Software Architecture Verification at MR

Architecture Improvement during the Race

René L. Krikhaar
Rene.Krikhaar@philips.com

Philips Medical Systems
Magnetic Resonance
Best, the Netherlands

Let's make things better.

PHILIPS

Overview

- Introducing myself
- Medical System: Magnetic Resonance
- Developing (SW) an MR system
- Software Architecture Verification
- Development Process
- Conclusions

Overview

- Introducing myself
- Medical System: Magnetic Resonance
- Developing (SW) an MR system
- Software Architecture Verification
- Development Process
- Conclusions

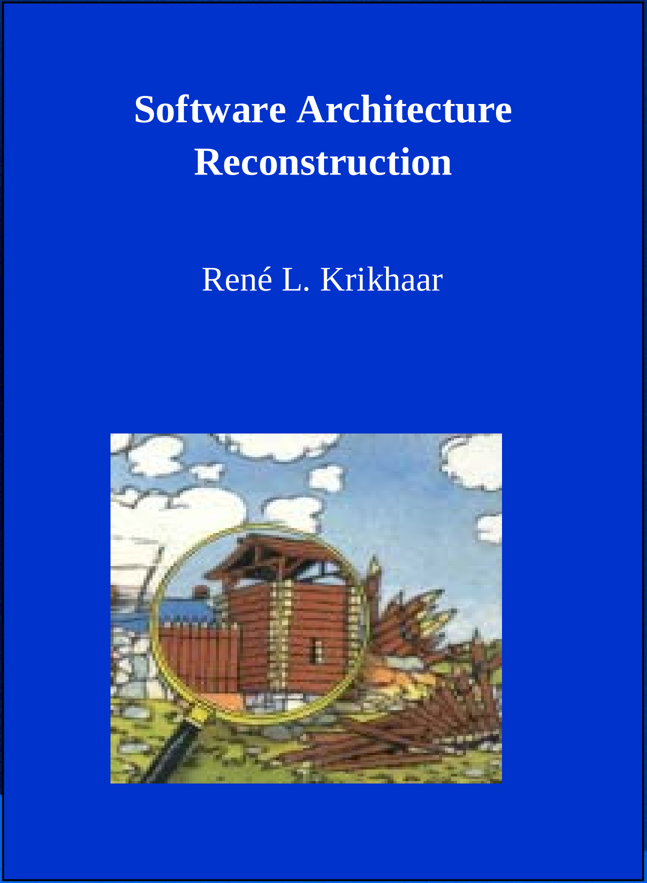
Introducing myself

- 1986: MSc. Computer Science KU Nijmegen
- 1994: MSc. Knowledge Engineering Uo Middlesex
- 1999: PhD. Computer Science Uv Amsterdam

Philips:

- 1987: VLSI Testing Software Engineer P-ASIC
- 1991: Logic Synthesis Software Engineer ED&T
- 1994: Research Scientist PRL-Eindhoven
- 1999: Software Architect MR Scan Software

Software Architecture Reconstruction



Software Architecture Reconstruction

René L. Krikhaar

Framework
- Described Architecture
- Redefined Architecture
- Managed Architecture

Software Architecture Verification

Overview

- Introducing myself
- Medical System: Magnetic Resonance
- Developing (SW) an MR system
- Software Architecture Verification
- Development Process
- Conclusions

Magnetic Resonance system

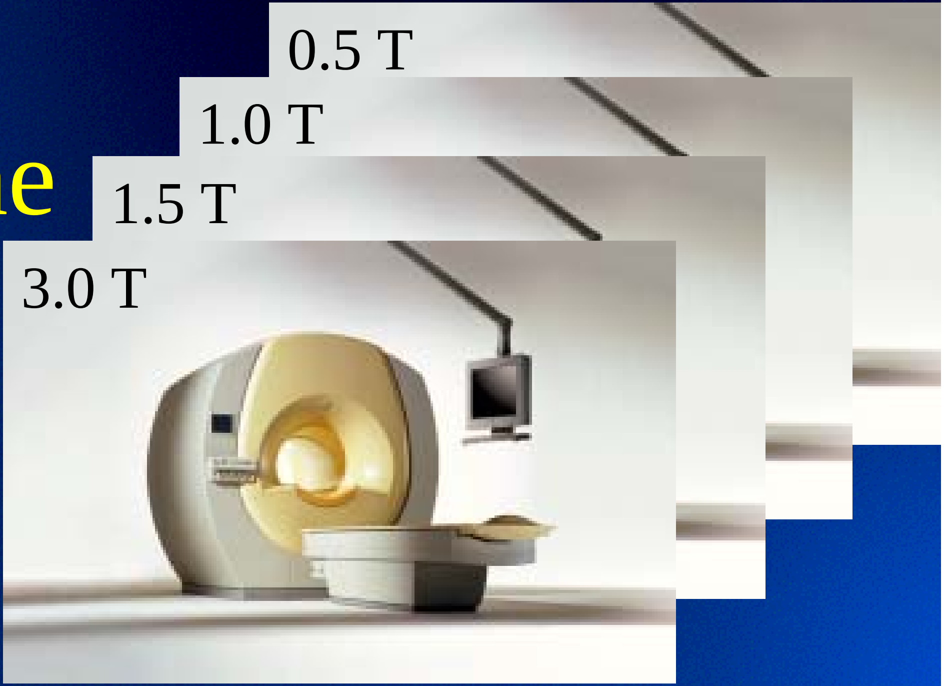
What?

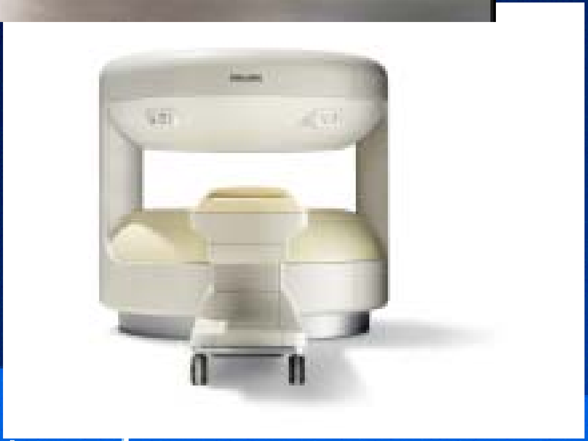
- (diagnostic) medical images

How?

