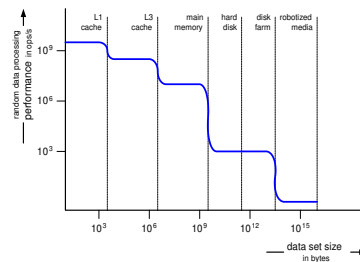


# Modeling and Analysis Fundamentals of Technology

-



Gerrit Muller

Embedded Systems Institute

Den Dolech 2 (Laplace Building 0.10) P.O. Box 513, 5600 MB Eindhoven The Netherlands

gerrit.muller@embeddedsystems.nl

## Abstract

This presentation shows fundamental elements for models that are ICT-technology related. Basic hardware functions are discussed: storage, communication and computing with fundamental characteristics, such as throughput, latency, and capacity. A system is build by layers of software on top of hardware. The problem statement is how to reason about system properties, when the system consists of many layers of hardware and software.

### 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.

All Gaudí documents are available at:  
<http://www.gaudisite.nl/>

version: 0.5

status: preliminary draft

July 20, 2011

# 1 Introduction

Figure 1 provides an overview of the content. In this article we discuss generic know how of computing technology. We will start with a commonly used decomposition and layering. We provide *figures of merit* for several generic computing functions, such as storage and communication. Finally we discuss caching as example of a technology that is related to storage figures of merit. We will apply the caching in a web shop example, and discuss design considerations.

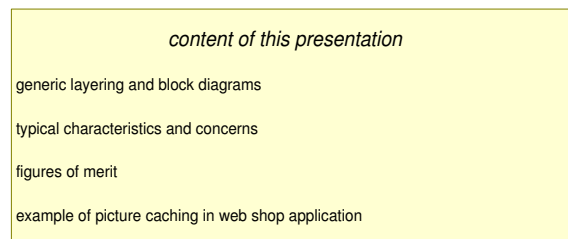


Figure 1: Overview Content *Fundamentals of Technology*

When we model technology oriented design questions we often need feasibility answers that are assessed at the level of non functional system requirements. Figure 2 shows a set of potential technology questions and the required answers at system level.

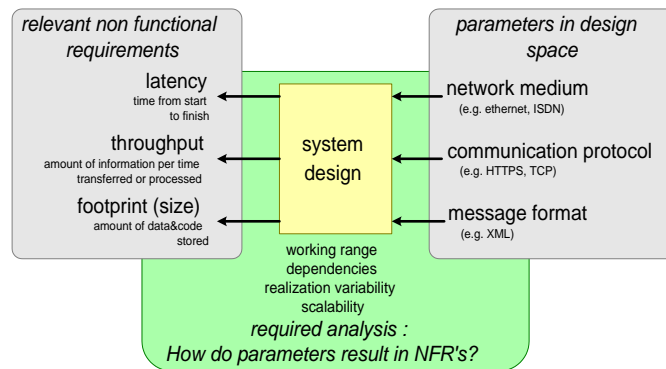


Figure 2: What do We Need to Analyze?

From design point of view we need, for example, information about the working range, dependencies, variability of the actual realization, or scalability.

## 2 Computing Technology Figures of Merit

In information and communication systems we can distinguish the following generic technology functions:

**storage** ranging from short term volatile storage to long term persistent storage. Storage technologies range from solid state static memories to optical disks or tapes.

**communication** between components, subsystems and systems. Technologies range from local interconnects and busses to distributed networks.

**processing** of data, ranging from simple control, to presentation to compute intensive operations such as 3D rendering or data mining. Technologies range from general purpose CPUs to dedicated I/O or graphics processors.

**presentation** to human beings, the final interaction point with the human users. Technologies range from small mobile display devices to large “cockpit” like control rooms with many flat panel displays.

