Architectural Refactoring; illustrated by MR

Abstract

The market of medical 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 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.

September 5, 2020 status: preliminary draft version: 0.1



Today's Medical Products



Architectural Refactoring; illustrated by MR 2 Gerrit Muller version: 0.1 September 5, 2020 ARMRproducts





Architectural Refactoring; illustrated by MR 3 Gerrit Muller version: 0.1 September 5, 2020 ARMRconvergence



Integration and Diversity

information handling			
administration billing scheduling logistics laboratory pharmaceutics			
MR image acquisition		image handling	
bore systems 7T	gradients	prepare	treatment
ЗТ	very fast	diagnosis	planning
1.5T	fast	diagnosis	research
open magnets	economy	report	education
0.6T	RF coils	authorise	demonstration
	integrated dedicated	clinical review	



Today's Medical Products





Distribution Scenario's





Problem: increasing SW size, decreasing reliability?



Architectural Refactoring; illustrated by MR 7 Gerrit Muller version: 0.1 September 5, 2020 ARMRproblem

The Holy Grail: Reuse



version: 0.1 September 5, 2020 ARtheHolyGrail



Simplistic Architecture



Architectural Refactoring; illustrated by MR 9 Gerrit Muller



Future Simplistic Architecture







Available Code Assets



total make: >1000 my

total buy: >1500 my





version: 0.1 September 5, 2020 ARmergeProblems



Refactoring

clear product clear value proposition

within short term business goals

feedback on direction

with limited but substantial refactoring goals

limited investment

based on long term architecture vision

Architectural Refactoring; illustrated by MR 13 Gerrit Muller version: 0.1 September 5, 2020 ARsolution



- + 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 Code Refactoring Function, Structure, Rationale move() optimize() old accelarate() add(a) add(a) remove() ... new remove() . . . Mechanisms, Technologies old old optimize() move() return(error) new accelarate() free, alloc add(a) • • • new . . . remove() raise(exception) garbage collection

Frequent feedback





version: 0.1 September 5, 2020 LWAfeedbackLarge



Feedback (2)



version: 0.1 September 5, 2020 LWAfeedbackMedium



Feedback (3)



Small feedback cycles result in Faster Time to Market

version: 0.1 September 5, 2020 LWAfeedbackSmall



Awareness of dynamics







Platform Evolution (Easyvision 1991-1996)





Long Term Vision



Example Long Term Vision







reuse





Conclusion: Refactoring the Architecture is a must



version: 0.1 September 5, 2020 ARMRrefactoring

