Abstract
This course provides submethods for the CAFCR views. Qualities are provided to integrate the views. Story telling is introduced as a means to explore requirements. In a second iteration over the CAFCR views a thread of reasoning over the views is created.

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.
Module CAFCR course info

by Gerrit Muller  Buskerud University College

e-mail: gaudisite@gmail.com

www.gaudisite.nl

Abstract
This module provides the information about the CAFCR course: “Multi-Objective Embedded Systems Design, based on CAFCR”.

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.
Multi-Objective Embedded Systems design, based on CAFCR

by Gerrit Muller       Buskerud University College

e-mail: gaudisite@gmail.com

www.gaudisite.nl

Abstract
The course Multi-Objective Embedded Systems Design, based on the CAFCR-views, is described. The program existing of 10 modules is described. The course format, iterating theory, illustration and interaction is explained. The course heavily emphasizes the practical application of the method. In every module the theory is applied on the participants products. Teams of 4 participants with the same background apply the method on their own product and report the results.

Distribution
This article or presentation is written as part of the Gaudi project. The Gaudi 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 24, 2014
status: preliminary draft
version: 0.1
Complementing Forms

Theory
- dull
- passive

Insight

Practical Illustration
- vivid
- passive

Interaction
- vivid
- active

Spin-off:
- cross-fertilization

Abstraction

Exercise
<table>
<thead>
<tr>
<th>Time</th>
<th>Subject</th>
</tr>
</thead>
<tbody>
<tr>
<td>Session 1</td>
<td>Method overview</td>
</tr>
<tr>
<td>Session 2</td>
<td>Functional View</td>
</tr>
<tr>
<td>Session 3</td>
<td>Customer Views</td>
</tr>
<tr>
<td>Session 4</td>
<td>Design Views</td>
</tr>
<tr>
<td>Session 5</td>
<td>Story telling</td>
</tr>
<tr>
<td>Session 6</td>
<td>Qualities</td>
</tr>
<tr>
<td>Session 7</td>
<td>Customer Views (2)</td>
</tr>
<tr>
<td>Session 8</td>
<td>Functional View (2), Cases</td>
</tr>
<tr>
<td>Session 9</td>
<td>Design Views (2)</td>
</tr>
<tr>
<td>Session 10</td>
<td>wrap up</td>
</tr>
</tbody>
</table>
Rules of the Broadcast Part

- Please write your questions/remarks/statements on yellow stickers and attach them at the end on the P-flip. 
  *These will be used in the interactive section for discussion and to increase insight.*

- Short clarification questions are welcome, 
  *Discussion will take place in the interactive part.*

- Stupid questions don’t exist. Learning is based on **safe** and **open** interaction.
  *Very individual-oriented questions can be referred to a break or after the session.*
Your contribution is essential.

Don’t monopolize the time. Everyone, also the quiet people, should have the opportunity to contribute.

*The facilitator will intervene if the contribution is limited to a small group of participants.*

Respect the contribution of others.

*Opinions can’t be wrong, difference of opinion is normal and called pluriformity.*

The course format is highly experimental and based on improvisation, constructive proposals are welcome.

*It is your course! Regular evaluations will give the opportunity to influence the rest of the course.*
Evaluation of the Expectations

Please write your name and expectations with a marker on one A4 page.

Describe your expectations as one-liner or in a few keywords.

These pages will be displayed on the wall of the room.

At the end of the course we will look back on these expectations, with the purpose of two-way learning.
Abstract
This module described the overview of the complete architecting method.
Overview of CAFCR and Threads of Reasoning

by Gerrit Muller  Buskerud University College

e-mail: gaudisite@gmail.com

www.gaudisite.nl

Abstract

The described architecting method uses the CAFCR model as starting point. Qualities are used as orthogonal dimension to integrate the CAFCR views. Story telling is used to add specifics. Threads of reasoning combine all the information into a coherent overview.
From vague notions to articulate and structured architecture description:

- articulated
- structured
- problem and solution know-how
- architecting

Vague notion of the problem

Vague notion of potential solutions

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

Basic methods
- framework
- submethods
- integration methods

Architecting method:

Overview of CAFCR and Threads of Reasoning

Gerrit Muller

version: 1.5
July 24, 2014
AMOmethod
Overview of architecting method

**method outline**

**framework**

- **Customer objectives**
- **Application**
- **Functional**
- **Conceptual**
- **Realization**

**submethods**

- + key drivers
- + value chain
- + business models
- + supplier map
- + stakeholders
- + context diagram
- + entity relationship models
- + dynamic models
- + use case
- + commercial, logistics decompositions
- + mapping technical functions and several more
- + construction decomposition
- + functional decomposition
- + information model and many more
- + budget
- + benchmarking
- + performance analysis
- + safety analysis and many more

**integration**

via qualities

- safety
- performance

**explore**

specific details

- story
- use case
- detailed design

**reasoning**

market vision

a priori solution know-how

- diagnosis
- mapping
- typical use case
- Moore’s law
- render engine
- processing power
- depth
Overview of CAFCR and Threads of Reasoning

14  Gerrit Muller

version: 1.5
July 24, 2014
CAFCRannotated

The “CAFCR” model

What does Customer need in Product and Why?

Customer What
Customer How
Application
Functional
Conceptual
Realization

drives, justifies, needs
enables, supports
Five viewpoints for an architecture

Customer objectives

What does Customer need in Product and Why?

C - Customer

Application

F - Functional

Conceptual

Product

How

Realization

context understanding

intention

objective driven

opportunities

constraint awareness

know how based

What does Customer need in Product and Why?

Overview of CAFCR and Threads of Reasoning
version: 1.5
July 24, 2014
AMOintegratingCAFCR
Short introduction to basic “CAFCR” model

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

Abstract
The basic “CAFCR” reference model is described, which is used to describe a system in relation to its context. The main stakeholder in the context is the customer. The question “Who is the customer?” is addressed.
The “CAFCR” model

What does Customer need in Product and Why?

Customer objectives
Application
Functional
Conceptual
Realization

drives, justifies, needs
enables, supports
What does Customer need in Product and Why?

Customer
- What
- How

Product
- What

Customer objectives
Application
Functional
Conceptual
Realization

context understanding
intention
objective driven
opportunity
constraint awareness
knowledge based

Short introduction to basic “CAFCR” model
version: 0.4
July 24, 2014
MSintegratingCAFCR
CAFCR can be applied recursively

Consumer

Drives

Customer's Business

Enables

Customer's Business

Drives

Value Chain

larger scope has smaller influence on architecture

Customer Business

Enables

Customer Business

Drives

System (producer)

Enables

System (producer)
<table>
<thead>
<tr>
<th>segmentation axis</th>
<th>examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>geographical</td>
<td>USA, UK, Germany, Japan, China</td>
</tr>
<tr>
<td>business model</td>
<td>profit, non profit</td>
</tr>
<tr>
<td>economics</td>
<td>high end versus cost constrained</td>
</tr>
<tr>
<td>consumers</td>
<td>youth, elderly</td>
</tr>
<tr>
<td>outlet</td>
<td>retailer, provider, OEM, consumer direct</td>
</tr>
</tbody>
</table>
Example of a small buying organization

Who is the customer?

decision maker(s)
purchaser
operator
maintainer
user

department head

cEO: Chief Executive Officer
CFO: Chief Financial Officer
CIO: Chief Information Officer
CMO: Chief Marketing Officer
CTO: Chief Technology Officer
CAFCR+ model; Life Cycle View

- **Customer objectives**
- **Application**
- **Functional**
- **Conceptual**
- **Realization**

**Life cycle**
- operations
- maintenance
- upgrades
- development
- manufacturing
- installation
- sales, service, logistics, production, R&D

Short introduction to basic “CAFCR” model

Gerrit Muller

version: 0.4
July 24, 2014
BCAFCRplusLifeCycle
• make a bottom-up analysis of your product:
  1. realization
  2. conceptual
  3. functional
  4. application
  5. customer objectives
  6. qualities

• use time boxes of 15 minutes per view

• show the most dominant decomposition of that view, as diagram or as a list
Module Functional View

by Gerrit Muller  Buskerud University College

e-mail: gaudisite@gmail.com

www.gaudisite.nl

Abstract
This module addresses the Functional View.
Abstract
The purpose of the functional view is described. A number of methods or models is given to use in this view: (use) case descriptions, commercial decomposition function and feature specifications performance models and specifications, information models. The role of standards is discussed.
### Example personal video recorder use case contents