- Magnetic field
- RF signals (receivers and transmitters)
- Gradient

MR Product Line



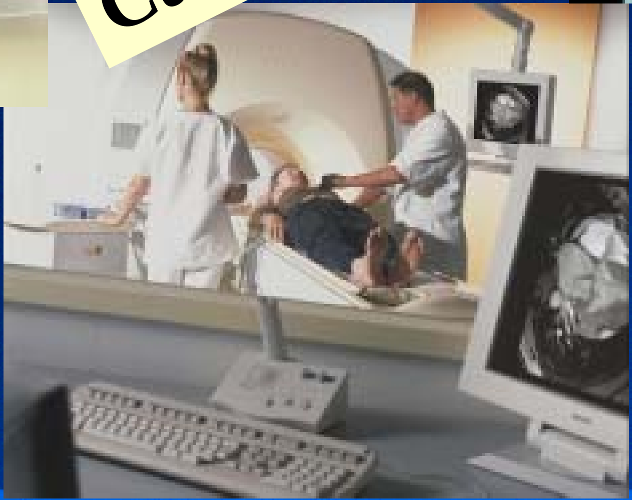


Functional Areas

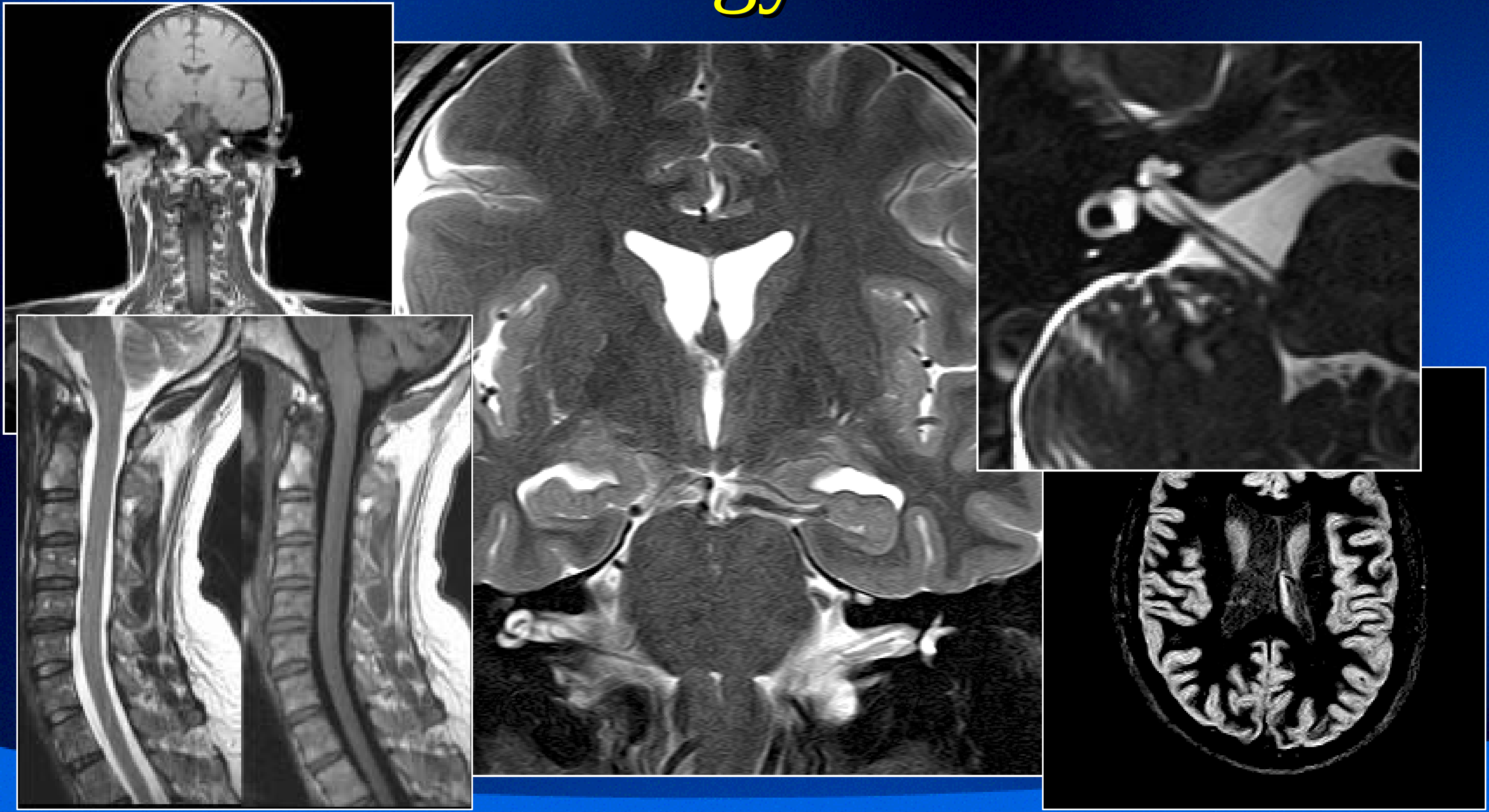
Radiology

Interventional

Cardiology



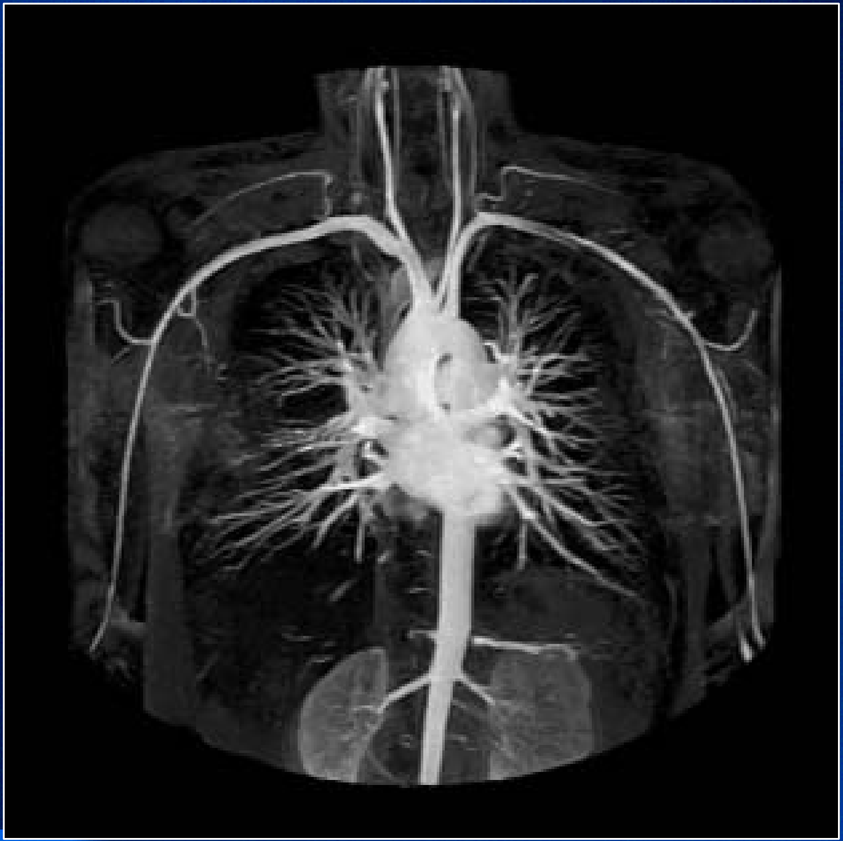
Neurology

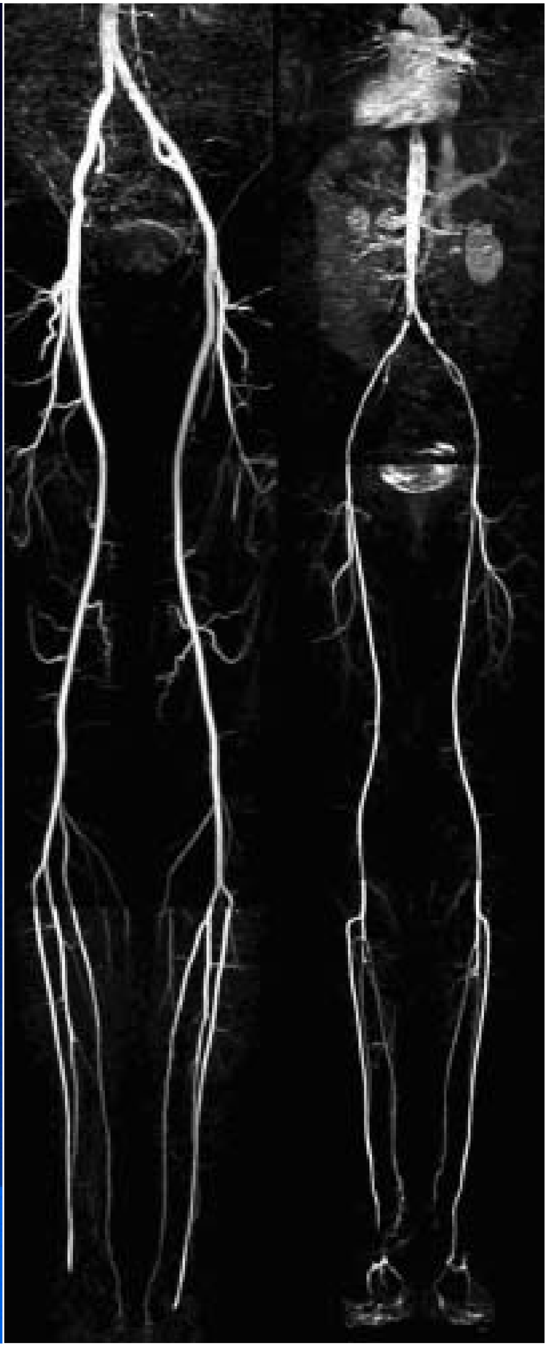


Let's make things better.

PHILIPS

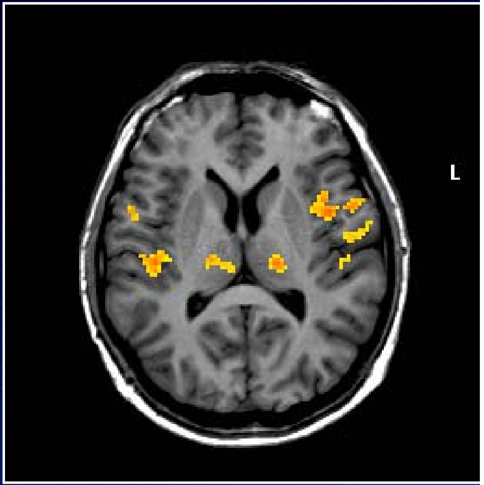
Angiography

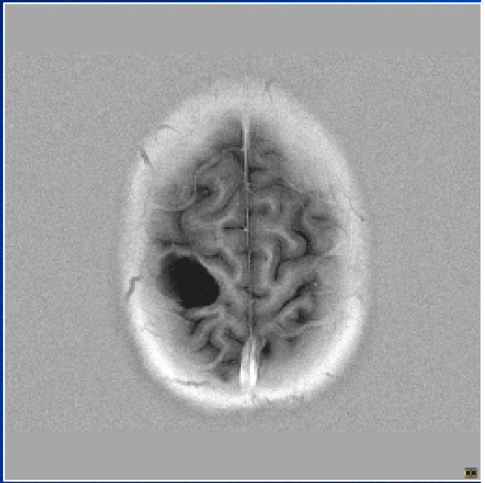




Let's make things better.

Functional Brain





Let's make things better.

PHILIPS

Overview

- Introducing myself
- Medical System: Magnetic Resonance
- Developing (SW) an MR system
- Software Architecture Verification
