Abstract
The Modeling and Analysis course is part of a broader Systems Architecting Curriculum. It addresses the skills and methods needed to model and analyse systems in their context. The purpose of modeling is to gain insight in the domain and potential solutions to facilitate the architecting process.
Module Modeling and Analysis course info

by Gerrit Muller     Buskerud University College

  e-mail: gaudisite@gmail.com

  www.gaudisite.nl

Abstract
This module provides the information about the “Modeling and Analysis” course.

The complete course MA 611™ is owned by TNO-ESI. To teach this course a license from Buskerud University College is required. This material is preliminary course material. The final material and course information can be found at: [www.esi.nl/cursus](http://www.esi.nl/cursus).
Modeling and Analysis Overview Content

**goal of this module**
Provide overview and context for complete course.
Understand and experience the connection between problem and solution.

**content of this module**
Positioning of Modeling and Analysis (M&A)
Why, what and how of M&A
Program of the complete course
Overview of M&A approach

**exercise**
Quick scan of one case
Electronic Patient Record, Video on Demand, or Health Care Archive
Abstract
The course Modeling and Analysis is described. The program consists of 10 modules. The course format, iterating theory, illustration and interaction is explained. The course heavily emphasizes the practical application of the method. This presentation shows the overview of the modeling and analysis approach and the methods and techniques that will be elaborated in the rest of the course.
Positioning Modeling and Analysis in Architecting

architecture description:
- articulated
- structured
problem and solution
know-how

vague notion of the problem
vague notion of potential solutions

architecting method:
- framework
- submethods
- integration methods

modeling and analysis

Report Spec Design
Modeling and Analysis supports:

- understanding
- exploration
- optimization
- verification

Type of model depends on project phase
Models have a goal
Goals evolve and models evolve
Techniques are used to reach this goal
Purpose of Modeling

- **facts** from research
- **measurements**
- **assumptions**

**Modeling**

**Uncertainties**
- unknowns
- errors

**Accuracy**
- working range
- credibility

**Project results**
- specification
- verification
- decisions

**Risk**
- customer satisfaction
- time, cost, effort
- profit margin
What to Model?

- **Business**: profit, etc.
- **Operational costs**
- **Stakeholder benefits**
- **Workload**
- **Risks**

- **Key performance**: throughput, response reliability, availability, scalability

- **(Emerging?) properties**: resource utilization, load, latency, throughput, quality, accuracy

- **Usage context**: enterprise & users
- **System**: requirements, black box view
- **Design**: realization, technology

- **Life cycle context**: creation, life cycle business

- **And their mutual relations**
## Program of Modeling and Analysis Course

<table>
<thead>
<tr>
<th>Day</th>
<th>Module</th>
</tr>
</thead>
</table>
| 1    | 1. Overall Approach  
intro, overall approach, exercise overall approach |
| 2    | 2. Input Facts, Data, Uncertainties  
quantification, measurements, modeling, validation, technology background, lifecycle and business input sources |
| 3    | 3. System Modeling  
purpose, approaches, patterns, modularity, parametrization, means, exploration, visualization, micro-benchmarking, characterization, performance as example |
| 4    | 4. Application, Life-Cycle Modeling  
reiteration of modeling approach (see module 3), applied on customer application and business, and life cycle |
| 5    | 5. Integration and Reasoning  
relating key driver models to design models, model based threads of reasoning, FMEA-like approach, modeling in project life-cycle |
| 6    | 6. Analysis, Using Models  
sensitivity, robustness, worst case, working range, scalability, exceptions, changes |
Overview of Approach

- Collect input data
- Model and analyse relevant issues for different stakeholders & concerns
- Integration and reasoning

**Usage Context**
- Enterprise & users
- Requirements (black box view)
- Creation, life cycle, business
- Life cycle context

**System**
- Design, realization, technology

**Facts**
- From research
- Measurements
- Assumptions

**Modeling and Analysis**

- Throughput
- Processing
- Library
- Diagnostic
- Quality
- Image quality
- IQ spec
- Pixel depth
- CPU budget
- Typical case
- Common
- Console
- Memory limit
- BoM
- Purchase price
- CoO
- Render engine
- Moore's law
- Profit margin
- Standard workstation
- Memory budget
- Moore's law
- Console
- Normalized limit
Iteration over viewpoints

usage context
- enterprise & users
- requirements
- black box view

system
- design
- realization
- technology

context
- understanding

intention
- driven

objective
- based

opportunities
- constraint
- awareness

know how
- based

life cycle business
- creation

life cycle context
Abstract
The background ideas of the Modeling and Analysis course are collected in a number of diagrams. These diagrams are provided solely as background and probably should not be shown during the course itself.

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.
How to Model?

- **back-of-the-envelope**
- **formula based**
- **executable**

**small, simple, goal-driven models**
What and Why to Model

- How well is the customer served?
- How credible becomes the solution?
- How much are time and effort reduced?
- How much is the risk reduced?
- How much is the solution improved?

Purpose and type of model depend on project life cycle
Type of model and views depend on purpose

Customer
- Key driver: e.g. productivity
- Risk

Business
- Key driver: e.g. SLA cost/price
- Risk

Business as usual
- Obvious
- Historic data
- Competitive data

Decision factors:
- Accuracy of model
- Credibility of results
- Level of abstraction
- Working range
- Calibration of model
- Robustness of model
- Time to first results and feedback
- Effort
- Evolvability (adaptation to new questions)

Modeling
- Feasibility
- Communication
- Risk mitigation
- Exploration
- Validation

How much effort is needed to create model(s)?
How much effort is needed to use and maintain model(s)?
How much time is needed to obtain useful result?
Unknows, Uncertainties, ...

**usage context**
- enterprise & users
- requirements
- black box view

**system**
- design
- realization
- technology

**unexpected use**
- assumptions
- uncertainties
- unknowns
- dynamics
- interference

**hidden properties**
- assumptions
- unknowns
- uncertainties
- dynamics
- interference
Model versus Reality

- complexity
- full of surprises

- simplifications
- assumptions
- implementation
Starting Points of the Course

- practical, immediately applicable in day-to-day work
- (inter)active: daily hands-on exercises on case(s)
- target: understanding, insight; way-of-working
- method, tool, language and domain agnostic
1. Why do we model? - what are indicators that modeling and analysis beyond "business as usual" architecture is needed. What questions trigger M & A.

2. What do we model? - what kinds of views do we need to consider (4+1, IBM GS Method, Zachman, CAFCR)

3. When do we model? - what models are needed at various points in the project lifecycle.

4. What is the appropriate type of model? - formula, visualization, executable, simulation

5. What is the required accuracy of the model? - when do we achieve the desired risk mitigation

6. What is the appropriate level of abstraction? - how much details have to be taken into account, versus how much effort can we afford

7. How to calibrate models? - models are based on facts and assumptions. The model outcome depends strongly on these input data. Note again the tension between effort to make and calibrate versus the value in terms of risk mitigation.

8. How to use models?
Recommendations as Red Thread

**principles**
- use feedback
- work incremental
- work evolutionary
- be explicit
- make issues tangible

**objectives**
- support communication
- facilitate reasoning
- support decision making
- create understanding
- maintain overview

**recommendations**
- Time-box
- Iterate
- Quantify early
- Measure and validate
- Multiple levels of abstraction
- (Simple) mathematical models
- Analysis of accuracy and credibility
- Multi-view
- System and its context
- Visualize

Translate into:
- help to achieve
- translate into

Modeling and Analysis: Background of the Course
©2006, Gerrit Muller
Cases for exercise

Electronic Patient Record:
+ relevant health care related information available at the right place for the right person

Long Term Health Care Archive:
+ extreme robust, persistent, high availability archive for large chain of hospitals

Video on Demand Backoffice:
+ large scale content database with fast response download capability including billing, DRM et cetera
Modeling and Analysis Exercise

• make a quick scan over the following views:
  0. what is this exercise about?
  1. context: stakeholders, concerns, application
  2. system design and realization
  3. requirements
  4. operational context
  5. qualities
• use time boxes of 15 minutes per view
• show the most dominant decomposition(s) of that view, as diagram or as a list; quantify whenever possible
Reflection on Exercise

+ collectively we know quite a lot
+ broad overview in short amount of time

~ some "hot" issues appear to be less relevant

- #questions >> #answers

baseline for next refining steps
Conclusions

Modeling and Analysis must provide more *in-depth* answers for questions that are *breadth* relevant.

Modeling and Analysis is a means that supports *requirements* management, *architecting* and *project* management.

Modeling and Analysis ranges from *business* aspects to *technical* decisions.

Good models are *small*, *simple* and *goal-driven*.

Techniques, Models, Heuristics of this module

Context viewpoints

Fast iteration based on time-boxing
Abstract
This module addresses Modeling and Analysis: Inputs and Uncertainties. The input for models comes from different sources: facts obtained from market and technology research, data from measurements, and assumptions. All these sources have uncertainties and may hide unknowns, or may even be wrong. We zoom in on commonly used technology.
**goal of this module**

Provide foundation and figures of merit for technology modeling

Provide insight in the inputs of models

Provide measurement fundamentals

**content of this module**

problem statement

generic layering and block diagrams

measuring HW and SW

**exercise**

measurement of loop and file open performance

participants may chose their own programming environment or Python
Where are we in the Course?

facts from research
measurements
assumptions
uncertainties
unknowns
errors

generic
technologies

usage context

enterprise & users
requirements
black box view

design
realization
technology

creation
life cycle business

life cycle context

system

modeling

results

project

specification
verification
decisions

risk
customer satisfaction
time, cost, effort
profit margin

accuracy
working range
credibility
Abstract
What is System Performance? Why should a software engineer have knowledge of the other parts of the system, such as the Hardware, the Operating System and the Middleware? The applications that he/she writes are self-contained, so how can other parts have any influence? This introduction sketches the problem and shows that at least a high level understanding of the system is very useful in order to get optimal performance.
Content of Problem Introduction

content of this presentation

Example of problem

Problem statements
Image Retrieval Performance

Sample application code:

for x = 1 to 3 {
    for y = 1 to 3 {
        retrieve_image(x,y)
    }
}

Alternative application code:

event 3*3 -> show screen 3*3

<screen 3*3>
    <row 1>
        <col 1><image 1,1></col 1>
        <col 2><image 1,2></col 2>
        <col 3><image 1,3></col 3>
    </row 1>
    <row 2>
        <col 1><image 1,1></col 1>
        <col 2><image 1,2></col 2>
        <col 3><image 1,3></col 3>
    </row 2>
    <row 3>
        <col 1><image 1,1></col 1>
        <col 2><image 1,2></col 2>
        <col 3><image 1,3></col 3>
    </row 3>
</screen 3*3>

Application need:
at event 3*3 show 3*3 images instantaneous

or

Design

Design

or

Design

Design

Introduction to System Performance Design
©2006, Embedded Systems Institute
©2006, Gerrit Muller

version: 0.5
July 31, 2014
PINTROsampleCode
Straight Forward Read and Display

What If....

Sample application code:

```java
for x = 1 to 3 {
    for y = 1 to 3 {
        retrieve_image(x,y)
    }
}
```

UI process

store

screen
More Process Communication

What If....

Sample application code:

```java
for x = 1 to 3 {
    for y = 1 to 3 {
        retrieve_image(x,y)
    }
}
```

UI process

update

screen server

retrieve

9 *

database

screen
Meta Information Realization Overhead

What If....

