J. P. Cohoon and J. W. Davidson

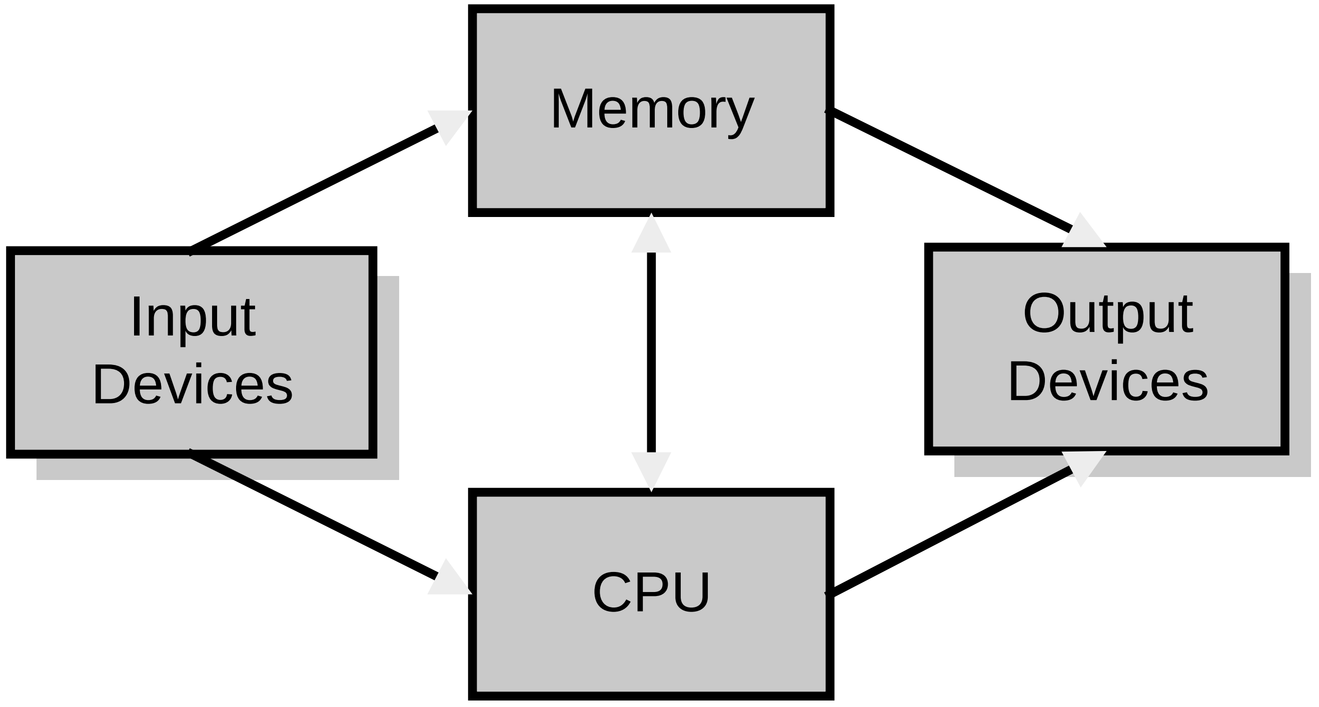
Introduction to Programming and Object-Oriented Design

Basics of machine, software, and program design

Computer Organization

- Every computer is organized roughly into four parts
  - CPU - central processing unit
    - Where decisions are made, computations are performed, and input/output requests are delegated
  - Memory
    - Stores information being processed by the CPU
  - Input devices
    - Allows people to supply information to computers
  - Output devices
    - Allows people to receive information from computers

Computer Organization

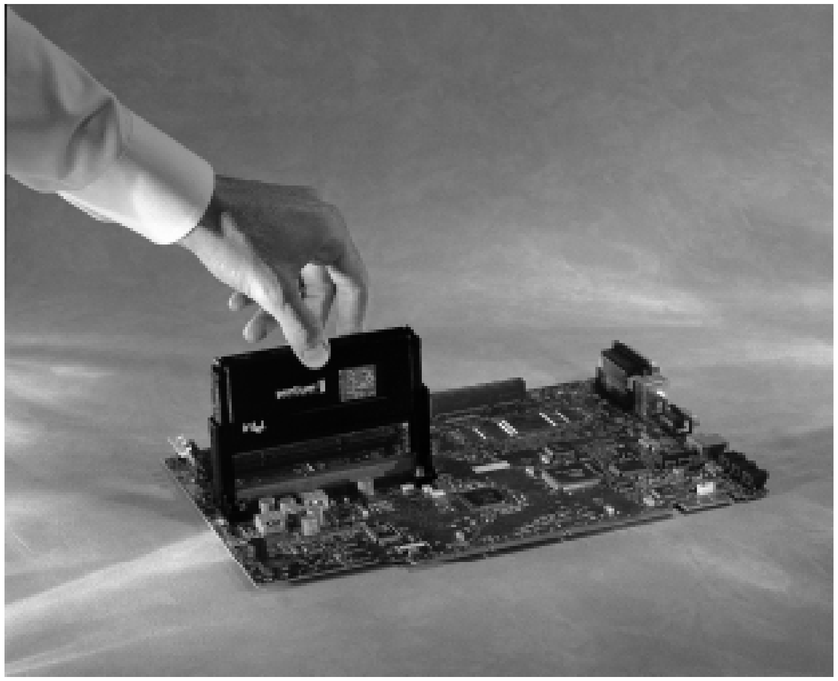


CPU

- “Brains” of the computer
  - Arithmetic calculations are performed using the Arithmetic/Logical Unit or ALU
  - Control unit decodes and executes instructions
- Arithmetic operations are performed using binary number system