- Development Process
- Conclusions

Product Characteristics

- High Tech Product:
  on the edge of possibilities in (MR) physics
- Each 0.5-1.0 yr new MR Products / Release
  - new functionality (e.g. SENSE)
  - new hardware (e.g. CPU, RF amplifier)
- Parallel Development
  - Multiple Projects

“Complicating” Factors for Development

- Large System
  - more than 3 MLOC (Lines Of Code)
  - many sw/hw developers (also multi-site)
  - third party software/hardware
- Many products in MR Family
  - deriving variants
- Incremental Development
  - includes code written 20 years ago

Making Life Easier -1-

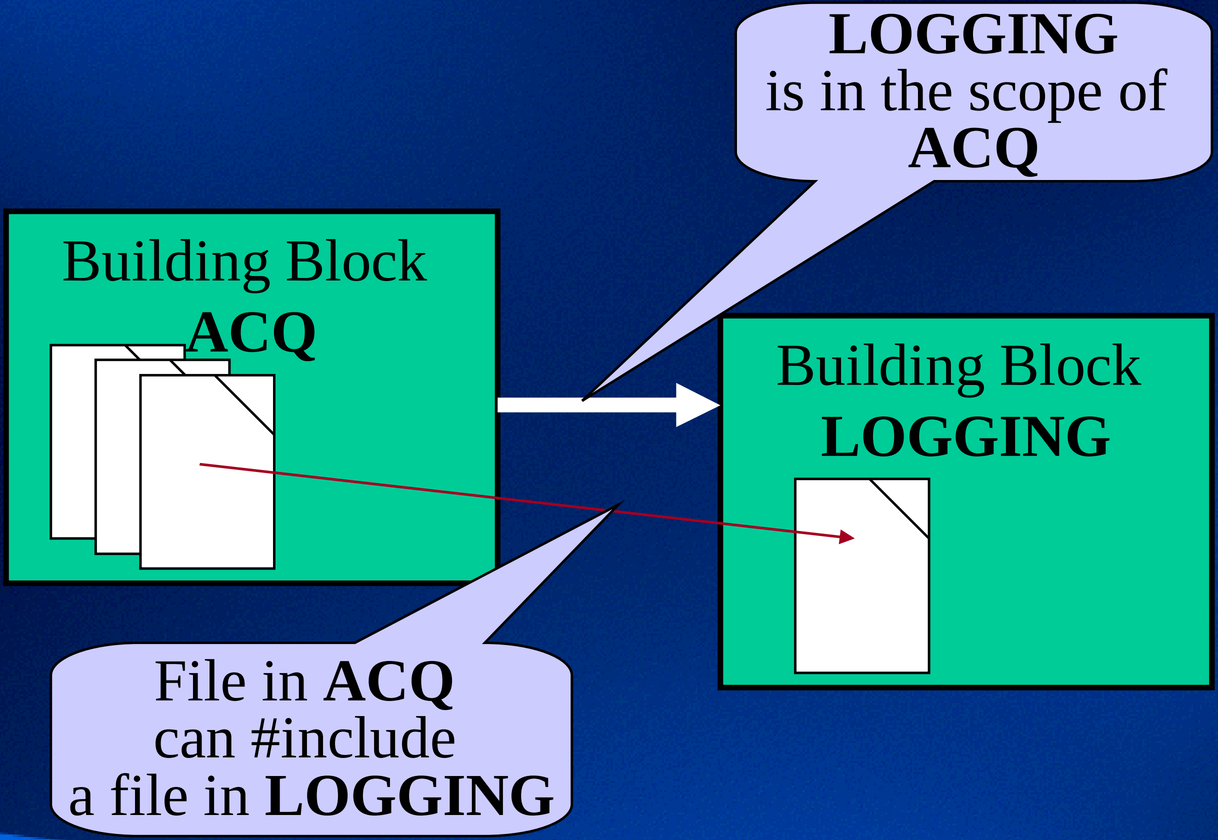
- *Daily-Build-and-Smoke-Test* (since 1984)