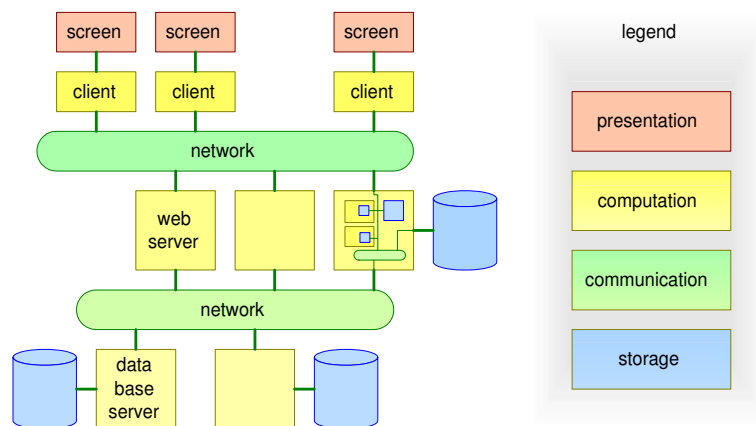


Figure 3: Typical Block Diagram and Typical Resources

Figure 3 shows these four generic technologies in the typical layering of a *Service Oriented Architecture* (SOA). In such an architecture the repositories, the bottom-tier of this figure, are decoupled from the business logic that is being handled in the middle layer, called *web server*. The client tier is the access and interaction layer, which can be highly distributed and heterogeneous.

The four generic technologies are recursively present: within a web-server, for example, communication, storage and processing are present. If we would zoom in further on the CPU itself, then we would again see the same technologies.

		latency	capacity
processor cache	<i>L1 cache</i>	sub ns	n kB
	<i>L2 cache</i>		
	<i>L3 cache</i>	ns	n MB
fast volatile	<i>main memory</i>	tens ns	n GB
persistent	<i>disks</i>		n*100 GB
	<i>disk arrays</i>	ms	
	<i>disk farms</i>		n*10 TB
archival	<i>robotized optical media tape</i>	>s	n PB

Figure 4: Hierarchy of Storage Technology *Figures of Merit*

For every generic technology we can provide *figures of merit* for several characteristics. Figure 4 shows a table with different storage technologies. The table provides typical data for latency and storage capacity. Very fast storage technologies tend to have a small capacity. For example, L1 caches, static memory as part of the CPU chip, run typically at processor speeds of several GHz, but their capacity is limited to several kilobytes. The much higher capacity main memory, solid state dynamic RAM, is much slower, but provides Gigabytes of memory. Non solid state memories use block access: data is transferred in chunks of many kilobytes. The consequence is that the access time for a single byte of information gets much longer, milliseconds for hard disks. When mechanical constructions are needed to transport physical media, such as robot arms for optical media, then the access time gets dominated by the physical transport times.

Figure 5 shows the same storage figures of merit in a 2-dimensional graph. The horizontal axis shows the capacity or the maximum data set size that we can store. The vertical axis shows the latency if we access a single byte of information in the data set in a random order. Note that both axes are shown as a logarithmic scale, both axes cover a dynamic range of many orders of magnitude! The resulting graph shows a rather non-linear behavior with step-like transitions. We can access data very fast up to several kilobytes; the access time increases significantly when we exceed the L1 cache capacity. This effect repeats itself for every technology transition.

The communication figures of merit are shown in the same way in Figure 6. In this table we show *latency*, *frequency* and *distance* as critical characteristics. The latency and the distance have a similar relationship as latency and capacity

