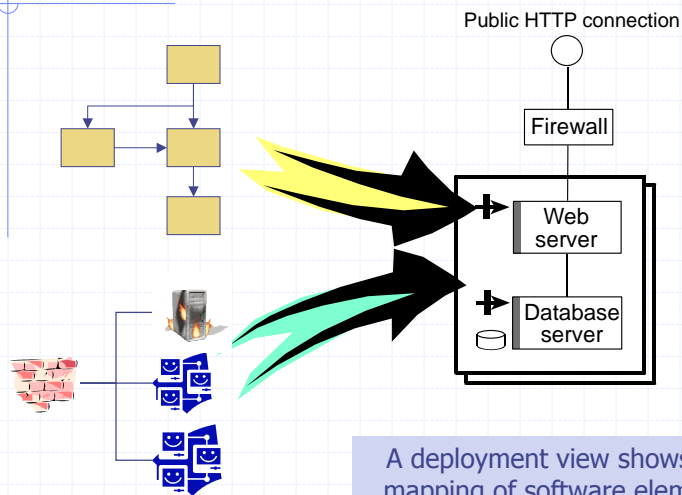
Physical architecture

- ◆ A simple audio-processing embedded system

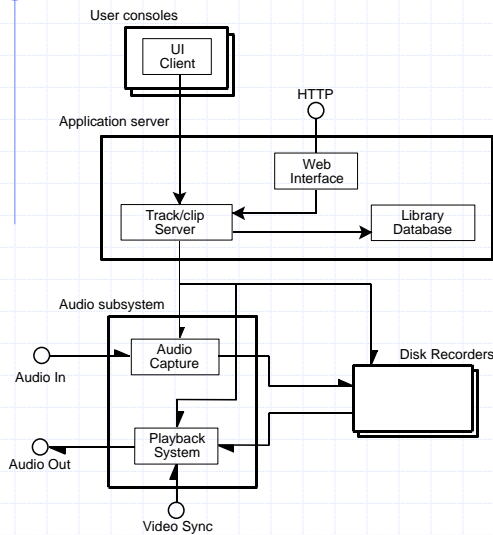


The physical architecture shows the key hardware elements of the system—as relevant to the software architecture

Deployment view



Deployment example



- ◆ A concurrent subsystems view overlaid on a (simplified) physical view
 - Processes map to a single processing node
 - Subsystems map to one or more processing nodes
 - Replication to show scalability

That's all folks!

◆ Questions or comments?