<table>
<thead>
<tr>
<th>typical use case(s)</th>
<th>worst case, exceptional, or change use case(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>interaction flow (functional aspects)</td>
<td>functional</td>
</tr>
<tr>
<td>select movie via directory</td>
<td>multiple inputs at the same time</td>
</tr>
<tr>
<td>start movie</td>
<td>extreme long movie</td>
</tr>
<tr>
<td>be able to pause or stop</td>
<td>directory behaviour in case of</td>
</tr>
<tr>
<td>be able to skip forward or backward</td>
<td>extreme many short movies</td>
</tr>
<tr>
<td>set recording quality</td>
<td></td>
</tr>
<tr>
<td>performance and other qualities</td>
<td>non-functional</td>
</tr>
<tr>
<td>(non-functional aspects)</td>
<td>response time with multiple inputs</td>
</tr>
<tr>
<td></td>
<td>image quality with multiple inputs</td>
</tr>
<tr>
<td></td>
<td>insufficient free space</td>
</tr>
<tr>
<td></td>
<td>response time with many directory entries</td>
</tr>
<tr>
<td></td>
<td>replay quality while HQ recording</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The functional view

version: 1.0
July 24, 2014
FVuseCase
Recommendations for working with use cases

+ combine related functions in one use case
- do not make a separate use case for every function
+ include non-functional requirements in the use cases

- minimise the amount of required worst case and exceptional use cases
- excessive amounts of use cases propagate to excessive implementation efforts
+ reduce the amount of these use cases in steps
- a few well chosen worst case use cases simplifies the design
Commercial Decomposition

The functional view

version: 1.0
July 24, 2014
FVcommercialTree
Logistic decompositions for a product

- **Commercial decomposition**
  - Saleable features

- **Service decomposition**
  - Replaceable items (such as consumables)

- **Goods flow decomposition**
  - Stockable items
  - Purchasable items
# Mapping technical functions on products

## Technical Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Home Cinema System</th>
<th>Flat Screen Cinema TV</th>
<th>Bedroom TV</th>
</tr>
</thead>
<tbody>
<tr>
<td>HD display</td>
<td>+</td>
<td>+</td>
<td>-</td>
</tr>
<tr>
<td>SD-&gt;HD up conversion</td>
<td>+</td>
<td>+</td>
<td>-</td>
</tr>
<tr>
<td>HD-&gt;SD down conversion</td>
<td>+</td>
<td>+</td>
<td>0</td>
</tr>
<tr>
<td>HD storage</td>
<td>0</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>SD storage</td>
<td>0</td>
<td>-</td>
<td>0</td>
</tr>
<tr>
<td>HD IQ improvement</td>
<td>+</td>
<td>+</td>
<td>-</td>
</tr>
<tr>
<td>SD IQ improvement</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>HD digital input</td>
<td>+</td>
<td>+</td>
<td>0</td>
</tr>
<tr>
<td>SD digital input</td>
<td>+</td>
<td>+</td>
<td>0</td>
</tr>
<tr>
<td>SD analog input</td>
<td>0</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>6 HQ channel audio</td>
<td>+</td>
<td>0</td>
<td>-</td>
</tr>
<tr>
<td>2 channel audio</td>
<td>-</td>
<td>+</td>
<td>+</td>
</tr>
</tbody>
</table>

### Legend
- + present
- 0 optional
- - absent
The functional view
31  Gerrit Muller

version: 1.0
July 24, 2014
FVUserInterface

Relation between user interface and functional specification

user interface
look & feel

functional behaviour

style guide
UI spec

prototype
as complement
to spec

stubs
simulators

artificial separation
from user point of view!
Layering of information definitions

human understanding and interpretation of the information

- **information model**, semantic defined in terms of:
  - entities
  - relations
  - operations

- **data model or data dictionary**
  - identifiers
  - types
  - ranges
Example partial internal information model
<table>
<thead>
<tr>
<th>12 bit Image:</th>
</tr>
</thead>
<tbody>
<tr>
<td>nx: 16 bit unsigned integer</td>
</tr>
<tr>
<td>ny: 16 bit unsigned integer</td>
</tr>
<tr>
<td>pixels[nx][ny]: 16 bit unsigned integers [0..4095]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>16 bit Image:</th>
</tr>
</thead>
<tbody>
<tr>
<td>nx: 16 bit unsigned integer</td>
</tr>
<tr>
<td>ny: 16 bit unsigned integer</td>
</tr>
<tr>
<td>pixels[nx][ny]: 16 bit unsigned integers</td>
</tr>
</tbody>
</table>

The functional view
Example of performance modelling

The functional view

required dose
field size
field map
alignment
procedure

throughput model

internal parameters from realisation:
max v,a
laser power
laser frequency
transmission factor
alignment time

user level throughput
The role of standards

well defined standards and legislation

- HL7
- DICOM
- HIPAA
- EMC
- FDA
- VDE
- ISO 9001

but many thousands of pages

realization consequences

conceptual assumptions

business objectives

application intention?
The functional view = What: externally observable
Exercise Functional View

- Make an overview of functions, performance figures, interfaces and optional features

- identify "most important" (related to CA-views)

- identify "most challenging" (related to CR-views)

- explain why "most important" or "most challenging"

- present in 5 minutes

Goals:

- create awareness of the breadth of the specification

- share the spec with the team
• create a "living" image of the Functional view
Exercise Functional View, second iteration

- Define a typical case, both functions and quantitative
- Create a single page product specification
- Define a worst case, suitable for design exploration and verification
Abstract
This module addresses The Customer Objectives and Application Views:
Abstract
The purpose of the customer objectives view is described. A number of methods or models is given to use in this view: customer key drivers to understand the essentials, value chains and business models to understand the position of the customer and a supplier map to understand the supply side of the customer.
Customer objectives overview

**Customer objectives**

**Key drivers**
- **Safety**
  - Reduce Accident rates
  - Enforce law
  - Improve Emergency Response
- **Effective Flow**
  - Reduce delay due to accident
  - Improve average speed
  - Improve total network throughput
  - Optimise road surface
  - Speed up target groups
  - Anticipate on future traffic condition
- **Smooth Operation**
  - Ensure Traceability
  - Ensure proper alarm handling
  - Ensure system health and fault indication
- **Environment**
  - Reduce emissions

**Derived application drivers**
- Early hazard detection with warning and signalling
- Maintain safe road condition
- Classify and track dangerous goods vehicles
- Detect and warn non compliant vehicles
- Enforce speed compliance
- Enforce red light compliance
- Enforce weight compliance

**Value chain and business models**

**Supplier map**

**Application**
- Fry's
- It's
- Retailers

**Functional**
- Consumers

**Conceptual**
- System Integrators

**Realisation**
- Providers
  - AOL
  - AT&T

**Component and Platform Suppliers**
- Intel
- Microsoft
- Micron
- Philips Semiconductors
- ST
- LG
- TI
- Samsung
- Philips Components
- Philips CE-DN
- Philips CE-PCC

**Competitors or complementors?**
- Sony
- Philips CE-TV
- Loewe
- Nokia
- Boeing
- Microsoft
- Intel
- Philips Semiconductors
- ST
### Example motorway management key drivers

<table>
<thead>
<tr>
<th>Key-drivers</th>
<th>Derived application drivers</th>
<th>Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Safety</td>
<td>Reduce accident rates</td>
<td>Early hazard detection</td>
</tr>
<tr>
<td></td>
<td>Enforce law</td>
<td>with warning and signaling</td>
</tr>
<tr>
<td></td>
<td>Improve emergency response</td>
<td>Maintain safe road condition</td>
</tr>
<tr>
<td>Effective Flow</td>
<td>Reduce delay due to accident</td>
<td>Classify and track dangerous</td>
</tr>
<tr>
<td></td>
<td>Improve average speed</td>
<td>goods vehicles</td>
</tr>
<tr>
<td></td>
<td>Improve total network throughput</td>
<td>Detect and warn noncompliant</td>
</tr>
<tr>
<td></td>
<td>Optimize road surface</td>
<td>vehicles</td>
</tr>
<tr>
<td></td>
<td>Speed up target groups</td>
<td>Enforce speed compliance</td>
</tr>
<tr>
<td></td>
<td>Anticipate on future traffic condition</td>
<td>Enforce red light compliance</td>
</tr>
<tr>
<td>Smooth Operation</td>
<td>Ensure traceability</td>
<td>Enforce weight compliance</td>
</tr>
<tr>
<td></td>
<td>Ensure proper alarm handling</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Ensure system health and fault indication</td>
<td></td>
</tr>
<tr>
<td>Environment</td>
<td>Reduce emissions</td>
<td></td>
</tr>
</tbody>
</table>

