Announcements

- 2\textsuperscript{nd} Folio Article due today

- **Read**
  - Messerschmitt Ch 11.1 - 11.2 (325-335)
  - Messerschmitt Ch 15.1 - 15.2 (415-425)
Architecture Example
Example Concept:
Architecture

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

- Break it into modules!
**Compound data types**

Programmer-defined composition of basic data types

Example:

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

PARAMETERS

N numbers of Float type

Computation of key statistics

Compute Mean and Variance

2 Numbers of float type that signify: Mean, Variance

INTERFACE

RETURNS
Implementation

One module should not be concerned with other module’s implementation

→ “Separation of concerns.”

One module should see the other only through its interface—implementation details hidden.

→ Abstraction

Compute Mean and Variance

\[ \text{MEAN} = \frac{1}{N} \sum_{i=1}^{N} x_i \]

\[ \text{VARIANCE} = \sum_{i=1}^{N} (x_i - \text{MEAN})^2 \]
Though different, this implementation is ok too.

We can choose the implementation details however we want, as long as we comply with the agreed interface.
Implementation

Module A

Computation of key statistics

“I need to get the sum, I’ll just take it from B”

Module B

Compute Mean and Variance

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

Should he use it?

- NO!!! Why??

Either A should compute “SUM” himself, or sit down with B and redesign the interface
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 module's implementation details inaccessible to other modules is called **encapsulation**.
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

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

(Might be passed As an array of a compound data type “passenger,” which in turn is composed of standard types like integer, and string)
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
- 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**
- 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!
More on layering

Slides modified from those by David G. Messerschmitt
Example 1

Bob sends a letter to Alice

Bob

Envelope

US Postal Service

Shipping Container

ABC Airlines

Alice

Envelope

UK Royal Mail

Shipping Container
Interaction of layers

Layer above is a client of the layer below

Each layer provides services to the layer above....

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

Layer below as as a server to the layer above
Example 2

Web server

Web browser

Screen

HTML

Application

Operating system

Network

File

Message

File system

Fragmentation

Collection of packets

Assembly
Package = file or message

Infrastructure deals with a package of data (non-standard terminology)
- collection of bits
- specified number and ordering

Infrastructure stores and communicates packages while maintaining data integrity
- File for storage
- message for communication
Data integrity

Retain the

- values
- order
- number

of bits in a package
Example 3

HHC Server Application

Windows OS

Networking Infrastructure (Contains: TCP/IP, WiFi)

Passenger Information

HHC Client Application

Palm OS

Networking Infrastructure (Contains: TCP/IP, WiFi)

Collection of Packets

Message
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
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
Architecture

HEADQUARTERS

Airline Dataserver

Airline Intranet

HHC Server

HHC

Wireless Link
Two ways to design a system

- Decomposition from system requirements
- Assembly from available components

System requirements

Requirements

Available components

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

Component: A subsystem purchased “as is” from an outside vendor

(Alternative – building your own subsystem)

A component implementation is encapsulated (although often configurable)

Slide adapted from slides for Understanding Networked Applications
By David G Messerschmitt. Copyright 2000. See copyright notice
The Palm OS we are buying “off the shelf” and integrating into our architecture. The Palm OS is a *component*. 
Other Examples of components

Computer
Disk drive
Network
Network router
Operating system
Integrated circuit
Database management system

Why is a component implementation encapsulated?

Slide adapted from slides for Understanding Networked Applications by David G Messerschmitt. Copyright 2000. See copyright notice.
Interoperability

- Components are interoperable when they interact properly to achieve some desired functionality.

- Increasingly component interoperability cannot be dependent on end-user integration:
  - PC and peripherals
  - Enterprise, inter-enterprise, consumer applications
  - Role for standardization
Outsourcing: A subsystem design is contracted to an outside vendor

Responsibility is delegated

Slide adapted from slides for *Understanding Networked Applications* by David G Messerschmitt. Copyright 2000. See copyright notice.
Suppose we choose to pay another firm to develop the user interface.

This is called **Outsourcing**.

Why would we do this?
System Integration

- Suppose we
  - Bring together all these subsystems
  - and test them...

- This is called **System Integration**
System integration

- Bring together subsystems;
- make them work together;
- to achieve a goal.

Requires
- Testing
- Making modifications to
  - architecture and/or
  - subsystem implementation
Can System Integration be Outsourced?