CPU

- Fundamental building block is a switch
  - Switches are made from ultrasmall transistors
- Example -- Pentium II



Binary Arithmetic

- The individual digits of a binary number are referred to as bits
  - Each bit represents a power of two
- Examples

$$01011 = 0 \cdot 2^4 + 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 = 11$$

$$00010 = 0 \cdot 2^4 + 0 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 = 2$$

Binary addition $\longrightarrow$

$$\begin{array}{r} 00010 \\ \underline{+\ 01011} \\ 01101 \end{array} \qquad \begin{array}{r} 2 \\ \underline{+\ 11} \\ 13 \end{array}$$

$\longleftarrow$ Equivalent decimal addition

Binary Arithmetic

Binary multiplication

```
     0101
  ×  0011
  -------
     0101
    0101
   0000
  0000
---------
  0001111
```

Equivalent decimal multiplication

```
   5
×  3
----
  15
```

Two’s Complement

- Convention for handling signed numbers in binary representation
  - The leading bit is a sign bit
    - Binary number with leading 0 is positive
    - Binary number with leading 1 is negative
- Magnitude of positive numbers is just the binary representation
- Magnitude of negative numbers is found by performing the two’s complement
  - Complement the bits
    - Replace all the 1's with 0's, and all the 0's with 1's
  - Add one to the complemented number
- Carry in most significant bit position is thrown away when performing arithmetic

Two's Complement Example

- Performing two's complement on the decimal 7 to get -7
  - Using a five-bit representation

```
7  =  00111   Convert to binary

      11000   Complement the bits

      11000   Add 1 to the complement
    + 00001
    -------
      11001   Is  -7 in two's complement
```

Two's Complement Arithmetic

- Computing 8 - 7 using a two's complement representation with five-bit numbers

8 - 7 = 8 + (-7) = 1

01000 Two's complement of 8

11001 Two's complement of -7

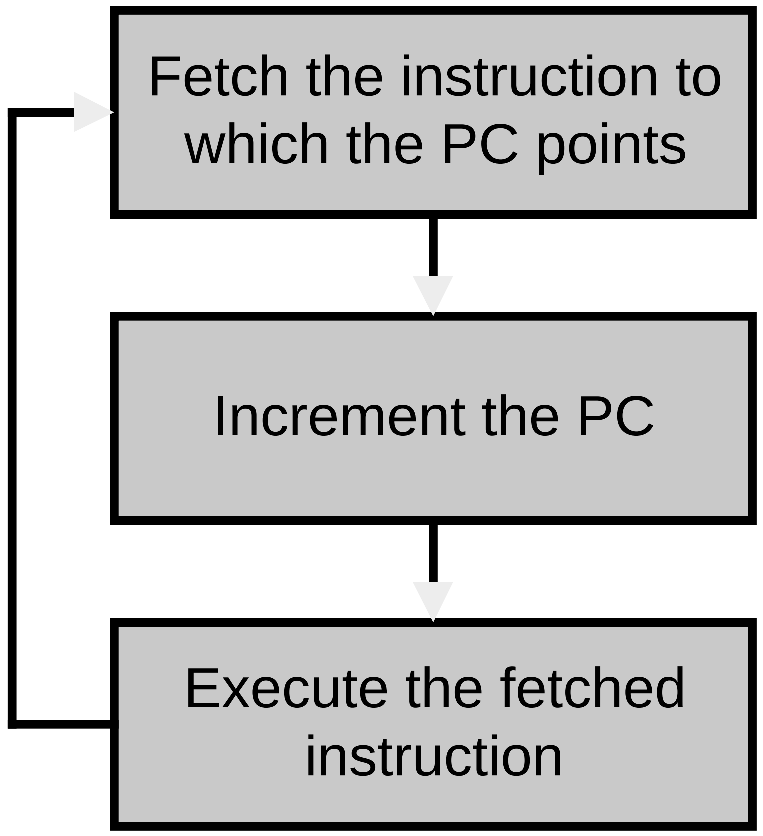
```
  01000  Add 8 and -7
+ 11001
 100001
```

Throw away the high-order carry as we are using a five bit representation

00001 Is the five-bit result

Control Unit

- The fetch/execute cycle is the steps the CPU takes to execute an instruction
- Performing the action specified by an instruction is known as “executing the instruction”
- The program counter (PC) holds the memory address of the next instruction


Fetch the instruction to which the PC points
Increment the PC
Execute the fetched instruction

Input and Output Devices

- Accessories that allow computer to perform specific tasks
  - Receiving information for processing
  - Return the results of processing
  - Store information
- Common input and output devices