Sample application code:

```java
for x = 1 to 3 {
    for y = 1 to 3 {
        retrieve_image(x,y)
    }
}
```

Attribute = 1 COM object
100 attributes / image
9 images = 900 COM objects
1 COM object = 80µs
9 images = 72 ms
I/O overhead

What If....

Sample application code:

```cpp
for x = 1 to 3 {
    for y = 1 to 3 {
        retrieve_image(x,y)
    }
}
```

- I/O on line basis (512^2 image)

\[ 9 \times 512 \times t_{I/O} \]

\[ t_{I/O} \approx 1 \text{ms} \]
Non Functional Requirements Require System View

Sample application code:

for x = 1 to 3 {
    for y = 1 to 3 {
        retrieve_image(x,y)
    }
}

can be:
- fast, but very local
- slow, but very generic
- slow, but very robust
- fast and robust
...

The emerging properties (behavior, performance) cannot be seen from the code itself!

Underlying platform and neighbouring functions determine emerging properties mostly.
Function in System Context

- Performance and behavior of a function depend on realizations of used layers, functions in the same context, and the usage context.

Usage Context

- Hardware (HW)
- Operating System (OS)
- Middleware (MW)

Functions & Services

- F & S
- F & S
- F & S
- F & S
- F & S
- F & S

Middleware

- OS
- HW

Operating Systems

- OS

Hardware

- HW
Challenge

Performance = Function (F&S, other F&S, MW, OS, HW)
MW, OS, HW >> 100 Manyear : very complex

Challenge: How to understand MW, OS, HW with only a few parameters
Summary of Introduction to Problem

Resulting System Characteristics cannot be deduced from local code.

Underlying platform, neighboring applications and user context:

- have a big impact on system characteristics
- are big and complex

Models require decomposition, relations and representations to analyse.
Why do we model?

- what are indicators that modeling and analysis beyond "business as usual" architecture is needed.

- What questions trigger Modeling and Analysis.

The answer to the question from business side is not evident

The answer is business critical (e.g. poor performance -> unusable service). We did not discuss business value for this case.

Past experience shows that design choices have big impact on the outcome, in other words this part of the design is critical
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.

July 31, 2014
status: preliminary
draft
version: 0.5
content of this presentation

generic layering and block diagrams

typical characteristics and concerns

figures of merit

example of picture caching in web shop application
What do We Need to Analyze?

required analysis:
How do parameters result in NFR's?

relevant non functional requirements

- latency
  time from start to finish

- throughput
  amount of information per time transferred or processed

- footprint (size)
  amount of data & code stored

parameters in design space

- network medium
  (e.g. ethernet, ISDN)

- communication protocol
  (e.g. HTTPS, TCP)

- message format
  (e.g. XML)

working range
dependencies
realization variability
scalability

Modeling and Analysis Fundamentals of Technology
©2006, Embedded Systems Institute
version: 0.5
July 31, 2014
MAFTcharacteristics
Typical Block Diagram and Typical Resources

**Legend**
- Presentation
- Computation
- Communication
- Storage

**Diagram**
- Screen
- Client
- Network
- Web server
- Data base server

**Typical Resources**
- Data base server
- Web server
- Client

**Fundamentals of Technology**
©2006, Embedded Systems Institute
©2006, Gerrit Muller

`version: 0.5
July 31, 2014`
## Hierarchy of Storage Technology

### Figures of Merit

<table>
<thead>
<tr>
<th></th>
<th>Latency</th>
<th>Capacity</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Processor Cache</strong></td>
<td>sub ns</td>
<td>n kB</td>
</tr>
<tr>
<td><em>L1 cache</em></td>
<td></td>
<td></td>
</tr>
<tr>
<td><em>L2 cache</em></td>
<td></td>
<td></td>
</tr>
<tr>
<td><em>L3 cache</em></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Main Memory</strong></td>
<td>tens ns</td>
<td>n GB</td>
</tr>
<tr>
<td><em>Disks</em></td>
<td></td>
<td>n*100 GB</td>
</tr>
<tr>
<td><em>Disk Arrays</em></td>
<td></td>
<td></td>
</tr>
<tr>
<td><em>Disk Farms</em></td>
<td></td>
<td>n*10 TB</td>
</tr>
<tr>
<td><strong>Archival</strong></td>
<td>&gt;s</td>
<td>n PB</td>
</tr>
<tr>
<td><em>Robotized Optical Media</em></td>
<td></td>
<td></td>
</tr>
<tr>
<td><em>Tape</em></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Performance as Function of Data Set Size

![Graph showing performance as a function of data set size. The x-axis represents data set size in bytes, ranging from $10^3$ to $10^{15}$, and the y-axis represents random data processing performance in ops/s, ranging from $10^3$ to $10^9$. The graph includes markers for L1 cache, L3 cache, main memory, hard disk, disk farm, and robotized media.]
## Communication Technology: Figures of Merit

<table>
<thead>
<tr>
<th></th>
<th>Latency</th>
<th>Frequency</th>
<th>Distance</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>on chip</strong></td>
<td><em>connection</em></td>
<td>sub ns</td>
<td>n GHz</td>
</tr>
<tr>
<td></td>
<td><em>network</em></td>
<td>n ns</td>
<td>n GHz</td>
</tr>
<tr>
<td><strong>PCB level</strong></td>
<td>tens ns</td>
<td>n 100MHz</td>
<td>n cm</td>
</tr>
<tr>
<td><strong>Serial I/O</strong></td>
<td>n ms</td>
<td>n 100MHz</td>
<td>n m</td>
</tr>
<tr>
<td><strong>network</strong></td>
<td><em>LAN</em></td>
<td>n ms</td>
<td>100MHz</td>
</tr>
<tr>
<td></td>
<td><em>WAN</em></td>
<td>n 10ms</td>
<td>n GHz</td>
</tr>
</tbody>
</table>

---

**Modeling and Analysis Fundamentals of Technology**

©2006, Embedded Systems Institute

©2006, Gerrit Muller

version: 0.5  
July 31, 2014

MAFTcommunication
### Multiple Layers of Caching

<table>
<thead>
<tr>
<th>Cache Type</th>
<th>Cache Miss Penalty</th>
<th>Cache Hit Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application Cache</td>
<td>1 s</td>
<td>10 ms</td>
</tr>
<tr>
<td>Network Layer Cache</td>
<td>100 ms</td>
<td>1 ms</td>
</tr>
<tr>
<td>File Cache</td>
<td>10 ms</td>
<td>10 ns</td>
</tr>
<tr>
<td>Virtual Memory</td>
<td>1 ms</td>
<td>100 ns</td>
</tr>
<tr>
<td>Memory Caches L1, L2, L3</td>
<td>100 ns</td>
<td>1 ns</td>
</tr>
</tbody>
</table>

---

![Diagram of caching layers](image-url)

The diagram illustrates the various levels of caching, from network layer cache to file cache, application cache, and finally to the client. Each level has its own cache miss penalty and cache hit performance, indicating the efficiency of data retrieval from these caches. The network layer cache has a high cache miss penalty and a low cache hit performance, while the memory caches (L1, L2, L3) have a low cache miss penalty and a high cache hit performance, making them efficient for data retrieval.
Why Caching?

- **Project Risk**
  - Performance
  - Response time

- **Life Cycle**
  - Cost

- **Long Latency**
  - Mass storage
  - Communication
  - Overhead

- **Resource Intensive Processing**

- **Limit Storage Needs to Fit in Fast Local Storage**

- **Frequently Used Subset**

- **Low Latency**

- **Fast Storage**

- **Local Storage**

- **Larger Chunks**

- **In (Pre)Processed Format**

- **Design Parameters**
  - Caching algorithm
  - Storage location
  - Cache size
  - Chunk size
  - Format

- **Overhead Once**

- **Processing Once**

- **Less Communication**

Modeling and Analysis Fundamentals of Technology
©2006, Embedded Systems Institute
version: 0.5
July 31, 2014
MAFTwhyCaching
Impact of Picture Cache

- fast response
- less load
- less server costs

- screen
- client
- mid office server
- network
- back office server
- product descriptions
- logistics ERP
- financial
- customer relations

- picture cache

Modeling and Analysis Fundamentals of Technology
©2006, Embedded Systems Institute
Risks of Caching

- frequently used subset
- robustness for application changes
- ability to benefit from technology improvements
- robustness for changing context (e.g., scalability)
- robustness for concurrent applications
- failure modes in exceptional user space
- local storage
- fast storage
- in (pre)processed format

Life cycle
- cost
- effort

Project risk
- cost
- effort
- performance

Modeling and Analysis Fundamentals of Technology
©2006, Embedded Systems Institute
©2006, Gerrit Muller
version: 0.5
July 31, 2014
MAFTrisksOfCaching
Summary

Conclusions

Technology characteristics can be discontinuous

Caches are an example to work around discontinuities

Caches introduce complexity and decrease transparency

Techniques, Models, Heuristics of this module

Generic block diagram: Presentation, Computation, Communication and Storage

Figures of merit

Local reasoning (e.g. cache example)
Abstract
This presentation addresses the fundamentals of measuring: What and how to measure, impact of context and experiment on measurement, measurement errors, validation of the result against expectations, and analysis of variation and credibility.

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.

July 31, 2014
status: preliminary
draft
version: 1.2
content

What and How to measure

Impact of experiment and context on measurement

Validation of results, a.o. by comparing with expectation

Consolidation of measurement data

Analysis of variation and analysis of credibility
Measuring Approach: What and How

**what**

1. What do we need to know?
2. Define quantity to be measured.
3. Define required accuracy
4A. Define the measurement circumstances
4B. Determine expectation
4C. Define measurement set-up
5. Determine actual accuracy
6. Start measuring
7. Perform sanity check

**how**

iterative process:

- expectation versus actual outcome
- uncertainties, measurement error
1. What do We Need? Example Context Switching

- ARM 9
- 200 MHz CPU
- 100 MHz bus
- VxWorks
- test program

What:
- Estimation of total lost CPU time due to context switching
- Context switch time of VxWorks running on ARM9

Guidance of concurrency design and task granularity
2. Define Quantity by Initial Model

What (original):
context switch time of VxWorks running on ARM9

What (more explicit):
The amount of lost CPU time, due to context switching on VxWorks running on ARM9 on a heavy loaded CPU

\[ t_{\text{context switch}} = t_{\text{scheduler}} + t_{p1, \text{loss}} \]

Legend:
- Scheduler
- Process 1
- Process 2

\[ t_{p1, \text{no switching}} \]

\[ t_{p1, \text{before}} t_{\text{scheduler}} t_{p2,\text{loss}} t_{p2} t_{\text{scheduler}} t_{p1,\text{loss}} t_{p1, \text{after}} \]

\[ p2 \text{ pre-empts } p1 \quad p1 \text{ resumes} \]

\[ = \text{lost CPU time} \]
3. Define Required Accuracy

Guidance of concurrency design and task granularity

Estimation of total lost CPU time due to context switching

- ~10%

Number of context switches depends on application

Cost of context switch depends on OS and HW

Purpose drives required accuracy
Intermezzo: How to Measure CPU Time?