- Of course!
Supplier Types

- Three types of suppliers:
  - Component Suppliers
  - Custom Subsystem Developers
  - System Integrators

- (Some suppliers are 2 or even 3 of above.)
Two ways to sell Software

**Product**
- Customer installed and operated
- Often (but not necessarily) sold or licensed at a fixed price

**Service**
- Functionality provided over a wide-area network
- Often (but not necessarily) sold by subscription

Slide adapted from slides for *Understanding Networked Applications* by David G Messerschmitt. Copyright 2000. See copyright notice.
Recall: Infrastructure and Applications

Infrastructure

- Equipment and/or software used by many applications

Applications

- Provide specific capabilities and features serving individual users.
Four possibilities

<table>
<thead>
<tr>
<th>Product</th>
<th>Service</th>
</tr>
</thead>
<tbody>
<tr>
<td>Microsoft Office</td>
<td>Hotmail</td>
</tr>
<tr>
<td>Application</td>
<td></td>
</tr>
<tr>
<td>Infrastructure</td>
<td>Internet DNS</td>
</tr>
<tr>
<td>Personal computer</td>
<td></td>
</tr>
</tbody>
</table>

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

- Two types
  - Bundled
    - An infrastructure provider bundles applications with their infrastructure
      - Example: AOL, telephony service providers
  - Unbundled
    - A provider of an application service without providing an infrastructure service
      - Examples?
Examples of unbundled ASP model

- Yahoo: Web-based calendar
- Hotmail: Web-based email
- Schwab: Web-based stock trading
Unbundled ASP model

Advantageous to user

- Proven way to reduce installation, integration, and maintenance costs
- Contractual obligation for availability and quality
- Location independence

Slide adapted from slides for *Understanding Networked Applications* by David G Messerschmitt. Copyright 2000. See copyright notice
Advantages to supplier

- Ongoing revenue stream supporting upgrade and maintenance
- Usage-based revenue better aligned with user’s value proposition
- Opportunity for price discrimination, advertising revenue, etc.
Some pricing alternatives

Price discrimination?
Usage dependent?
Terms and conditions
  - fixed, leasing, per-use, subscription
  - warrantee, service level agreements
Bundles
  - maintenance, support, releases, provisioning and operations
Who pays?
  - sometimes not the end user

Slide adapted from slides for *Understanding Networked Applications* by David G Messerschmitt. Copyright 2000. See copyright notice.
Infrastructure acquisition

Infrastructure

- Build and operate
- Build but do not operate
- Do not build but operate
- Neither

Trend

- Outsourced operations
- System integrator
- Service provider

Slide adapted from slides for *Understanding Networked Applications* by David G Messerschmitt. Copyright 2000. See copyright notice.
Application acquisition

Application

\{ Develop internally \}
\{ Buy as product \}
\{ Contract development \}
\{ Product w/ customization \}

Trend

Software supplier
Outsource developer
Supplier, consultants
Stovepipe vs. Integrated Infrastructure

**Stovepipe Architecture**

---or---

**Turnkey Solution**

- Single supplier provides all encompassing solution
- (complete with infrastructure)

**Integrated Infrastructure**

- Separate infrastructure that can support many applications

---

Slide adapted from slides for *Understanding Networked Applications* by David G Messerschmitt. Copyright 2000. See copyright notice.
From stovepipe to layering

Data Voice Video

Application-dependent infrastructure

Many applications

Integrated Infrastructure
(Maybe broken into Additional layers.)

Application-independent
Stovepipe vs. Integrated Infrastructure

- What are some examples of each?

- What are the advantages of each approach?
Vertical Integration vs. Diversification

- A company is **vertically integrated** when it makes rather than buys the subsystems in its products.

- A **diversified** company produces products across different industry segments.

Slide adapted from slides for *Understanding Networked Applications* by David G Messerschmitt. Copyright 2000. See copyright notice.
Vertical Integration vs. Diversification

Why do customers favor less vertical integration?

- Prefer competition amongst component suppliers
- Mix and match components
- Reduced lock in

Disadvantages??

- Customer needs to integrate components from different suppliers.
Vertical Integration vs. Diversification

Why do customers favor diversification?

- Reduce coordination costs by having to deal with fewer suppliers.
General Trend

- Less Vertical Integration

- More Diversification

- Of course there are exceptions...
Today's supplier structure

- Applications
- Frameworks and components
- Middleware
- Infrastructure (network, OS) software
- Equipment (network, computers)
- Semiconductors, components

Slide adapted from slides for *Understanding Networked Applications* by David G Messerschmitt. Copyright 2000. See copyright notice.
Role of Venture Capital in Computing.

