

Module Role of Software in Complex Systems

by *Gerrit Muller* University of South-Eastern Norway-NISE

e-mail: `gaudisite@gmail.com`

`www.gaudisite.nl`

Abstract

This module addresses the role of software in complex systems

Distribution

This article or presentation is written as part of the Gaudí project. The Gaudí project philosophy is to improve by obtaining frequent feedback. Frequent feedback is pursued by an open creation process. This document is published as intermediate or nearly mature version to get feedback. Further distribution is allowed as long as the document remains complete and unchanged.

September 9, 2018

status: preliminary

draft

version: 1.0



The Role of Software in Systems

by *Gerrit Muller* University of South-Eastern Norway-NISE

e-mail: `gaudisite@gmail.com`

`www.gaudisite.nl`

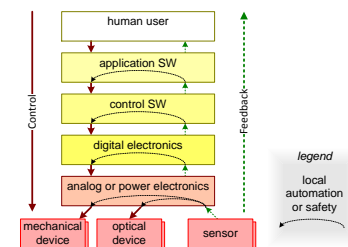
Abstract

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 significant 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.

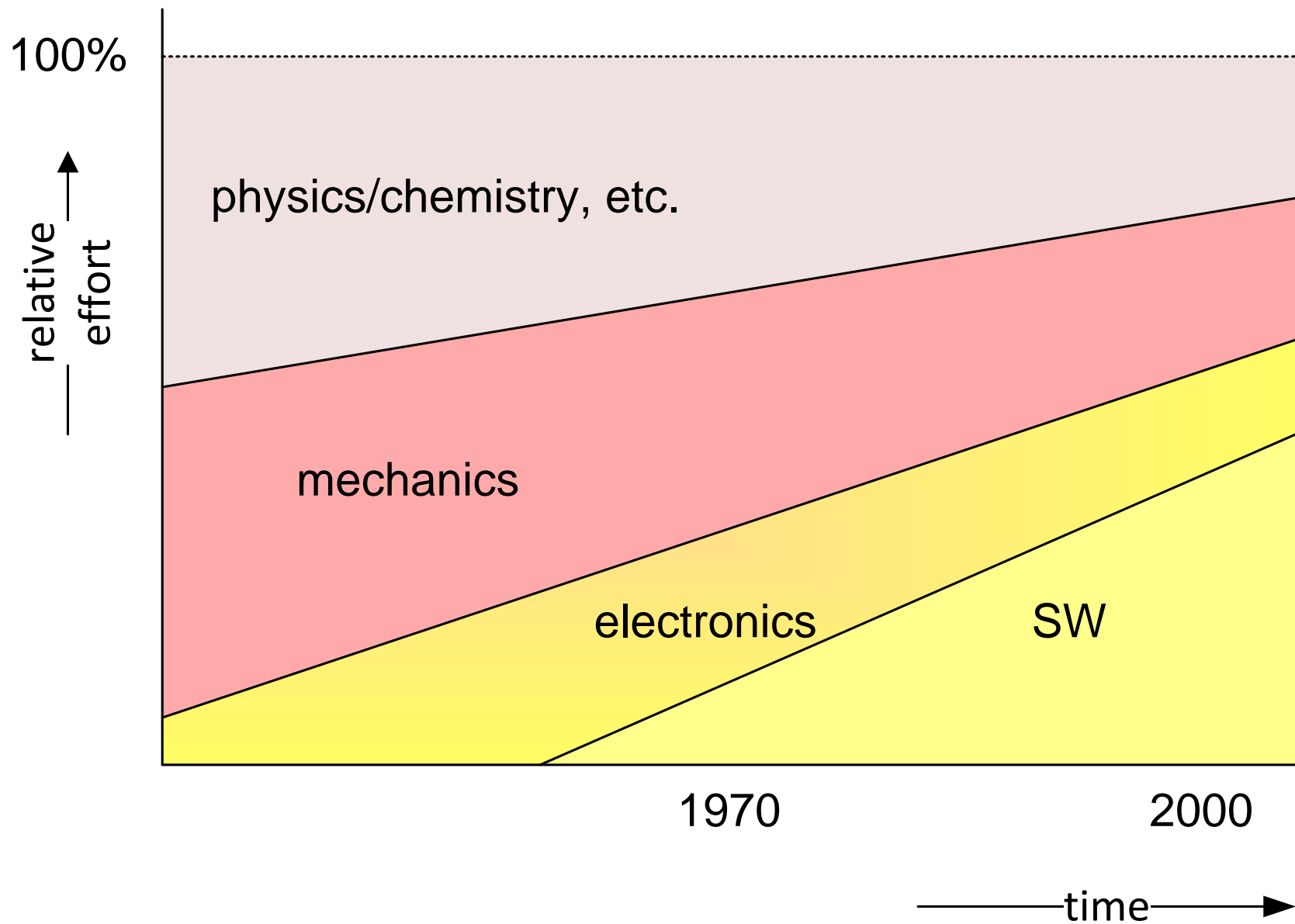
Distribution

This article or presentation is written as part of the Gaudí project. The Gaudí project philosophy is to improve by obtaining frequent feedback. Frequent feedback is pursued by an open creation process. This document is published as intermediate or nearly mature version to get feedback. Further distribution is allowed as long as the document remains complete and unchanged.