Making Life Easier -2-

- Define Coding Standards (since 1985)
- Enforce Check Coding Standards (since 1990)
- Improve Code for Coding Standards (since 1994)

Code Architecture Verification

Scoping


LOGGING is in the scope of ACQ
Building Block ACQ
Building Block LOGGING
File in ACQ can #include a file in LOGGING

Making Life Easier -3-

- Define Scoping rules (since 1988)
- Enforce Scoping rules (since 1990)
- Improve Scoping rules (since 1994)

Module Architecture Verification

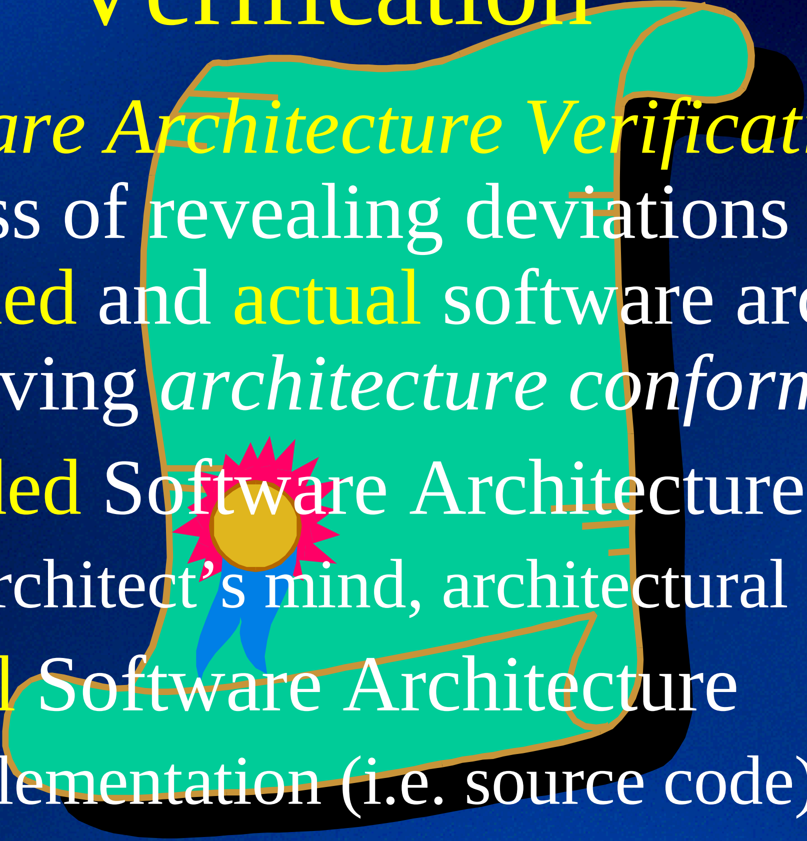
What did we achieve?

- Improvement of code comprehension
  - coding standard
  - scoping
- Reduction of coding errors
  - coding standard
- Incremental Testing
  - scoping
- Easier introduction of an OSAL
  - scoping

Overview

- Introducing myself
- Medical System: Magnetic Resonance
- Developing (SW) an MR system
- Software Architecture Verification
- Development Process
- Conclusions

Software Architecture Verification

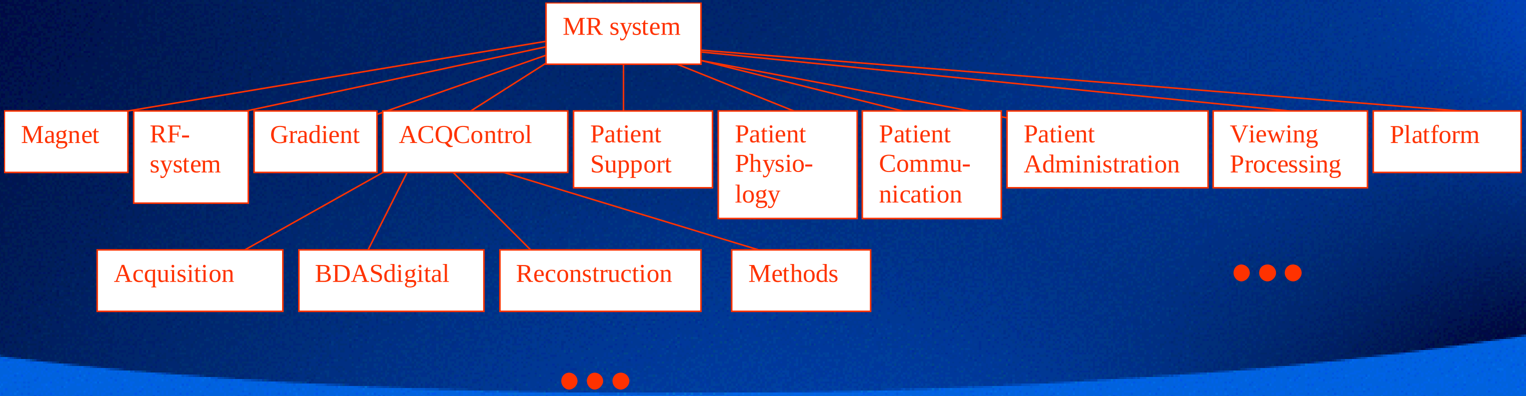


- *Software Architecture Verification* is the process of revealing deviations between intended and actual software architecture (achieving *architecture conformance*)
- Intended Software Architecture
  - In architect's mind, architectural documents
- Actual Software Architecture
  - Implementation (i.e. source code)

