Class Announcements

- Midterm 5/5
  - Study guide to be posted soon
- Assignment 3 due Thursday 4/30

E-commerce Marketing

- Internet provides marketers with new ways of identifying and communicating with customers.
- Long tail marketing:
  - Sell large number of unique items
  - Relatively few of each item sold
- Behavioral targeting: tracking online behavior of individuals on thousands of Web sites.
- Advertising formats include search engine marketing, display ads, rich media, and e-mail.

Networked Computing in B2B

- History predates Internet
- **Electronic Data Interchange (EDI)**
  - Exchange order information between firms involved in direct procurement
  - Originally done over private links
  - Only large firms could afford
  - Initially order and invoice
  - Existed since 70’s

Business-to-Business Electronic Commerce: New Efficiencies and Relationships

- Electronic data interchange (EDI)
  - Major industries have EDI standards that define structure and information fields of electronic documents for that industry.
  - More companies increasingly moving away from private networks to Internet for linking to other firms.
Data and information

by

David G. Messerschmitt

Key concept

The key commodity manipulated by information technology is information. To be manipulated in a computing/networking environment, information must be represented by data.

What is information?

Data

A bit is “0” or “1” — the atom of the information economy.

Data is a collection of bits, like:

- “0101110111010110”
- “0000011”
- “11101101101101011110111010110”

Note: the terms data and information are not always used consistently!

Representation

- Take the place of the original
- Equivalent to, in the sense that the original can be reconstructed from its representation
- Often the original can only be approximately reconstructed, although it may be indistinguishable to the user

- e.g. audio or video

Information

From a user (human) perspective...

...recognizable patterns that influence you in some way (perspective, understanding, behavior...)

In the computing infrastructure, information has a somewhat different connotation as structure and interpretation added to data.
ASCII

<table>
<thead>
<tr>
<th>Alphabet</th>
<th>Hex</th>
<th>Binary</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;7&gt;</td>
<td>x37</td>
<td>00110111</td>
</tr>
<tr>
<td>&lt;8&gt;</td>
<td>x38</td>
<td>00111000</td>
</tr>
<tr>
<td>&lt;9&gt;</td>
<td>x39</td>
<td>00111001</td>
</tr>
<tr>
<td>&lt;:&gt;</td>
<td>x3A</td>
<td>00111010</td>
</tr>
<tr>
<td>&lt;;&gt;</td>
<td>x3B</td>
<td>00111011</td>
</tr>
<tr>
<td>&lt;&lt;&gt;</td>
<td>x3C</td>
<td>00111100</td>
</tr>
<tr>
<td>&lt;-&gt;</td>
<td>x3D</td>
<td>00111101</td>
</tr>
<tr>
<td>&lt;/&gt;&gt;</td>
<td>x3E</td>
<td>00111110</td>
</tr>
<tr>
<td>?&gt;?</td>
<td>x3F</td>
<td>00111111</td>
</tr>
<tr>
<td>&lt;At&gt;</td>
<td>x40</td>
<td>01000000</td>
</tr>
<tr>
<td>&lt;A&gt;</td>
<td>x41</td>
<td>01000001</td>
</tr>
<tr>
<td>&lt;B&gt;</td>
<td>x42</td>
<td>01000010</td>
</tr>
<tr>
<td>&lt;C&gt;</td>
<td>x43</td>
<td>01000011</td>
</tr>
<tr>
<td>&lt;D&gt;</td>
<td>x44</td>
<td>01000100</td>
</tr>
</tbody>
</table>

Note that this representation is not unique…

…this one happens to be a standard (ANSI X3.110-1983)

A picture

This picture conveys information

This information is represented in this computer, but how?

Representation of picture: image

Expanding a small portion of the picture, we see that it is represented by square pixels…

….300 tall by 200 wide….

….with a range of 256 intensities per pixel

300 • 200 • 8 bits = 480,000 bits (but it can be compressed)

Color picture

A color picture can be represented by three monochrome images…

At the expense of three times as many bits

Representation needs to be standardized

If the representation is not standardized, the information is garbled!