September 9, 2018
status: concept
version: 1.3



Relative Contribution of SW



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

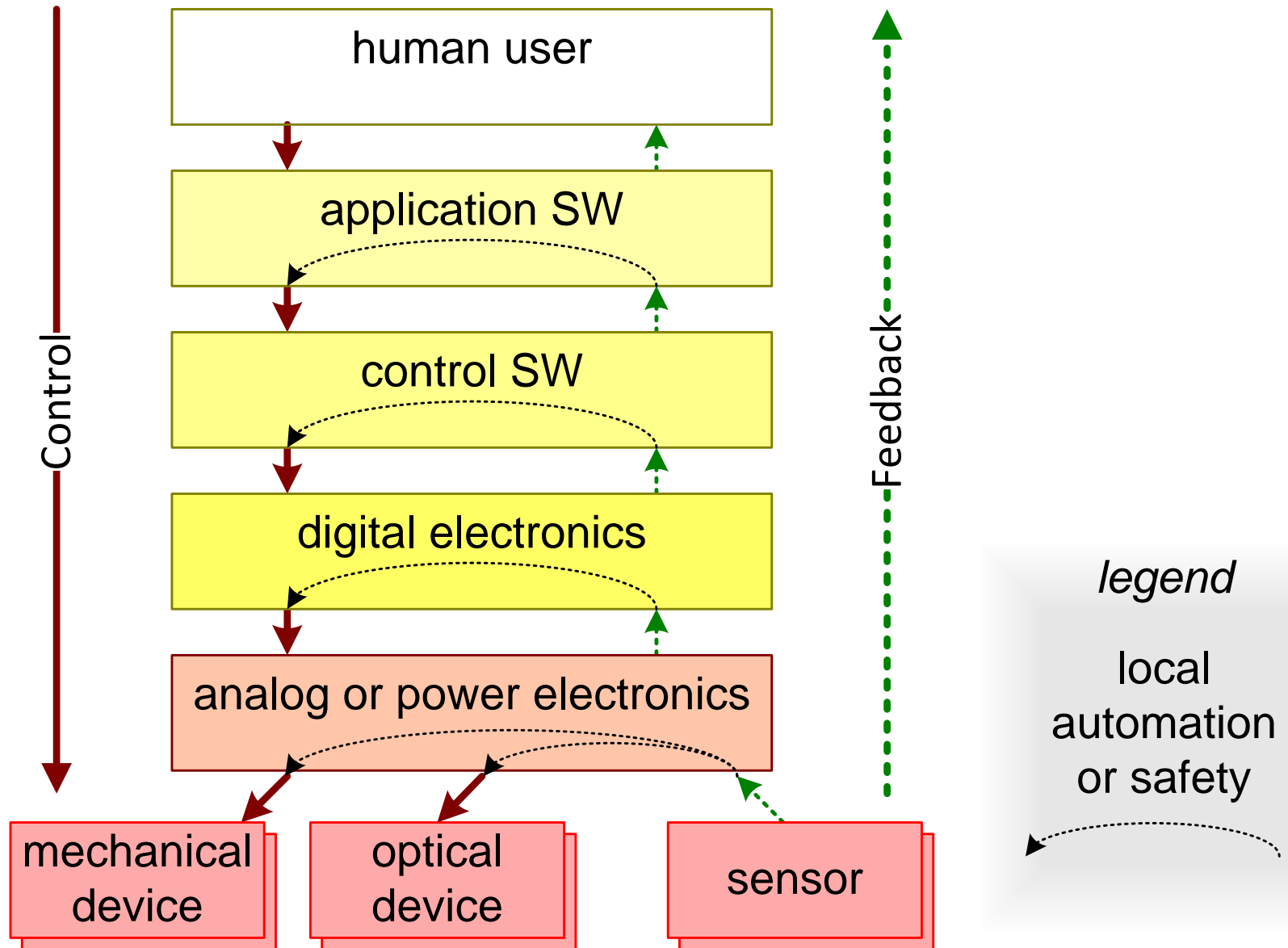