Low resolution ( ~ µs - ms)
Easy access
Lot of instrumentation

High resolution ( ~ 10 ns)
requires
HW instrumentation

High resolution ( ~ 10 ns)
Cope with limitations:
- Duration (16 / 32 bit counter)
- Requires Timer Access
**4A. Define the Measurement Set-up**

*Mimick relevant real world characteristics*

**real world**
- many concurrent processes, with
  - # instructions >> I-cache
  - # data >> D-cache

**experimental set-up**
- P1
- P2
- pre-empts
- cache flush
- no other CPU activities

- \( t_{p1,\text{before}} \)
- \( t_{\text{scheduler}} \)
- \( t_{p2,\text{loss}} \)
- \( t_{p2} \)
- \( t_{\text{scheduler}} \)
- \( t_{p1,\text{loss}} \)
- \( t_{p1,\text{after}} \)

- \( p2 \) pre-empts \( p1 \)
- \( p1 \) resumes
  = *lost CPU time*
200 MHz

CPU

on-chip bus

Instruction cache

Data cache

cache line size: 8 32-bit words

memory bus

memory

100 MHz

chip

PCB
Key Hardware Performance Aspect

memory request → 22 cycles → memory response

---

memory access time in case of a cache miss
200 Mhz, 5 ns cycle: 190 ns

---

38 cycles

memory access time in case of a cache miss
200 Mhz, 5 ns cycle: 190 ns
Determine Expectation

simple SW model of context switch:
- save state P1
- determine next runnable task
- update scheduler administration
- load state P2
- run P2

input data HW:
- \( t_{\text{ARM instruction}} = 5 \text{ ns} \)
- \( t_{\text{memory access}} = 190 \text{ ns} \)

Estimate how many instructions and memory accesses are needed per context switch.

Calculate the estimated time needed per context switch.
**Determine Expectation Quantified**

simple SW model of context switch:
- save state P1
- determine next runnable task
- update scheduler administration
- load state P2
- run P2

input data HW:
- \( t_{\text{ARM instruction}} = 5 \text{ ns} \)
- \( t_{\text{memory access}} = 190 \text{ ns} \)

<table>
<thead>
<tr>
<th>Instructions</th>
<th>Memory Accesses</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>1</td>
</tr>
<tr>
<td>50</td>
<td>2</td>
</tr>
<tr>
<td>20</td>
<td>1</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
</tr>
<tr>
<td><strong>100</strong></td>
<td><strong>6</strong></td>
</tr>
</tbody>
</table>

\[
500 \text{ ns} + 1140 \text{ ns} = 1640 \text{ ns}
\]

round up (as margin) gives expected \( t_{\text{context switch}} = 2 \mu\text{s} \)

**Estimate** how many instructions and memory accesses are needed per context switch

**Calculate** the estimated time needed per context switch
4C. Code to Measure Context Switch

Task 1
- Time Stamp End
- Cache Flush
- Time Stamp Begin
- Context Switch

Task 2
- Time Stamp End
- Cache Flush
- Time Stamp Begin
- Context Switch
Measuring Task Switch Time

Diagram illustrating the sequence of events during task switching, including:
- Time Stamp Begin
- Time Stamp End
- Context switch
- Start Cache Flush

Legend:
- Scheduler
- Process 1
- Process 2
Understanding: Impact of Context Switch

Clock cycles Per Instruction (CPI)

Task 1
Task 2

Scheduler

Process 1
Process 2

Based on figure diagram by Ton Kostelijk

Version: 1.2
July 31, 2014
PHRTarmCacheTaskSwitch
5. Accuracy: Measurement Error

- **noise**
- **resolution**
- **offset**
- **calibration**
- **characteristics**

Measurements have stochastic variations and systematic deviations resulting in a range rather than a single value.
Accuracy 2: Be Aware of Error Propagation

\[ t_{\text{duration}} = t_{\text{end}} - t_{\text{start}} \]

\[ t_{\text{start}} = 10 \pm 2 \, \mu s \]

\[ t_{\text{end}} = 14 \pm 2 \, \mu s \]

\[ t_{\text{duration}} = 4 \pm ? \, \mu s \]

systematic errors: add linear

stochastic errors: add quadratic
Measurements have stochastic variations and systematic deviations resulting in a range rather than a single value.

The inputs of modeling, "facts", assumptions, and measurement results, also have stochastic variations and systematic deviations.

Stochastic variations and systematic deviations propagate (add, amplify or cancel) through the model resulting in an output range.
6. Actual ARM Figures

ARM9  200 MHz

<table>
<thead>
<tr>
<th>cache setting</th>
<th>$t_{\text{context switch}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>From cache</td>
<td>2 µs</td>
</tr>
<tr>
<td>After cache flush</td>
<td>10 µs</td>
</tr>
<tr>
<td>Cache disabled</td>
<td>50 µs</td>
</tr>
</tbody>
</table>
7. Expectation versus Measurement

<table>
<thead>
<tr>
<th>Instructions</th>
<th>Memory Accesses</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>1</td>
</tr>
<tr>
<td>50</td>
<td>2</td>
</tr>
<tr>
<td>20</td>
<td>1</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
</tr>
</tbody>
</table>

Input data HW:
- \( t_{\text{ARM instruction}} = 5 \text{ ns} \)
- \( t_{\text{memory access}} = 190 \text{ ns} \)

Simple SW model of context switch:
1. Save state P1
2. Determine next runnable task
3. Update scheduler administration
4. Load state P2
5. Run P2

Expected context switch time: \( t_{\text{context switch}} = 2 \mu s \)

Measured context switch time: \( t_{\text{context switch}} = 10 \mu s \)

How to explain?

Potentially missing in expectation:
- Memory accesses due to instructions:
  - \( \sim 10 \) instruction memory accesses \( \sim 2 \mu s \)
- Memory management (MMU context)
- Complex process model (parents, permissions)
- Bookkeeping, e.g., performance data
- Layering (function calls, stack handling)

However, measurement seems to make sense.
Conclusion Context Switch Overhead

\[ t_{\text{overhead}} = n_{\text{context switch}} \times t_{\text{context switch}} \]

<table>
<thead>
<tr>
<th>( n_{\text{context switch}} ) (s(^{-1}))</th>
<th>( t_{\text{context switch}} )</th>
<th>CPU load overhead</th>
<th>( t_{\text{context switch}} )</th>
<th>CPU load overhead</th>
</tr>
</thead>
<tbody>
<tr>
<td>500</td>
<td>5ms</td>
<td>0.5%</td>
<td>1ms</td>
<td>0.1%</td>
</tr>
<tr>
<td>5000</td>
<td>50ms</td>
<td>5%</td>
<td>10ms</td>
<td>1%</td>
</tr>
<tr>
<td>50000</td>
<td>500ms</td>
<td>50%</td>
<td>100ms</td>
<td>10%</td>
</tr>
</tbody>
</table>
### goal of measurement

Guidance of concurrency design and task granularity

Estimation of context switching overhead

Cost of context switch on given platform

### examples of measurement

Needed: context switch overhead ~10% accurate

Measurement instrumentation: HW pin and small SW test program

Simple models of HW and SW layers

Measurement results for context switching on ARM9
Conclusions

Measurements are an important source of factual data.

A measurement requires a well-designed experiment.

Measurement error, validation of the result determine the credibility.

Lots of consolidated data must be reduced to essential understanding.

Techniques, Models, Heuristics of this module

experimentation

error analysis

estimating expectations
This work is derived from the EXARCH course at CTT developed by *Ton Kostelijk* (Philips) and *Gerrit Muller*. The Boderc project contributed to the measurement approach. Especially the work of *Peter van den Bosch* (Océ), *Oana Florescu* (TU/e), and *Marcel Verhoef* (Chess) has been valuable.
Abstract
A simple measurement exercise is described. Purpose of this exercise is to build up experience in measuring and its many pitfalls. The programming language Python is used as platform, because of its availability and low threshold for use.

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.
Select a programming environment, where loop overhead and file open can be measured in 30 minutes.

If this environment is not available, then use Python.
Active State Python (Freeware distribution, runs directly)
http://www.activestate.com/Products/ActivePython/

Python Language Website
http://www.python.org/

Python Reference Card
http://admin.oreillynet.com/python/excerpt/PythonPocketRef/examples/python.pdf
import time

for n in (1,10,100,1000,10000,100000,1000000):
    a = 0
    tstart = time.time()
    for i in xrange(n):
        a = a+1
    tend=time.time()
    print n, tend-tstart, (tend-tstart)/n

def example_filehandling():
    f = open("c:\temp\test.txt")
    for line in f.readlines():
        print line
    f.close()

tstart = time.time()
example_filehandling()
tend=time.time()
print "file open, read & print, close: ",tend-tstart,"s"
Exercise

• Perform the following measurements
  1. loop overhead
  2. file open

• Determine for every measurement:
  What is the expected result?
  What is the measurement error?
  What is the result?
  What is the credibility of the result?
  Explain the result.
  (optional) What is the variation? Explain the variation.
Reflection on Exercise

+ measuring is easy
+ measuring provides data and understanding

~ result and expectation often don't match

- sensible measuring is more difficult
Abstract

This module addresses Modeling and Analysis Performance. What are the customer performance needs, what are the operational performance considerations? What are the performance related design choices? How to analyze feasibility, explore design options, and how to validate performance?
goal of this module

provide a stepwise approach to system modeling
provide concrete examples of system models

content of this module

web shop system model
Non Functional requirements (NFR), System Properties and Critical Technologies
zero order and first order load models
budgeting

exercise

model one NFR in relation to a critical technology choice
Where are we in the Course?

- Facts from research
- Measurements
- Assumptions
- Uncertainties
- Unknowns
- Errors

**Module Modeling and Analysis:**
- System model
Abstract
This presentation uses a web shop service as example system to construct a system model. The caching of pictures of the products in the shop is modeled to analyze performance, robustness, scalability and reliability of the system.

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.
content

What to model of the system

Stepwise approach to system modeling

Non Functional requirements (NFR), System Properties and Critical Technologies

Examples of web shop case
What to Model in System Context?

**usage context**

*NFR's:*
- performance
- reliability
- availability
- scalability
- maintainability

...  

**system**

*(emerging?) properties:*
- critical technologies
  - caching
  - load balancing
  - firewalls
  - virtual networks
  - XML for customization and configuration

...  