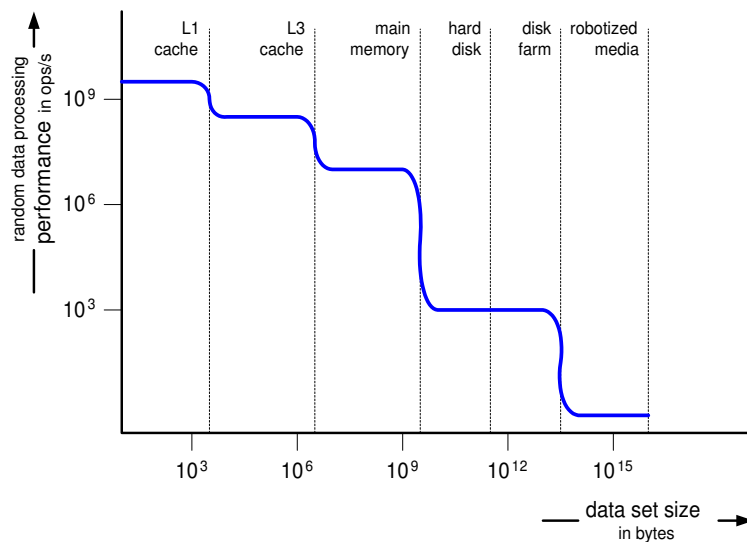


Figure 5: Performance as Function of Data Set Size

for storage: longer distance capabilities result in longer latencies. The frequency behavior, which relates directly to the transfer capacity, is different. On chip very high frequencies can be realized. Off chip and on the printed circuit board these high frequencies are much more difficult and costly. When we go to the long-distance networks optical technologies are being used, with very high frequencies.

### 3 Caching in Web Shop Example

The speed differences in storage and communication often result in the use of a cache design pattern. The cache is a local fast storage, where frequently used data is stored to prevent repeated slow accesses to slow storage media. Figure 7 shows that this caching pattern is applied at many levels within a system, for example:

**network layer cache** to avoid network latencies for distributed data. Many communication protocol stacks, such as http, have local caches.

**file cache** as part of the operating system. The file cache caches the stored data itself as well as directory information in main memory to speed up many file operations.

**application cache** application programs have dedicated caches based on application know how.

**L1, L2, L3 memory caches** A multi-level cache to bridge the speed gap between on-chip speed and off chip dynamic memory.

		latency	frequency	distance
on chip	connection	sub ns	n GHz	n mm
	network	n ns	n GHz	n mm
PCB level		tens ns	n 100MHz	n cm
Serial I/O		n ms	n 100MHz	n m
network	LAN	n ms	100MHz	n km
	WAN	n 10ms	n GHz	global

Figure 6: Communication Technology Figures of Merit

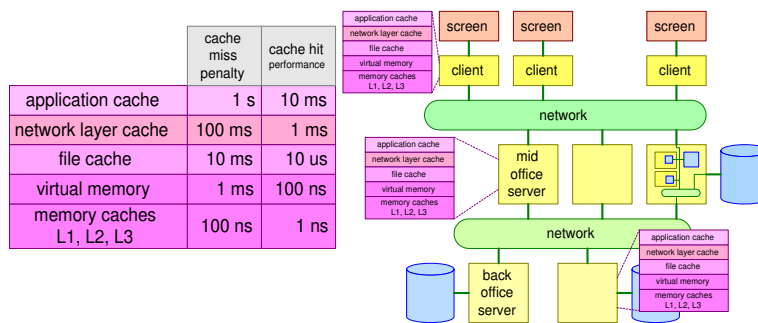


Figure 7: Multiple Layers of Caching

**virtual memory** where the physical main memory is cache for the much slower virtual memory that resides mostly on the hard disk.

Note that in the 3-tier SLA approach these caches are present in most of the tiers.

In Figure 8 we analyze the introduction of caches somewhat more. At the left hand side we show that *long latencies of storage and communication*, *communication overhead*, and *resource intensive processing* are the main reasons to introduce caching. In the background the project needs for performance and cost are seen as driving factors. Potential performance problems could also be solved by over-dimensioning, however this might conflict with the cost constraints on the project.

The design translates these performance reasons into a number of design choices:

**frequently used subset** enable the implementation to store this subset in the low capacity, but faster type of memory.