data bases

operating systems

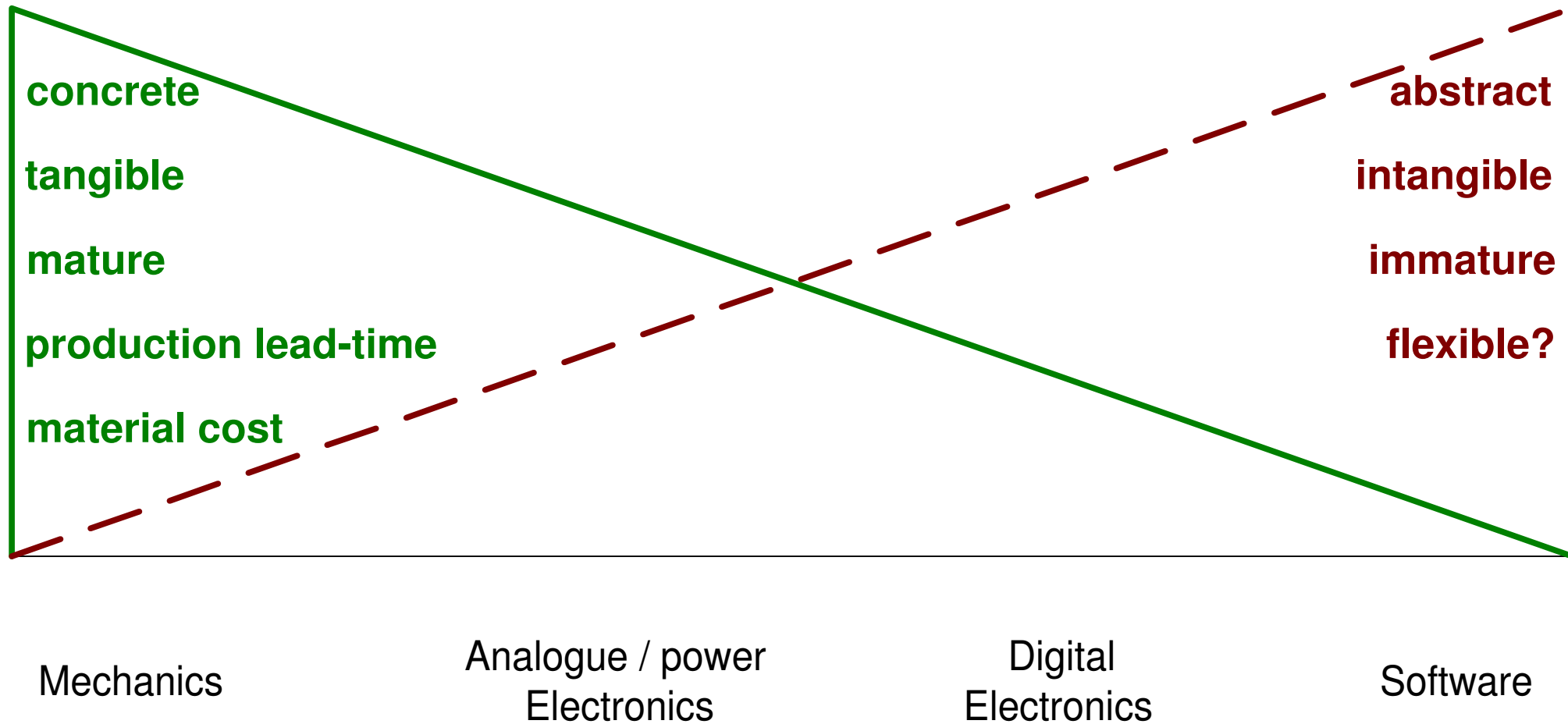
component technologies

engineering practices

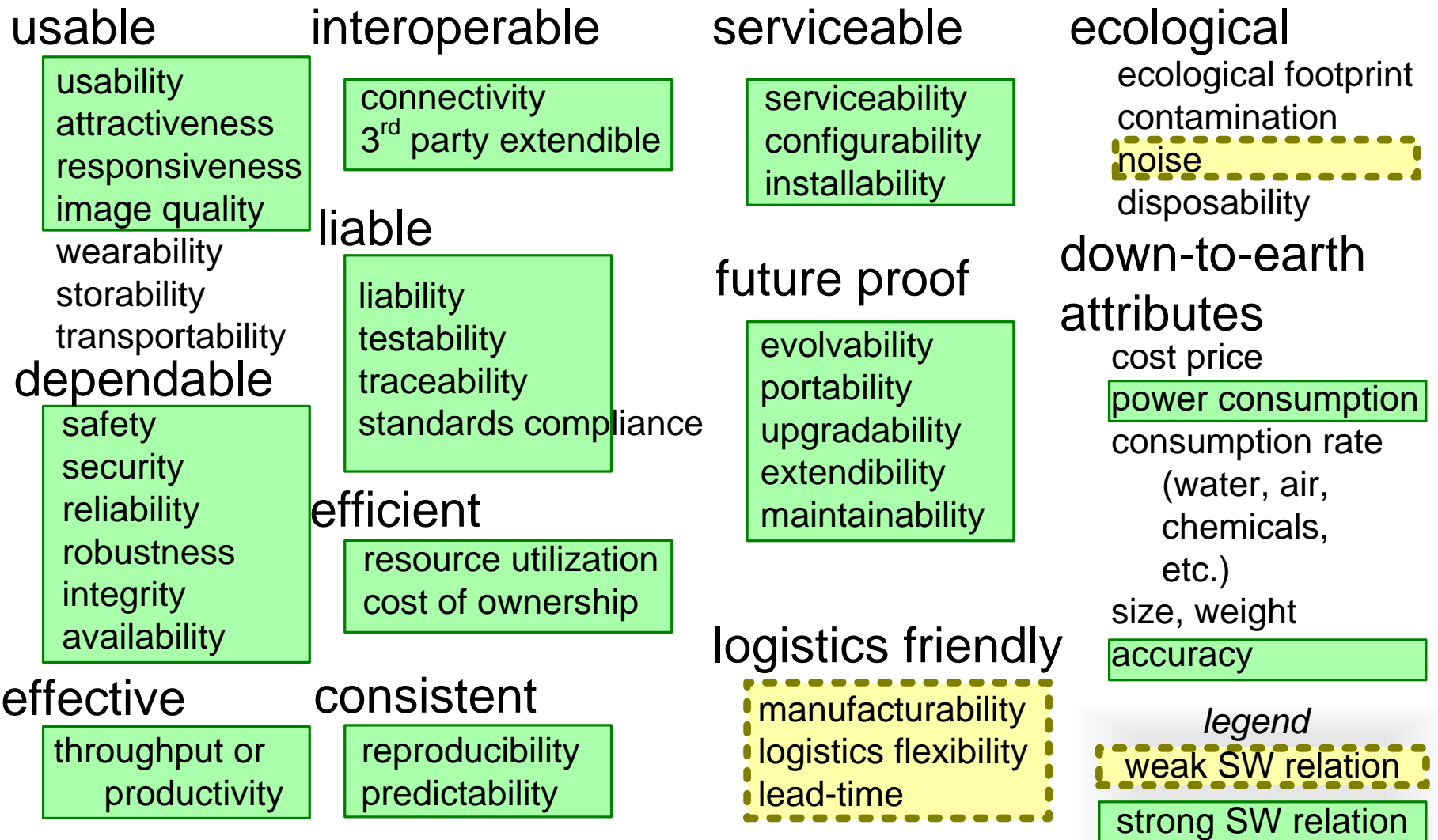
Control Hierarchy along Technology axis



Characterization of disciplines



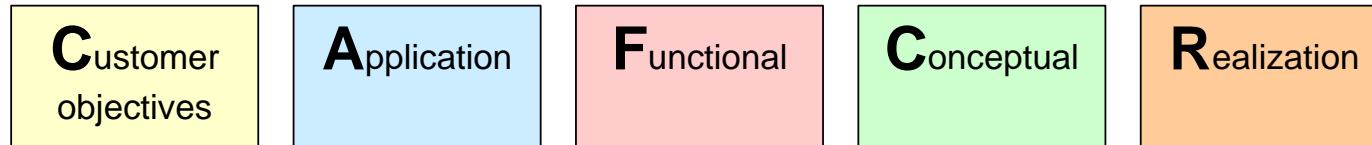
Quality Attributes annotated with SW relation