*(changes, inputs)*
1. determine relevant Non Functional Requirements (NFR's)

2. determine relevant system design properties

3. determine critical technologies

4. relate NFR's to properties to critical technologies

5. rank the relations in relevancy and criticality

6. model relations with a high score
Web Shop: NFR’s, Properties and Critical Technologies

NFR's:
- performance browsing
- initial cost
- running costs
- reliability/availability
- scalability order rate
- maintainability
  - effort product changes
  - effort staff changes
- security

(emerging?) properties:
- resource utilization
  - server load, capacity
  - memory load, capacity
- response latency
- redundancy
- order throughput
- product data quality
- product definition flow
- staff definition flow
- security design
  - compartmentalization
  - authentication
  - encryption

Critical technologies:
- caching
- load balancing
- pipelining
- virtual memory
- memory management
- data base transactions
- XML for customization and configuration
- firewalls
- virtual networks
- ...
4. Determine Relations

**NFR's:**
- performance browsing
- initial cost
- running costs
- reliability/availability
- scalability order rate
- maintainability
- effort product changes
- effort staff changes
- security

**Emerging properties:**
- resource utilization
- server load, capacity
- memory load, capacity
- response latency
- redundancy
- order throughput
- product data quality
- product definition flow
- staff definition flow
- security design
- compartmentalization
- authentication
- encryption

**Critical technologies:**
- caching
- load balancing
- pipelining
- virtual memory
- memory management
- data base transactions
- XML for customization
- and configuration
- firewalls
- virtual networks
- ...
5. Rank Relations

NFR’s:
- performance browsing
- initial cost
- running costs
- reliability/availability
- scalability order rate
- maintainability
- effort product changes
- effort staff changes
- security

(emerging?) properties:
- resource utilization
- server load, capacity
- memory load, capacity
- response latency
- redundancy
- order throughout

Critical technologies:
- caching
- load balancing
- pipelining
- virtual memory
- memory management

Security design:
- compartmentalization
- authentication
- encryption

Ranking will be discussed in Modeling and Analysis: Reasoning
Purpose of Picture Cache Model in Web Shop Context

- **Client**
- **Screen**
- **Network**
- **Web server**
- **Database server**
- **Product descriptions**
- **Exhibit products**
- **Browse products**
- **Response time**
- **Required server capacity**
- **Picture cache**
- **Logistics ERP**
- **Financial**
- **Customer relations**

**Modeling and Analysis: System Model**

© 2006, Embedded Systems Institute

© 2006, Gerrit Muller

version: 0.4
July 31, 2014
MASMwebShopPictureCache
zero order web server load model

Load = n_a * t_a

n_a = total requests

\( t_a = \text{cost per request} \)
First Order Load Model

\[
\text{Load} = n_{a,h} \cdot t_h + n_{a,m} \cdot t_m
\]

- \(n_{a,h}\) = accesses with cache hit
- \(n_{a,m}\) = accesses with cache miss
- \(t_h\) = cost of cache hit
- \(t_m\) = cost of cache miss

\[
n_{a,h} = n_a \cdot h
\]
\[
n_{a,m} = n_a \cdot (1-h)
\]
- \(n_a\) = total accesses
- \(h\) = hit rate

\[
\text{Load}(h) = n_a \cdot h \cdot t_h + n_a \cdot (1-h) \cdot t_m = n_a \cdot t_m - n_a \cdot h \cdot (t_m - t_h)
\]
quantified mid office server example

\[ t_h = 0.02 \text{ ms} \]
\[ t_m = 2 \text{ ms} \]
\[ n_a = 1000 \]

Load(h) = 1000 \times 2[\text{ms}] - 1000 \times h \times 1.98[\text{ms}]

Load(h) = 2000 - 1980 \times h [\text{ms}]

utilizable capacity

working range

hit rate

load in seconds
Hit Rate Considerations

Quantified mid office server example:

- \( t_h = 0.02 \text{ ms} \)
- \( t_m = 2 \text{ ms} \)
- \( n_a = 1000 \)

Load(h) = 1000 * 2[ms] - 1000 * h * 1.98[ms]
Load(h) = 2000 - 1980 * h [ms]

Hit rate is context dependent. Life cycle changes or peak loads may degrade hit rate.

Hit rate of well designed system is ample within working range (e.g. 95%)

0th order formula is valid:
Load = 0.12 * n_a [ms]

Modeling and Analysis: System Model
version: 0.4
July 31, 2014
MASMdiscussion
Response Time

- Human customer: press next
- Web server:
  - Request picture
  - Check cache
  - Retrieve picture
  - Store in cache
  - Transfer to client
- Client:
  - Transfer process
  - Display

Time in milliseconds in optimal circumstances:
- $t_0$
- $t_0 + 10$
- $t_0 + 20$
- $t_0 + 30$
- $t_0 + 40$
- $t_0 + 50$
- $t_0 + 60$
- $t_0 + 70$
What Memory Capacity is Required for Picture Transfers?

required memory capacity?
Process View of Picture Flow in Web Server

- **one copy per process**
  - mid office server
  - web server
  - picture cache server
  - back office access

- **multiple copies per process**
  - mid office server
  - web server
  - picture cache server
  - back office access

- **multiple copies per process and thread**
  - mid office server
  - web server
  - picture cache server
  - back office access
picture memory =

\[ 3 \times n \times s + 5 \times m \times s + c \times s + 3 \times k \times s \]

where

- \( n \) = number of database access threads
- \( m \) = number of picture cache threads
- \( k \) = number of web server threads
- \( s \) = picture size in bytes
- \( c \) = in-memory cache capacity in number of pictures
Web Server Memory Capacity

<table>
<thead>
<tr>
<th>use case</th>
<th>n</th>
<th>m</th>
<th>k</th>
<th>s</th>
<th>c</th>
<th>MB</th>
<th>storage type</th>
</tr>
</thead>
<tbody>
<tr>
<td>small shop</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>100</td>
<td>10</td>
<td>1.5</td>
<td>L3</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>4</td>
<td>10</td>
<td>100</td>
<td>20</td>
<td>5.3</td>
<td>main</td>
</tr>
<tr>
<td>highly concurrent</td>
<td>2</td>
<td>4</td>
<td>1000</td>
<td>100</td>
<td>100</td>
<td>296</td>
<td>main</td>
</tr>
<tr>
<td>large pictures</td>
<td>2</td>
<td>4</td>
<td>1000</td>
<td>1000</td>
<td>100</td>
<td>2,962</td>
<td>main+disk</td>
</tr>
<tr>
<td>many pictures</td>
<td>2</td>
<td>4</td>
<td>10</td>
<td>100</td>
<td>100,000</td>
<td>9,540</td>
<td>main+disk</td>
</tr>
<tr>
<td>all at once</td>
<td>2</td>
<td>4</td>
<td>1000</td>
<td>1000</td>
<td>100,000</td>
<td>98,234</td>
<td>disk</td>
</tr>
</tbody>
</table>

picture memory = \(3n + 5m + cs + 3ks\)

where
- \(n\) = # back office access threads
- \(m\) = # picture cache threads
- \(k\) = # web server threads
- \(s\) = picture size in bytes
- \(c\) = in memory cache capacity in # pictures

memory use
- product browsing only
- pictures only
- single server

What is the performance impact of memory use on other processing?
We Have only Modeled a Small Part of the System...

| function | browse/exhibit products | sales, order intake, payments
|          |                        | track, order handling
|          |                        | stock handling
|          |                        | financial bookkeeping
|          |                        | customer relation management
|          |                        | update catalogue
|          |                        | advertise
|          |                        | after sales support

| data | picture | structured (product attributes, logistics, ...)
|      |         | program code

| aspect | server memory use | network use
|        | response time     | reliability
|        | server load       | any resource, any NFR

aspect(d, f)

ignoring other dimensions such as applications, users, circumstances
static

mostly assumptions and coarse estimates

some insight in:

what are key design issues

what are relevant use case areas
Refinement After Context Modeling

usage context

enterprise & users

NFR's:
- performance
- reliability
- availability
- scalability
- maintainability
...

(system)

creation
life cycle business

life cycle context

(system)

NFR's:
- (emerging?) properties
- resource utilization
- load
- latency, throughput
- quality, accuracy
- sensitivity
- (changes, inputs)
...

critical technologies
- caching
- load balancing
- firewalls
- virtual networks
- XML for customization
- and configuration
...

version: 0.4
July 31, 2014
MASMfollowUp
Conclusions

Non Functional Requirements are the starting point for system modeling

Focus on highest ranking relations between NFR's and critical technologies

Make simple mathematical models

Evaluate quantified instantiations

Techniques, Models, Heuristics of this module

Non functional requirements

System properties

Critical technologies

Graph of relations
Abstract
This presentation addresses the fundamentals of budgeting: What is a budget, how to create and use a budget, what types of budgets are there. What is the relation with modeling and measuring.
content of this presentation

What and why of a budget

How to create a budget (decomposition, granularity, inputs)

How to use a budget
What is a Budget?

A budget is a quantified instantiation of a model.

A budget can prescribe or describe the contributions by parts of the solution to the system quality under consideration.
Why Budgets?

- to make the design explicit
- to provide a baseline to take decisions
- to specify the requirements for the detailed designs
- to have guidance during integration
- to provide a baseline for verification
- to manage the design margins explicitly
Visualization of Budget Based Design Flow

- can be more complex than additions

- model

- measurements existing system

- design estimates; simulations

- budget

- feedback

- spec

- SRS

- $t_{\text{boot}}$: 0.5s
- $t_{\text{zap}}$: 0.2s

- $t_{\text{proc}}$
- $t_{\text{over}}$
- $t_{\text{disp}}$

- tuning

- measurements new (proto) system

- micro benchmarks
- aggregated functions
- applications
- profiles
- traces

- budget version: 1.0
  July 31, 2014
  EAAbudget

- micro benchmarks
- aggregated functions
- applications

- existing system

- model

- measurements existing system

- design estimates; simulations

- $V_{\text{4aa}}$

- feedback
### Stepwise Budget Based Design Flow

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>1A</td>
<td>Measure old systems</td>
<td>Micro-benchmarks, aggregated functions, applications</td>
</tr>
<tr>
<td>1B</td>
<td>Model the performance starting with old systems</td>
<td>Flow model and analytical model</td>
</tr>
<tr>
<td>1C</td>
<td>Determine requirements for new system</td>
<td>Response time or throughput</td>
</tr>
<tr>
<td>2</td>
<td>Make a design for the new system</td>
<td>Explore design space, estimate and simulate</td>
</tr>
<tr>
<td>3</td>
<td>Make a budget for the new system:</td>
<td>Models provide the structure measurements and estimates provide initial numbers specification provides bottom line</td>
</tr>
<tr>
<td>4</td>
<td>Measure prototypes and new system</td>
<td>Micro-benchmarks, aggregated functions, applications profiles, traces</td>
</tr>
<tr>
<td>5</td>
<td>Iterate steps 1B to 4</td>
<td></td>
</tr>
</tbody>
</table>
Budgets Applied on Waferstepper Overlay

- process overlay 80 nm
  - matched machine 60 nm
    - process dependency sensor 5 nm
  - single machine 30 nm
    - matching accuracy 5 nm
    - stage overlay 12 nm
      - stage grid accuracy 5 nm
      - metrology stability 5 nm
      - alignment repro 5 nm
      - position accuracy 7 nm
    - global alignment accuracy 6 nm
      - system adjustment accuracy 2 nm
    - off axis pos. meas. accuracy 4 nm
      - off axis Sensor repro 3 nm
  - lens matching 25 nm
    - reticule 15 nm
      - blue align sensor repro 3 nm
      - interferometer stability 1 nm
  - stage Al. pos. meas. accuracy 4 nm
    - frame stability 2.5 nm
      - tracking error WS 2 nm
      - tracking error X, Y 2.5 nm
      - tracking error RS 1 nm
      - tracking error phi 75 nrad

Modeling and Analysis: Budgeting
©2006, Embedded Systems Institute
114 ©2006, Gerrit Muller

version: 1.0
July 31, 2014
ASMLOverlayBudget
### Budgets Applied on Medical Workstation Memory Use

