Legacy is sexy and this room is hot

Story

Drivers

Tips

Agile / Scrum

Solution

SOA / Integration

Coaching

Quality

Enterprise Java

Viktor Grgić @vgrgic, LeanArch.eu

Why this talk?

* Remove from tech dictionary words:
  * generic
  * reusable
  * flexible

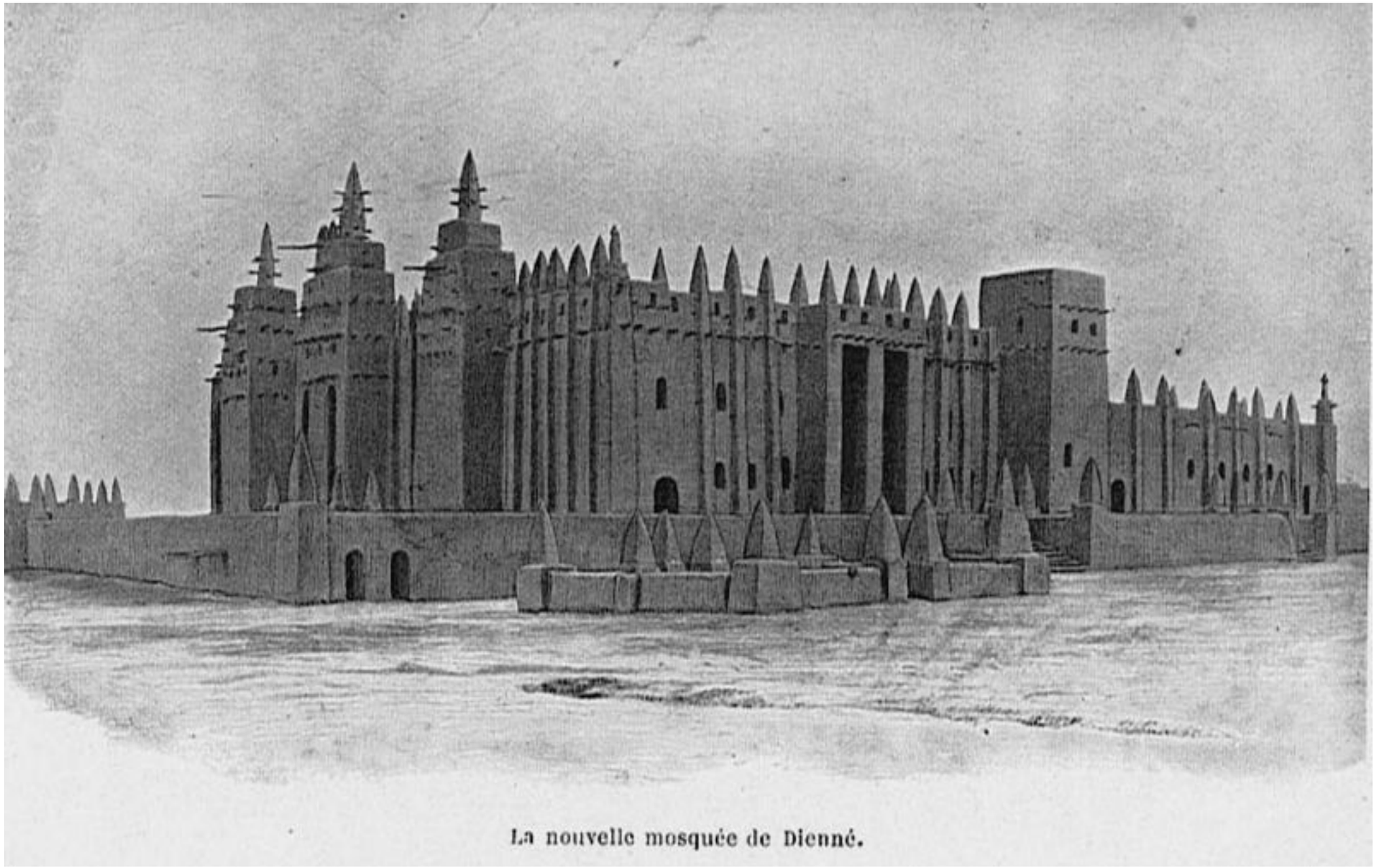
Large software projects today

* ...are replacing something **old for new**.
* Many of them have single goal: **Replace** outdated **technology** with new one.
* Many of them are huge investments with long term promises of so called **flexibility in future**.
* ...and if your project is not one of these, then you're maybe creating a **christmas tree**. Looks nice, but it is dead very soon.

