Abstract

Software is a dominating factor in the development of complex systems. It plays a crucial role in the performance of the final product at the one hand, while it contributes significantly to the development cost and elapsed time of development. This paper will discuss the role of software in the broader system context. An improved understanding of the role of software enables the system architect, and the other stakeholders of the product creation process, to integrate the software development better. In this way hardware-software tradeoffs can be made, balancing performance, costs and risks.
Relative Contribution of SW

The diagram illustrates the relative effort contributed by different fields over time. By 1970:
- Electronics (red) was contributing some effort.
- Mechanics (pink) was contributing more effort.
- Physics/chemistry (grey) was contributing the least.

By 2000:
- Electronics continued to grow, becoming more significant.
- Mechanics showed a significant increase, surpassing electronics.
- Physics/chemistry also increased, but remained the least prominent.

The trend indicates a growing importance of mechanics and electronics, with physics/chemistry lagging behind.
Mismatch between Role and Discipline

**role of software**

- integration technology
- captures *application* functionality
- defines lot of *system* behavior
- determines how much of potential *system* performance is achieved
- acts as director

**mismatch!**

**focus of software discipline**

- software technologies, such as:
  - programming languages
  - data bases
  - operating systems
  - component technologies
  - engineering practices
Control Hierarchy along Technology axis

human user

application SW

control SW

digital electronics

analog or power electronics

mechanical device

optical device

sensor

Feedback

Control

legend

local automation or safety
Characterization of disciplines

Mechanics  Analogue / power Electronics  Digital Electronics  Software

concrete  abstract

tangible  intangible

mature  immature

production lead-time  flexible?

material cost
Quality Attributes annotated with SW relation

usable
- usability
- attractiveness
- responsiveness
- image quality
- wearability
- storability
- transportability

interoperable
- connectivity
- 3rd party extendible

liable
- liability
- testability
- traceability
- standards compliance

serviceable
- serviceability
- configurability
- installability

future proof
- evolvability
- portability
- upgradability
- extendibility
- maintainability

down-to-earth attributes
- cost price
- power consumption
- consumption rate
  (water, air, chemicals, etc.)
- size, weight
- accuracy

logistics friendly
- manufacturability
- logistics flexibility
- lead-time

ecological
- ecological footprint
- contamination
- noise
- disposability

dependable
- safety
- security
- reliability
- robustness
- integrity
- availability

efficient
- resource utilization
- cost of ownership

effective
- throughput or productivity

consistent
- reproducibility
- predictability

consistent
- throughput or productivity

The Role of Software in Systems
6 Gerrit Muller

version: 1.3
October 22, 2014
RSWqualities
Design Aspects related to SW

- **Customer objectives**
  - Performance, safety, security, ...
  - e.g., distributed or centralized control

- **Application**
  - Design philosophy per quality attribute
  - Granularity, scoping, containment, cohesion, coupling
  - Interfaces, allocation, budgets

- **Functional**
  - Information model (entities, relations, operations)
  - Identification, naming
  - HAL_message_acknowledge_status versus ACK

- **Conceptual**
  - Static characteristics, dynamic behavior
  - System-level infrastructure

- **Realization**
  - Software development process, environment, repository, and tools
  - Life cycle, configuration management, upgrades, obsolescence
  - Feedback tools, for instance monitoring, statistics, and analysis
  - Persistence
  - Licensing, SW-keys

- **Technology choices**
  - Make, outsource, buy, or interoperability decisions

---

The Role of Software in Systems
7  Gerrit Muller

Version: 1.3
October 22, 2014
RSWdesignAspects
SW Mechanisms

Customer objectives
Application
Functional
Conceptual
Realization

error handling, exception handling, logging
processes, tasks, threads
configuration management; packages, components, files, objects, modules, interfaces
automated testing: special methods, harness, suites
signaling, messaging, callback scheduling, notification, active data, watchdogs, timeouts
locking, semaphores, transactions, checkpoints, deadlock detection, rollback
identification, naming, data model, registry, configuration database, inheritance, scoping
resource management, allocation, fragmentation prevention, garbage collection
persistence, caching, versioning, prefetching, lazy evaluation
licensing, SW-keys
bootstrap, discovery, negotiation, introspection
call graphs, message tracing, object tracing, etc.
distribution, allocation, transparency; component, client/server, multitier model