<table>
<thead>
<tr>
<th></th>
<th>code</th>
<th>obj data</th>
<th>bulk data</th>
<th>total</th>
</tr>
</thead>
<tbody>
<tr>
<td>shared code</td>
<td>11.0</td>
<td></td>
<td></td>
<td>11.0</td>
</tr>
<tr>
<td>User Interface process</td>
<td>0.3</td>
<td>3.0</td>
<td>12.0</td>
<td>15.3</td>
</tr>
<tr>
<td>database server</td>
<td>0.3</td>
<td>3.2</td>
<td>3.0</td>
<td>6.5</td>
</tr>
<tr>
<td>print server</td>
<td>0.3</td>
<td>1.2</td>
<td>9.0</td>
<td>10.5</td>
</tr>
<tr>
<td>optical storage server</td>
<td>0.3</td>
<td>2.0</td>
<td>1.0</td>
<td>3.3</td>
</tr>
<tr>
<td>communication server</td>
<td>0.3</td>
<td>2.0</td>
<td>4.0</td>
<td>6.3</td>
</tr>
<tr>
<td>UNIX commands</td>
<td>0.3</td>
<td>0.2</td>
<td>0</td>
<td>0.5</td>
</tr>
<tr>
<td>compute server</td>
<td>0.3</td>
<td>0.5</td>
<td>6.0</td>
<td>6.8</td>
</tr>
<tr>
<td>system monitor</td>
<td>0.3</td>
<td>0.5</td>
<td>0</td>
<td>0.8</td>
</tr>
<tr>
<td><strong>application SW total</strong></td>
<td><strong>13.4</strong></td>
<td><strong>12.6</strong></td>
<td><strong>35.0</strong></td>
<td><strong>61.0</strong></td>
</tr>
<tr>
<td>UNIX Solaris 2.x</td>
<td></td>
<td></td>
<td></td>
<td>10.0</td>
</tr>
<tr>
<td>file cache</td>
<td></td>
<td></td>
<td></td>
<td>3.0</td>
</tr>
<tr>
<td><strong>total</strong></td>
<td></td>
<td></td>
<td></td>
<td><strong>74.0</strong></td>
</tr>
</tbody>
</table>
Power Budget Visualization for Document Handler

- scanner and feeder
- procedé
- paper path
- finisher
- UI and control
- paper input module
- cooling

Legend:
- physical layout
- size proportional to power

--

Modeling and Analysis: Budgeting
©2006, Embedded Systems Institute
116 ©2006, Gerrit Muller

version: 1.0
July 31, 2014
MDMpowerProportions
Alternative Power Visualization

- Power supplies
- UI and control
- Paper input module
- Finisher
- Paper path
- Cooling
- Electrical power
- Heat
- Procedure
- Paper

Modeling and Analysis: Budgeting
© 2006, Embedded Systems Institute
© 2006, Gerrit Muller

version: 1.0
July 31, 2014
MDMpowerArrows
**Evolution of Budget over Time**

<table>
<thead>
<tr>
<th>Evolution Stage</th>
<th>Start/End</th>
</tr>
</thead>
<tbody>
<tr>
<td>fact finding through details</td>
<td><strong>start</strong></td>
</tr>
<tr>
<td>aggregate to end-to-end performance</td>
<td></td>
</tr>
<tr>
<td>search for appropriate abstraction level(s)</td>
<td></td>
</tr>
<tr>
<td>from coarse guesstimate</td>
<td><strong>time</strong></td>
</tr>
<tr>
<td>to reliable prediction</td>
<td></td>
</tr>
<tr>
<td>from typical case</td>
<td></td>
</tr>
<tr>
<td>to boundaries of requirement space</td>
<td></td>
</tr>
<tr>
<td>from static understanding</td>
<td></td>
</tr>
<tr>
<td>to dynamic understanding</td>
<td></td>
</tr>
<tr>
<td>from steady state</td>
<td></td>
</tr>
<tr>
<td>to initialization, state change and shut down</td>
<td></td>
</tr>
<tr>
<td>from old system</td>
<td></td>
</tr>
<tr>
<td>to prototype</td>
<td></td>
</tr>
<tr>
<td>to actual implementation</td>
<td></td>
</tr>
<tr>
<td>only if needed</td>
<td></td>
</tr>
</tbody>
</table>

Modeling and Analysis: Budgeting
©2006, Embedded Systems Institute
118 ©2006, Gerrit Muller

version: 1.0
July 31, 2014
MABU increments
Potential Applications of Budget based design

• resource use  (CPU, memory, disk, bus, network)

• timing  (response, latency, start up, shutdown)

• productivity  (throughput, reliability)

• Image Quality parameters  (contrast, SNR, deformation, overlay, DOF)

• cost, space, time
What kind of budget is required?

<table>
<thead>
<tr>
<th>static</th>
<th>dynamic</th>
</tr>
</thead>
<tbody>
<tr>
<td>typical case</td>
<td>worst case</td>
</tr>
<tr>
<td>global</td>
<td>detailed</td>
</tr>
<tr>
<td>approximate</td>
<td>accurate</td>
</tr>
</tbody>
</table>

is the budget based on wish, empirical data, extrapolation, educated guess, or expectation?
Summary of Budgetting

A budget is a quantified instantiation of a model

A budget can prescribe or describe the contributions by parts of the solution to the system quality under consideration

A budget uses a decomposition in tens of elements

The numbers are based on historic data, user needs, first principles and measurements

Budgets are based on models and estimations

Budget visualization is critical for communication

Budgeting requires an incremental process

Many types of budgets can be made; start simple!
The Boderc project contributed to Budget Based Design. Especially the work of

**Hennie Freriks, Peter van den Bosch** (Océ),

**Heico Sandee** and **Maurice Heemels** (TU/e, ESI)

has been valuable.
Abstract
This module addresses Modeling and Analysis Fundamentals of Application.
### goal of this module

Tangible understanding of the customer enterprise and life cycle aspects

Provide useful views on customer application

Simplify and demystify customer concerns

### content of this module

Example financial computations

views on customer application:

- stakeholders and concerns
- simple cost models
- simple life cycle models

### exercise

Make context and application models
Where are we in the Course?

Module Modeling and Analysis: Application and Life Cycle Modeling
©2006, Embedded Systems Institute
125 ©2006, Gerrit Muller
version: 0.3
July 31, 2014
MMAFAPosition
Abstract

The enterprise and its application is a complex system in itself. Specification and design decisions can have a significant impact on this system. We show a number of relevant application models with the purpose to be able to reason about specification and design in relation to the impact on the enterprise.

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.
Understanding Usage and Life Cycle Context

High Level Visual Models
+ value chain
+ map of competitors, partners, suppliers
+ context diagram
+ stakeholder diagram
+ infrastructure diagram
+ aspect diagrams e.g. security, data integrity,..
+ customer key driver graph
+ life cycle key driver graph

relations beyond actual system!

commercial
financial
legal
strategic
tactical
operational
social
technical
Simplified Web Shop Value Chain

suppliers

management
products
customers
finance

virtual shop

sales

order

goods

flow

goods

payment

customers

$\downarrow$

service level

$\uparrow$

IT services

web services

IT infrastructure

version: 0.1
July 31, 2014
MAAMvalueChain
Simplistic Customer Key Driver Graph

- high sales
  - right pricing
  - up to date product offering
- fast delivery
  - fast goods processing
  - goods flow tuning
- low capital use
  - minimal stocks
- predictable (low) cost level
  - strategic outsourcing
  - service level agreement
  - reduce running cost
- accurate financial status
  - financial processes
- prevention of virtual crime
  - security management
  - access control
- easy change of product database
  - easy update of product attributes
  - responsive portal
- reporting & analysis
  - easy update of flow
- service cost / volume
  - financial reporting & analysis
  - work flow support
  - access administration
  - authentication
  - auditing, logging
  - security reporting
Example Assessment of Design Choices

Are these concerns: relevant? significant?

What is the impact at enterprise level?
Example Zero Order Problem Statement

How does the picture cache design impact

- responsive portal
- service cost/volume
- predictable (low) cost level
- high sales
- right pricing
- right presentation
- fast delivery
- goods flow tuning
- fast goods processing
- low capital use
- minimal stocks
- strategic outsourcing
- service level agreement
- reduce running cost
- accurate financial status
- financial processes
- access control
- security management
- prevention of virtual crime
- easy update of product database
- easy update of product attributes
- easy change of flow
- financial reporting & analysis
- work flow support
- access administration
- authentication
- auditing, logging
- security reporting
- security management

Modeling and Analysis: Application Models
©2006, Embedded Systems Institute
131 ©2006, Gerrit Muller
Zero Order Cost Model

total cost = f + s(v) + p * v + g * v
where
f = fixed base cost
s = service cost, see below
p = personnel cost including overheads
v = volume
g = goods flow handling

service cost s(v) = b + c * v
where
b = fixed base cost
c = cost / volume
v = volume
all including provider margin
low volume, labor intensive, shop

fixed costs and personnel cost dominate:
service cost changes have negligible impact on total cost!

total cost = f + s(v) + p * v + g * v
where
f = fixed base cost
s = service cost, see below
p = personnel cost including overheads
v = volume
g = goods flow handling

service cost s(v) = b + c * v
where
b = fixed base cost
c = cost / volume
v = volume
all including provider margin

\[ f = 100k \]
\[ p = 1 \]
\[ v = 100k \]
\[ g = 0.1 \]
\[ s(100k) = 101k \]
\[ b = 100k \]
\[ c = 0.1 \]
Example High Volume, Highly Automated, Shop

**high volume, highly automated, shop**

variable service costs dominate:
service cost changes have big impact on total cost!

\[
\text{total cost} = f + s(v) + p \times v + g \times v
\]
where
- \(f\) = fixed base cost
- \(s = \text{service cost, see below}\)
- \(p = \text{personnel cost including overheads}\)
- \(v = \text{volume}\)
- \(g = \text{goods flow handling}\)

\[
\text{service cost } s(v) = b + c \times v
\]
where
- \(b = \text{fixed base cost}\)
- \(c = \text{cost} / \text{volume}\)
- \(v = \text{volume}\)
all including provider margin

\[
\begin{align*}
\text{fixed base} & = 1M \\
\text{personnel} & = 0.01 \\
\text{goods flow} & = 0.01 \\
\text{service} & = 100M \\
\text{variable service} & = 101k \\
\end{align*}
\]
Very simple, very coarse, zero order models provide insight in relevance of specification and design issues.

These models are used to identify relevant issues.
Abstract
Products and enterprises evolve over time. This presentation explores the impact of these changes on the system and on the business by making (small and simple) models of life cycle aspects.

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.

July 31, 2014
status: preliminary
draft
version: 0.7
Product Related Life Cycles

- Individual systems
- Service
  - System
    - Production
    - Sales
  - Upgrades and options
    - Production
    - Sales
- Creation
- Upgrades and options
  - Creation
- Disposal
System Life Cycle

- System order
- Ordering components
- Manufacturing
- Shipping installation
- Using
- Add option
- Maintenance
- Upgrade
- Using
- Sales shipping refurbishing shipping installation secondary use maintenance dispose

Local changes, e.g., accounts procedures
### Approach to Life Cycle Modeling

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Identify potential life cycle changes and sources</td>
<td></td>
</tr>
<tr>
<td>Characterize time aspect of changes</td>
<td>how often, how fast</td>
</tr>
<tr>
<td>Determine required effort</td>
<td>amount, type</td>
</tr>
<tr>
<td>Determine impact of change on system and context</td>
<td>performance, reliability</td>
</tr>
<tr>
<td>Analyse risks</td>
<td>business</td>
</tr>
</tbody>
</table>

See reasoning for further details.
What May Change During the Life Cycle?