Building Blocks

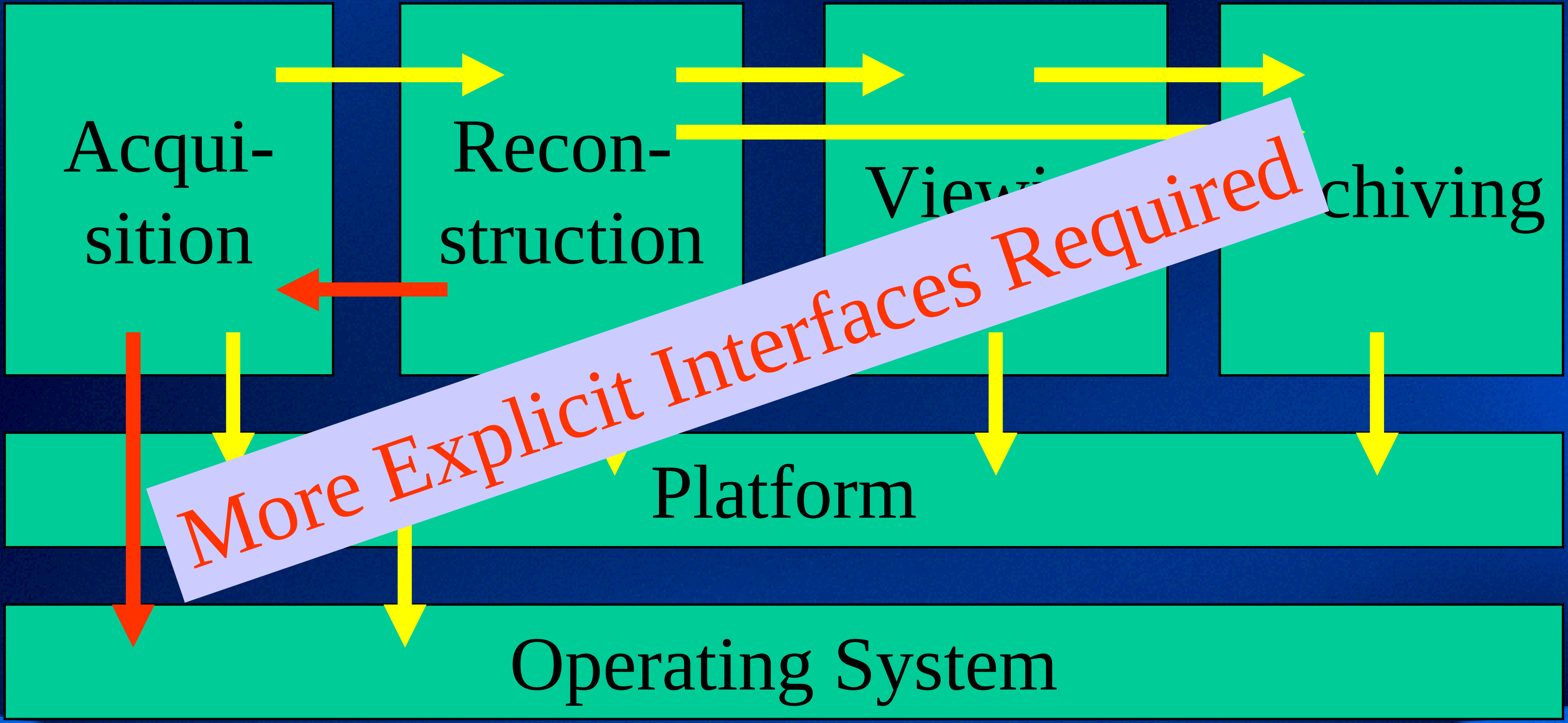
– A functional unit of the MR system.

– Building blocks are hierarchically organized, meaning that a building block may consist of a number of building blocks.



Let's make things better.

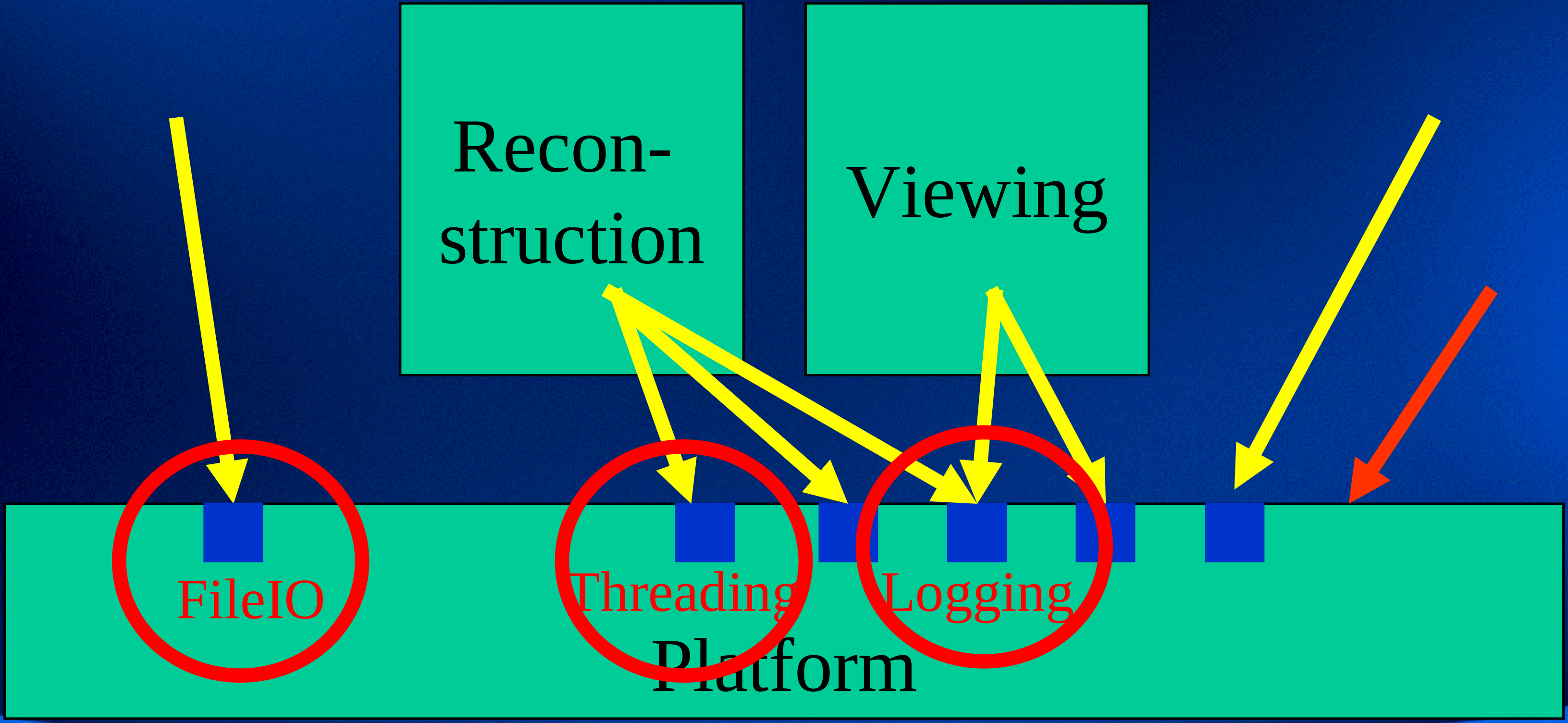
MR System


Acqui-
sition
Recon-
struction
Viewi
chiving
Platform
Operating System
More Explicit Interfaces Required