Communicate data to another user or organization

Communicate data to another user or organization
**Regeneration**

- Make a precise copy of the data (copy bit by bit)
- If you know the representation, this is equivalent to making a precise copy of the information
- Each such precise copy is called a generation
- Process is called regeneration

**Replication of information**

Anything that can be regenerated can be replicated any number of times. This is a blessing and a curse.

**Analog information cannot be regenerated**

Analog information can be copied, but not regenerated.

We will never know exactly what the original of this Rembrandt looked like.

**Discrete information can be regenerated**

Regeneration can preserve data (but not its original physical form).

Regeneration is possible for information represented digitally (which is tolerant of physical deterioration).

\[ 0 + \text{noise} = 0 \]
\[ 1 + \text{noise} = 1 \]

**Replication of information requires knowledge of representation**

Replication preserves the integrity of the data, but that is not sufficient.

Every .xxx DOS file is a representation.

**Implications**

Digitally represented information can be preserved over time or distance in its precise original form by occasional regeneration:
- digital library
- digital telephony

Replication of data is easy and cheap.
Implications (con’t)

- Replication of information requires knowledge of the structure and interpretation
  - Standardization or some other means
- Extreme supply economies of scale
- You can give away or sell and still retain
- Unauthorized replication or piracy relatively easy

Architecture

by

David G. Messerschmitt

What is Architecture?

How do you architect a solution?

Copyright notice

©Copyright David G. Messerschmitt, 2000. This material may be used, copied, and distributed freely for educational purposes as long as this copyright notice remains attached. It cannot be used for any commercial purpose without the written permission of the author.

Architecture

A system is decomposed into interacting subsystems

Each subsystem may have a similar internal decomposition

Three elements of architecture

- Decomposition
- Organization
- Functionality
- Responsibility
- Interaction
- Cooperation
System examples

Let’s quickly look at some system decomposition examples

- Quick tour of information technology systems

---

Time sharing

- ASCII terminal (no graphics)
- Mainframe (database and application server)
- Time sharing
- Point-to-point wire (no network)

---

Two-tier client/server

- Local-area network
- Server
- Mainframe

---

Three-tier client/server

- Client
- Application server
- Enterprise data server

---

System integration

1. Architecture
2. Subsystem implementation
3. System integration
   - Bring together subsystems and make them achieve desired system functionality
     - Testing
     - Modifications often needed

---

Emergence

- Subsystems are specialized
- Have simple functionality
- Higher-level system functionality arises from the interaction of subsystems
  - Called: Emergence
- E.g. airplane flies, but subsystems can’t...
Why system decomposition?

- Divide and conquer approach to containing complexity
- Reuse
- Consonant with industry structure (unless system is to be supplied by one company)
- Others?

Networked computing infrastructure

by

David G. Messerschmitt

Layering

Existing layers

Elaboration or specialization

Services

Example of Layering: networking

Application

Messages

Transport

Packets

Network

Frames

Link

Bits

Physical

Signals

Software Layering

Application

Middleware

Operating System

Operating system functions

- Graphical user interface (client only)
- Hide details of equipment from the application
- Multitasking
- Resource management
  - Processing, memory, storage, etc
- etc
Middleware Functions

- Capabilities that can be shared by many applications, but that is not part of OS
  - Example: Database Management System (DBMS)
- Hide details of OS from application
  - Java Virtual Machine
- More purposes we’ll talk about later.

What’s a database?