*Note: the graph is only partially elaborated for application drivers and requirements*
## Submethod to Link Key Drivers to Requirements

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Define the scope specific.</td>
<td>in terms of stakeholder or market segments</td>
</tr>
<tr>
<td>Acquire and analyze facts</td>
<td>extract facts from the product specification</td>
</tr>
<tr>
<td></td>
<td>and ask why questions about the specification of existing products.</td>
</tr>
<tr>
<td>Build a graph of relations between drivers and requirements</td>
<td>where requirements may have multiple drivers</td>
</tr>
<tr>
<td>by means of brainstorming and discussions</td>
<td></td>
</tr>
<tr>
<td>Obtain feedback</td>
<td>discuss with customers, observe their reactions</td>
</tr>
<tr>
<td>Iterate many times</td>
<td>increased understanding often triggers the move of issues</td>
</tr>
<tr>
<td></td>
<td>from driver to requirement or vice versa and rephrasing</td>
</tr>
</tbody>
</table>
### Key Driver Recommendations

- Limit the number of key-drivers
  - minimal 3, maximal 6

- Don’t leave out the obvious key-drivers
  - for instance the well-known **main function** of the product

- Use short names, recognized by the customer.

- Use market-/customer- specific names, no generic names
  - for instance replace “ease of use” by “minimal number of actions for experienced users”
    - or “efficiency” by “integral cost per patient”

- Do not worry about the exact boundary between Customer Objective and Application
  - create clear **goal means** relations
Example value chain

The customer objectives view

version: 0.3
July 24, 2014
LWAValueChain
competitors or complementers?

Suppliers of appliances, services and content are colour coded.

The customer does business with many suppliers, and has to integrate the products of many suppliers.
The application view

by *Gerrit Muller*  
Buskerud University College

e-mail: gaudisite@gmail.com

www.gaudisite.nl

---

**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 24, 2014

status: preliminary draft

version: 0.2
The application view overview

Customer objectives

Application

Functional
Conceptual
Realisation

Stakeholders and concerns

Context diagrams

Entity relationship models

Dynamic models

The application view
Gerrit Muller

version: 0.2
July 24, 2014

AVoverview
### Stakeholders and concerns MRI scanner

<table>
<thead>
<tr>
<th>Government</th>
<th>Financial Director</th>
<th>Insurance</th>
<th>Administration</th>
</tr>
</thead>
<tbody>
<tr>
<td>cost of care</td>
<td>cash flow</td>
<td>cost of care</td>
<td>patient id invoice</td>
</tr>
<tr>
<td>General Practitioner</td>
<td>ref. physician</td>
<td>radiologist</td>
<td>nurse</td>
</tr>
<tr>
<td>patient</td>
<td>diagnosis</td>
<td>diagnosis</td>
<td>patient ease of work</td>
</tr>
<tr>
<td>Patient</td>
<td>comfort</td>
<td>health</td>
<td></td>
</tr>
<tr>
<td>Family</td>
<td>support</td>
<td></td>
<td></td>
</tr>
<tr>
<td>IT Department</td>
<td>conformance</td>
<td>facility man.</td>
<td>cleaner</td>
</tr>
<tr>
<td>security</td>
<td></td>
<td>space</td>
<td>accessibility</td>
</tr>
<tr>
<td></td>
<td></td>
<td>service supp.</td>
<td>safety</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Legend**
- Administrative
- Clinical
- Patient
- Support

**The application view**

50 Gerrit Muller

version: 0.2
July 24, 2014
AVV stakeholders
Context of motorway management system

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

The application view

version: 0.2
July 24, 2014
AVcontextMotorwayManagement
Example of simple TV application model

channel \(\text{transmits}\) content

- soaps
- movies
- sports
- news

- live

- canned

- described by

- age, sex, violence attributes

- informs

parents

children

The application view

version: 0.2
July 24, 2014
AVsimpleTVmodel
Examples of dynamic models

Flow models:
- People
- Goods
- Information
- Wait for screening
- Wait for diagnosis
- Problem

State diagrams:
- Wait for screening
- Wait for exam
- Exam
- Acute exam
- No problem

Timeline:
- 20:00
- 20:30
- 21:00
- 21:30
- 22:00
- 22:30
- Start movie
- Broadcast
- End movie
- View
- Talk
- Record
- Play
- Pause viewing
- Resume viewing
- Finish conversation

The application view
Gerrit Muller
version: 0.2
July 24, 2014
AVDynamicModels
Productivity and Cost models

The application view

version: 0.2
July 24, 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

URF examination room
Changing room
Waiting room
• Determine stakeholders, key drivers and context of the product.
• Translate these drivers into application drivers and link them to the requirements.
Create a (max) 8 sheet presentation describing the customer objectives and application.
Module Design Side

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

Abstract
This module addresses the Conceptual and Realization Views.
Abstract
The purpose of the conceptual view is described. A number of methods or models is given to use in this view: construction decomposition, functional decomposition, class or object decomposition, other decompositions (power, resources, recycling, maintenance, project management, cost, ...), and related models (performance, behavior, cost, ...); allocation, dependency structure; identify the infrastructure (factoring out shareable implementations), classify the technology in core, key and base technology; integrating concepts (start up, shutdown, safety, exception handling, persistency, resource management,...).
Example construction decomposition simple TV

- **Applications**:
  - View
  - PIP
  - Adjust
  - View TXT

- **Services**:
  - Viewport
  - Menu

- **Toolboxes**:
  - Audio
  - Video
  - TXT
  - Etc.

- **Driver**:
  - Drivers
  - Scheduler

- **Hardware**:
  - Tuner
  - Frame-buffer
  - MPEG
  - DSP

- **Signal Processing Subsystem**
- **Control Subsystem**

- **Domain Specific**
- **Generic**

- **Control Subsystem**
  - CPU
  - RAM
  - etc.

- **Operating System**

- **Networking**
- **File System**

- **Browse**

- **Audio**
- **Video**
- **TXT**

- **Version**: 0.7
  
  **Date**: July 24, 2014
**Characterization of the construction decomposition**

<table>
<thead>
<tr>
<th>management of design</th>
<th>SW example</th>
<th>HW example</th>
</tr>
</thead>
<tbody>
<tr>
<td>unit of creation</td>
<td>file</td>
<td>PCB</td>
</tr>
<tr>
<td>storage</td>
<td></td>
<td>IP cells</td>
</tr>
<tr>
<td>update</td>
<td></td>
<td>IP core</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>unit of aggregation for organisation test release</th>
<th>package module</th>
<th>box</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>IP core</td>
</tr>
<tr>
<td></td>
<td></td>
<td>IC</td>
</tr>
</tbody>
</table>
How; what is the flow of internal activities to realise external functionality?

Some keywords:
- activities
- transformation
- input output
- data flow
- control flow

Multiple functional decompositions are possible and valuable!
How about the <characteristic> of the <component> when performing <function>?

