Chapter 7

by

David G. Messerschmitt
Components

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

A component implementation is encapsulated (although often configurable)

Examples of components

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

Why is a component implementation encapsulated?
Understanding Networked Applications

Interoperability

- Components are interoperable when they interact properly to achieve some desired functionality.
- Increasingly component interoperability cannot be dependent on integration, or is dependent on end-user integration.
  - PC and peripherals
  - Enterprise, inter-enterprise, consumer applications
  - Role for standardization

Outsourcing

Outsourcing: A subsystem design is contract to an outside vendor.
Responsibility is delegated.
**System integration**

Architecture ➔ subsystem implementation ➔ system integration

- Bring together subsystems and make them cooperate properly to achieve desired system functionality
  - Always requires testing
  - May require modifications to architecture and/or subsystem implementation

**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?
Components

A component implementation is **encapsulated** (although often configurable).

Examples of components

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

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

**Why is a component implementation encapsulated?**
Interoperability

- Components are interoperable when they interact properly to achieve some desired functionality.
- Increasingly component interoperability cannot be dependent on integration, or is dependent on end-user integration.
  - PC and peripherals
  - Enterprise, inter-enterprise, consumer applications
  - Role for standardization

Two ways to design a system

- System requirements
- Available components
- Decomposition from system requirements
- Assembly from available components
Outsourcing

Outsourcing: A subsystem design is contract to an outside vendor

Responsibility is delegated

Three types of software

Application

- Components and frameworks:
  What is in common among applications

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

by
David G. Messerschmitt

Outline

• Motivation for standards
• Elements of a standard
• Types of standards
• Process to develop a standard
Network effects

- The value of a product to the adopter depends on the number of other adopters
  - Direct
    - e.g. fax machine
  - Indirect, through common content or software
    - e.g. Windows, CD music

Lock-in

- Consumer:
  - Switching costs make consumer reluctant to adopt a new product
- Supplier:
  - Switching costs or cannibalization of existing products make supplier reluctant to pursue new product opportunity
Consumer lock-in

- Prevalence increases as the industry fragments, and consumer has to purchase complementary products to get a “complete solution”
  - Switching costs discourage moving to complete new solution
  - Supplier with a “better mousetrap” can’t penetrate market unless product is compatible with existing complementary product

Purpose of a standard

- Infrastructure:
  - Allow products or services from different suppliers or providers to be interoperable
- Application:
  - Enable applications to run across uncoordinated administrative domains
Scope of a standard

- **Included:**
  - architecture (reference model)
  - interfaces (physical, electrical, information)
  - 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
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

- **de jure**
  - Ada, VHDL
- **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 (protocols, SGML)
- “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

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
Lock-in

• (Particularly open) standards reduce consumer lock-in
  – Consumers can mix and match complementary products
  – e.g. IBM (in their day) and Microsoft are perceived to be lock-in problems, other agendas in addition to pleasing customers

• Increase supplier lock-in
  – Innovation limited by backward compatibility
  – e.g. IP/TCP, x86, Hayes command set

Question

• What are some examples of open standards that reduce consumer lock-in?
  – Intranet applications
    • WWW, newsgroups, calendar, etc
  – Linux
  – PC peripherals
    • ISA, serial/parallel port, etc
  – Others?
Network effects

- Standards can harness network effects to the industry advantage
  - Revenue = (market size) x (market share)
- Increases value to customer
- Increases competition
  - Only within confines of the standard
  - But forces customer integration or services of a system integrator

Question

- What are examples of standards that serve to tame network effects?
  - Internet protocols
  - XML
  - CORBA
  - DVD
  - others?
Why standards?

- *de jure* standards 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 lock-in
- Governments like them because they
  - promote competition in some circumstances
  - May believe they can be used to national advantage

Voluntary standards process

Sanctioning organization(s)

Ongoing committees

Participating companies
Approaches

- Consensus
  - ISO
- Collaborative design
  - MPEG
- Competitive “bake off”
  - ITEF
- Coordination of vendors
  - OMG

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