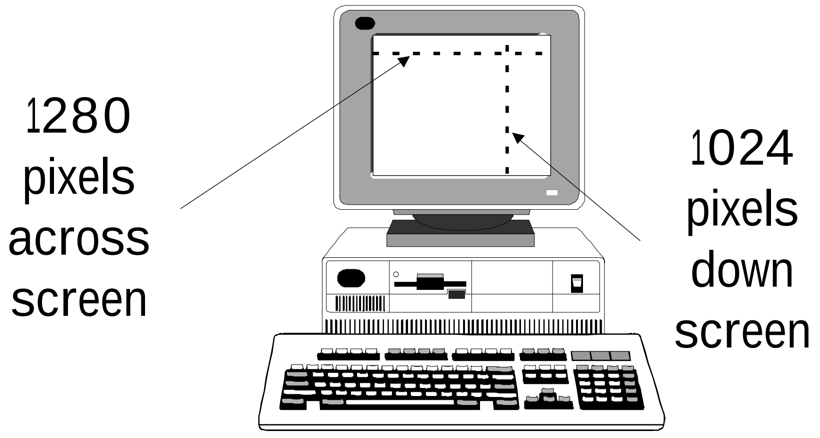
■ Speakers	Mouse	Scanner
■ Printer	Joystick	CD-ROM
■ Keyboard	Microphone	

- Some devices are capable of both input and output
  - Floppy drive
  - Hard drive
  - Magnetic tape units

Monitor

- Display device
- Also known as CRT (cathode ray tube)
- Operates like a television
- Controlled by an output device called a “graphics card”

Monitor and Card Characteristics



- Refresh rate
  - How fast image is updated on the screen
- Resolution
  - Displayable area
    - Measured in dots per inch, dots are often referred to as pixels (short for picture element)
  - Standard resolution is 640 by 480
  - Some cards support resolution up to 1280 by 1024
- Number of colors supported

Software

- Application software
  - Programs designed to perform specific tasks that are transparent to the user
- System software
  - Programs that support the execution and development of other programs
  - Two major types
    - Operating systems
    - Translation systems

Application Software

- Application software is the software that has made using computers indispensable and popular
- Common application software
  - Word processors
  - Desktop publishing programs
  - Spreadsheets
  - Presentation managers
  - Drawing programs

- Learning how to develop application software is our focus

Operating System

- Controls and manages the computing resources
- Important services that an operating system provides
  - File system
    - Directories, folders, files
  - Commands that allow for manipulation of the file system
    - Sort, delete, copy
  - Ability to perform input and output on a variety of devices
  - Management of the running systems
- Examples
  - MSDOS ®, Windows ®, UNIX ®