- business volume
- product mix
- product portfolio
- product attributes (e.g. price)
- customers
- personnel
- suppliers
- application, business processes
- et cetera

- www.homes4sale.com
- www.apple.com/itunes/
- www.amazon.com
- www.ebay.com
- www.shell.com
- www.stevens.edu
- www.nokia.com
- stock market
- insurance company
- local Dutch cheese shop
Simple Model of Data Sources of Changes

usage context

other systems

system

requirements

design realization

other systems

life cycle context

legend

automated data inputs
interoperability

human inputs
error prone!
~3% error rate

change request
problem report
Data Sources of Web Server

- Content preparation
- Content provider
- Data quality?
- Web server
- Shop configuration (e.g., staff, roles)
- System configuration (e.g., resource allocation)
- Client
new books per year

<table>
<thead>
<tr>
<th>Country</th>
<th>Sales 2005</th>
<th>Sales 1996</th>
</tr>
</thead>
<tbody>
<tr>
<td>UK (1)</td>
<td>206k</td>
<td>107k</td>
</tr>
<tr>
<td>USA (2)</td>
<td>172k</td>
<td>68k</td>
</tr>
<tr>
<td>China (3)</td>
<td>101k</td>
<td></td>
</tr>
<tr>
<td>India (21)</td>
<td>12k</td>
<td></td>
</tr>
</tbody>
</table>

source: http://en.wikipedia.org/wiki/Books_published_per_country_per_year

```
new books per year

<table>
<thead>
<tr>
<th>Country</th>
<th>Sales 2005</th>
<th>Sales 1996</th>
</tr>
</thead>
<tbody>
<tr>
<td>UK (1)</td>
<td>206k</td>
<td>107k</td>
</tr>
<tr>
<td>USA (2)</td>
<td>172k</td>
<td>68k</td>
</tr>
<tr>
<td>China (3)</td>
<td>101k</td>
<td></td>
</tr>
<tr>
<td>India (21)</td>
<td>12k</td>
<td></td>
</tr>
</tbody>
</table>
```

```
<table>
<thead>
<tr>
<th>Country</th>
<th>Sales 2005</th>
<th>Sales 1996</th>
</tr>
</thead>
<tbody>
<tr>
<td>UK (1)</td>
<td>206k</td>
<td>107k</td>
</tr>
<tr>
<td>USA (2)</td>
<td>172k</td>
<td>68k</td>
</tr>
<tr>
<td>China (3)</td>
<td>101k</td>
<td></td>
</tr>
<tr>
<td>India (21)</td>
<td>12k</td>
<td></td>
</tr>
</tbody>
</table>
```

source: http://en.wikipedia.org/wiki/Books_published_per_country_per_year
internet: broadband penetration

<table>
<thead>
<tr>
<th></th>
<th>Q1 '04</th>
<th>Q2 '04</th>
<th>growth in Q2 '04</th>
</tr>
</thead>
<tbody>
<tr>
<td>Asia Pacific total</td>
<td>48M</td>
<td>54M</td>
<td>12.8%</td>
</tr>
<tr>
<td>China</td>
<td>15M</td>
<td>19M</td>
<td>26.1%</td>
</tr>
<tr>
<td>India</td>
<td>87k</td>
<td>189k</td>
<td>116.8%</td>
</tr>
</tbody>
</table>


What is the expected growth of # customers?
What is the impact on system and infrastructure?
What is the impact on CRM (Customer Relation Management)?
What is the impact on customer, sales support staff?
How much time/effort is needed for content updates?
How much staff is needed?
What is the impact of errors in content updates?
How many errors can be expected?
What is the impact of content updates on server loads?
Web Shop Content Change Effort

\[
\text{effort}_{\text{changes}} = n_{\text{changes}} \cdot (t_{\text{prepare}} + t_{\text{verify}}) + t_{\text{commit}}
\]

\[
\#fte = \frac{\text{effort}_{\text{changes}}}{\text{hours per day}}
\]

<table>
<thead>
<tr>
<th>(n_{\text{changes}}) per day</th>
<th>10</th>
<th>100</th>
<th>1000</th>
</tr>
</thead>
<tbody>
<tr>
<td>effort changes</td>
<td>1 uur</td>
<td>10 uur</td>
<td>100 uur</td>
</tr>
<tr>
<td>#fte</td>
<td>0.1</td>
<td>1</td>
<td>12</td>
</tr>
</tbody>
</table>

with

\[
\begin{align*}
    t_{\text{prepare}} &= 4 \text{ min} \\
    t_{\text{verify}} &= 2 \text{ min} \\
    t_{\text{commit}} &= 1 \text{ min}
\end{align*}
\]

hours per day = 8 hours
Example of Client Level Changes

Up-to-date information:
Bestsellers
What Other Customers Are Looking At Right Now

catalogue entries
main access through search
personalization
styling: frequently updated, fashion!

standard boilerplate

snapshot of www.amazon.com
Example of Time Scale Model for Changes

- Problem response
  - Clinical prototype
    - Procedural change
      - Legislation change
        - Workstation useful life
          - MR scanner useful life

- Commodity hardware and software
  - Minor SW release
  - Major SW release
    - New generation of magnets
      - Gradients
        - Detectors

- Useful life
  - MR scanner: 10 years
  - Workstation: 1 year

- Useful life
  - Workstation: 10 years
  - Problem response: 3 months
Web Shop Security and Changes

What is the security model?

What is the impact on server loads?

What is the impact on staffing?

What is the impact of changes in staff?

What is the impact of changes on security?

public internet

protected production area

very secure intranet

screen
client
web server
network

data base server
product descriptions

logistics ERP
financial
customer relations

content definition

Modeling and Analysis: Life Cycle Models
©2006, Embedded Systems Institute
149 ©2006, Gerrit Muller
version: 0.7
July 31, 2014
MALCwebShopSecurity
new faults = average fault density * #changes

#errors = \sum_{faults} f(\text{severity, hit probability, detection probability})

<table>
<thead>
<tr>
<th>severity</th>
<th>hit probability</th>
<th>detection probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jansen iso Janssen</td>
<td>low</td>
<td>high</td>
</tr>
<tr>
<td>operator iso sales repr</td>
<td>high</td>
<td>high</td>
</tr>
</tbody>
</table>
Simplistic Financial Computations for System Architects.

by Gerrit Muller       HBV-NISE

e-mail: gaudisite@gmail.com

www.gaudisite.nl

Abstract
This document explains how simple financial estimates can be made by system architects. These simplistic estimates are useful for an architect to perform sanity checks on proposals and to obtain understanding of the financial impact of proposals. Note that architects will never have full fledged financial controller know how and skills. These estimates are zero order models, but real business decisions will have to be founded on more substantial financial proposals.

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.
Product Margin = Sales Price - Cost

- **Material**
- **Labour**
- **Miscellaneous**
- **Margin**
- **Cost Price**
- **Sales Price**
- **Retailer Margin and Costs**

**Margin per product.**
The margin over the sales volume, must cover the fixed costs, and generate profit transportation, insurance, royalties per product, ...

**Cost per product,**
excluding fixed costs
purchase price of components may cover development cost of supplier
Profit as function of sales volume

\[
\text{Sales Volume (in units)} \rightarrow \text{Income} \quad \text{expenses} \quad \text{break even point} \quad \text{fixed costs} \quad \text{variable}
\]

\[
\text{Sales Volume (in units)} \rightarrow \text{Expected sales volume}
\]
Investments, more than R&D

<table>
<thead>
<tr>
<th>financing</th>
</tr>
</thead>
<tbody>
<tr>
<td>marketing, sales</td>
</tr>
<tr>
<td>training sales&amp;service</td>
</tr>
<tr>
<td>NRE: outsourcing, royalties</td>
</tr>
<tr>
<td>research and development</td>
</tr>
</tbody>
</table>

- **business dependent:**
  - pharmaceuticals industry
  - sales cost >> R&D cost

- **strategic choice:**
  - NRE or per product

- **including:**
  - staff, training, tools, housing
  - materials, prototypes
  - overhead
  - certification

- Often a standard staffing rate is used that covers most costs above:
  - R&D investment = Effort * rate
Income, more than product sales only

- Other recurring income
- Services
  \[ \sum_{\text{services}} \text{income}_{\text{service}} \]
- Options, accessories
  \[ \sum_{\text{options}} \text{sales price}_{\text{option}} \times \text{volume}_{\text{option}} \]
- Products
  \[ \text{sales price}_{\text{product}} \times \text{volume}_{\text{product}} \]

License fees, pay per movie, content, portal updates, maintenance.
## The Time Dimension

<table>
<thead>
<tr>
<th>Q1</th>
<th>Q2</th>
<th>Q3</th>
<th>Q4</th>
<th>Q1</th>
<th>Q2</th>
<th>Q3</th>
</tr>
</thead>
<tbody>
<tr>
<td>investments</td>
<td>100k$</td>
<td>400k$</td>
<td>500k$</td>
<td>100k$</td>
<td>100k$</td>
<td>60k$</td>
</tr>
<tr>
<td>sales volume (units)</td>
<td>-</td>
<td>-</td>
<td>2</td>
<td>10</td>
<td>20</td>
<td>30</td>
</tr>
<tr>
<td>material &amp; labour costs</td>
<td>-</td>
<td>-</td>
<td>40k$</td>
<td>200k$</td>
<td>400k$</td>
<td>600k$</td>
</tr>
<tr>
<td>income</td>
<td>-</td>
<td>-</td>
<td>100k$</td>
<td>500k$</td>
<td>1000k$</td>
<td>1500k$</td>
</tr>
<tr>
<td>quarter profit (loss)</td>
<td>(100k$)</td>
<td>(400k$)</td>
<td>(440k$)</td>
<td>200k$</td>
<td>500k$</td>
<td>840k$</td>
</tr>
<tr>
<td>cumulative profit</td>
<td>(100k$)</td>
<td>(500k$)</td>
<td>(940k$)</td>
<td>(740k$)</td>
<td>(240k$)</td>
<td>600k$</td>
</tr>
</tbody>
</table>

\[
\text{cost price / unit} = 20k$
\]
\[
\text{sales price / unit} = 50k$
\]

\[
\text{variable cost} = \text{sales volume} \times \text{cost price / unit}
\]

\[
\text{income} = \text{sales volume} \times \text{sales price / unit}
\]

\[
\text{quarter profit} = \text{income} - (\text{investments} + \text{variable costs})
\]
The “Hockey” Stick

profit

loss
time

(1M$)
(0.5M$)
0.5M$
1M$
What if ...?

- Early more expensive product + follow-on
- Delay of 3 months
- Original model

Simplistic Financial Computations for System Architects.
©2006, Gerrit Muller
Fashionable financial yardsticks

Return On Investments (ROI)

Net Present Value

Return On Net Assets (RONA)  leasing reduces assets, improves RONA

turnover / fte  outsourcing reduces headcount, improves this ratio

market ranking (share, growth)  "only numbers 1, 2 and 3 will be profitable"

R&D investment / sales  in high tech segments 10% or more

cash-flow  fast growing companies combine profits with negative cash-flow, risk of bankruptcy
Abstract
The purpose of the application view is described. A number of methods or models is given to use in this view: stakeholder and concerns, context diagram, static entity relationship models and dynamic flow models.

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.