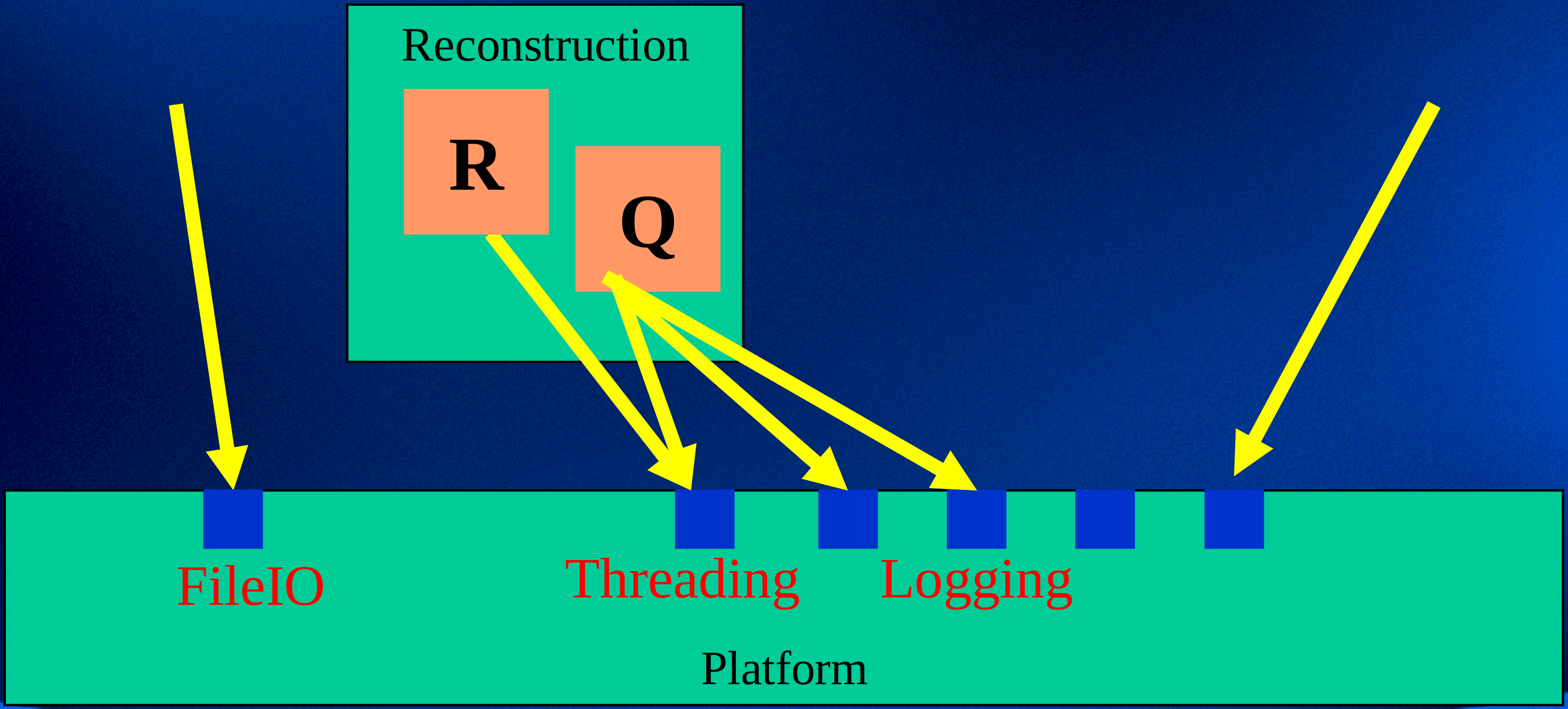
Why interfaces?

- Separation of concerns
  - maintenance
  - new employees
  - development (planning & tracking)
  - testing
  - parallel development
  - product variants
  - outsourcing
  - …

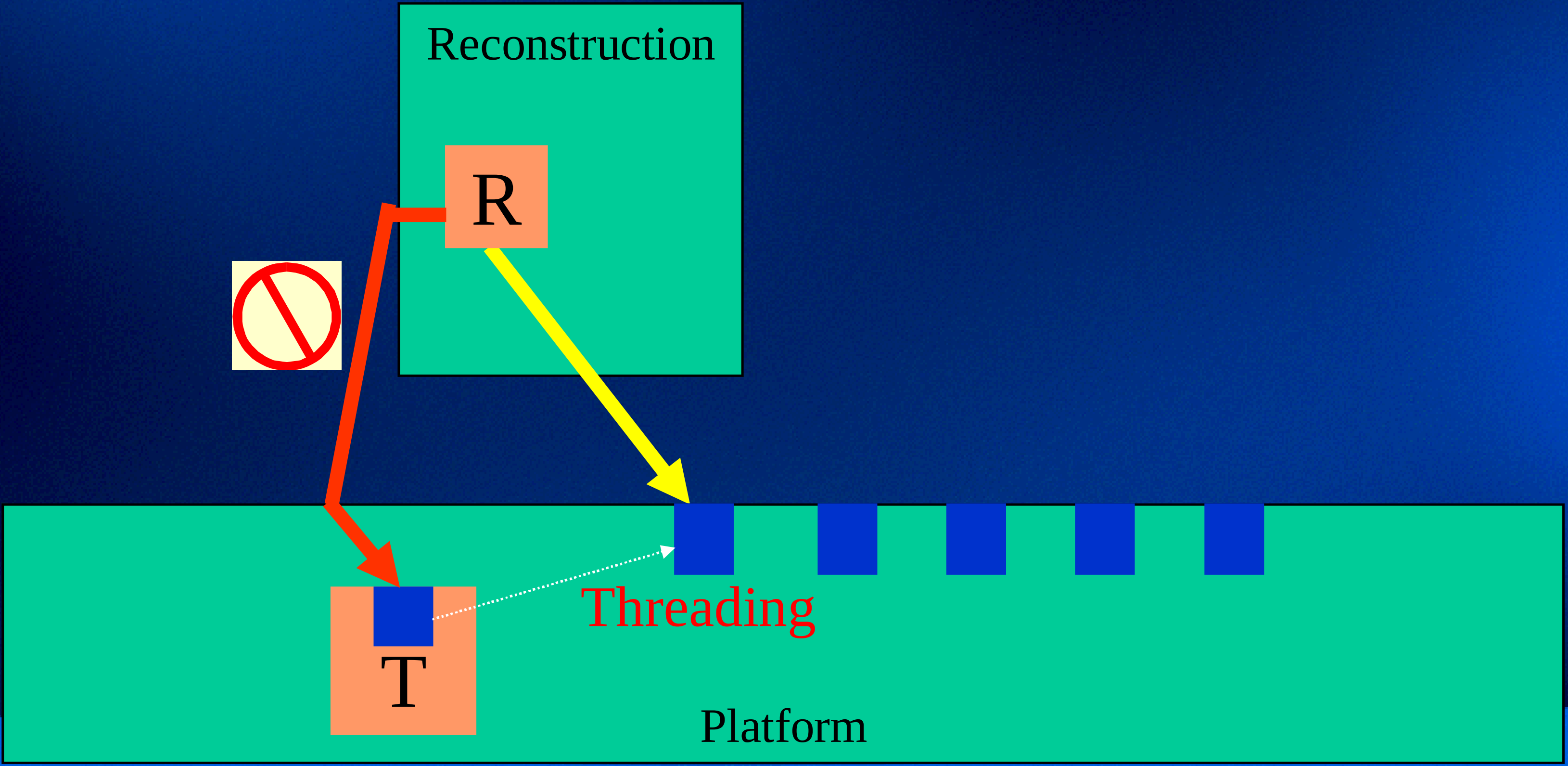
Building Blocks and Interfaces



Hierarchy in Interface Usage



Hierarchy Interface Rule



Making Life Easier -4-

- Define Interface Management (2000)
- Enforce Interface Management (2002)
- Improve Interfaces (> 2003)

