Partial Ordering of Events

- Sequential programs give use a total ordering of events: everything happens in a determined order.
- Concurrency gives us a partial ordering of events: we know some things happen before other things, but not total order.

Value of Concurrency

- As an abstraction: hides the details of exactly when things happen.
  - Program thinking about objects
  - Interleaving of events
- Opportunity for implementations:
  - Execute faster by using multiple cores.

How is concurrency a kind of abstraction?

A Picture of the Multicore Crisis


The Free Lunch Is Over: A Fundamental Turn Toward Concurrency in Software, Herb Sutter

http://www.gotw.ca/publications/concurrency-ddj.htm
Where is most of the processing power on your PC?

nVIDIA GeForce GTX 470
448 Cores

nVIDIA Quadro Plex 7000 DHIC
1792 cores

public class BankAccount {
    private int balance;
    ...
    public void transfer(BankAccount b, int amount) throws InsufficientFundsException {
        // MODIFIES: this, b
        // EFFECTS: If this account has more than amount value, transfers amount from
        //     this account to b.  Otherwise, throws InsufficientFundsException.
        synchronized(this) {
            synchronized(b) {
                if (this.getBalance() > amount) {
                    b.addFunds(amount);
                    this.balance = this.balance – amount;
                } else { throw new InsufficientFundsException(); }
            }
        }
    }
}

Request 1:
Request 2:
Request 3:

Challenge of Concurrency

Server
Listener
Request 1
Request 2
Request 3

Store
Shared Data
Instruction Streams

Why are threads hard?

Too few ordering constraints: race conditions
Too many ordering constraints: deadlocks
poor performance, livelocks, starvation
Hard/impossible to reason modularly
— If an object is accessible to multiple threads, need to think
about what any of those threads could do at any time!
Testing is even more impossible than it is for sequential
code
— Even if you test all the inputs, don’t know it will work if
threads run in different order

Solutions

No shared store

Functional programming
Scheme without set!, set-car!, set-cdr!

Use analysis tools and locking discipline
http://findbugs.sourceforge.net/

Force determinism
Require thread interleavings to happen in a predictable way

PS5 Designs [Throughout the Day]

1. Blanton, James Kalish, Michael
2. Borja, Joseph Oh, Uyn Noh, Brian
3. Brown, Jeremy Hearn, Charles
4. Chen, Jiamin Sparkman, Elisabeth Sun, Yixin
5. Dewey-Vogt, Michael Lopez, Erick
6. Dilorenzo, Jonathan Featherston, Joseph
7. Dollhopf, Niklaus Marion, John
8. Herder, Samuel Wallace, Alexander
Buzzword Description

“A simple, object-oriented, distributed, interpreted, robust, secure, architecture neutral, portable, high-performance, multithreaded, and dynamic language.” [Sun95]

As the course proceeds, we will discuss how well it satisfies these “buzzwords”. You should especially be able to answer how well it satisfies each of the blue ones in your final interview.

What does it mean for a language to be “Object-Oriented”?

What is an Object?

- Packaging state and procedures
  - state: the rep
    - What a thing is
  - procedures: methods and constructors
    - What you can do with it

Bjarne Stroustrup (C++)’s Answer

“Object-oriented programming is programming with inheritance. Data abstraction is programming using user-defined types. With few exceptions, object-oriented programming can and ought to be a superset of data abstraction. These techniques need proper support to be effective. Data abstraction primarily needs support in the form of language features and object-oriented programming needs further support from a programming environment. To be general purpose, a language supporting data abstraction or object-oriented programming must enable effective use of traditional hardware.”

“I invented the term Object-Oriented and I can tell you I did not have C++ in mind.”