July 31, 2014
status: preliminary
draft
version: 0.2
The application view
©2006, Gerrit Muller

version: 0.2
July 31, 2014
AVoverview
Stakeholders and concerns MRI scanner

- **government**
  - cost of care
- **financial dir.**
  - cash flow
  - cost of op.
- **insurance**
  - cost of care
- **administration**
  - patient id
  - invoice

- **general practitioner**
  - patient
- **ref. physician**
  - diagnosis
  - treatment
- **radiologist**
  - diagnosis
  - reimbursement
- **nurse**
  - patient
  - ease of work

- **patient**
  - comfort
  - health
- **family**
  - support
- **IT dep.**
  - conformance
  - security
- **facility man.**
  - space
  - service supp.
- **maintainer**
  - accessibility
  - safety
- **cleaner**
  - accessibility
  - safety

**Legend**:
- **administrative**
- **clinical**
- **patient**
- **support**
Context of motorway management system

- Maintenance contractors
- Fleet management
- Urban traffic control
- Advanced vehicle control
- Environmental monitoring
- Third party
- Other concerns
- Special applications
- Bus lanes
- Torry lanes
- Restaurants
- Gas stations
- Specialized segments
- Contingencies
- "Add-ons"
- Special applications
- Other concerns
- Administrative
- Special destinations
- Toll tunnel
- Car repair
- Towing service
- Airports
- Railways
- Competing or cooperating?

The application view

©2006, Embedded Systems Institute
AVcontextMotorwayManagement
Example of simple TV application model

channel \(\xrightarrow{transmits}\) 

transmits 

selects 

tuner 

TV screen 

TV 

soaps 

movies 

sports 

news 

content 

canned 

described by 

age, sex, violence attributes 

informs 

parents 

children 

live 

The application view

©2006, Gerrit Muller
Examples of dynamic models

flow models

state diagrams

time line
Productivity and Cost models

The application view
©2006, Embedded Systems Institute
167 ©2006, Gerrit Muller
version: 0.2
July 31, 2014
AVcostBenefitModels
Dynamics of an URF examination room

8:30  9:00  9:30  10:00  10:30

- patient 1, intestinal investigation
- patient 2, simple X-ray
- patient 3, intestinal investigation
- patient 4, intestinal investigation
- patient 5, intestinal investigation

waiting room
changing room
URF examination room
Make a context diagram:

What other related systems and applications are used?
How do these relate with our system
Visualize the context as diagram
Make supporting diagrams for main application views
Reflection on Exercise

+ Context diagram lift insight to a higher level

~ Our system or application is only a fraction of the customers world

- Application models can become too generic or abstract
Conclusions

Real requirements are driven by understanding of the customer's application.

Complexity of finance is no excuse for ignoring all financial aspects; simplified models provide a lot of insight.

Techniques, Models, Heuristics of this module

Simplistic financial models

TBD
Module Modeling and Analysis: Integration and Reasoning

by Gerrit Muller Buskerud University College

e-mail: gaudisite@gmail.com
www.gaudisite.nl

Abstract
This module addresses the integration of small or partial models into bigger models. We also discuss how multiple models are used and how to reason using multiple models.
Where are we in the Course?

- **facts** from research
- **measurements**
- **assumptions**

- **uncertainties**
- **unknowns**
- **errors**

- **modeling**
- **analysis**

- **reasoning**

- **specification**
- **verification**
- **decisions**

- **risk**
- customer satisfaction
- time, cost, effort
- profit margin

- **usage context**
  - enterprise & users
- **system**
  - requirements
    - black box view
  - design
    - realization
    - technology

- **creation**
- life cycle context

- **life cycle context**
- **system**
- **usage context**

- **enterprise & users**
- **requirements**
- **black box view**
- **design**
- **realization**
- **technology**

- **specification**
- **verification**
- **decisions**

- **module**
- **modeling and analysis**
- **integration and reasoning**

©2006, Embedded Systems Institute
©2006, Gerrit Muller

version: 0.3
July 31, 2014
MMAREposition
Abstract
Models are made to facilitate decision making. These decisions range from business decisions, such as Service Level Agreements, to requirements, and to detailed design decisions. The space of decisions is huge and heterogeneous. The proposed modeling approach is to use multiple small and simple models. In this paper we discuss how to reason by means of multiple models.
content

From chaos to order: inputs, assumptions, models and decisions

Reasoning approach: stepwise top-down and bottom-up

Life cycles of models in relation to project life cycle
Purpose of Modeling

How to use multiple models to facilitate decisions?
How to get from many fragments to integral insight?
How many models do we need?
At what quality and complexity levels?
Graph of Decisions and Models

usage context

system

life cycle context

legend

- assumption
- input e.g. measurement
- decision
- model
1. Explore usage context, life cycle context and system

   **top-down**

   t2. Determine main Threads-of-Reasoning
   t3. Make main Threads-of-Reasoning SMART
   t4. Identify "hottest" issues
   t5. Model hottest, non-obvious, issues

   **learn**

   **bottom-up**

   b2a. "Play" with models
   b2b. Investigate facts
   b2c. Identify assumptions
   b3. Model significant, non-obvious, issues

6. Capture overview, results and decisions

7. Iterate and validate

*all steps time-boxed between 1 hour and a few days*

early in project  later in project
1. Explore usage context, life cycle context and system

Populate with "known" facts, numbers, issues from preceding projects, available work products and stakeholders
t2. Determine main Threads-of-Reasoning

Architecting and System Design

e.g. http://www.gaudisite.nl/ModuleTORSslides.pdf
t3. Make main Threads-of-Reasoning SMART

Quantify in terms of Key Performance Indicators, Key Performance Measures, Critical Resources

- market dynamics
- service costs

**usage context**
- products & attributes
- customers
- order rate
- personnel

**system**
- SW/HW platform(s)
- resources
- cost
- transaction initiated

**life cycle context**
- infrastructure
- margin services
- maintenance projection

**Market Volume**
- Best case
- Worst case

**Order Rate**
- server cost
- 1M$
- 2M$

**Time**
- JFMAMJJASOND

**System Load**
- time

version: 0.6
July 31, 2014
Intermezzo: the acronym SMART

<table>
<thead>
<tr>
<th>Specific</th>
<th>quantified</th>
</tr>
</thead>
<tbody>
<tr>
<td>Measurable</td>
<td>verifiable</td>
</tr>
</tbody>
</table>

Assignable (Achievable, Attainable, Action oriented, Acceptable, Agreed-upon, Accountable)

Realistic (Relevant, Result-Oriented)

Time-related (Timely, Time-bound, Tangible, Traceable)

*variation of meaning*
### t4. Identify "hottest" issues

<table>
<thead>
<tr>
<th>assess explored landscape:</th>
<th>1..5 scale,</th>
<th>1 = low risk</th>
<th>5 = high risk</th>
<th>et cetera</th>
</tr>
</thead>
<tbody>
<tr>
<td>highest (perceived) risk</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>most important/valuable</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>most discussed</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>historic evidence</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>urgency</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Rank issues according to aggregated assessment

<table>
<thead>
<tr>
<th>server cost</th>
<th>2</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>3</th>
<th>ranking</th>
</tr>
</thead>
<tbody>
<tr>
<td>order rate</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>transactions</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>response time</td>
<td>3</td>
<td>5</td>
<td>1</td>
<td>4</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>availability</td>
<td>2</td>
<td>5</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>network bandwidth</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>storage capacity</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>
t5. Model hottest, non-obvious, issues

non-obvious: desired lead time 2 mnth versus actual lead time 6 months
From *top-down* to *bottom-up*

1. Explore usage context, life cycle context and system

2. Determine main Threads-of-Reasoning

3. Make main Threads-of-Reasoning SMART

4. Identify "hottest" issues

5. Model hottest, non-obvious, issues

6. Capture overview, results and decisions

7. Iterate and validate

---

*all steps time-boxed between 1 hour and a few days*

- early in project
- later in project

---

Modeling and Analysis: Reasoning
©2006, Embedded Systems Institute
187 ©2006, Gerrit Muller

version: 0.6
July 31, 2014
MAREmethod
## b2abc: Bottom-up

<table>
<thead>
<tr>
<th>b2a. &quot;Play&quot; with models</th>
<th>b2b. Investigate facts</th>
<th>b2c. Identify assumptions</th>
</tr>
</thead>
<tbody>
<tr>
<td>vary inputs</td>
<td>market research</td>
<td>What is the basis for</td>
</tr>
<tr>
<td>vary model structure</td>
<td>measurements</td>
<td>model structure, design</td>
</tr>
<tr>
<td>to understand <strong>model</strong></td>
<td>preceeding systems</td>
<td>decision, specification,</td>
</tr>
<tr>
<td><strong>applicability</strong>, <strong>design quality</strong></td>
<td>micro benchmarks</td>
<td>quantification et cetera?</td>
</tr>
<tr>
<td>and <strong>specification feasibility</strong></td>
<td>literature, supplier info</td>
<td><strong>Most assumptions are</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>implicit and hidden!</strong></td>
</tr>
</tbody>
</table>

\[
\text{n}_{\text{CPU}} = \frac{t_{\text{required total}}}{t_{1 \text{ CPU}}}
\]
\[
t_{\text{required total}} = n_{\text{transactions}} \cdot t_{1 \text{ transaction}} + t_{\text{other}}
\]
\[
t_{1 \text{ transaction}} = 1 \text{ ms (on 1 CPU)}
\]

---

\[n_{\text{CPU}} = \frac{t_{\text{required total}}}{t_{1 \text{ CPU}}}
\]
\[t_{\text{required total}} = n_{\text{transactions}} \cdot t_{1 \text{ transaction}} + t_{\text{other}}
\]
\[t_{1 \text{ transaction}} = 1 \text{ ms (on 1 CPU)}
\]

---

**IBM System p5 595**

TPC-C Throughput 4,033,378
Total # of Processors: 32
Total # of Cores: 64

\[
1/t_{1 \text{ transaction}} = \frac{4 \times 10^{6}}{60} / 64
\]

\[t_{1 \text{ transaction}} \approx 1 \text{ ms}
\]

---

http://www.tpc.org/tpcc/results/tpcc_perf_results.asp

server load dominated by transactions
transaction load scales linear
TPC-C is representative

**what is the effect of other TPC-C workload?**
Bottom-up, more detailed steps

<table>
<thead>
<tr>
<th>Make a list of technologies, components and resources to be used</th>
</tr>
</thead>
<tbody>
<tr>
<td>transactions, data base engine, memory, disk</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Make a list of important qualities</th>
</tr>
</thead>
<tbody>
<tr>
<td>performance, reliability, security, maintainability</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Make a characterization matrix of technologies, components and resources versus qualities</th>
</tr>
</thead>
<tbody>
<tr>
<td>1..5 scale, 1 = low risk 5 = high risk et cetera</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Perform step 2abc on most critical</th>
</tr>
</thead>
<tbody>
<tr>
<td>class 4 and 5 risks</td>
</tr>
</tbody>
</table>

Modeling and Analysis: Reasoning
©2006, Embedded Systems Institute
189 ©2006, Gerrit Muller

version: 0.6
July 31, 2014
MAREbottomUpHowTo
b3. Model significant, non-obvious, issues

for example, memory use in server(s) for picture transfers and buffering

<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>n</td>
<td>m</td>
<td>k</td>
<td>s</td>
<td>c</td>
<td>MB</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1.E+05</td>
<td>10</td>
<td>1.5</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>10</td>
<td>1.E+05</td>
<td>20</td>
<td>5.3</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>1000</td>
<td>1.E+05</td>
<td>100</td>
<td>296.2</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>1000</td>
<td>1.E+06</td>
<td>100</td>
<td>2962.1</td>
<td></td>
</tr>
</tbody>
</table>

picture memory = 
3 * n * s + 
5 * m * s + c * s + 
3 * k * s

where
- n = # back office access threads
- m = # picture cache threads
- k = # web server threads
- s = picture size in bytes
- c = in memory cache capacity

in # pictures

What is the performance impact of memory use on other processing?

memory use
product browsing only
pictures only
single server

L1 capacity  L2  L3  main memory  disk
kB  MB  GB

modelSignificant
Learning Concurrent Bottom-up and Top-down

**top-down:** what is *hidden* in details?

top-down: do we address the *relevant* decomposition?

bottom-up: do we address relevant details?

bottom-up: what details have *significant* impact?