Module Architecture Verification

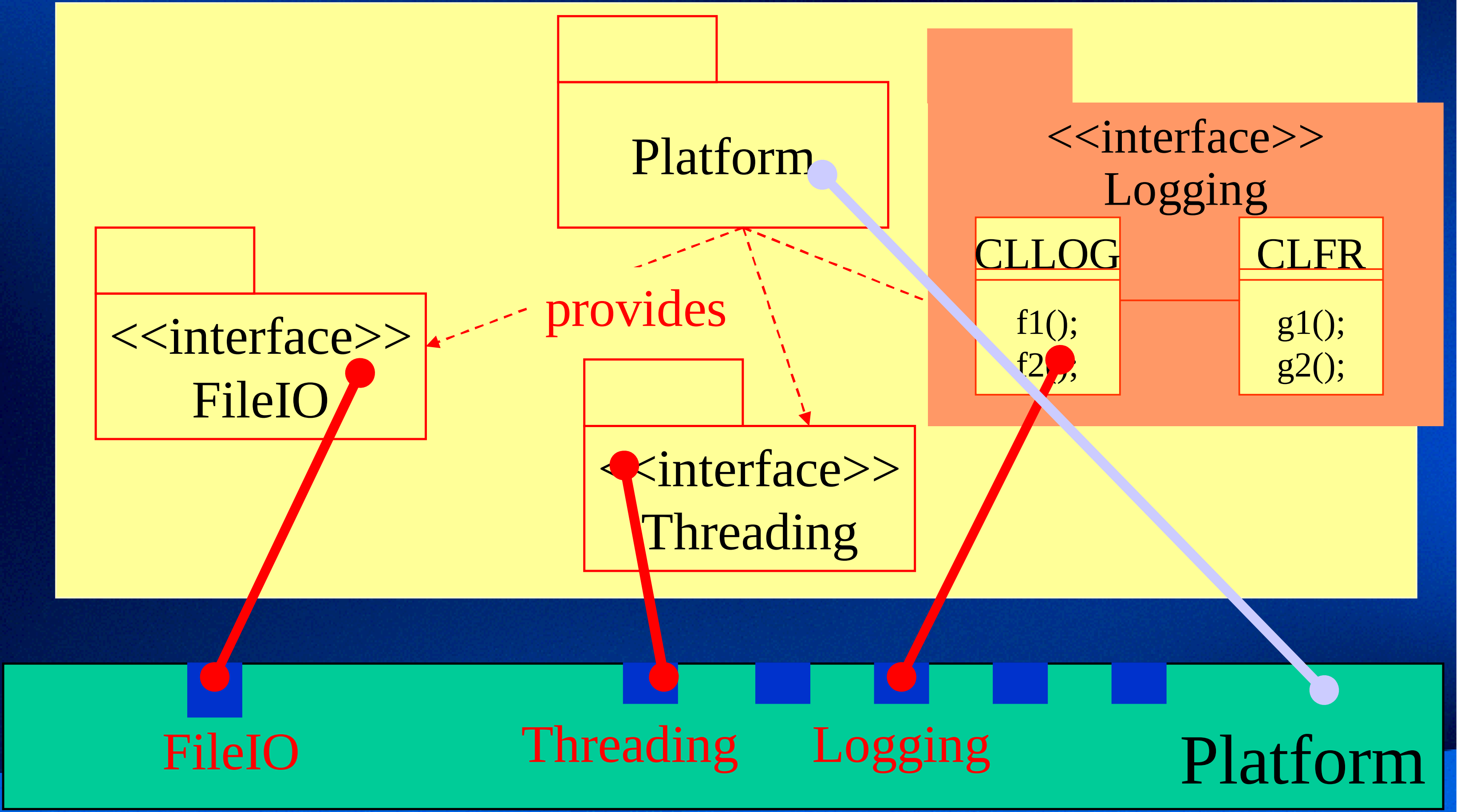
Managing the Development Process

- Daily Build and Smoke Test
  - **quality & stability** of code base
- Coding Standards
  - **comprehensability** of code base
- Scoping Rules
  - **complexity** of code base
- Interface Management
  - **life cycle independency** in code base