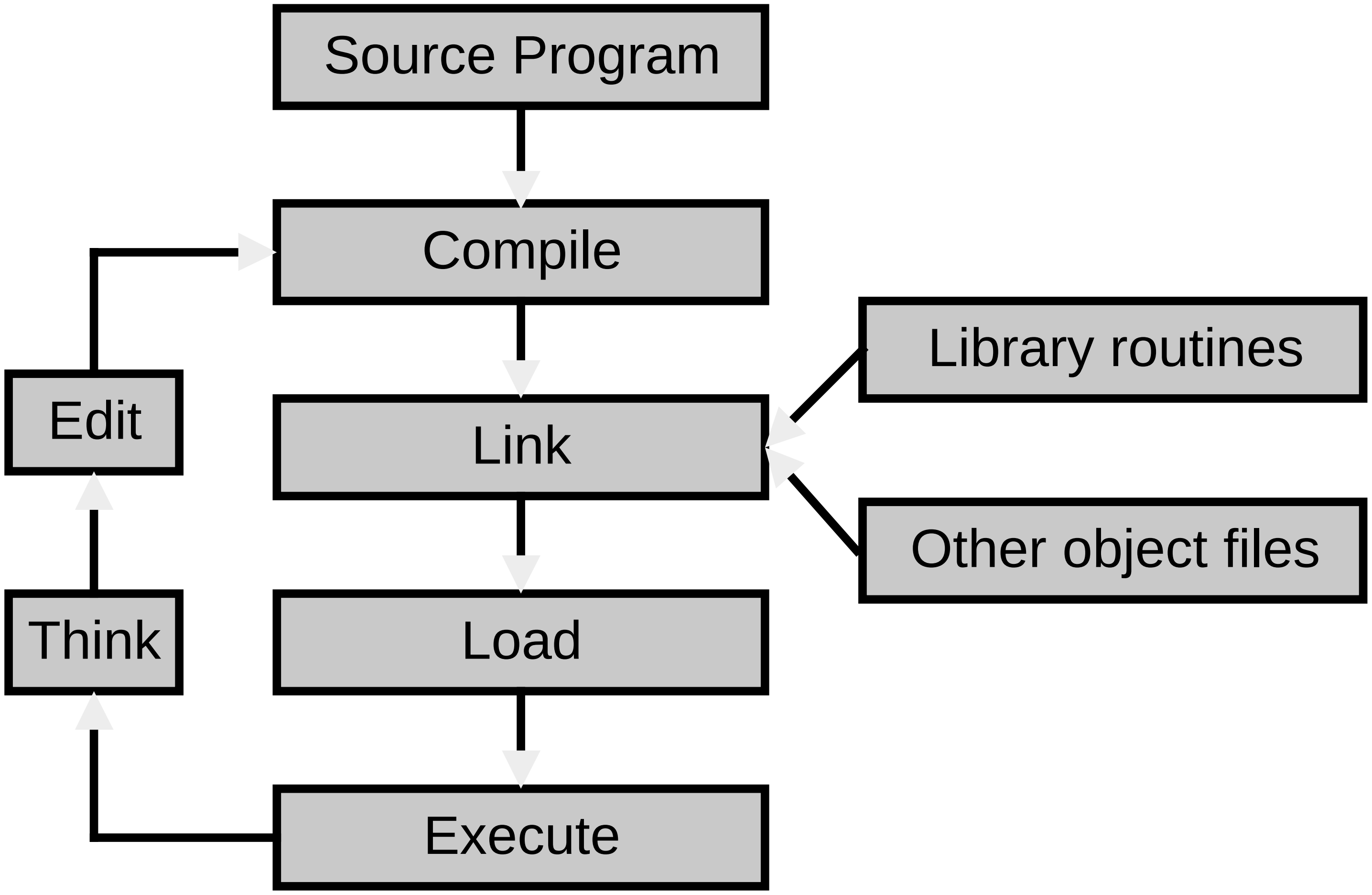
Translation System

- Set of programs used to develop software
- A key component of a translation system is a translator
- Types of translators
  - Compiler
    - Converts from one language to another
  - Linker
    - Combines resources
- Examples
  - Borland C++ ®, Microsoft Visual C++ ®, g++, Code Warrior ®
    - Performs compilation, linking, and other activities.

Software Development

- Major activities
  - Editing
  - Compiling
  - Linking with precompiled files
    - Object files
    - Library modules
  - Loading and executing
  - Viewing the behavior of the program

Software Development Cycle



IDEs

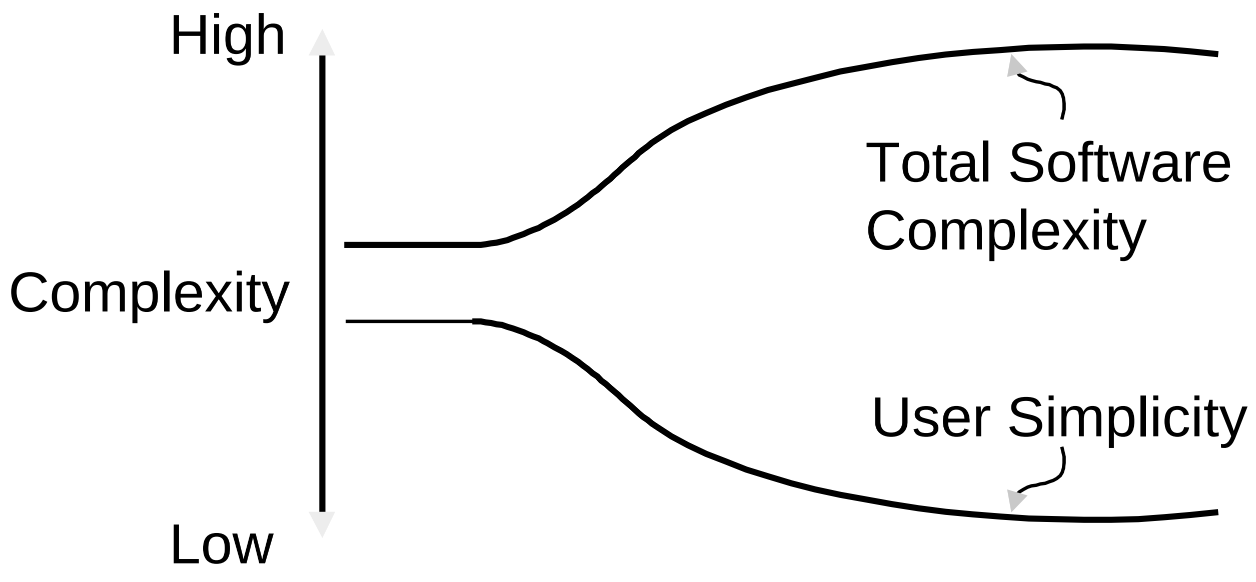
- Integrated Development Environments or IDEs
  - Translation systems that support the entire software development cycle
    - E.g., Borland, MS Visual C++, Code Warrior
- Combine all of the capabilities that a programmer would want handy while developing software
  - Editor
  - Compiler
  - Linker
  - Loader
  - Debugger
  - Viewer

Engineering Software

- Software engineering
  - Area of computer science concerned with building large software systems
- Challenge
  - Tremendous advances in hardware have not been accompanied by comparable advances in software

Complexity Trade-off

- System complexity tends to grow as the system becomes more user friendly



Software Engineering Goals

- Reliability
  - An unreliable life-critical system can be fatal
- Understandability
  - Future development becomes very difficult if software is hard to understand
- Cost Effectiveness
  - Cost to develop and maintain should not exceed profit
- Adaptability
  - System that is adaptive is easier to alter and expand
- Reusability
  - Improves reliability and maintainability, and reduces development costs