**fast storage** relates immediately to low latency of the storage itself

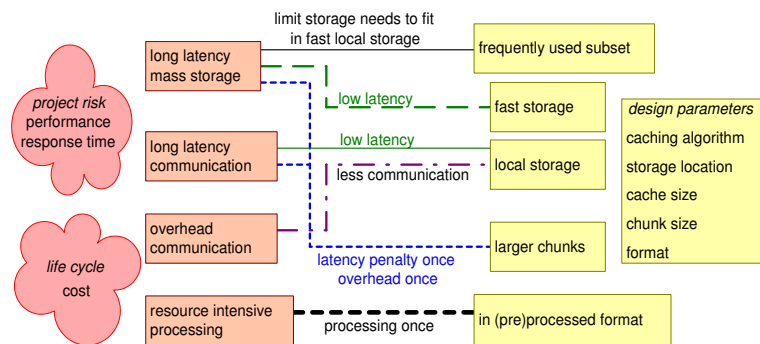


Figure 8: Why Caching?

**local storage** gives low latency for the communication with the storage (sub)system

**larger chunks** reduces the number of times that storage or communication latency occurs and reduces the overhead.

**cache in (pre)processed format** to reduce processing latency and overhead

These design choices again translate in a number of design parameters:

- caching algorithm
- storage location
- cache size
- chunk size
- format

As an example of caching we look at a web shop, as shown in Figure 9. Customers at client level should be able to browse the product catalogue, to order products, to pay, and to track the progress of the order. Other stakeholders at client level have logistics functions, financial functions, and can do product and customer management. The web server layer provides the logic for the exhibition of products, the sales and order intake, the order handling, the stock handling, and the financial bookkeeping. Also at the web server layer is the logic for customer relation management, the update of the product catalogue, the advertisements, and the after sales support. The data base layer has repositories for product descriptions, logistics and resource planning, customer relations, and financial information.

We will zoom in on the product browsing by the customers. During this browsing customers can see pictures of the products in the catalogue. The originals of these pictures reside in the product catalogue repository in the data base layer.

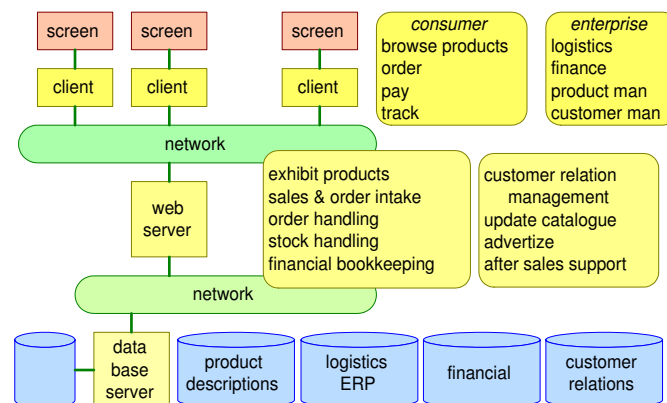


Figure 9: Example Web Shop

The web server determines when and how to show products for customers. The actual pictures are shown to many customers, who are distributed widely over the country.

The customers expect a fast response when browsing. Slow response may result in loss of customer attention and hence may cause a reduced sales. A picture cache at the web server level decreases the load at web server level, and at the same time improves the response time for customer browsing. It also reduces the server load of the data base.

So far, the caching appears to be a no-brainer: improved response, reduces server loads, what more do we want? However, Figure 11 shows the potential risks of caching, caused mostly by increased complexity and decreased transparency. These risks are:

- The robustness for application changes may decrease, because the assumptions are not true anymore.
- The design becomes specific for this technology, impacting the ability to benefit from technology improvements.
- The robustness for changing context (e.g. scalability) is reduced
- The design is not robust for concurrent applications
- Failure modes in exceptional user space may occur

All of these technical risks translate in project risks in terms of cost, effort and performance.