What is the memory usage of the user interface when querying the DB?
Selection factors to improve the question generator

Critical for system performance

Risk planning wise

Least robust part of the design

Suspect part of the design
- experience based
- person based
Addressing planes or lines

The conceptual view

66 Gerrit Muller

version: 0.7
July 24, 2014
CVquestionGeneratorPlanes
Example partial internal information model

- **Patient**
  - Attributes

- **Examination**
  - Attributes

- **Scan**
  - Attributes
    - Pictorial index
  - 3D volume
    - Volume index
  - 2D images
    - Image index

- **Exam Procedures**
  - Attributes

- **Scan Procedures**
  - Attributes
    - Precompiled

- **Work-list**
  - Attributes

Data elements additional to the external information model.
Example process decomposition

The conceptual view

version: 0.7
July 24, 2014
CVprocessDecomposition
The conceptual view

The conceptual view

Execution architecture

dead lines
timing, throughput requirements

functional model
receive
demux
display
process
store

execution architecture
Map
process
task
thread
interrupt handlers

hardware
CPU
DSP
RAM
tuner
drive

repository structure
Applications
play
zap
ot
UI toolkit
menu
processing
DCT
foundation classes
def
set
hardware abstraction
tuner

repository structure

execution architecture issues:
concurrency
scheduling
synchronisation
mutual exclusion
priorities
granularity

The conceptual view

version: 0.7
July 24, 2014
CVexecutionArchitecture
The conceptual view
70 Gerrit Muller

Performance Model

\[ t_{\text{recon}} = t_{\text{filter}}(n_{\text{raw-x}}, n_{\text{raw-y}}) + n_{\text{raw-x}} \cdot (t_{\text{fft}}(n_{\text{raw-y}}) + n_{\text{raw-x}} \cdot (t_{\text{fft}}(n_{\text{raw-x}}) + n \cdot (t_{\text{corrections}}(n_{x}, n_{y}) + t_{\text{control-overhead}}) + t_{\text{col-overhead}}) + t_{\text{row-overhead}}) + t_{\text{fft}}(n) = c_{\text{fft}} \cdot n \cdot \log(n) \]
Safety, Reliability and Security concepts

- containment (limit failure consequences to well defined scope)
- graceful degradation (system parts not affected by failure continue operation)
- dead man switch (human activity required for operation)
- interlock (operation only if hardware conditions are fulfilled)
- detection and tracing of failures
- black box (log) for post mortem analysis
- redundancy
Simplified start up sequence

1. **Discover kernel HW**
2. **Initialise kernel data structures**
3. **Determine next layer**
4. **Load and initialise loader**
5. **Determine loading HW**
6. **Bring in initial state**
7. **Load and initialise firmware**
8. **Configure services**
9. **Allocate resources**
10. **Load, initialise and start services**
11. **Discover kernel HW**
12. **Initialise kernel data structures**
13. **Determine next layer**
14. **Load and initialise loader**
15. **Determine loading HW**
16. **Determine next layer**
17. **Load**
18. **Configure**
19. **Initiate, start**

**Power**

**HW**

**Boot-loader**

**Kernel**

**Services**

**User interface**

**Connect to outside**

**Application**

**Stop in safe sequence**

**Flush ongoing activities**

**Close connections**

**Save persistent data**

**Free resources**

**Stop**

**Start up**

**HW SW interface**

**Shut down**

The conceptual view

Gerrit Muller

version: 0.7
July 24, 2014
CVstartUp
Example work breakdown

The conceptual view

73 Gerrit Muller

version: 0.7
July 24, 2014
CVworkBreakdown

The conceptual view

73 Gerrit Muller

version: 0.7
July 24, 2014
CVworkBreakdown
Core, Key or Base technology

- **Core**, Critical for final performance
- **Key**, Critical for final performance
- **Base**, Critical for final performance

- Own value IP
- Commodity

- Technology life cycle
- Partnering
- Total Product
- make
- outsource
- buy
- refer customer to 3rd party

The conceptual view version: 0.7 July 24, 2014 SSScoreKeyBase
Example integration plan

The conceptual view

July 24, 2014

version: 0.7

CVIntegrationPlan
The realization view

by Gerrit Muller        Buskerud University College

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

Abstract
The realization view looks at the actual technologies used and the actual implementation. Methods used here are logarithmic views, micro-benchmarks and budgets.

Analysis methods with respect to safety, reliability and security provide a link back to the functional and conceptual views.
The realization view

Gerrit Muller
Example of a memory budget

<table>
<thead>
<tr>
<th>memory budget in Mbytes</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>application SW total</td>
<td>13.4</td>
<td>12.6</td>
<td>35.0</td>
<td>61.0</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>total</td>
<td></td>
<td></td>
<td></td>
<td>74.0</td>
</tr>
</tbody>
</table>
Actual timing on logarithmic scale

- Application needs
  - Light travels 1 cm
- Time scales:
  - $10^{-12}$ ps
  - $10^{-9}$ ns
  - $10^{-6}$ s
  - $10^{-3}$ ms
  - 1 s

- From low to high level processing times

- FO4 inverter delay
- Cycle 2 GHz CPU
- DRAM cycle time
- DRAM latency
- 1 byte transfer fast ethernet
- 1 package transfer fast ethernet
- Disk seek
- Appliance message exchange
- Appliance level function response
- Human 1st irritation threshold
- Human 2nd irritation threshold
- Human eye-hand co-ordination
- Human reaction time
- 100 Hz TV frame
- 100 Hz video line
- 100 Hz video pixel time
- appl level message exchange
- appl level function response
- appl level network message exchange

The realization view

Gerrit Muller

version: 0.1
July 24, 2014
RVtimeAxis
Typical micro benchmarks for timing aspects

<table>
<thead>
<tr>
<th></th>
<th>infrequent operations, often time-intensive</th>
<th>often repeated operations</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>database</strong></td>
<td>start session</td>
<td>perform transaction</td>
</tr>
<tr>
<td></td>
<td>finish session</td>
<td>query</td>
</tr>
<tr>
<td><strong>network, I/O</strong></td>
<td>open connection</td>
<td>transfer data</td>
</tr>
<tr>
<td></td>
<td>close connection</td>
<td></td>
</tr>
<tr>
<td><strong>high level construction</strong></td>
<td>component creation</td>
<td>method invocation</td>
</tr>
<tr>
<td></td>
<td>component destruction</td>
<td>same scope</td>
</tr>
<tr>
<td></td>
<td></td>
<td>other context</td>
</tr>
<tr>
<td><strong>low level construction</strong></td>
<td>object creation</td>
<td>method invocation</td>
</tr>
<tr>
<td></td>
<td>object destruction</td>
<td></td>
</tr>
<tr>
<td><strong>basic programming</strong></td>
<td>memory allocation</td>
<td>function call</td>
</tr>
<tr>
<td></td>
<td>memory free</td>
<td>loop overhead</td>
</tr>
<tr>
<td></td>
<td></td>
<td>basic operations (add, mul, load, store)</td>
</tr>
<tr>
<td><strong>OS</strong></td>
<td>task, thread creation</td>
<td>task switch</td>
</tr>
<tr>
<td></td>
<td></td>
<td>interrupt response</td>
</tr>
<tr>
<td><strong>HW</strong></td>
<td>power up, power down</td>
<td>cache flush</td>
</tr>
<tr>
<td></td>
<td>boot</td>
<td>low level data transfer</td>
</tr>
</tbody>
</table>
The transfer time as function of blocksize

- The graph shows the transfer time as a function of blocksize.
- The y-axis represents time, and the x-axis represents block size.
- There are two lines on the graph:
  - A magenta line labeled "worst case".
  - A blue line labeled "optimal block-size".
- The vertical arrow labeled "rate ^{-1}" points upwards from the blue line.
- The horizontal arrow labeled "t_{overhead}" points to the left from the blue line.

The realization view

Gerrit Muller

version: 0.1
July 24, 2014
RVparametrizedTransferRate
Performance evaluation

\[
\begin{align*}
\text{t}_{\text{recon}} &= \text{t}_{\text{filter}}(n_{\text{raw-x}}, n_{\text{raw-y}}) + n_{\text{raw-x}} \cdot (\text{t}_{\text{fft}}(n_{\text{raw-y}}) + \text{t}_{\text{col-overhead}}) + n_{y} \cdot (\text{t}_{\text{fft}}(n_{\text{raw-x}}) + \text{t}_{\text{row-overhead}}) + \text{t}_{\text{corrections}}(n_{x}, n_{y}) + \text{t}_{\text{read I/O}} + \text{t}_{\text{transpose}} + \text{t}_{\text{write I/O}} + \text{t}_{\text{control - overhead}} \\
\text{t}_{\text{fft}}(n) &= c_{\text{fft}} \cdot n \cdot \log(n)
\end{align*}
\]