Software Engineering Principles

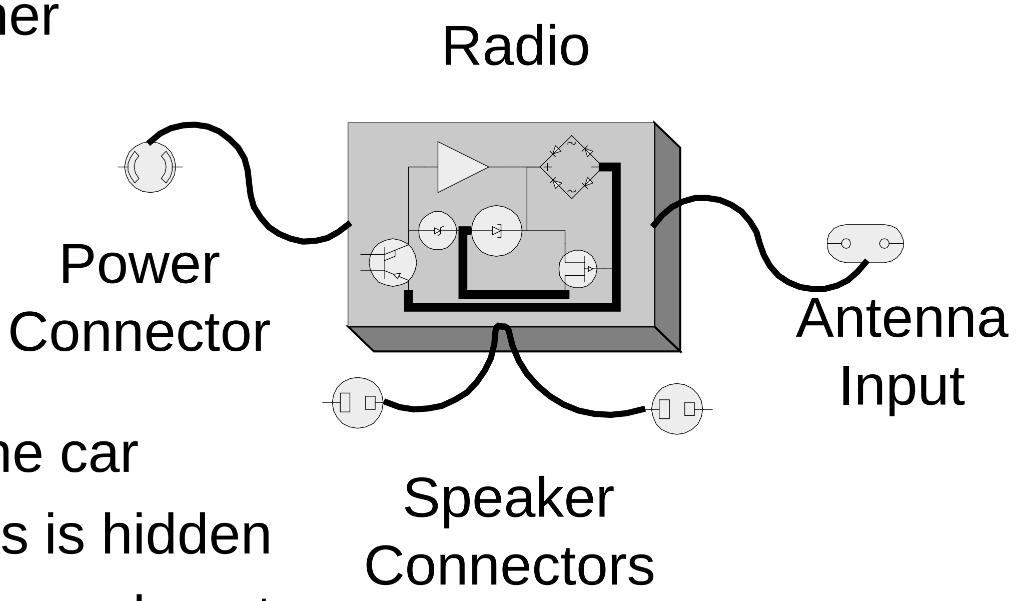
- Abstraction
  - Extract the relevant properties of an object while ignoring inessential details
- Encapsulation
  - Breaking down an object into parts, hiding and protecting its essential information, and supplying an interface to modify the information in a controlled and useful manner
- Modularity
  - Dividing an object into smaller pieces or modules such that the object is easier to understand and manipulate
- Hierarchy
  - Ranking or ordering of objects based on some relationship between them

Abstraction

- Process of extracting only the relevant properties of an object
- Extracted properties define a view of the object
- Example
  - Car dealer views a car from selling features standpoint
    - E.g., price, length of warranty, color, optional equipment
  - Mechanic views a car from systems maintenance standpoint
    - E.g., type of oil, size of the oil filter, type of spark plugs

Encapsulation

- Breaking down an object into parts, hiding and protecting its essential information, and supplying an interface to modify the information in a controlled and useful manner
- By hiding the information its representation and content can be changed without affecting other parts of the system
- Example - car radio
  - Interface consists of the controls and types of connectors for connecting the radio to the car
  - The details of how it works is hidden
  - To install and use a radio, we do not need to know anything about the radio’s electrical system

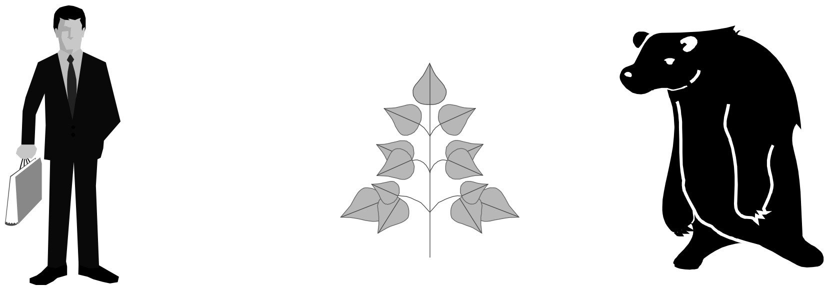


Modularity

- Dividing an object into smaller pieces or modules such that the modules hold useful information and the object is easier to understand and manipulate
- Most complex systems are modular
- Example - Automobile can be decomposed into subsystems
  - Cooling system
    - Radiator
    - Thermostat
    - Water pump
  - Ignition system
    - Battery
    - Starter
    - Spark plugs

Hierarchy

- Ranking or ordering of objects based on some relationship between them
- Hierarchies help us understand complex systems
  - Example - a company hierarchy helps employees understand the company and their positions within it
- For complex systems, a useful way of ordering similar abstractions is from least general to most general
  - Scientists use this technique to identify and classify species
- Hierarchical ordering based on natural relationships is called a taxonomy