- Database
  - File with specified structure
  - Example: relational table

A Database

<table>
<thead>
<tr>
<th>Year</th>
<th>City</th>
<th>Accommodation</th>
<th>Tourists</th>
</tr>
</thead>
<tbody>
<tr>
<td>2002</td>
<td>Oakley</td>
<td>Bed&amp;Breakfast</td>
<td>14</td>
</tr>
<tr>
<td>2002</td>
<td>Oakland</td>
<td>Resort</td>
<td>196</td>
</tr>
<tr>
<td>2002</td>
<td>Oakland</td>
<td>Resort</td>
<td>250</td>
</tr>
<tr>
<td>2002</td>
<td>Berkeley</td>
<td>Camping</td>
<td>120000</td>
</tr>
<tr>
<td>2002</td>
<td>Berkeley</td>
<td>Bed&amp;Breakfast</td>
<td>34000</td>
</tr>
<tr>
<td>2002</td>
<td>Berkeley</td>
<td>Resort</td>
<td>390800</td>
</tr>
<tr>
<td>2002</td>
<td>Albany</td>
<td>Camping</td>
<td>87000</td>
</tr>
<tr>
<td>2003</td>
<td>Albany</td>
<td>Bed&amp;Breakfast</td>
<td>52470</td>
</tr>
<tr>
<td>2003</td>
<td>Oakland</td>
<td>Bed&amp;Breakfast</td>
<td>86120</td>
</tr>
<tr>
<td>2003</td>
<td>Oakland</td>
<td>Resort</td>
<td>210</td>
</tr>
<tr>
<td>2003</td>
<td>Berkeley</td>
<td>Camping</td>
<td>118800</td>
</tr>
<tr>
<td>2003</td>
<td>Berkeley</td>
<td>Bed&amp;Breakfast</td>
<td>95500</td>
</tr>
<tr>
<td>2003</td>
<td>Berkeley</td>
<td>Resort</td>
<td>415000</td>
</tr>
<tr>
<td>2003</td>
<td>Albany</td>
<td>Camping</td>
<td>7800</td>
</tr>
<tr>
<td>2003</td>
<td>Albany</td>
<td>Bed&amp;Breakfast</td>
<td>6750</td>
</tr>
</tbody>
</table>

Storage Middleware example: DBMS

- Database Management System (DBMS)
  - Manage Multiple databases
  - Allow multiple applications to access common databases
  - Implement standard data “lookup” (query) functions.

Client - Server Computing

Client Server Example

Client

“i want to see www.google.com”

Server
Client Server Example - Layers Revealed

Client
Application:
Infrastructure
Internet
Packet
Packet
Server
Application
Infrastructure
Packet
Packet

3-Tier Client Server Architecture example

Client
Clicks, keystrokes
Application Server
What is Bob’s balance?
Shared data
Balance $0.50

Shared data

3-Tier Client Server Architecture example

Client
Application Server
Web Server
Common Gateway Interchange
Application Logic
Shared data

3-Tier Client Server Architecture example

Client
Application Server
Web Server
Java Servlet
Application Logic
Shared data

3-Tier Client Server Architecture example

Client
Application Server
Web Server
Common Gateway Interchange
Application Logic
Database Management System (DBMS)
Database
Shared data

Relational Database

<table>
<thead>
<tr>
<th>Customer</th>
<th>Balance</th>
<th>Customer Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice</td>
<td>$527</td>
<td>Silver</td>
</tr>
<tr>
<td>Bob</td>
<td>$0.50</td>
<td>Bronze</td>
</tr>
<tr>
<td>Charles</td>
<td>$1000000</td>
<td>Gold</td>
</tr>
</tbody>
</table>

In some implementations, Application Logic and Web Server can be put on different machines.
**DBMS Responsibilities**

- Hide Changes in the Database hardware from the Application
- Standard operations on the data, including searches, such a search is called a query.
- Separate Database Management from Applications, so that many applications can access the same data.
- Security, Integrity, Backup, fault tolerance, etc.

---

**3-Tier Client Server Architecture in General**

- Takes inputs from client
- Decides what to be done next
- Decides what shared data to access and manipulates it
- Processes shared data
- Takes inputs from user
- Make requests of server
- Display responses of server
- Support multiple applications with common data
- Protect critical data
- Decouple data administration and application administration

---

**Peer to peer**

- Client
- Server
- Peer

---

**Architecture Example**

- Financial institution
- Book distribution centers
- Customers
- Enterprise
- Inter-enterprise

---

*Slide adapted from slides for Understanding Networked Applications*  
By David G. Messerschmitt. Copyright 2000. See copyright notice
**Architecture**

- **Conceptualization**
  - What is it you are trying to do?

- **Example Concept:**
  - Small HHC for flight attendants.
  - HHC tells flight attendants which passengers are higher priority.
    - Who paid the highest fares
    - Who has been a more valuable customer in past
  - Flight attendant discriminates based on this
    - Free drinks, meals, and pillows to valuable customers
    - Ignore less valuable customers

**Example Concept:**

**Architecture**

- How do you begin to architect a solution for a problem like this?

- Break it into modules!

**HHC Architecture**

When a module is composed of sub-modules, the architecture is **hierarchical**.