<table>
<thead>
<tr>
<th>overhead</th>
<th>focus on overhead reduction is more important than faster algorithms this is not an excuse for sloppy algorithms</th>
</tr>
</thead>
<tbody>
<tr>
<td>correction computations</td>
<td>number crunching</td>
</tr>
<tr>
<td>row overhead</td>
<td></td>
</tr>
<tr>
<td>FFT computations</td>
<td></td>
</tr>
<tr>
<td>column overhead</td>
<td></td>
</tr>
<tr>
<td>FFT crunching</td>
<td></td>
</tr>
<tr>
<td>overhead</td>
<td></td>
</tr>
<tr>
<td>transpose</td>
<td></td>
</tr>
<tr>
<td>malloc, free</td>
<td></td>
</tr>
<tr>
<td>write I/O</td>
<td></td>
</tr>
<tr>
<td>bookkeeping</td>
<td></td>
</tr>
<tr>
<td>read I/O</td>
<td></td>
</tr>
<tr>
<td>write I/O</td>
<td></td>
</tr>
<tr>
<td>transpose</td>
<td></td>
</tr>
<tr>
<td>overhead</td>
<td></td>
</tr>
<tr>
<td>FFT computations</td>
<td></td>
</tr>
</tbody>
</table>

The realization view

82 Gerrit Muller
Performance Cost, choice based on sales value

- 5400 rpm
- 7200 rpm
- 7200 rpm, 8 MB buffer
- pentium4

The realization view
84 Gerrit Muller

source: http://www.mpcomp.com/
September 5, 2002
Performance Cost, effort consequences

The realization view
85  Gerrit Muller
But many many other considerations

- processing
- performance
- storage
- capacity
- user value
- effort
- time to market
- risk
- future evolution
- system context

The realization view
Safety, Reliability and Security analysis methods

<table>
<thead>
<tr>
<th>Safety hazard analysis</th>
<th>potential hazards</th>
<th>probability severity</th>
<th>measures</th>
</tr>
</thead>
<tbody>
<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>
</tbody>
</table>

(systematic) brainstorm → analysis and assessment → improve design
Make a first design:

- decomposition in functions
- decomposition in building blocks
- budgets for most important quality requirements
• Make a design:
  • that covers the most critical design aspects
  • that fulfills the most important and valuable customer needs

• Make a presentation of the design of maximal 8 sheets.
Module Qualities

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

Abstract
This module addresses the Qualities.
Abstract
Many stakeholder concerns can be specified in terms of qualities. These qualities can be viewed from all 5 “CAF CR” viewpoints. In this way qualities can be used to relate the views to each other.

The meaning of qualities for the different views is described. A checklist of qualities is provided as a means for architecting. All qualities in the checklist are described briefly.
Quality needles as generic integrating concepts

Customer objectives
Application
Functional
Conceptual
Realization

usability
safety
evolvability

Qualities as Integrating Needles
version: 1.3
July 24, 2014
QNneedles
<table>
<thead>
<tr>
<th>Customer objectives</th>
<th>Application</th>
<th>Functional</th>
<th>Conceptual</th>
<th>Realization</th>
</tr>
</thead>
<tbody>
<tr>
<td>sensitive information</td>
<td>selection</td>
<td>functions</td>
<td>cryptography</td>
<td>specific algorithms</td>
</tr>
<tr>
<td>trusted</td>
<td>classification</td>
<td>for</td>
<td>firewall</td>
<td>algorithms</td>
</tr>
<tr>
<td></td>
<td>people</td>
<td>administration</td>
<td>security zones</td>
<td>interfaces</td>
</tr>
<tr>
<td></td>
<td>information</td>
<td>authentication</td>
<td>authentication</td>
<td>libraries</td>
</tr>
<tr>
<td></td>
<td>authentication</td>
<td>intrusion detection</td>
<td>registry</td>
<td>servers</td>
</tr>
<tr>
<td></td>
<td>badges</td>
<td>logging</td>
<td>logging</td>
<td>storage</td>
</tr>
<tr>
<td></td>
<td>passwords</td>
<td>quantification</td>
<td></td>
<td>protocols</td>
</tr>
</tbody>
</table>

**desired characteristics, specifications & mechanisms**

<table>
<thead>
<tr>
<th></th>
<th>social contacts</th>
<th>missing</th>
<th>holes between</th>
<th>bugs</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>open passwords</td>
<td>functionality</td>
<td>concepts</td>
<td>buffer overflow</td>
</tr>
<tr>
<td></td>
<td>blackmail</td>
<td>wrong</td>
<td></td>
<td>non encrypted</td>
</tr>
<tr>
<td></td>
<td>burglary</td>
<td>quantification</td>
<td></td>
<td>storage</td>
</tr>
<tr>
<td></td>
<td>fraud</td>
<td></td>
<td></td>
<td>poor exception</td>
</tr>
<tr>
<td></td>
<td>unworkable procedures</td>
<td></td>
<td></td>
<td>handling</td>
</tr>
</tbody>
</table>

**threats**
Quality Checklist

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

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

**efficient**
- resource utilization
- cost of ownership

**effective**
- throughput or productivity

**consistent**
- reproducibility
- predictability

**interoperable**
- connectivity
- 3rd party extendible

**liable**
- liability
- testability
- traceability
- standards compliance

**serviceable**
- serviceability
- configurability
- installability

**future proof**
- evolvability
- portability
- upgradeability
- extendibility
- maintainability

**logistics friendly**
- manufacturability
- logistics flexibility
- lead time

**ecological**
- ecological footprint
- contamination
- noise
- disposability

**down to earth attributes**
- cost price
- power consumption
- consumption rate
- (water, air, chemicals, et cetera)
- size, weight
- accuracy

Qualities as Integrating Needles

Gerrit Muller

version: 1.3
July 24, 2014
QNchecklist
Exercise Qualities

• Determine most important qualities.

• Annotate 3 most important qualities in every CAFCR view
Abstract
This module addresses Scenarios, Story Telling and Use Cases. Scenarios are used to cope with multiple alternatives for specification or design. Story telling is a means to explore customer needs and as a means for communication. Use Cases are used to analyze the design for specific circumstances.
## Goal of this Module

- Be able to apply story telling technique.
- Be able to use scenario analysis.
- Be able to use use-cases for design.

## Content of this Module

- Format and criteria for stories
- Elements of scenarios
- Role of scenarios in decision making
- Quantified use cases

## Exercise

Create a story and translate story via use cases in design
Abstract

A story is an easily accessible story or narrative to make an application live. A good story is highly specific and articulated entirely in the problem domain: the native world of the users. An important function of a story is to enable specific (quantified, relevant, explicit) discussions.
From story to design

What does Customer need in Product and Why?

Customer
  What
  objectives

Customer
  How

Product
  What

Product
  How

Application

Functional

Conceptual

Realization

Story

market vision

a priori solution knowledge

analyze design
case

analyze design

design
A day in the life of Bob

bla blah bla, rabarber music
bla bla composer bla bla
qwwwety30 zeps.
nja nja njet njippie est quo
vado? Fijir jaleski bla bla
bla bree tfly sgs fhrg
mrm bas engel heeft een
interessant excusus, lex stelt
voor om vanavond door te
werken.

In the middle of the night he is awake and decides to change the world forever.

The next hour the great event takes place:

This brilliant invention will change the world forever because it is so unique and valuable that nobody believes the feasibility. It is great and WOW at the same time, highly exciting.

Vtables are seen as the solution for an indirection problem. The invention of Bob will obsolete all of this in one incredible move, which will make him famous forever.

He opens his PDA, logs in and enters his private secure unique non trivial password, followed by a thorough authentication. The PDA asks for the fingerprint of this little left toe and to pronounce the word shit. After passing this test Bob can continue.
Points of attention

- purpose
- scope
- viewpoint, stakeholders
- visualization
- size (max 1 A4)
- recursive decomposition, refinement
Criteria for a good story

- accessible, understandable
  "Do you see it in front of you?"

- valuable, appealing
  attractive, important
  "Are customers queuing up for this?"

- critical, challenging
  "What is difficult in the realization?"
  "What do you learn w.r.t. the design?"

- frequent, no exceptional niche
  "Does it add significantly to the bottom line?"

- specific
  names, ages, amounts, durations, titles, ...
Betty is a 70-year-old woman who lives in Eindhoven. Three years ago her husband passed away and since then she lives in a home for the elderly. Her 2 children, Angela and Robert, come and visit her every weekend, often with Betty’s grandchildren Ashley and Christopher. As so many women of her age, Betty is reluctant to touch anything that has a technical appearance. She knows how to operate her television, but a VCR or even a DVD player is way to complex.