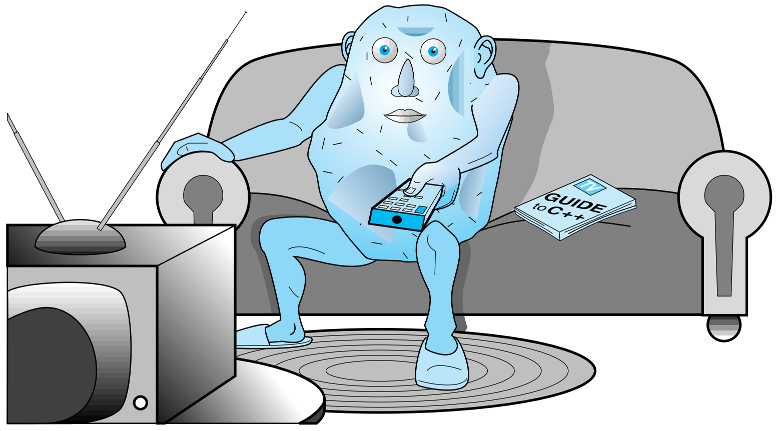


Northern Timber Wolf Taxonomy

Kingdom Animalia
Subkingdom Metaoza
Phylum Chordata
Subphylum Vertabrata
Superclass Tetrapoda
Class Mammalia
Subclass Theria
Infraclass Eutheria
Cohort Ferungulata
Superorder Ferae
Order Carnivora
Suborder Fissipeda
Superfamily Canoidea
Family Caninae
Subfamily Caninae
Genus Canis
Subspecies Canis lupus occidentalis
(Northern Timber Wolf)

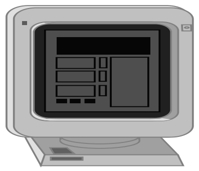
OO Design and Programming

- Object-oriented (OO) design and programming is a methodology that supports good software engineering
- Object-oriented design promotes thinking about software in a way that models the way we think and interact with the real world
- Example - watching television
  - The remote is a physical object with properties
    - Weight, size, can send messages to the television
  - The television is also a physical object with various properties



Objects

- An object is almost anything that can be attributed with the following characteristics
  - Name
  - Properties
  - The ability to act upon receiving a message
    - – Basic message types
      - Directive to perform an action
      - Request to change one of its properties



Object-Oriented Programming

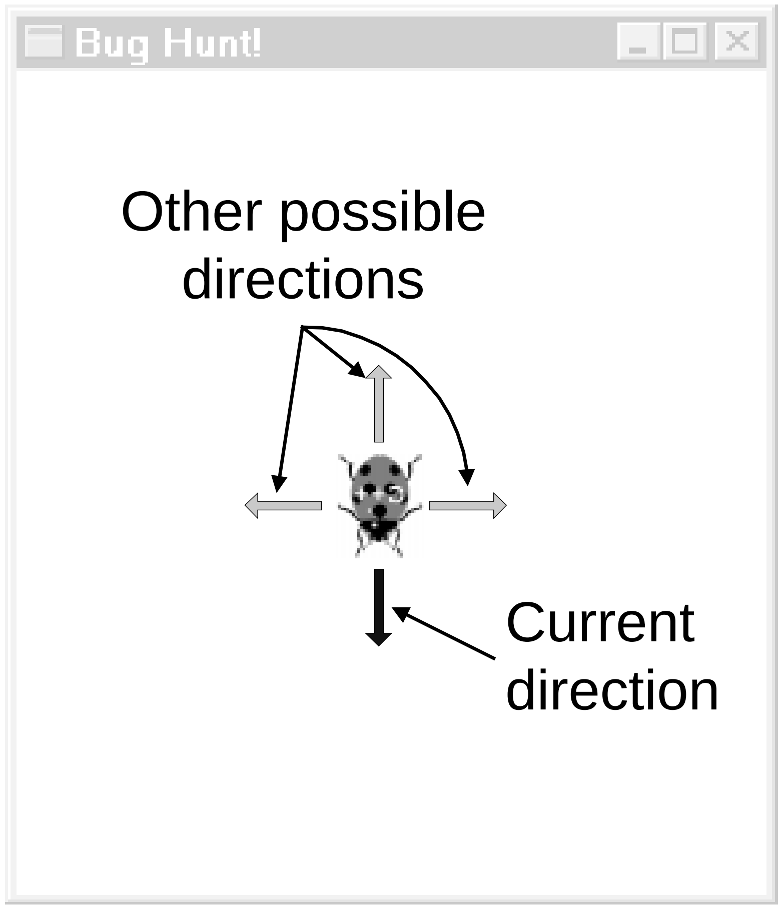
- Example
  - Sketch the design of a simple computer game called Bug Hunt
    - Goal of game is to eliminate the bugs on the screen
  - Features of game
    - A moving bug is displayed in a window on the screen
    - The bug changes directions randomly
    - A bug is eliminated by “swatting” it several times
    - If a bug is eliminated, a faster one pops up
    - If an attempted swat misses the bug, the game ends