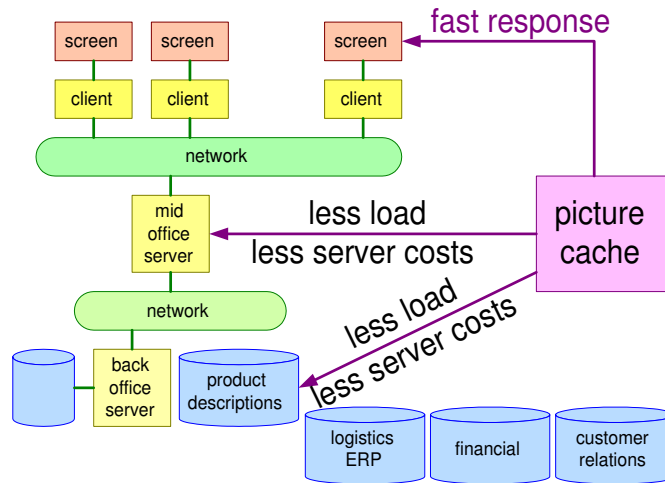


Figure 10: Impact of Picture Cache

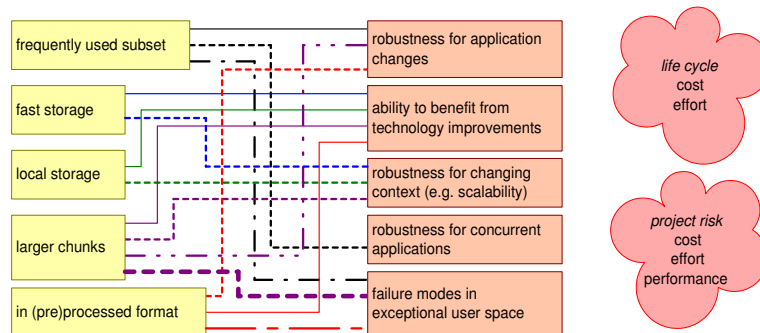


Figure 11: Risks of Caching

## 4 Summary

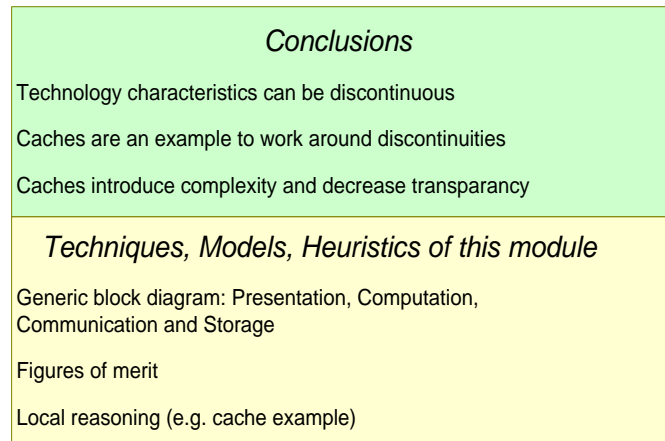


Figure 12: Summary

Figure 12 shows a summary of this paper. We showed a generic block diagram with *Presentation*, *Computation*, *Communication* and *Storage* as generic computing technologies. Technology characteristics of these generic technologies have discontinuous characteristics. At the transition from one type of technology to another type of technology a steep transition of characteristics takes place. We have provided *figures of merit* for several technologies. Caches are an example to work around these discontinuities. However, caches introduce complexity and decrease the transparency of the design. We have applied local reasoning graphs to discuss the reasons of introduction of caches and the related design parameters. later we applied the same type of graph to discuss potential risks caused by the increased complexity and decreased transparency.

## References

- [1] Gerrit Muller. The system architecture homepage. <http://www.gaudisite.nl/index.html>, 1999.

## History

**Version: 0.5, date: 23 February 2007 changed by: Gerrit Muller**

- updated summary

- Version: 0.4, date: 22 February 2007 changed by: Gerrit Muller**
- added text
- Version: 0.3, date: 2 January 2007 changed by: Gerrit Muller**
- added sheet about cache levels
  - added sheet why caching
  - added web shop example
  - removed layering diagram
- Version: 0.2, date: 17 November 2006 changed by: Gerrit Muller**
- completely reworked characteristics slide
  - many small improvements
  - changed status to preliminary draft
- Version: 0.1, date: 10 November 2006 changed by: Gerrit Muller**
- added introductory slide
- Version: 0, date: 6 November 2006 changed by: Gerrit Muller**
- Created, no changelog yet