When Betty turned 60, she stopped working in a sewing studio. Her work in this noisy environment made her hard-of-hearing with a hearing-loss of 70dB around 2kHz. The rest of the frequency spectrum shows a loss of about 45dB. This is why she had problems understanding her grandchildren and why her children urged her to apply for hearing aids two years ago. Her technophobia (and her first hints or arthritis) inhibit her to change her hearing aids’ batteries. Fortunately her children can do this every weekend.

This Wednesday Betty visits the weekly Bingo afternoon in the meetingplace of the old-folk’s home. It’s summer now and the tables are outside. With all those people there it’s a lot of chatter and babble. Two years ago Betty would never go to the bingo: “I cannot hear a thing when everyone babbles and clatters with the coffee cups. How can I hear the winning numbers?!”. Now that she has her new digital hearing instruments, even in the bingo cacophony, she can understand everyone she looks at. Her social life has improved a lot and she even won the bingo a few times.

That same night, together with her friend Janet, she attends Mozart’s opera The Magic Flute. Two years earlier this would have been one big low rumbly mess, but now she even hears the sparkling high piccolos. Her other friend Carol never joins their visits to the theaters. Carol also has hearing aids, however hers only “work well” in normal conversations. “When I hear music it’s as if a butcher’s knife cuts through my head. It’s way too sharp!”. So Carol prefers to take her hearing aids out, missing most of the fun. Betty is so happy that her hearing instruments simply know where they are and adapt to their environment.
## Value and Challenges in this story

### Value proposition in this story:
- **quality of life:** active participation in different social settings
- **usability for nontechnical elderly people:** "intelligent" system is simple to use
- **loading of batteries**

### Challenges in this story:
- **Intelligent hearing instrument**
- **Battery life:** at least 1 week
- **No buttons or other fancy user interface on the hearing instrument,** other than a robust On/Off method
- **The user does not want a technical device but a solution for a problem**
- **Instrument can be adapted to the hearing loss of the user**
- **Directional sensitivity** (to prevent the so-called cocktail party effect)
- **Recognition of sound environments and automatic adaptation** (adaptive filtering)

---

Source: Roland Mathijssen, Embedded Systems Institute, Eindhoven
Scenario How To

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

Abstract
Good designers keep multiple alternatives open in parallel. This improves the specification and design quality. Scenarios can be used to cope with these alternatives and as a means for communication with stakeholders.
content of this presentation

Decision making

Multiple propositions

Scenarios
Decision Making Process

1. Problem understanding
2. Analysis
3. Decision
4. Monitor, verify, validate

- vague problem statement
- insufficient data
- no satisfying solution
- invalidated solution
- conflicting other decision
- invalid solution
Flow from problem to solution

1. Problem understanding by
   exploration and simple models

2. Analysis by
   + exploring multiple propositions (specification + design proposals)
   + exploring decision criteria (by evaluation of proposition feedback)
   + assessment of propositions against criteria

3. Decision by
   + review and agree on analysis
   + communicate and document

4. Monitor, verify, validate by
   + measurements and testing
   + assessment of other decisions

vague problem statement

conflicting other decision

insufficient data

no satisfying solution

invalidated solution
Example of Multiple Propositions

<table>
<thead>
<tr>
<th>Proposition 1</th>
<th>Proposition 2</th>
<th>Proposition 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Throughput</td>
<td>20 p/m</td>
<td>25 p/m</td>
</tr>
<tr>
<td>Cost</td>
<td>5 k$</td>
<td>7 k$</td>
</tr>
<tr>
<td>Safety</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Performance</td>
<td>high-speed</td>
<td>high-speed</td>
</tr>
<tr>
<td></td>
<td>moves</td>
<td>moves</td>
</tr>
<tr>
<td></td>
<td>additional</td>
<td></td>
</tr>
<tr>
<td>Pipeline</td>
<td>pipelining</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Latency</td>
<td>350 ns</td>
<td>300 ns</td>
</tr>
<tr>
<td>Speed</td>
<td>9 m/s</td>
<td>10 m/s</td>
</tr>
</tbody>
</table>

**low cost and performance 1**

**low cost and performance 2**

**high cost and performance**
Recursive and concurrent application of flow

1. Problem understanding
2. Analysis
3. Decision
4. Monitor, verify, validate

system level

1. Problem understanding
2. Analysis
3. Decision
4. Monitor, verify, validate

subsystem level

1. Problem understanding
2. Analysis
3. Decision
4. Monitor, verify, validate

component level

1. Problem understanding
2. Analysis
3. Decision
4. Monitor, verify, validate

atomic level

legend

decision flow

analysis flow
Graph of Decisions and Alternatives

Legend:
- Past decision
- Most probable decision
- Potential alternative
- Less probable alternative

Communication
Scope
Scope of architect's considerations

Now
Time
Different Types of Decisions

Understanding Why
Describing What
Guiding How

- basic principles
- requirements
- architecture rules implementation choices f.i. technology
Elements of a Scenario

scenario: <clear title>

story

key specification and design decisions

case

design

scenario: <clear title>

story

key specification and design decisions

case

design

scenario: <clear title>

story

key specification and design decisions

case

design
Summary of Scenarios

Exploration and analysis require multiple propositions.

Architects continuously work with multiple alternatives.

Scenarios have a clear title, story, use case and design.

Scenarios are differentiated by key specifications and design decisions.
Abstract
Use cases are frequently used in Software Engineering. Use cases support specification and facilitate design, analysis, verification and testing. Many designers, unfortunately, apply use cases in a rather limited way. This presentation provides recommendations for effective use cases.
Why Use Cases?

Supports or is part of specification
by providing specific data in user perspective

Facilitates analysis and design

Facilitates verification and testing
Example Time Shift recording

<table>
<thead>
<tr>
<th>Time</th>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>20:00</td>
<td>start movie</td>
</tr>
<tr>
<td>21:00</td>
<td>broadcast</td>
</tr>
<tr>
<td>22:00</td>
<td>record</td>
</tr>
<tr>
<td>23:00</td>
<td>end movie</td>
</tr>
<tr>
<td></td>
<td>phone rings</td>
</tr>
<tr>
<td></td>
<td>pause viewing</td>
</tr>
<tr>
<td></td>
<td>finish conversation</td>
</tr>
<tr>
<td></td>
<td>resume viewing</td>
</tr>
</tbody>
</table>

Start movie at 20:00, broadcast at 21:00, record at 22:00, end movie at 23:00. Timeline includes events such as phone ringing, pausing viewing, finishing conversation, and resuming viewing.
Construction limits intrude in User Experience

- number of tuners
- number of simultaneous streams (recording and playing)
- amount of available storage
- management strategy of storage space
What if?

20:00  21:00  22:00  23:00

start movie    broadcast    end movie

view

talk

phone rings   pause viewing  

record

play

view

play

1. programmed recording of other station

2. very long phone call

3. Dad zaps

phone rings

pause viewing

finish conversation

resume viewing
Content of a Use Case

**use case**

- user or system specified functionality
- behavior
- interfaces
- qualities (NFR's)

**Input data**
- format
- size
- content

**Output data**
- format
- size
- content

**Context**
- interaction

**Subsystem or component**
Example personal video recorder use case contents

<table>
<thead>
<tr>
<th>Typical use case(s)</th>
<th>Worst case, exceptional, or change use case(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Interaction flow (functional aspects)</strong></td>
<td><strong>Functional</strong></td>
</tr>
<tr>
<td>select movie via directory</td>
<td>multiple inputs at the same time</td>
</tr>
<tr>
<td>start movie</td>
<td>extreme long movie</td>
</tr>
<tr>
<td>be able to pause or stop</td>
<td>directory behaviour in case of</td>
</tr>
<tr>
<td>be able to skip forward or backward</td>
<td>extreme many short movies</td>
</tr>
<tr>
<td>set recording quality</td>
<td></td>
</tr>
<tr>
<td><strong>Performance and other qualities</strong> (non-functional aspects)</td>
<td><strong>Non-functional</strong></td>
</tr>
<tr>
<td>response times for start / stop</td>
<td>response time with multiple inputs</td>
</tr>
<tr>
<td>response times for directory browsing</td>
<td>image quality with multiple inputs</td>
</tr>
<tr>
<td>end-of-movie behaviour</td>
<td>insufficient free space</td>
</tr>
<tr>
<td>relation recording quality and storage</td>
<td>response time with many directory entries</td>
</tr>
<tr>
<td></td>
<td>replay quality while HQ recording</td>
</tr>
</tbody>
</table>
Example of Quantification of Typical Use Case