Object-Oriented Programming

- First step
  - Determine the objects
    - Mouse
    - Window
    - Bug

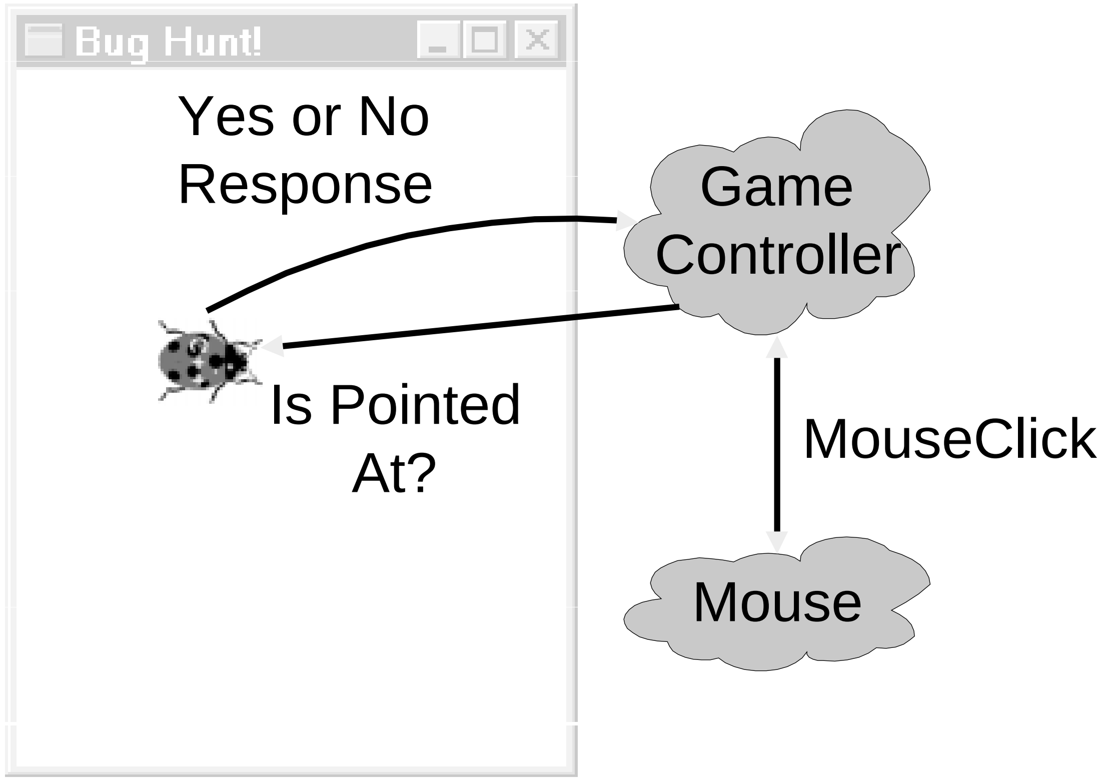
Bug Hunt

- To implement Bug Hunt, a bug needs the following properties
  - Position in the window
  - A display image or picture
  - Current speed
  - Current direction
  - Strength (the number of swats it takes to eliminate the bug)

