Software Design III

Concurrent System Design

WOLFRAM KAHL

kahl@cas.mcmaster.ca

Department of Computing and Software
Faculty of Engineering
McMaster University

Course Description (Calendar)

- Processes, threads, concurrency;
- Synchronization mechanisms, resource management and sharing;
- Objects and concurrency;
- Design, architecture and testing of concurrent systems.

The New Concept: Concurrency

So far (Software Design I & II):

- **Imperative Programming:**
  telling the computer what to **do**

  — as opposed to **declarative programming**: telling the computer what to **achieve**

- **Sequential Programming:**

  specifying a **sequence of actions**

  — as opposed to specifying actions that may be performed **potentially** in **parallel**

- **Single-threaded Programming:**

  specifying a **single flow of control**

Non-Sequential Systems

- **Client-Server Systems:**
  - ATM-Banking systems
  - WWW
- **Peer-to-peer computing:**
  - UNIX `talk`
  - some other information sharing systems …

Description of Such Systems

- **Distributed:**

  several **processes** or **threads of control** execute on physically disjoint **computers** (with no shared clock and no shared memory).

- **Parallel:**

  several **processes** or **threads of control** execute on physically disjoint processors (but usually with shared memory) at the same time.

- **Concurrent:**

  several **processes** or **threads of control** can be considered as ***active at the same time***.

An Important Distinction

- “Parallel execution” and “distributed systems” are **implementation concepts**.

- **Concurrency** is a **design concept**!

- A **concurrent design** *may* be **implemented** as a distributed or parallel system.

Concurrency

A **concurrent system** consists of a set (with at least two elements) of **threads of control** or **processes** that

- **execute** (essentially) **independently**
- may access **common resources**
- may **communicate** with each other

Concurrency in Music: Polyphony

- A **pianist** plays ***one piece*** of music, say a **fugue by Bach**
  - Each of her **two hands** plays a different segment of that fugue;
  - each of the **three voices** of that fugue may move from hand to hand.
- **Conceptual structure** of the fugue: **three voices**
- **Implementation: two hands**

Concurrent Systems

- **Concurrent System Design**

  is concerned with **structuring** a task into concurrent processes.

- **Concurrent System Implementation**

  is concerned with **mechanisms** that **efficiently implement** certain (classes of) concurrent systems

Goals

- Know **what to expect** from a computer
- Know **what *not* to expect** from a computer
- Know **what can go wrong**, and **why**
- Know **what the public expects from your software**
- Know **how to achieve what you need**

Implementations of Concurrent Systems

- **Timesharing:** interleaving the actions of the concurrent processes on a single processor
- **Parallelizing/Distributing:** distributing the actions of the concurrent processes onto separate processors/systems.

The essential support for the implementation of concurrency is usually found in **operating systems**.

Interaction with a Computer

Different interaction paradigms:

- **Point-and-click**
  - ***High* intuitivity** (sometimes)
  - ***Low* expressivity** and **abstraction** capabilities
- **Command line** — ***linguistic***
  - ***High* expressivity, abstraction** capabilities
  - ***High* intuitivity** — once you know the language

Skills

- OS Interaction — UNIX command line
- Programming — C, Java
- System-Level Programming — UNIX system calls
- Analysis of Concurrent Processes
- Analysis of Possible Uses of Concurrent Processes
- Design of Concurrent Systems
- Implementation of Concurrent Systems

More Skills

- Read the Documentation!
- Read the Questions!