3 examination rooms connected to 1 medical imaging workstation + printer

examination room: average 4 interleaved examinations / hour

image production: 20 1024 2 8 bit images per examination

film production: 3 films of 4k*5k pixels each

high quality output (bi-cubic interpolation)
Timing of this Use Case

Use Case How To
123 Gerrit Muller

version: 0.1
July 24, 2014
MICAF typical Timing
Recommendations for working with use cases

+ combine related functions in one use case
- do not make a separate use case for every function
+ include non-functional requirements in the use cases

+ minimise the amount of required worst case and exceptional use cases
- excessive amounts of use cases propagate to excessive implementation efforts
+ reduce the amount of these use cases in steps
- a few well chosen worst case use cases simplifies the design
1. Create a story
   • use the criteria
2. Transform the story into a case
   • functional, as well as quantitative
3. Perform a short design exploration
   • based on the case.
4. Improve the story
   • first iteration based on feedback from case and design.
   • Use time boxes to ensure that you make all the indicated steps.
+ stories make discussions much more specific
+ implicit assumptions are identified

~ creating relevant stories is far from trivial

- too much fun

starting point for generalization: specification and design
Conclusions

Stories help to focus early design discussions

Scenarios help to cope with multiple alternatives

Use cases address integral use: functional and quantitative

Techniques, Models, Heuristics of this module

Story telling, criterias

Scenarios

Quantified use cases

Worst case, exceptional and change use cases
Abstract
This module addresses *Threads of Reasoning* as a means to connect business and operational needs to design and technology choices.
### goal of this module

Be able to relate *Customer* and *Operational* objectives to design and technology choices.

Be able to provide rationale for design decisions.

### content of this module

Key driver method and recommendations

Threads of reasoning approach

Example in Health Care domain

### exercise

Key driver graph
Abstract
The notion of "business key drivers" is introduced and a method is described to link these key drivers to the product specification.
Example Motorway Management Analysis

Key-drivers
- Safety
  - Reduce accident rates
  - Enforce law
  - Improve emergency response
- Effective Flow
  - Reduce delay due to accident
  - Improve average speed
  - Improve total network throughput
  - Optimize road surface
  - Speed up target groups
  - Anticipate on future traffic condition
- Smooth Operation
  - Ensure traceability
  - Ensure proper alarm handling
  - Ensure system health and fault indication
- Environment
  - Reduce emissions

Derived application drivers
- Early hazard detection with warning and signaling
- Maintain safe road condition
  - Classify and track dangerous goods vehicles
  - Detect and warn noncompliant vehicles
  - Enforce speed compliance
  - Enforce red light compliance
  - Enforce weight compliance

Requirements
- Automatic upstream accident detection
- Weather condition dependent control
- Traffic condition dependent speed control
- Cameras
  - Deicing
  - Traffic speed and density measurement

Note: the graph is only partially elaborated for application drivers and requirements
## Method to create Key Driver Graph

- Define the scope specific. in terms of stakeholder or market segments
- Acquire and analyze facts extract facts from the product specification and ask why questions about the specification of existing products.
- Build a graph of relations between drivers and requirements by means of brainstorming and discussions where requirements may have multiple drivers.
- Obtain feedback discuss with customers, observe their reactions.
- Iterate many times increased understanding often triggers the move of issues from driver to requirement or vice versa and rephrasing.
## Recommendation for the Definition of Key Drivers

- **Limit the number of key-drivers**
  - minimal 3, maximal 6

- **Don’t leave out the obvious key-drivers**
  - for instance the well-known main function of the product

- **Use short names, recognized by the customer.**

- **Use market-/customer- specific names, no generic names**
  - for instance replace “ease of use” by “minimal number of actions for experienced users”, or “efficiency” by “integral cost per patient”

- **Do not worry about the exact boundary between Customer Objective and Application**
  - create clear goal means relations
Transformation of Key Drivers into Requirements

- **Customer**
  - What: Customer objectives

- **Derived Application Drivers**
  - Derived Application

- **Product**
  - What: Functional

- **Key (Customer) Drivers**
  - Goal: means
    - May be skipped or
      - Articulated by several intermediate steps

**Key Drivers How To**

134 Gerrit Muller

version: 0.2
July 24, 2014
REQfromDriverToRequirement
Abstract
A method of reasoning is described, which addresses cross-cutting issues. The basis is fast iteration in the problem and solution space.

A thread of reasoning is a set of highly relevant related issues, which are addressed by articulating the problem in terms of tension and analyzing it in the CAFCR framework.
Overview of the reasoning approach

1. select starting point:
   ! actual dominant need or problem

2. create insight:
   + submethod in one of CAFCR views
   + qualities checklist

3. deepen insight via facts:
   + via tests, measurements, simulations
   + story telling

4. broaden insight via questions:
   + why
   + what
   + how

5. define and extend the thread:
   ? what is the most important / valuable
   ? what is the most critical / sensitive
   ! look for the conflicts and tension

continuously

consolidate in simple models
communicate to stakeholders
refactor documentation
From starting point to insight

step 1 starting point

slow response

C - Customer objectives
A - Application
F - Functional
C - Conceptual
R - Realization
Creating Insight

step 2 creating insight

C
A
F
C
R

Customer objectives
Application
Functional
Conceptual
Realization

performance
response
time model

Threads of Reasoning
138  Gerrit Muller

version: 2.4
July 24, 2014
TORcreatingInsight
Deepening Insight

- Customer objectives
- Application
- Functional
- Conceptual
- Realization

- specific needs
- story
- simulations, test, measurements
- specific facts

step 3 deepening insight

version: 2.4
July 24, 2014
TORdeepeningInsight

Threads of Reasoning
139 Gerrit Muller

Embedded Systems
Institute
step 4 broadening insight

- Customer objectives
- Application
- Functional
- Conceptual
- Realization

why? - what? - how?

why? - what? - how?
Problem identification and articulation

important
critical
valuable
difficult
valuable

need and problem selection criterions

Customer objectives
Application
Functional
Conceptual
Realization

definition in terms of tension

throughput
high performance sensor
cost
high speed moves
safety
Iteration during the analysis

- Solution
- Problem
- Objective criteria
- Objective ranking
- Intuitive ranking
- Architect intuition
- Detect mismatch
- Improve solution understanding
- Adjust intuition
- Improve criteria

Threads of Reasoning

Version: 2.4
July 24, 2014
TORanalysisIteration
Threads of related issues

Elements:
- **Customer objectives**
- **Application**
- **Functional**
- **Conceptual**
- **Realization**

Connections between elements show the relationships and dependencies.
Documentation and communication structure

C   A   F   C   R
Customer objectives  Application  Functional  Conceptual  Realization

IQ  key drivers  context  case  zap  IQ spec  functional model

IQ  spec  store  target  pipeline design

CoO  response time  cost budget  time budget

perfor-
mance  processing library  micro benchmarks

cost
Threads of reasoning illustrated by medical imaging case

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

Abstract
The medical imaging workstation case is introduced. An architecting method based on the CAFCR viewpoints is explained, consisting of 4 elements:

- the CAFCR viewpoints
- qualities as integrating needles
- story telling
- threads of reasoning

A thread of reasoning is build up in steps, based on this case. The underlying reasoning is explained.
Easyvision serving three URF examination rooms

URF-systems

EasyVision: Medical Imaging Workstation

typical clinical image (intestines)
X-ray rooms from examination to reading around 1990

Threads of reasoning illustrated by medical imaging case
147   Gerrit Muller

version: 0
July 24, 2014
XRayRoomsOld
X-ray rooms with Easyvision applied as printserver

Examination Room
- X-ray source
- detector

Control Room
- console

Corridor or closet
- printer

Reading Room
- light box

Threads of reasoning illustrated by medical imaging case

version: 0
July 24, 2014
XRrayRoomsPlusPrintServer
Comparison screen copy versus optimized film

old: screen copy

new: SW formatting

20 to 50% less film needed
Challenges for product creation

- **Print throughput**
  - ca 1 film / minute
  - film = 4k*5k pixels

- **View response time**
  - subsecond retrieve
  - screen = 1k*1k

- **Image quality**
  - image processing

- **Product policy:**
  - standard HW
  - SW "only"
  - 40 MHz CPU
  - 64 MByte memory
  - 10 MBit/s ethernet
  - 1 GByte disk

Threads of reasoning illustrated by medical imaging case
150 Gerrit Muller
Top level decomposition

SW
- application
  - framework, libraries
    - operating system

HW
- optical disc
- workstation
  - desk, cabinet
- network
- laser printer
- remote control

Legend:
- tools
- make
- buy
CAFCR viewpoints

What does Customer need in Product and Why?