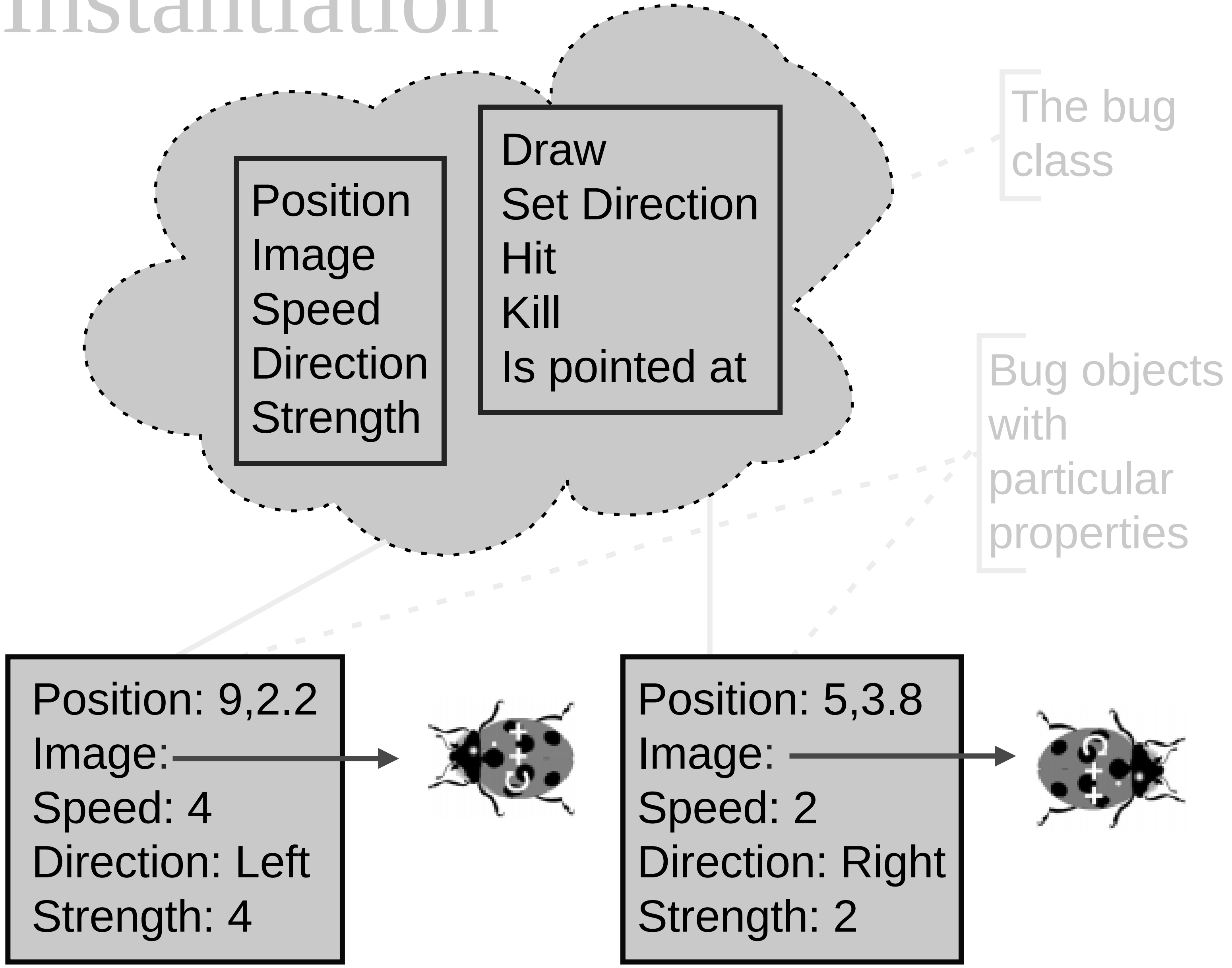


Bug Hunt

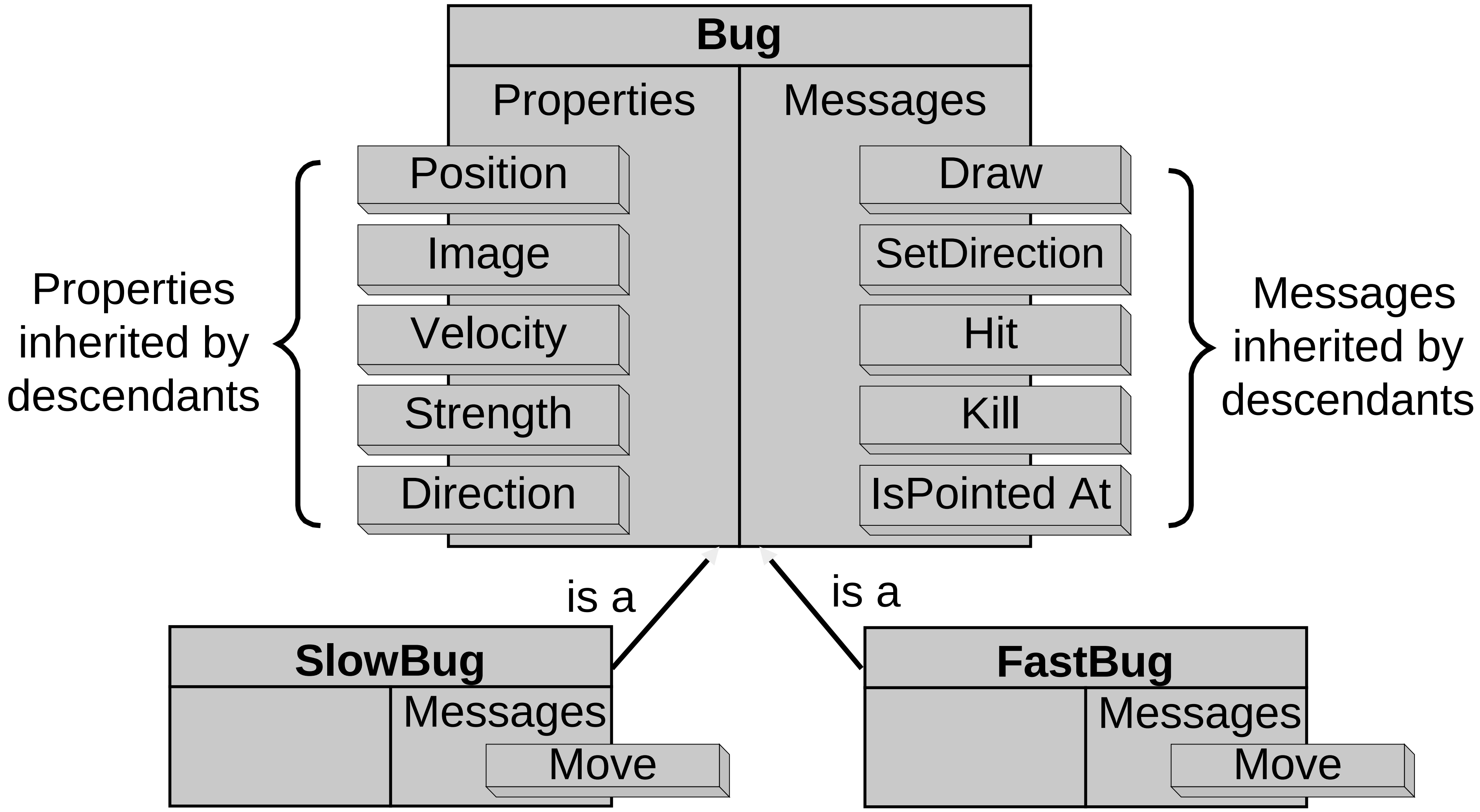
- A bug needs to handle the following messages
  - Draw
  - Move
  - Change direction
  - Hit
  - Kill
  - Is-pointed-at



Bug Instantiation



Inheritance


Bug
Properties
Messages
Position
Image
Velocity
Strength
Direction
Draw
SetDirection
Hit
Kill
IsPointed At
Properties inherited by descendants
Messages inherited by descendants
is a
is a
SlowBug
Messages
Move
FastBug
Messages
Move