David M. Keil, Framingham State University

CSCI 252 Computer Science II Using Java

7. Concurrency in mobile and web computing

1. Concurrency and Java threads
2. Software for mobile computing
3. Web development and concurrency
4. Ethical and security issues

Inquiry

- Computing today is characterized by *multiple I/O streams.* How do software developers handle such requirements?
- What is special about mobile and web computing?
- How do smartphones handle multiple I/O streams?
- How do robots coordinate multiple sensor and effectors?

Topic objective

Describe concurrent features of mobile and web environments, and implications for programming, ethics, and security.

1. Concurrency and Java threads

- What's meant by *concurrency*?
- How do computers execute multiple operations at the same time?
- Are *Java threads* an adequate solution to problems of concurrency?

Subtopic objectives

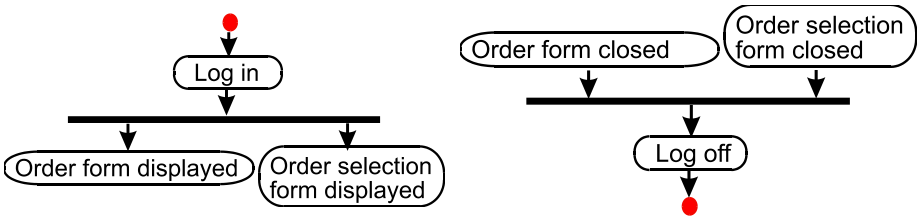
7.1a Describe multi-threading

7.1b Use Java threads†

Multitaking and concurrency

- A processor may run two or more programs at the same time (multitasking)
- To do so, it saves its current state in one program's fetch/execute cycle, and loads the state of another program's cycle, to run a time slice of that program, numerous times per second
- Concurrency includes parallel and distributed computing

Concurrent activity diagrammed in UML: *fork* and *join*



- A *fork* (two forms display at once)

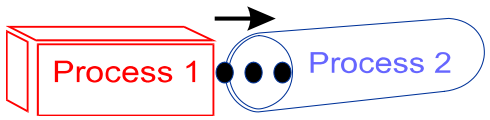
A *join*, where execution converges

Processes

- *Process*: a programming abstraction that simulates ownership of all computer resources
- Operating systems support multi-tasking with multiple processes

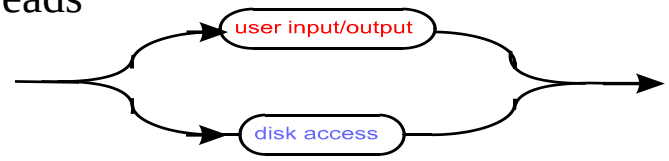
A *pipe* simulates interprocess communication

- A process that produces data may make results available before it completes
- A second process works immediately on the first data that enters the pipeline
- Familiar example: an assembly line



A *thread* simulates ownership of a CPU

- One program may run multiple threads, e.g., for disk access and user I/O
- The processor executes multiple threads concurrently
- Multiple processors may share execution of threads



Examples of multithreading

- *Downloading video* while playing parts already downloaded
- *Printing* document concurrently with editing it
- *Garbage collection*: the Java garbage collector runs as a low-priority thread to free up memory previously allocated to objects but no longer used

Memory allocation and garbage collection

- Java and C++ *new* operator (function *malloc* in C) causes *dynamic allocation* of memory for objects
- *References* and *pointers* provide access to object [pic]
- When reference variable goes out of scope (e.g., enclosing method terminates), memory space is still allocated but object is inaccessible
- Memory must be *deallocated*
  - automatically by Java garbage collector or
  - using *delete* operator in C++, *free* in C

Time slicing

- Threads are each allocated a time quantum (slice) to execute
- After one thread's time slice, the thread waits for other processes to take their turns
- All threads of equal priority take turns, round-robin style
- Threads are *pre-empted* by higher-priority threads or by threads whose turn has come

Java threads

- A *Thread* object is constructed within a running method and is launched by a call to the *start* method
- Upon launch, the thread executes concurrently with the calling thread
- *sleep(n)* method causes thread to sleep for *n* milliseconds
- *suspend* and *resume* methods cause thread to pause and restart

Life cycle of a thread

[pic, Deitel and Deitel, 1997, p. 674]

Java methods for *Thread* class

- *start*
- *run*
- *sleep*
- *interrupt*
- *suspend*
- *stop*
- *isAlive*
- *currentThread*

Thread states

- born → ready ↔ running
- *ready* means waiting for processor resource
- running →
  - waiting
  - sleeping
  - suspended
  - blocked (waiting for I/O)
  - dead

Priorities and scheduling

- Each thread has priority in 1 .. 10; 5 by default (*Thread.MIN_PRIORITY* = 1)
- A thread inherits priority of its creator thread
- Java scheduler ensures that highest priority thread executes at all times
- If OS platform supports time slicing and multiple highest-priority threads are running, then they execute *round robin*

Synchronization

- Objects with methods declared as *synchronized* may have these methods execute only under one thread at a time
- This is enforced by a thread obtaining a *lock* on the object
- Monitor objects enforce locking
- *Example:* a bank-account transaction requires locking the account object

Producers and consumers

- *Example:* Producer thread places data in a buffer, consumer thread prints it
- Producer and consumer may need to be synchronized to avoid excess data or loss of data to consumer
- Hence these methods are declared with the *synchronized* keyword

Daemon threads

- *Definition:* threads that run in background to serve other threads
- *Example:* Java garbage collector
- *setDaemon(true)* causes a thread to be a daemon
- Program terminates when only daemons are running

Java interface *Runnable*

Multithreading is enabled only in classes that

- extend *Thread*, or
- implement the Java interface *Runnable*

Multithreading vs. parallel processing

- Threading is a *programming abstraction*
- Parallelism refers to multi-processor *hardware* environment
- Multithreading can occur on a single processor or multiple processors
- Multicore and multi-processor systems can speed up performance of multithreaded software

2. Software for mobile computing

- How does mobile computing differ from desktop computing?
- Do standards exist for mobile apps as they do for desktop apps?

