The conceptual view

by Gerrit Muller University of South-Eastern Norway-NISE
e-mail: gaudisite@gmail.com
www.gaudisite.nl

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
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>module</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 realize 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

version: 0.7
September 9, 2018
CVquestionGeneratorPlanes
The conceptual view
9  Gerrit Muller

version: 0.7
September 9, 2018
CVInformationModel

data elements additional to the external information model
Example process decomposition

scan control
scan control
scan control
acq control
recon control
xDAS
recon

image handling
image handling
UI
image handling
UI
archiving control
import control
export control
display control
display
media
network
device hardware

server process

UI process

legend
Execution architecture

Issues:
- concurrency
- scheduling
- synchronisation
- mutual exclusion
- priorities
- granularity
Performance Model

\[ t_{\text{recon}} = n_{\text{raw-x}} \times (t_{\text{fft}}(n_{\text{raw-y}}) + n_{\text{raw-x}} \times (t_{\text{fft}}(n_{y}) + n_{y} \times (t_{\text{fft}}(n_{\text{raw-x}}) + t_{\text{corrections}}(n_{x}, n_{y}) + t_{\text{control-overhead}}) + t_{\text{col-overhead}}) + t_{\text{row-overhead}}) + t_{\text{fft}}(n) = c_{\text{fft}} \times n \times \log(n) \]
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

- Power
  - Start up
  - Shut down

- HW
  - Bring in initial state
  - Load and initialise firmware

- Boot-loader
  - Load and initialise loader
  - Determine loading HW
  - Determine next layer

- Kernel
  - Discover kernel HW
  - Initialise kernel data structures
  - Determine next layer

- Services
  - Load, initialise and start services
  - Discover kernel HW
  - Initialise kernel data structures
  - Determine next layer

- User interface
  - Configure UI
  - Allocate resources
  - Load, initialise and start UI

- Connect to outside
  - Detect external services
  - Publish internal services
  - Connect where needed

- Application
  - Load
  - Configure
  - Initialise
  - Start

- HW SW interface

- Stop in safe sequence
  - Flush ongoing activities
  - Close connections
  - Save persistent data
  - Free resources
  - Stop
Example work breakdown

project organization

work packages

TIP:NBE R1

scanning

preparation conversion run time acq algorithms

xDAS xFEC reconstruction hardware

VDU console algorithms gfx UI

database

database engine clinical bulk data import export archive

computing

host OS foundation classes start up shutdown exception handling

SPS SDS TPS integration alfa test betta test conf man

system

make SW

make HW

buy SW

buy HW

system

version: 0.7
September 9, 2018
CVworkBreakdown
Core, Key or Base technology

- **Core**: Own value IP
- **Key**: Critical for final performance
- **Base**: Commodity

Technology life cycle:
- **Make**
- **Outsource**
- **Buy**
- **Refer customer to 3rd party**

Partnering

The conceptual view

version: 0.7
September 9, 2018

Gerrit Muller
Example integration plan

- **Existing base system**
- **New HW subsystem**
- **SW dev system**
- **Test HW subsystem**
- **Test SW for new HW subsystem**
- **New application**
- **Integrate subsystem**
- **Integrate and refine application**
- **Integrate HW system**
- **Integrate system**

**2 partial systems for SW testing**

**2 existing base systems**

**New base systems**

**Time**

---

The conceptual view

17 Gerrit Muller

version: 0.7

September 9, 2018

CVIntegrationPlan