---

*order rate* (and evolution over time)

*is highly speculative*

---

*needs input* ----

*transactions dominate* server load & cost

---

*induces risk* ----

Modeling and Analysis: Reasoning

©2006, Embedded Systems Institute

version: 0.6
July 31, 2014
MARElearn

©2006, Gerrit Muller
top-down:
what is impact of
catalogue size and changes?

bottom-up:
what is relevant concurrency (k), cache size (c),
or picture size (s)?
6. Capture overview, results and decisions

- re-usable assets from previous projects
- major milestone
- transfer of relevant insights
- re-usable assets from previous projects
- major milestone
- transfer of relevant insights
- re-usable assets to next projects
- re-usable assets to next projects

- definition
- execution
- deployment

- customer business
- proposal
- specification
- design
- realization
- configuration
- consolidation baseline
7. Iterate and Validate

- Stakeholders
- Questions
- Results
- Data
- Experiments
- Measurements
- Data
- Systems, models, components

Iteration timeline:
- Iteration 1
- Iteration 2
- Iteration 3
- Iteration 4

Time
Focus is Shifting during Project
Models Support Communication

number of involved people

10

1

fast but intangible

in the architect's head

tangible

model based architecting

1 minute 1 hour 1 day 1 week 1 month

time per iteration cycle

Modeling and Analysis: Reasoning
©2006, Gerrit Muller
Frequency of Assumptions, Decisions and Modeling

![Frequency of Assumptions, Decisions and Modeling](image)

Legend:
- **a**: Assumption
- **i**: Input e.g. measurement
- **d**: Decision
- **m**: Model

- **Implicit (trivial?)**
- **Explicit**
- **Try-outs**
- **Very simple**
- **Small**
- **Key**
- **Substantial**

Scale:
- $10^0$
- $10^2$
- $10^4$
- $10^6$
Life Cycle of Models

- **understanding**
- **exploration**
- **optimization**
- **verification**

**try out models**
- most try out models never leave the desk or computer of the architect!
- many small and simple models are used only once;
- some are re-used in next projects

**simple and small models**
- re-use
- archived not maintained
- re-used in next project

**substantial models**
- re-use
- archived not maintained
- re-used in next project
- many small and simple models are used only once;
- some are re-used in next projects
- substantial models capture core domain know how;
- they evolve often from project to project.
Examples of Life Cycle of Models

understanding > exploration > optimization > verification

- try out models
  - load/cost
  - function mix
  - load/cost peak impact

- simple and small models
  - customer global distribution
  - integral load model

- substantial models
  - (IP assets)
  - global customer demographics

- web server performance
  - load/stress test suite
  - webshop benchmark suite
## Conclusions

Top-down and bottom-up provide complementary insights

Key words for selection: hottest, non-obvious, significant, relevant

Multiple small models are used in combination

Some models evolve from very simple to more substantial

### Techniques, Models, Heuristics of this module

- Threads-of-reasoning
- SMART
- Key Performance Indicators, Key Performance Measures, Critical Resources
- Ranking matrices
Abstract
More substantial models are created step by step. We will discuss the order of creation and modularity considerations. The modules have to be integrated into the desired substantial model.

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.
Example of (Partial) Flow Simulator

Modeling and Analysis: Modularity and Integration
©2006, Embedded Systems Institute

©2006, Gerrit Muller

version: 0
July 31, 2014
MAINflowSimulator
Example of Incremental Model Creation
### Approach for Incremental Model Creation

<table>
<thead>
<tr>
<th>Recommendation</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Start with the hottest issue</strong></td>
</tr>
<tr>
<td>what creates the most discussion or uncertainty?</td>
</tr>
<tr>
<td><strong>Ensure immediate feedback</strong></td>
</tr>
<tr>
<td>does this model help to answer the questions that we have?</td>
</tr>
<tr>
<td><strong>Keep flexible decoupling point</strong></td>
</tr>
<tr>
<td>e.g. human readable/editable files</td>
</tr>
<tr>
<td><strong>Extend model only for a good purpose</strong></td>
</tr>
<tr>
<td>don't integrate models because it can be done</td>
</tr>
<tr>
<td><strong>Create effective visual outputs</strong></td>
</tr>
<tr>
<td>simple animations, graphs, tables, ...</td>
</tr>
<tr>
<td><strong>Refactor regularly</strong></td>
</tr>
<tr>
<td>based on increasing insight, feedback and purpose</td>
</tr>
</tbody>
</table>
Attention Points for Every Integration Step

Does the output of the integrated model match your expectation?
Can you explain the model behavior?
Can you explain the variation of the output?

<table>
<thead>
<tr>
<th>time behavior</th>
<th>value</th>
<th>variation</th>
<th>value</th>
<th>variation</th>
<th>or</th>
<th>?</th>
</tr>
</thead>
<tbody>
<tr>
<td>load</td>
<td>time</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Module Modeling and Analysis: Analysis and Using Models

by Gerrit Muller  Buskerud University College
e-mail: gaudisite@gmail.com
www.gaudisite.nl

Abstract
This module addresses the analysis of models and discusses how to use models.
Where are we in the Course?

facts from research
measurements
assumptions

uncertainties
unknowns
errors

accuracy
working range
credibility

risk
customer satisfaction
time, cost, effort
profit margin

_module

Module Modeling and Analysis: Analysis and Using Models
©2006, Embedded Systems Institute
version: 0.1
July 31, 2014
MMAANposition

©2006, Gerrit Muller
Abstract
Models only get value when they are actively used. We will focus in this presentation on analysis aspects: accuracy, credibility, sensitivity, efficiency, robustness, reliability and scalability.

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.

July 31, 2014
status: planned
version: 0.2
What Comes out of a Model

- varying inputs
- varying circumstances
- varying design options
- varying realizations
- specification changes

and ripple through model(s)

- accuracy
- credibility
- working range

worst case behavior

exceptional behavior

- sensitivity
- robustness
- efficiency

- working range

- sensitivity

- robustness

- efficiency

- performance
- reliability
- scalability
- other system qualities

specification feasibility

design quality

- model applicability

- design quality

- specification feasibility

- design quality

life cycle

- understanding
- exploration
- optimization
- verification

- design

Modeling and Analysis: Analysis
©2006, Embedded Systems Institute
209  ©2006, Gerrit Muller

version: 0.2
July 31, 2014
MAANaspects
Applicability of the Model

+ $\mathcal{E}_1$
- $\mathcal{E}_2$

Input
- accuracy
- credibility

Measurements
Assumptions
Facts

Abstraction

Model(s)

Accuracy
Credibility
Working range

Model realization
Credibility propagation

Usage context
Specifications
Designs
Realizations

Version: 0.2
July 31, 2014
MAANmodelApplicability
How to Determine Applicability

**try out models**
be aware of accuracy, credibility and working range

**simple and small models**

1. **Estimate accuracy of results**
   - based on most significant inaccuracies of inputs and assumed model propagation behavior

2. **Identify top 3 credibility risks**
   - identify biggest uncertainties in inputs, abstractions and realization

3. **Identify relevant working range risks**
   - identify required (critical) working ranges and compare with model working range

**substantial models**

systematic analysis and documentation of accuracy, credibility and working range
A system design assumption is often:
the performance of this function
{is constant | is linear | doesn't exceed x | ...}

The working range is the interval where this assumption holds
The models are simple as long as working ranges are obeyed. If the system operates outside the working range then more complex models need to be used (e.g. from 0th order to 1st order).
Common Pitfalls

- Discrete events in continuous world
  - Discretization artefacts
    - E.g. stepwise simulations

- (Too) systematic input data
  - Random data show different behavior
    - E.g. memory fragmentation

- Fragile model
  - Small model change results in large shift in results

- Self fulfilling prophecy
  - Price erosions + cost increase (inflation) -> bankruptcy
Example of Worst Case Picture Cache

What is the system behavior and performance for worst case access patterns?
Worst Case Questions

Which design assumptions have a big impact on system performance?

What are the worst cases for these assumptions?

How does the system behave in the worst case?

  a. poor performance within spec
  b. poor performance not within spec
  c. failure -> reliability issue
## FMEAlke Analysis Techniques

### (systematic) brainstorm

1. **FMEA-like Analysis Techniques**
   - Potential hazards
   - Safety
   - Hazard analysis
   - Reliability
   - FMEA
   - Failure modes
   - Exceptional cases
   - Security
   - Vulnerability risks
   - Maintainability
   - Change cases
   - Performance
   - Worst cases

### Analysis and Assessment
- Probability
- Severity
- Propagation

### Improve
- Spec, design, process, procedure, ...

### Table: Analysis Techniques

<table>
<thead>
<tr>
<th>Category</th>
<th>Focus</th>
<th>Data</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Safety hazard analysis</td>
<td>Potential hazards</td>
<td>Damage</td>
<td>Measures</td>
</tr>
<tr>
<td>Reliability FMEA</td>
<td>Failure modes</td>
<td>Effects</td>
<td>Measures</td>
</tr>
<tr>
<td>Security</td>
<td>Vulnerability risks</td>
<td>Consequences</td>
<td>Measures</td>
</tr>
<tr>
<td>Maintainability</td>
<td>Change cases</td>
<td>Impact, effort, time</td>
<td>Decisions</td>
</tr>
<tr>
<td>Performance</td>
<td>Worst cases</td>
<td>System behavior</td>
<td>Decisions</td>
</tr>
</tbody>
</table>
Brainstorming Phases

wave 1: the obvious

wave 2: more of the same

wave 3: the exotic, but potentially important

don't stop too early with brainstorming!
Different Viewpoints for Analysis

usage context
- new product
  - e.g. WoW extension
- merger
- automated access

system
- new functions
- new interfaces
- new media
- new standards
- cache/memory trashing
- garbage collection
- critical sections
- local peak loads
- intermittent HW failure
- power failure
- network failure
- new SW release
- roll back to old SW release

life cycle context
sensitivity: how sensitive is the system output for small changes in input or realization?
Example of CPU Utilization and Efficiency

CPU utilization is "only" 8%
what is the efficiency?
Efficiency is Context Dependent!

*low volume, labor intensive, shop*

- Fixed costs and personnel cost dominate: service cost changes have negligible impact on total cost!

*high volume, highly automated, shop*

- Variable service costs dominate: service cost changes have big impact on total cost!

---

Modeling and Analysis: Analysis
©2006, Gerrit Muller
version: 0.2
July 31, 2014
MAANefficiencyInContext