Subtopic objectives

7.2a Describe features of software for mobile computing

7.2b Design and test a multi-threaded robotic program†

Cases of concurrency in mobile devices

- Your mobile phone may be listening to/for incoming calls, text messages, clock, GPS location, motion sensor, and sending out data concurrently to multiple destinations
- NAO robot has a dozen or more input sources to listen to, including sound, vision, touch; has multiple effectors, e.g., head, arms, hands, legs

Mobile vs. desktop computing

- Desktop GUIs work mostly *synchronously*; that is, output is generated as fast as user can respond and GUIs waits for user inputs
- Mobile devices such as smart phones and tablets interact *asynchronously* with their environments; that is, there is no waiting turns
- Devices interact with multiple other devices and initiate communication autonomously

Concurrency in robotics

Examples:

- *Waiting for a spoken command* and responding is an event-driven process
- *Speaking and walking* at the same time
- [pic of NAO Choregraph screen]

3. Web development and concurrency

- How does development for Web 2.0 differ from earlier web development?
- How do *client* and *server* applications differ?

Subtopic objectives

7.3a Describe issues in web software development

7.3b Write a Java applet†

Concurrency and web software

- Web and other networked systems support access to data by multiple users concurrently
- *Example:* In online banking, how does web-based software assure integrity of *atomic transactions* (e.g., bank-account transfers) when multiple users may be requesting access to the same accounts?

Java applets

- Java compilers may generate *applets* that run in browser
- The browser contains a *virtual machine* that *interprets* the byte code and enables distribution of *processor independent* compiled code on the Web
- HTML tag: **`<applet>`**
- Programmer uses public class of applets that extends *JApplet* (package *javax.swing.JApplet*)

Interoperability and XML

- XML is a standard markup language that supports *semantic tagging* of data items

How web-based apps handle storage of user data

4. Ethical and security issues

- What *privacy policies* for web services are ethical?
- What security measures are needed to ensure Internet and web user privacy?
- What transparency are developers obliged to provide about vulnerabilities?

Subtopic objectives

7.4a Describe a network security issue

7.4b Describe ethical issues for computer professionals*

Special characteristics of network communication

- *Scope* (speed/immediacy; vastness of reach; interactivity)
- *Anonymity* (implying diminished trust)
- *Reproducibility* (enables violation of privacy and copyright)

Why IT raises new issues

- Ease of collecting data
- Ease of copying
- Ease of communication
- Power of processing data
- Storage capacity

Network security measures

- *Authentication* (password, biometrics, passcard) requires users to prove they have permission to access network
- *Levels of access privileges* segment users according to need to know
- *Firewalls* may be layered:
  - External screening router to examine packets
  - Bastion host and proxy server screens Internet traffic
  - Internal screening router limits acceptable Internet requests

Security and encryption

- *Secure web sites* are password-protected
- *Secure connections* are those that encrypt the data communicated; protocol is *https*
- *Encryption* transforms a message into a difficult-to-read text; decryption transforms encrypted text to readable form
- Encryption/decryption use algorithms and a private key; some algorithms also permit a public key
- *Strong encryption* is hard to "break"
- Public keys enable client browsers to encrypt messages for sending securely to recipients

Desired security features

- *Availability:* ability to receive and process data
- *Integrity:* assurance of accuracy and against unauthorized changes
- *Authentication:* protection against fraudulent transmission through assurance of identity of sender
- *Confidentiality:* assurance of nondisclosure of info to entities not authorized to receive
- *Nonrepudiation:* a way of certifying that sender and receiver have participated in a transaction

Effect of data collection on privacy

- Online data is collected and linked by computers
- User is often uninformed of collection
- "Secure" data may be leaked
- Data online is copied and re-published even if removed
- Data provided for one purpose is often used for other purposes

Data collection

- *Secondary use:* Use of personal information for purposes other than those for which the user provided it
- *Computer matching:* Comparing and combining information from different databases, using identifying information
- *Data mining:* Analyzing databases to find patterns and to support decisions
- *Computer profiling:* Predicting behavior of individual based on data analysis

Processing of personal data

Some provisions of a 1995 directive of the European Parliament:

- Data quality should be high
- Legitimacy of purpose
- Use for intended purpose
- No processing of "sensitive" data (ethnic, political, religious, trade-union affiliations)
- Right of object to be informed and to correct errors

Software developers' responsibilities

- Consider costs and benefits to end users, including safety
- Whistle blowing may help the public and employer; some issues are high-priority
- To assess risk one must have sufficient expertise
- Disclosure of conflicts of interest is crucial
- Testing should be independent of product development
- Maintenance of systems requires the same professionalism as initial development

Codes of ethics for IT professionals

- Central concern: the public good, including human rights and diversity of culture
- Honesty and fairness in communication about software and related topics
- Use client or employer property only as authorized
- Assure high quality, reasonable cost and schedule
- Respect privacy, intellectual property
- Disclose conflicts of interest
- Address software errors
- Honor agreements and assigned responsibilities

Some guidelines

- Define objectives reasonably
- Involve users in design and testing
- Plan, estimate and schedule carefully
- Design for human users, validating input
- Validate components and default settings
- Speak honestly of risks and limitations
- Disclose possible conflicts of interest

References

H. Deitel and P. Deitel. *Java How to Program.* Prentice Hall, 1997.