**Architecture**

- We are using a **layered architecture** as well.
  - Allows reuse of previously built infrastructure.
Granularity tradeoff.

- How big should we make the modules
  - Many simple small ones
  - Or a few complicated big ones...
- This aspect of modularity is called granularity.
- Which is better?

HHC Server

*HHC Server Application*
  - Computation of key statistics
  - Communication with HHC

*Networking Infrastructure*
  - Communication with airline database

- Again, we see layering and hierarchy.
- Between each module we specify an interface

Data server

*Standard Database* "queries" (SQL) from HHC Server

Database

Our architecture makes use of the existing interface of the airline database, so we don’t need to redesign it!

Interfaces

Compute Mean and Variance

*Computation of key statistics*

N numbers of float type

Compute Mean and Variance

2 Numbers of float type that signify: Mean, Variance

More on Data types

- Data passing an interface is often specified in terms of a limited number of standard data types
- Data type = range of values and allowable manipulation
- Data type does not presume a specific representation, to allow heterogeneous platforms
- Representation must be known when data passes a specific module interface

Interfaces specifications are often made precise by using data types.

- Example type: float
  - A number with a decimal place
  - Has a certain allowable range, and precision.
Example data types

Integer
- "natural number between -32,767 and +32,768"
- Could be represented (in many ways) by 16 bits
  - since $2^{15} = 65,536$

Float
- "number of the form $m \times 10^n/32768$, where $m$ is in the range -32,767 to +32,768 and $n$ is in the range -255 to +256"
- Could be represented by $16 + 8 = 24$ bits

More data types

Character
- "values assuming a-z and A-Z plus space and punctuation marks"
  - could be represented by 7 or 8 bits

Character string
- "collection of $n$ characters, where $n$ is customizable"
  - could be represented by $7 \times n$ bits

Compound data types

Programmer-defined composition of basic data types
Example:

```java
Employee {
    String name;
    String address;
    Integer year_of_birth;
    etc.
}
```
**Implementation**

Module B

Compute Mean and Variance

\[ \text{SUM} = \sum_{i=1}^{N} x_i \]
\[ \text{MEAN} = \frac{\text{SUM}}{N} \]
\[ \text{VARIANCE} = \frac{\sum_{i=1}^{N} (x_i - \text{MEAN})^2}{N} \]

Implementation 1:

- Should he use it?
  - NO!!!! Why??
- Either A should compute “SUM” himself, or sit down with B and redesign the

**Encapsulation**

- The designer of B might take measures to hide “SUM” from A so that A is not able to violate the agreed interface.
  - Example: B does not declare “SUM” as a global variable.
- Making a modules implementation details inaccessible to other modules is called **encapsulation**.

**Interfaces**

Module B

Compute Mean and Variance

This simple interface example allows for only one action of module B.
- Action is “Compute mean and variance.”
- Other examples are possible.

**Possible software interface**

Menu of actions

- action-1
- action-2
- action-3
- ...

Example:

- Action 1: Compute mean
- Action 2: Compute variance
- Action 3: Compute mode
- Etc..

**Protocol**

In addition to atomic actions, an interface may define protocols
- Protocol == finite sequence of actions required to achieve a higher level function
- One action can be shared by multiple protocols
- Multiple modules may participate in a protocol

**Protocol Example**

Hello: I’m the HHC of Airplane#1234
Hello: I’m the gate 32 server
These were the unruly passengers on last flight “Passengers noted”
(Might be passed As an array of a compound data type “passenger,” which in turn is composed of standard types like integer, and string)

Tell me about the passengers of my next flight
- Return Passenger Data

Tell me about the weather at my next destination.
- Return Weather Data
Another Interface Example: Automatic teller machine (ATM)

What is the interface between this machine and the customer?

