From Legacy to State-of-the-art; Architectural Refactoring

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

Abstract
The market of electronic appliances shows a fast increasing diversity. Manufacturers must be able to combine existing functions and new applications in a short time frame. A large amount of accumulated SW code (legacy) has to be reused in new ways.

The architecture(s) must be adapted to these new ways of working. Revolutionary adaptations have proven to be extremely risky. Opportunistic extension and integration decrease the quality of the code base, making it increasingly more difficult to continue. Architectural refactoring is a feedback based method to evolve an architecture.

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.
Today’s Audio Video Consumer Products
Trend: Convergence of separate worlds

Telecom

Consumer

Computer
Integration and Diversity
Today’s Video Products

- Cable Modem
- Set Top Box
- DVD Player
- Video Recorder
- Conventional TV
- PC
Evolution of Video Products

From Legacy to State-of-the-art; Architectural Refactoring

version: 1.3
March 6, 2013
ARproductEvolution
Distribution Scenario’s

A "Thin Servers"

Network

B "All-in-one" Combi's

All-in-one Combi's

Network

C "All-in-one" Home server

Thin Clients

Network

D "Modular"

Client

Client

Server

Server

From Legacy to State-of-the-art; Architectural Refactoring
version: 1.3
March 6, 2013
ARproductScenarios

Gerrit Muller
Moore’s law

1965
1979
2000
1990
1 kB
64 kB
2 MB

From: COPA tutorial, Rob van Ommering
Problem: increasing SW size, decreasing reliability?

![Graph showing increasing SW size and decreasing reliability over time.](image)
The Holy Grail: Reuse

From Legacy to State-of-the-art; Architectural Refactoring

Gerrit Muller

version: 1.3
March 6, 2013
ARtheHolyGrail
Simplistic Architecting: Digital TV

analog TV

<table>
<thead>
<tr>
<th>TV applications</th>
<th>TV domain platform</th>
<th>TV domain HW</th>
<th>Computing HW</th>
</tr>
</thead>
<tbody>
<tr>
<td>TV computing Infra-structure</td>
<td>Set Top Box Platform</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Set top box

<table>
<thead>
<tr>
<th>3rd party stack(s)</th>
<th>Set Top Box functions</th>
<th>MHP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set Top Box domain HW</td>
<td>Computing HW</td>
<td></td>
</tr>
</tbody>
</table>

Digital TV

<table>
<thead>
<tr>
<th>Digital TV UI</th>
<th>TV applications</th>
<th>TV computing Infra-structure</th>
<th>MHP</th>
</tr>
</thead>
<tbody>
<tr>
<td>TV domain platform</td>
<td></td>
<td>Set Top Box Platform</td>
<td></td>
</tr>
</tbody>
</table>

Digital Video Platform SW

<table>
<thead>
<tr>
<th>TV domain HW</th>
<th>Set Top Box domain HW</th>
<th>Computing HW</th>
</tr>
</thead>
</table>

Digital Video Platform

From Legacy to State-of-the-art; Architectural Refactoring

version: 1.3
March 6, 2013
ArdigitalTelevisionSimplisticArchitecture
Available Code Assets

---

From Legacy to State-of-the-art; Architectural Refactoring
Gerrit Muller
version: 1.3
March 6, 2013
ARdigitalTelevisionCodeAssets
Merge problems

Architectural mismatch:
wrappers, translators, conflicting controls

Additional code and complexity, no added value

Poor performance; additional resource usage

Problems

Architecture
Reuse
non problem
Refactoring

within short term business goals

with limited but substantial refactoring goals

limited investment

based on long term architecture vision

clear product

clear value proposition

feedback on direction

limited investment

based on long term architecture vision
Example of Refactoring Goals

+ Decrease Code Size

+ Decrease Resource Usage
  * power
  * memory
  * silicon area

+ Increase Performance
  * response time
  * throughput

+ Increase quality
  * decrease fault density
Architectural vs Code refactoring

Architectural Refactoring
Function, Structure, Rationale

Mechanisms, Technologies

Code Refactoring

From Legacy to State-of-the-art; Architectural Refactoring
17 Gerrit Muller
Frequent feedback
stepsize: 3 months
elapsed time: 25 months
stepsize: elapsed time

3 months
25 months

2 months
12 months
Small feedback cycles result in Faster Time to Market
Awareness of dynamics
Myth: Platforms are Stable

Dynamic Market

How stable is a platform or an architecture?

Architecture Components Platform

Fast changing Technology
Growth and change continues, some "old" components become obsolete.

3rd generation components are mature, active maintenance needed.
Long Term Vision
Example Long Term Vision

Long Term Vision: Reference Architecture + Sample implementation of Framework and Components

Applications

Services

Computing Infrastructure

Domain Infrastructure

Reference Architecture

Framework

personalization

i.e. tunes, themes

Configuration

i.e. Internationalization

From Legacy to State-of-the-art; Architectural Refactoring

version: 1.3
March 6, 2013
ARlongTermVisionExample

Gerrit Muller
Don’t do

Digital Cable  Set top  DVR  DVD RW  trans mitter  portable media-screen

Digital TV  Opportunistic Legacy Integration

Proclaimed reuse

Opportunistic Legacy Integration

From Legacy to State-of-the-art; Architectural Refactoring

version: 1.3
March 6, 2013
ARtheWrongWay
Conclusion: Refactoring the Architecture is a must