Design Aspects related to SW



SW Mechanisms



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

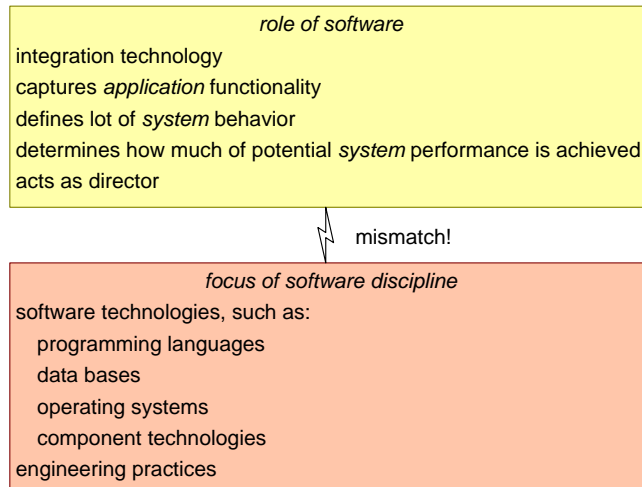
Exercise Role of Software in a complex product

Describe the SW in a complex product, from different viewpoints for instance:

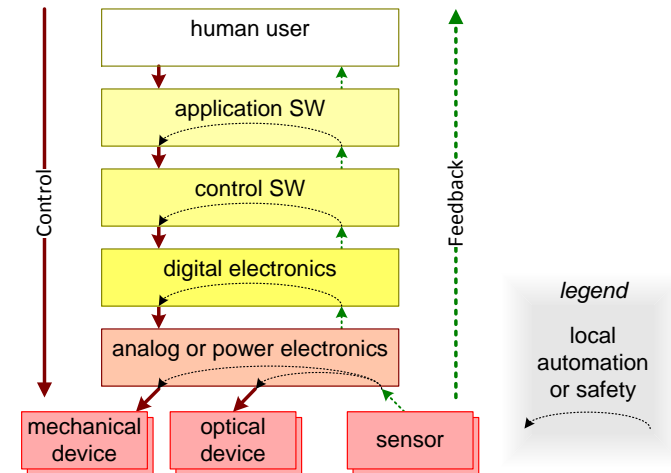
- Give an indication of the size/complexity
- Outline the SW architecture
- Identify the top 3 critical characteristics
- Identify potential improvements
- Process
- Development environment

Software

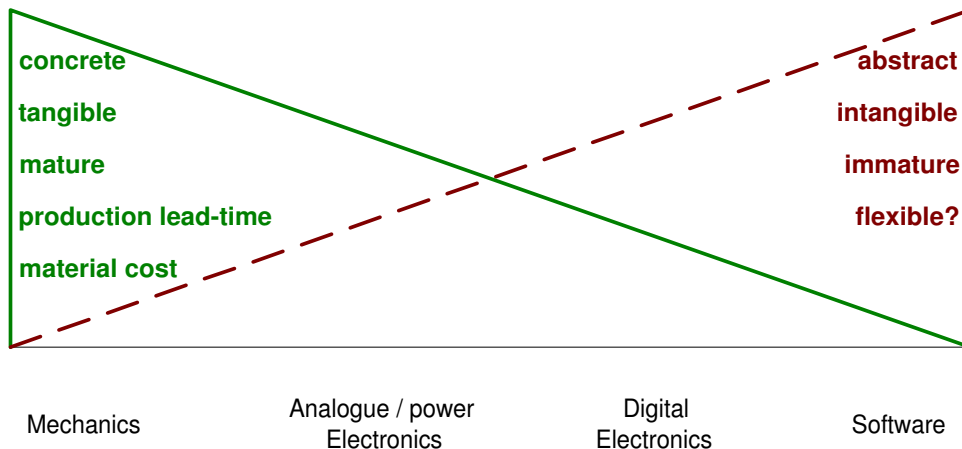
Role of Software



Control Hierarchy



Discipline Characteristics



intentionally left blank