Steps

Define available actions
Define, for each higher level function, a protocol
  - Single action or a finite sequence of actions

Interface building blocks

Message on screen or printed
  - Menu of actions or returns from an action
  - Touch selection of action
Keypad
  - Input parameters to an action
Card reader
  - Authentication, input parameters
Money output slot
  - Returns money

Action: authentication

Parameters
Internal functionality
Returns

Action: authentication

Parameters
  - Identity (card in slot)
  - Institution (card in slot)
  - PIN (typed on keypad)

Internally, it contacts institution and matches against its database, institution noted for all subsequent actions (example of state)

Returns
  - Screen message ("Invalid PIN" or menu of available actions)

Action: specify_account

Parameters
Internal functionality
Returns
**Action: specify_account**

Parameters
- Account (touch screen from menu of choices)

Internally, choice noted for all subsequent actions (another example of state)

Returns
- None

---

**Action: amount**

Parameters
- Dollars_and_cents (typed on keypad)

Internally, amount noted (another example of state)

Returns
- Success or failure (state dependent, for example for a withdraw failure when dollars_and_cents exceeds balance)

---

**Protocol: cash_withdrawal**

What is the sequence of actions?

---

**Protocol: cash_withdrawal**

```
authentication → failure
|
choose objective → other objectives
|   
account → no accounts
|       
amount → balance exceeded!
```

---

**Goals**

Understand better
- how layering is used in the infrastructure
- how it contains complexity
- how it coordinates suppliers
- how it allows new capabilities to be added incrementally

---

**More on layering**

by
David G. Messerschmitt
Interaction of layers

Layer above is a client of the layer below.

Layer below as a server to the layer above.

Each layer provides services to the layer above.

...by utilizing the services of the layer below and adding capability.

Layering

Elaboration or specialization

Existing layers

Layering builds capability incrementally by adding to what exists.

Three types of software

Application

Components and frameworks:
  What is in common among applications

Infrastructure:
  Basic services (communication, storage, concurrency, presentation, etc.)

Part of Microsoft vs. DOJ dispute

Microsoft position

DOJ position

Major layers

Applications

Application frameworks and components

Middleware

Operating system

Network
**Data and information**

**Application**
Deals with information

Assumes structure and interpretation

**Infrastructure**
Deals with data

Ignores structure and interpretation

---

**Data and information in layers**

- The infrastructure should deal with data, or at most minimal structure and interpretation
- The application adds additional structure and interpretation
- This yields a separation of concerns

---

**Package = file, message**

In the simplest case, the infrastructure deals with a package of data (non-standard terminology)

- collection of bits
- specified number and ordering

The objective of the infrastructure is to store and communicate packages while maintaining data integrity

File for storage, message for communication

---

**Data integrity**

Retain the

- values
- order
- number of bits in a package

---

**Example 1**

Bob sends a letter to Alice

Bob
US Postal Service
Shipping Container
ABC Airlines

Alice
UK Royal Mail
Shipping Container

---

**Example 2**

**Application**

Web server
Web browser
Screen

**Operating system**

Operating system
File system

**Network**

Fragmentation
Collection of packets
Assembly

---
Example 3

HHC Server Application  Passenger Information  HHC Client Application
Windows OS  message  Palm OS  message
Networking Infrastructure (Contains: TCP/IP, WiFi)
Collection of Packets  Networking Infrastructure (Contains: TCP/IP, WiFi)

Example 3: Network Infrastructure Expanded

HHC Server Application  Passenger Information  HHC Client Application
Windows OS  message  Palm OS  message
TCP transport layer  Packets  TCP transport layer
WiFi Link Layer  Packets  WiFi Link Layer
WiFi Physical Layer  WiFi Physical Layer
Networking Infrastructure  Radio Signals  Networking Infrastructure

Example 4

HHC Server Application  Passenger Information  HHC Client Application
Windows OS  message  Palm OS  message
Networking Infrastructure (Contains: TCP/IP, WiFi)
Collection of Packets  Networking Infrastructure (Contains: TCP/IP, WiFi)

HEADQUARTERS
Airline Dataserver

“Send me today’ s  flight information”  DBMS  message
Windows OS  message  Linux OS
Networking Infrastructure  Layers within TCP/IP, WiFi
Collection of Packets  Networking Infrastructure  Layers within TCP/IP, WiFi

Information in the infrastructure

Sometimes it is appropriate for the infrastructure to assume structure and interpretation for data
• to add capabilities widely useful to applications
• to help applications deal with heterogeneous platforms, where representations differ

At most, data types

Data and information

Application
Deals with information
Assumes structure and interpretation
Assumes standard data types

Infrastructure
Deals with data types