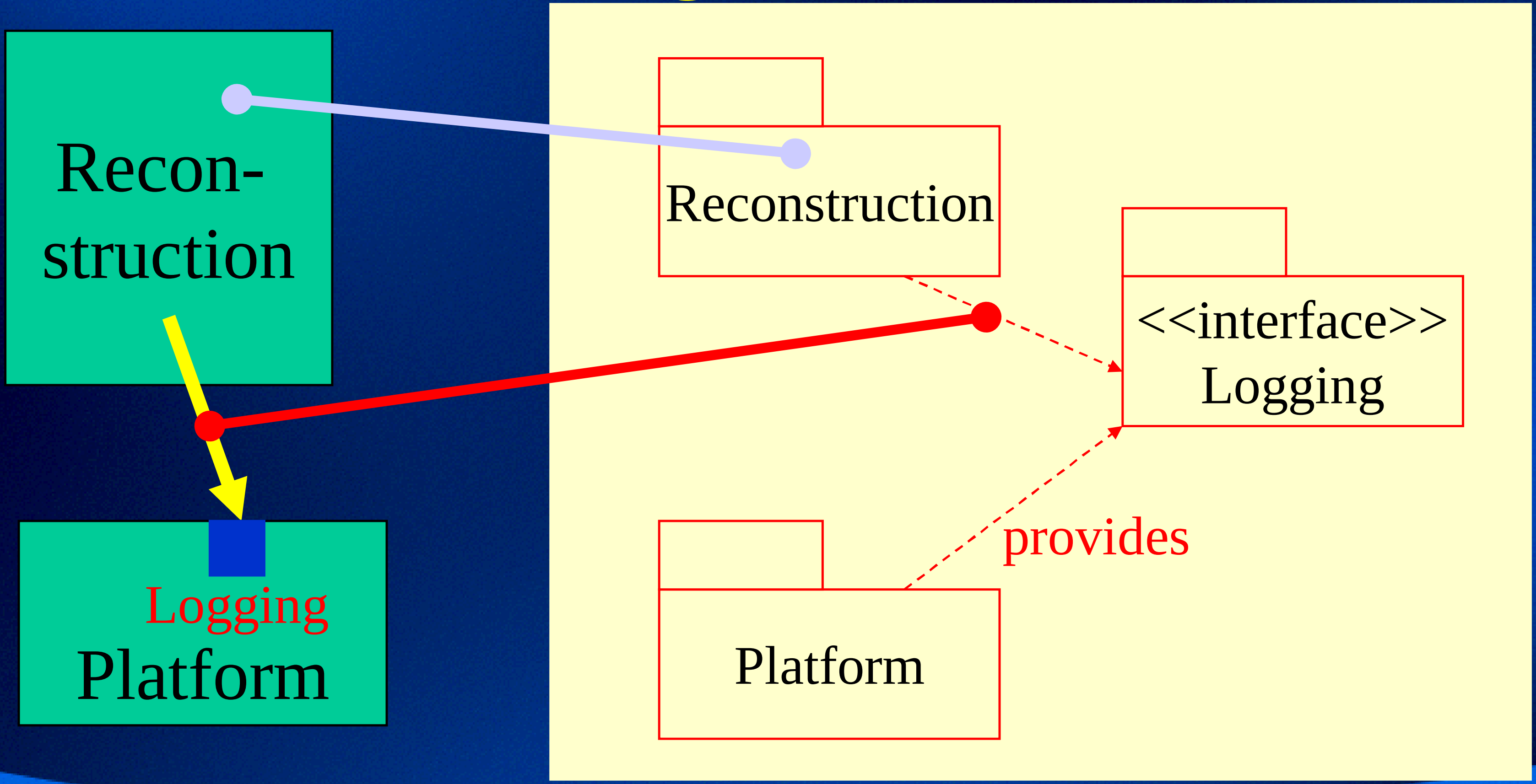
Overview

- Introducing myself
- Medical System: Magnetic Resonance
- Developing (SW) an MR system
- Software Architecture Verification
- Development Process
- Conclusions

Interfaces in UML

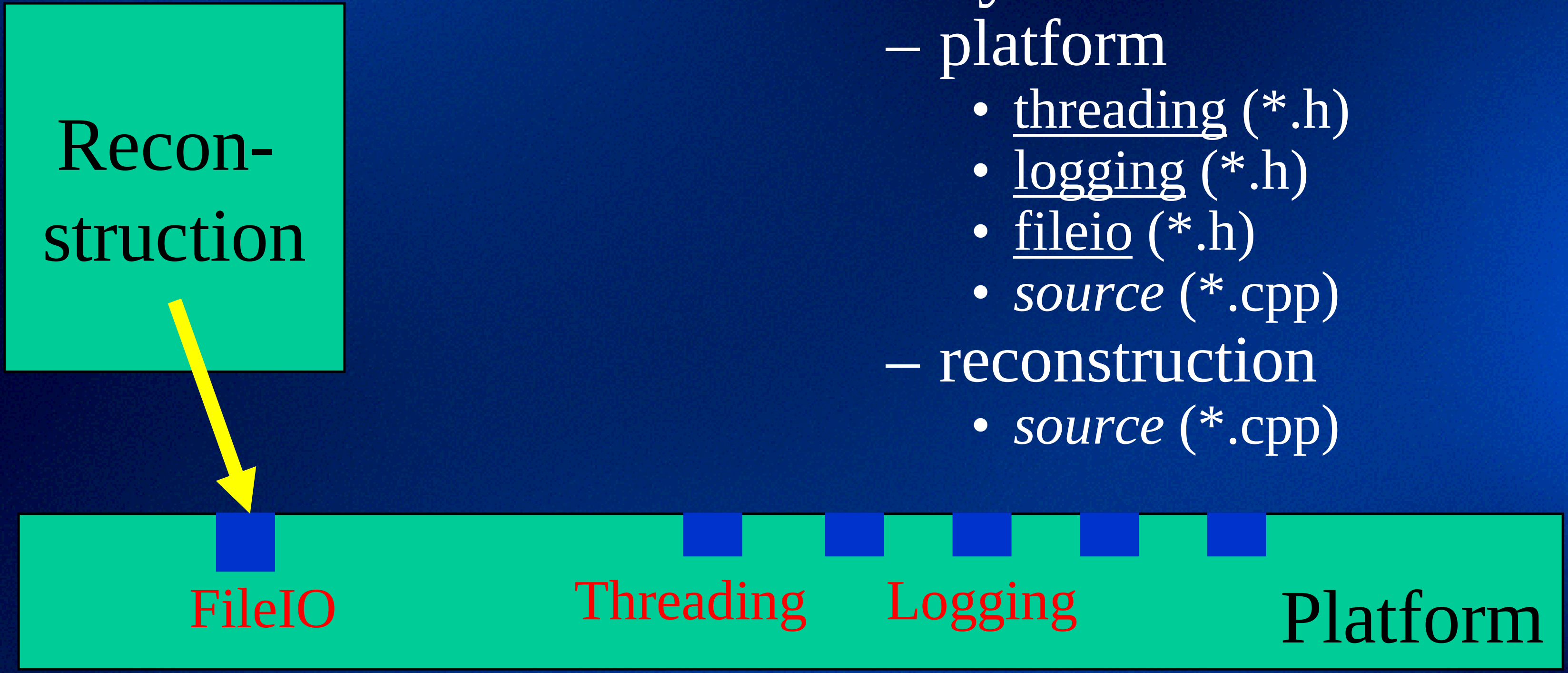


Interface Usage in UML



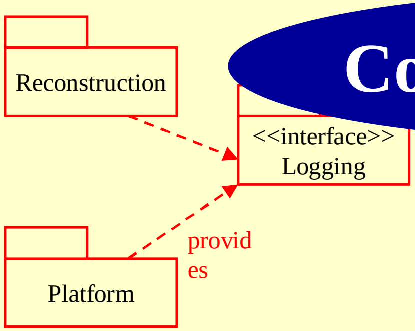
Interfaces in Code Archive

- mrsystem
  - platform
    - threading (*.h)
    - logging (*.h)
    - fileio (*.h)
    - *source* (*.cpp)
  - reconstruction
    - *source* (*.cpp)



Interface Verification

Rational Rose



ClearCase

m

n

- threading (*.h)
- logging (*.h)
- fileio (*.h)
- source (*.cpp)

– reconstruction
- source (recon.cpp)

Compile recon.cpp

Reconstruction

platform/logging

+ Coding standards

CC -Iplatform/logging recon.cpp

Let's make things better.

PHILIPS

Development Deliverables

"close the chain"

- For each Building Block:
  - Interface Specification (UML)
  - Dependencies / Usage between sub-Building Blocks (UML)
  - Implementation of Building Block
    - source code
  - mr_build command

Review + Authorisation

Review + Coding Standards

SW Architecture Verification

Overview

- Introducing myself
- Medical System: Magnetic Resonance
- Developing (SW) an MR system
- Software Architecture Verification
- Development Process
- Conclusions

Introduction in Organisation

Preparation phase

1 Define the required architectural rule

2 Define a way to (automatically) enforce it

3 Measure “*status*” (get a *threshold* value)

Execution phase

1 Accept violations < *threshold* value

2 Decrease continously *threshold*

3 Solve rest of violations

Experience -1-

- Introduction on separate development stream
- Code Base analysis not completely okay
  - missing parts of the code base
  - action: fix in a separate action
- Nested include statement
  - crossing subsystem borders
  - action 1: adapt the mechanism
  - action 2: fix in a separate action

Experience -2-

- Hard to find a project that took the ‘risk’
  - be very early
- Deployment in organisation
  - carefully planned and executed
  - accepted by engineers
- New projects starting to use I/f management
  - project control

Why does it succeed at Philips MR?

- Management & Project Support
- Evolutionary Introduction Strategy
- Verification Mechanism
  - automatic verification tools **AND**
  - embedded in the organisation’s process