Alan Kay

Programming Language History

- Before 1954: twidling knobs, machine code, assembly code
- FORTRAN (John Backus, UVa dropout, 1954) – Formula Translation
- Algol (Peter Naur, Alan Perlis, et. al., 1958-1960) – Most influential programming language
  - Many of the things Algol did first (types, while, blocks) are in Java
Programming Language History

**Simula** (Dahl and Nygaard, 1962-7)
First language with subtyping and inheritance

**CLU** (Liskov et. al., 1970s)
First language with good support for data abstraction (but no subtyping or inheritance)

**Smalltalk** (Kay et. al., 1970s)
First successful language and programming system to support subtyping and inheritance

Simula
- Considered the first “object-oriented” programming language
- Language designed for simulation by Kristen Nygaard and Ole-Johan Dahl (Norway, 1962)
- Had special syntax for defining classes that packages state and procedures together

Counter in Simula

```
class counter;
    integer count;
begin
    procedure reset(); count := 0; end;
    procedure next();
        count := count + 1; end;
    integer procedure current();
        current := count; end;
end
```

XEROX Palo Alto Research Center (PARC)

1970s:
- Bitmapped display
- Graphical User Interface — Steve Jobs paid $1M to visit and PARC, and returned to make Apple Lisa/Mac
- Ethernet
- First personal computer (Alto)
- PostScript Printers
- Object-Oriented Programming

Dynabook, 1972

- Tablet computer intended as tool for learning
- Alan Kay wanted children to program it also
- Hallway argument, Kay claims you could define “the most powerful language in the world in a page of code”

**Proof:** Smalltalk
- Scheme is as powerful, but takes two pages
- Simple Java compiler and VM requires thousands of pages

Don’t worry about what anybody else is going to do… The best way to predict the future is to invent it. Really smart people with reasonable funding can do just about anything that doesn’t violate too many of Newton’s Laws!” — Alan Kay, 1971
Smalltalk

- Everything is an object
- Objects communicate by sending and receiving messages
- Objects have their own state (which may contain other objects)
- How do you do 3 + 4?
  
  send the object 3 the message “+ 4”

CLU

- Developed by Barbara Liskov and colleagues at MIT (1973-1978)
- First language to provide real support for data abstraction
  - Encapsulation
  - Abstract data types: hide representation outside data type implementation
  - Parameterized types (generics), iteration abstraction, exceptions

Counter in Smalltalk

class name counter

instance variable names count

new count <- 0

next count <- count + 1

current ^ count

Counter in CLU

counter = cluster is create, next, current

rep = record [count: int];

create = proc () returns (counter);
  return up(rep$[count: 0]);
end create;

next = proc (c: counter);
  val : rep := down(c);
  val.count := val.count + 1;
end next;

current = proc (c: counter) returns (int);
  return (down(c).count);
end current;
end counter;

Problems in Simula that motivated CLU

1. “Simula did not support encapsulation, so its classes could be used as a data abstraction mechanism only if programmers obeyed rules not enforced by the language.”
2. Simula did not provide support for user-defined type “generators.” These are modules that define groups of related types, e.g., a user-defined set module that defines set[int], set[real], etc.
3. It associated operations with objects, not with types.
4. “It treated built-in and user-defined types non-uniformly. Objects of user-defined types had to reside in the heap, but objects of built-in type could be in either the stack or the heap.”

CLU “solved” all of these...how many do Java solve?
Object-oriented programming is an exceptionally bad idea which could only have originated in California.

So, who really was the first object-oriented programmer?

Edsger Dijkstra (1930-2002)
1972 Turing Award Winner

Object-Oriented Programming

Object-Oriented Programming is a state of mind where you program by thinking about objects. It is difficult to reach that state of mind if your language doesn’t have:

- Mechanisms for packaging state and procedures
- Subtyping: Java has extends and implements
- Other things can help: dynamic dispatch, inheritance, automatic memory management, everything is an object, mixins, good donuts, etc.

Who was the first object-oriented programmer?

Ada, Countess of Lovelace, around 1843

By the word operation, we mean any process which alters the mutual relation of two or more things, be this relation of what kind it may. This is the most general definition, and would include all subjects in the universe. Again, it might act upon other things besides number, were objects found whose mutual fundamental relations could be expressed by those of the abstract science of operations... Supposing, for instance, that the fundamental relations of pitched sounds in the science of harmony and of musical composition were susceptible of such expression and adaptations, the engine might compose elaborate and scientific pieces of music of any degree of complexity or extent.