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.
Budget based design flow

Can be more complex than additions

SRS
- t\textsubscript{boot} = 0.5s
- t\textsubscript{zap} = 0.2s

spec

feedback

measurements
new (proto) system

micro benchmarks
aggregated functions
applications
profiles
traces

tuning

model

measurements
existing system

micro benchmarks
aggregated functions
applications

budget

definition

V4aa

10

design estimates; simulations

feedback

form

The realization view

version: 0.1
March 6, 2013
EAAbudget
### Example of a memory budget

<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></td>
<td></td>
<td></td>
<td><strong>61.0</strong></td>
</tr>
</tbody>
</table>

#### UNIX Solaris 2.x
- file cache: 3.0
- **total**: 74.0
Actual timing on logarithmic scale

- **Application needs**
  - FO4 inverter delay
  - Cycle 2 GHz CPU
  - Light travels 1 cm

- **Processing Times**
  - *10^-12* (ps)
  - *10^-9* (ns)
  - *10^-6* (s)
  - *10^-3* (ms)
  - *1* (s)

- **From low to high level processing times**
  - From low level to high level processing times

- **100 Hz video**
  - Pixel time
  - 1 byte transfer fast ethernet

- **100 Hz video frame**
  - Human eye
  - Human reaction time
  - Human 1st irritation threshold
  - Human 2nd irritation threshold

- **Storage/network application needs**
  - Light travels 1 cm

- **Eye-hand co-ordination**
  - 1 package transfer fast ethernet
  - Application level function response
  - Application level message exchange
  - 1 byte transfer fast ethernet

- **DRAM cycle time**
  - 1 byte transfer fast ethernet

- **DRAM latency**
  - 1 byte transfer fast ethernet

- **Disk seek**
  - 1 package transfer fast ethernet
  - Application level message exchange
  - Application level function response

- **from low to high level storage/network**
  - Light travels 1 cm
Typical micro benchmarks for timing aspects

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

![Graph showing the transfer time as a function of blocksize. The graph includes points for worst case, optimal block-size, and t_{overhead}.]
Performance evaluation

The realization view

Gerrit Muller

version: 0.1
March 6, 2013
RVreconstructionPerformanceAnalysis

overhead

number crunching

focus on overhead reduction

is more important than faster algorithms

this is not an excuse for sloppy algorithms

\[ t_{\text{recon}} = t_{\text{filter}}(n_{\text{raw-x}}, n_{\text{raw-y}}) + n_{\text{raw-x}} \cdot (t_{\text{fft}}(n_{\text{raw-y}}) + t_{\text{col-overhead}}) + n_{y} \cdot (t_{\text{fft}}(n_{\text{raw-x}}) + t_{\text{row-overhead}}) + t_{\text{corrections}}(n_{x}, n_{y}) + t_{\text{read I/O}} + t_{\text{transpose}} + t_{\text{write I/O}} + t_{\text{control-overhead}} \]

\[ t_{\text{fft}}(n) = c_{\text{fft}} \cdot n \cdot \log(n) \]
Performance Cost, choice based on sales value

The realization view

Gerrit Muller
Performance Cost, effort consequences

The realization view

version: 0.1
March 6, 2013

RVperformanceCostEffort
But many many other considerations

The realization view

version: 0.1
March 6, 2013
RVcostPerformanceIssues
Safety, Reliability and Security analysis methods

- Safety: hazard analysis
- Reliability: FMEA
- Security: vulnerability risks

<table>
<thead>
<tr>
<th></th>
<th>(systematic) brainstorm</th>
<th>analysis and assessment</th>
<th>improve design</th>
</tr>
</thead>
<tbody>
<tr>
<td>potential hazards</td>
<td></td>
<td>probability severity</td>
<td>measures</td>
</tr>
<tr>
<td>failure modes</td>
<td></td>
<td>effects</td>
<td>measures</td>
</tr>
<tr>
<td>vulnerability risks</td>
<td></td>
<td>consequences</td>
<td>measures</td>
</tr>
</tbody>
</table>