Rethinking Components: From Hardware and Software to Systems

Article by David Messerschmitt
Summary Slides prepared by John Musacchio

Introduction

Components
- Elements of functionality sold as a unit, incorporated into multiple uses
- Origin in interchangeable parts

“ICT” industries hitting a wall
- OS and cell networks evolve in generations
  - Heavy redesign
  - Difficult to justify development
- Can componentization be used to make improvements easier?

Background

• Modularity
  - Decompose; assign functionality and interactions
  - Goals in decomposition
    - Weak coupling across modules
    - Strong cohesion within modules
  - Granularity
    - Hierarchy
• Modules vs. components
  - Components: Ability to combine in different ways
  - Modules: designed from decomposing system requirements
  - Components: design for composability

Background

• Reference Architecture
  - Achieve minimum coordination between component designers to achieve composability
  - Fine grain example: transistor components
  - Coarse grain example: comm. Network
• Infrastructure
  - Captures operational commonalities; make available to all

Rationale for new view

What Has Changed:
• Large-Scale Integration
  - Old days: component unit of manufacture
  - Today: manufacturing granularity a complex subsystem
• Software
  - Key part of ICT products
• Outsourcing
  - Move from purchasing parts to purchasing services
  - Outsourcing often “handcrafted” vs. componentized
• Increase Scale:
  - Mass Replication of similar elements: i.e. memory chips
  - At higher levels, systems more complex thus more context specific
  - Higher grain more handcrafted

Rationale for New View

Generalization of Component concept
• Example:
  - Communication link as a component
  - End-to-end network connection as component
• Physical Realization
  - not tied to a physical package
• Element of Design
  - Intermingled Implementation
    - I.e. components sharing a common i.c. package
• Dynamic Replication and Deployment
  - Assembled and torn down as needed
• Transparent to Implementation
• Fragmented Control and Shared Design responsibility
  - As in the “network connection” component.
Rationale for New View

Opportunities
- Managing complexity
  - Evolution of stable intermediate forms
  - Biological examples: Cell, eyes, liver
- Specialization
  - Division of labor by system function
  - Division by granularity
- Supplier coordination
  - Ensuring compatibility between components of different firms.
  - Hierarchical component design structured way to handle

Opportunities (cont’d)
- Product Diversity
  - Increase variety for mix and match
- Design quality
  - Fewer distinct designs. Easier to debug
- Time to Market
  - Less handcrafting; faster
- Design Flexibility
  - Hand crafted designs tightly bound to current requirements; not thinking of future uses
- Economies of Substitution
  - Changing needs met by replacing or upgrading components rather than total redesign
- Operational Flexibility
- Design Costs

Rationale for New View

Challenges
- Design Costs
  - Making components that meet more diverse requirements more expensive
- Replication costs
- Innovation
  - Compatibility with existing components discourages innovation
- Competition
  - Components with open interfaces invite competition
  - Firms who build systems from components lose differentiation opportunity
- Homogeneity vulnerabilities

What Components are Not
- Modularity (they are a form of modularity)
- Infrastructure
  - Components emphasize composition
  - Infrastructure emphasizes extension
  - ICT grows by adding new layers
    - WEB – started as application, now infrastructure
- Reuse
  - Modify old designs for new needs

System Components

Today: Component communities lie in hardware or software communities
However, ideally
  - Component modularity design ought to just consider functionality provided to higher level of hierarchy

Hardware-Software components (cont’d)
- Communication Link Example
  - Goals
    - Self contained
    - Composable in variety of contexts
    - Substituted freely in variety of technologies
  - Web services
System Components

• Distributed Components
  - Need not be in same physical location
  - Example: Network Connection as component
    • Why?
      - Strong internal coupling
  - Connection as Component vs. TCP Socket
    • interface focused on communication needs
    • End-to-end
    • No specific transport mechanism assumed within

System Components

• Application-Infrastructure
  - Example: Web Service Components
• Social-Technology
  - Example: insurance policy-as-component

Historical Perspective

• Interchangeable parts
  - Not embraced for mass production until 1860s
• Combinatorial innovation
  - Made possible by interchangeable parts
• Fitting
  - Remachining of parts to make them fit together
  - Perfected by Ford
• Standardization
  - Interchangeability across manufactures
  - Promoted by SAE

Design for Componentization

• Finer Grain Components
  - Produced by single firm
  - Sources of value:
    • Multiple uses; composition; context agnostic; encapsulated; independent deployability
• Coarse Grained Components
  - Coordination within firm or industry

Design for Componentization

• Decomposition \(\Rightarrow\) component realization \(\Rightarrow\) component composition
• Outcome: reference architecture
Design for Componentization

- Reference architecture
  - Supplier group
    - Example: (IETF)
  - End-user organizations
    - ARPA driving Internet
    - Universities making Sakai project
- Complex systems
  - High-level components that have fragmented control → industry responsibility
  - Lower-level Components → single firm and/or consortia

Component Markets

- Pressure to be on-time and under budget
  - Managers may not have incentive to develop components
  - Longer term opportunities might justify extra costs
- Componentization push may take
  - Pressure from professional orgs
  - Government
  - Users