- Open interfaces allow small firms to contribute components without having to develop entire solution.

- Fast decision making and no supplier lock-in.

- Other Advantages?
Standardization
Purpose of a standard?

- Allow products or services from different suppliers or providers to be interoperable
Scope of a standard

Included:

- interfaces (physical, electrical, information)
- architecture (reference model)
- formats and protocols (FAP)
- compliance tests (or process)

Excluded:

- implementation
- (possibly) extensions
Reference model

Decide decomposition of system
- where interfaces fall

Defines the boundaries of competition and ultimately industrial organization
- competition on the same side of an interface
- complementary suppliers on different sides
- hierarchical decomposition at the option of suppliers
- (possibly) optional extensions at option of suppliers

Slide adapted from slides for *Understanding Networked Applications* by David G Messerschmitt. Copyright 2000. See copyright notice.
Some issues

Once a standard is set

- becomes possible source of industry lock-in; overcoming that standard requires a major (~10x?) advance
- may lock out some innovation

In recognition, some standards evolve

- IETF, CCITT (modems), MPEG
- backward compatibility
Types of standards

**de jure**
- Sanctioned and actively promoted by some organization with jurisdiction, or by government

**de facto**
- Dominant solution arising out of the market
- Voluntary industry standards body

Industry consortium
Common or best practice

Examples?

---

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

*de jure*
- GSM, ISDN Telephone interface

*de facto*
- Hayes command set, Windows API, Pentium instruction set, Ethernet

Voluntary industry standards body
- OMG/CORBA, IAB/IETF, IEEE

Industry consortium
- W3C/XML, SET

Best practice
- Windowed GUI
The changing process

- As technology and industry move more quickly, the global consensus standards activity has proven too unwieldy
  - e.g. ISO
- “New age” standards activities are more informal, less consensus driven, a little less political, more strategic, smaller groups
  - e.g. OMG, IETF, ATM Forum, WAP

Programmable/extensible approaches for flexibility
- e.g. XML, Java
Old giving way to the new

Slide adapted from slides for *Understanding Networked Applications* by David G Messerschmitt. Copyright 2000. See copyright notice.
Reasons for change

- From government sanction/ownership to market forces
  - Increasing fragmentation
  - Importance of time to market

Greater complexity

- Less physical/performance constraint for either hardware or software

Slide adapted from slides for *Understanding Networked Applications* by David G Messerschmitt. Copyright 2000. See copyright notice.
Lock-in

(Particularly open) standards reduce consumer lock-in

- Consumers can mix and match complementary products

Increase supplier lock-in

- Innovation limited by backward compatibility
- e.g. IP/TCP, x86, Hayes command set

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

Standards can harness network effects to the industry advantage

- Revenue = (market size) \times (market share)

Increases value to customer

Increases competition

- Only within confines of the standard
- But forces customer integration or services of a system integrator

Slide adapted from slides for *Understanding Networked Applications* by David G Messerschmitt. Copyright 2000. See copyright notice
Why standards?

*de jure* are customer driven to reduce confusion and cost
*de facto* standards are sometimes the result of positive feedback in network effects

Customers and suppliers like them because they
- increase value
- reduce lockin

Governments like them because they
- promote competition in some circumstances
- May believe they can be used to national advantage
Approaches

Consensus
- ISO

Collaborative design
- MPEG

Competitive “bake off”
- IETF

Coordination of vendors
- OMG

Slide adapted from slides for Understanding Networked Applications by David G Messerschmitt. Copyright 2000. See copyright notice.
Open Standards

- Open standard - a standard that is well documented, unencumbered by intellectual property rights and restrictions, and available to any vendor.

- What are the advantages?

- What are the disadvantages?
Why companies participate

Pool expertise in collaborative design
- e.g. MPEG

Have influence on the standard

Get technology into the standard
- Proprietary, with expectation of royalties
- Non-proprietary

Reduced time to market
Standards applied to Business Processes?

- Can you standardize business processes?

  Yes!

  - ISO 9000
    - A set of standardized business processes for Quality Management.
    - Supports TQM (Total Quality Management)
  
  - RosettaNet
    - A set of standardized business processes, and accompanying standardized data interfaces/formats for conducting e-business.