Customer objectives
Application
Functional
Conceptual
Realization

drives, justifies, needs
enables, supports
Quality needles as generic integrating concepts

Threads of reasoning illustrated by medical imaging case

Gerrit Muller
From story to design

What does Customer need in Product and Why?

Customer
  What
  objectives

Customer
  How

Product
  What

Product
  How

Customer

Application

Functional

Conceptual

Realization

market vision

story

analyze design

case

analyze design

design

a priori solution knowledge
Chronology of Easyvision RF R1 development

1991
- basic application
- toolboxes
- 100 kloc
- interactive viewing

1992
- performance problems
- IQ problems
- Easyvision RF integrated product
- 360 kloc
- print server + communication + interactive viewing

Marketing opinion: "All the functionality is available, we only have to provide a clinical UI"
Thread of reasoning based on efficiency-quality tension

- **Customer objectives**
  - time efficient
  - diagnostic quality
  - safety (liability)

- **Specification issues**
  - system response
  - system throughput

- **Concepts**
  - resource management
  - processor, memory
  - internal logistics
  - concurrency, processes
  - image processing
  - algorithms

- **Threads of reasoning illustrated by medical imaging case**

Threads of reasoning illustrated by medical imaging case
156       Gerrit Muller

version: 0
July 24, 2014
MITORthread
Technology innovations

- standard UNIX based workstation
- full SW implementation, more flexible
- object oriented design and implementation (Objective-C)
- graphical User Interface, with windows, mouse etcetera
- call back scheduling, fine-grained notification
- data base engine, fast, reliable and robust
- extensive set of toolboxes
- property based configuration
- multiple coordinate spaces

Performance

Cost
Introvert view: cost and impact of new technologies
Memory usage half way R1

Threads of reasoning illustrated by medical imaging case

159  Gerrit Muller

version: 0
July 24, 2014
MSmemoryZeroMeasurement
Solution of memory performance problem

- Threads of reasoning illustrated by medical imaging case
- 200 MB measured
- 74 MB budget
- Code
- Bulk data
- Data
- OS

- Anti-fragmenting
- Budget based
- Awareness, measurement
- DLLs
- Tuning

- Version: 0
- July 24, 2014
- MSmemoryUsageReduction
Visualization memory use per process

Threads of reasoning illustrated by medical imaging case

161 Gerrit Muller
Typical case URF examination

3 examination rooms connected to

1 medical imaging workstation + printer

examination room: average 4 interleaved examinations / hour

image production: 20 1024 ² 8 bit images per examination

film production: 3 films of 4k*5k pixels each

high quality output (bi-cubic interpolation)
Thread of reasoning; phase 2

**Philips operational view**
(manufacturing, service, sales)

How to measure memory, how much is needed?
from introvert to extrovert
Radiologist workspots and activities

 Threads of reasoning illustrated by medical imaging case

supervision of the examination
view and diagnose, dictate report
verify and authorise report

activities of the radiologist
Diagnosis in tens of seconds

- Films loaded by clinical personnel during the day
- Looks at images
- Moves head forward / backward
- Moves head or eyes left/right/up/down
- Zooms in
- Moves head or eyes left/right/up/down
- Overview
- Mumbles a few Latin words or clinical codes in recorder
- Presses next button
- New films
- Old films
- Report
- Tens of seconds
- Image selection, panning
- Light-box
- Auto-loader
Rendered images at different destinations

**Screen:**
- low resolution
- fast response

**Film:**
- high resolution
- high throughput

**Network:**
- medium resolution
- high throughput
SW Process structure 1991

Threads of reasoning illustrated by medical imaging case

167 Gerrit Muller

version: 0
July 24, 2014
MITORsoftwareProcess1991
Threads of reasoning illustrated by medical imaging case

168   Gerrit Muller

version: 0
July 24, 2014
MITORswLayers1991
Print server is based on banding

Threads of reasoning illustrated by medical imaging case

version: 0
July 24, 2014
MICVbanding
Server CPU load

Threads of reasoning illustrated by medical imaging case

version: 0
July 24, 2014
MICVserverCPUload
Thread of reasoning; phase 3

Radiologists diagnose from film, throughput is important. Extrovert view shows conceptual and realization gaps!

Philips operational view (manufacturing, service, sales)
Image quality and safety problem

Threads of reasoning illustrated by medical imaging case

version: 0
July 24, 2014
MITORfalseContouring
Presentation pipeline for X-ray images

```
image from database

spatial enhancement

interpolate

Look up table
inverted contrast / brightness

graphics merge

colour LUT

monitor

bi-linear bi-cubic

legend

SW
HW

output

input

brightness

contrast

```

Threads of reasoning illustrated by medical imaging case

173   Gerrit Muller
What you see at one work-spot is what you get at another work-spot.

X-ray system image generation presentation

Easyvision application processing presentation

3rd party workstation

Monitor film network, storage

Version: 0
July 24, 2014
MICVwysiwyg
Safety problem

for user readability the font-size was determined "intelligently"; causing a dangerous mismatch between text and image

URF monitor output: fixed size letters at fixed grid

EV output: scaleable fonts in graphics overlay

Threads of reasoning illustrated by medical imaging case

MITORfontScaling
Thread of reasoning; phase 4

Philips operational view
(manufacturing, service, sales)

from extrovert diagnostic quality, via image quality, algorithms and load, to extrovert throughput
Threads of reasoning illustrated by medical imaging case

177 Gerrit Muller

version: 0
July 24, 2014
MITORthread50

cost revisited in context of clinical needs and realization constraints; note: original threads are significantly simplified

Philips operational view (manufacturing, service, sales)
Overview of architecting method

**Method outline**

**Framework**
- Customer objectives
- Application
- Functional
- Conceptual
- Realization

**Submethods**
- + key drivers
- + value chain
- + business models
- + supplier map
- + stakeholders
- + context diagram
- + entity relationship models
- + dynamic models
- + use case
- + commercial, logistics decompositions
- + mapping technical functions and several more
- + construction decomposition
- + functional decomposition
- + information model and many more
- + budget
- + benchmarking
- + performance analysis
- + safety analysis and many more

**Integration via qualities**
- Safety
- Performance

**Explore specific details**
- A priori solution know-how
- Market vision
- Story
- Analyse design
- Use case
- Analyse design
- Detailed design

**Reasoning**
- Dual path
- Diagnostic: RoM: Moores
- Memory budget: standard workstation
- Profit margin: standard workstation
- Render engine: pixel depth
- Processing: pixel depth
- Memory budget: standard workstation
- RoM: Moores
- Profit margin: standard workstation

Threads of reasoning illustrated by medical imaging case

178 Gerrit Muller
• Make a key driver graph

Use the key driver approach

Take the recommendations into account
Reflection on Exercise

+ Key drivers put requirements in broader perspective
+ Discussion creates shared understanding
~ The graph needs external feedback
- Are the key drivers really from the customer?
- Are the key drivers sharp enough?
Conclusions

Key Driver graph connects customer objectives to system requirements

Threads of Reasoning connects Customer and Operational Objectives to design and technology choices

The overview is maintained by focusing on valuable, important, critical or sensitive aspect; Look for tensions!

Techniques, Models, Heuristics of this module

Key driver graph
Thread of reasoning
Why, What and How
Tensions
The Boderc project contributed to Key drivers and Threads of Reasoning. Especially the work of

Lou Somers, Peter van den Bosch, Zhaouri Yuan (Océ),

Berry van der Wijst (Philips),

Adriaan van den Brand (Centric TSolve),

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

has been valuable.
Abstract
This module addresses what we did so far and what still has to be done.
Exercise Wrap Up

- Determine architecture status:
  - What do we have?
  - What are the important gaps?
  - What are the urgent gaps?

- How to obtain architecture feedback?

- Determine an integration sequence.