Viktor Grgić @vgrgic, LeanArch.eu

Great Mosque of Djenné, Mali ruins
1828

Great Mosque of Djenné rebuilt
1895



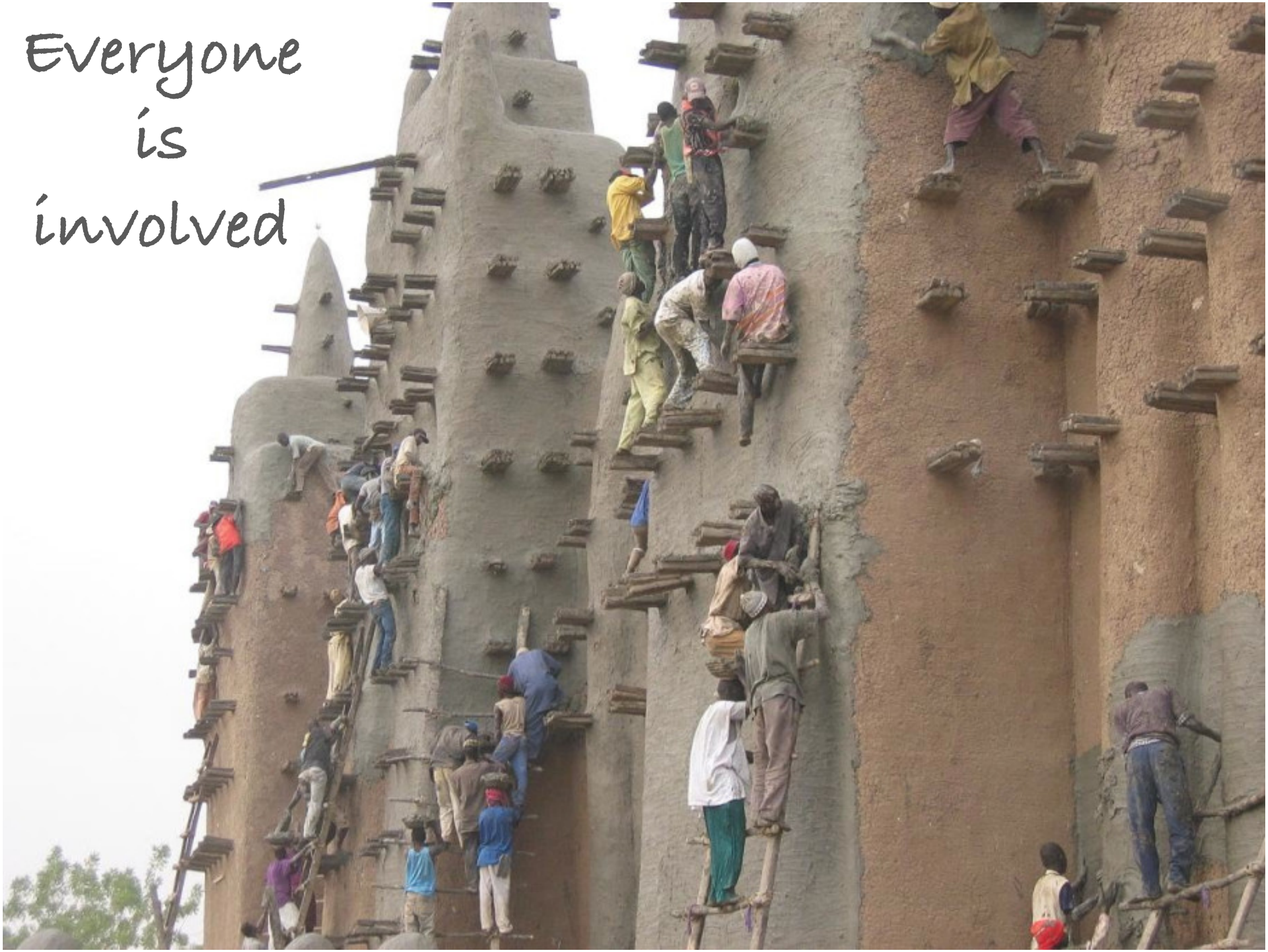
Great Mosque of Djenné
1910

Esha Chiocchio / National Geographic

Viktor Grgić @vgrgic, LeanArch.eu

Viktor Grgić @vgrgic, LeanArch.eu

Everyone is involved



Viktor Grgić @vgrgic, LeanArch.eu

Viktor Grgić @vgrgic, LeanArch.eu

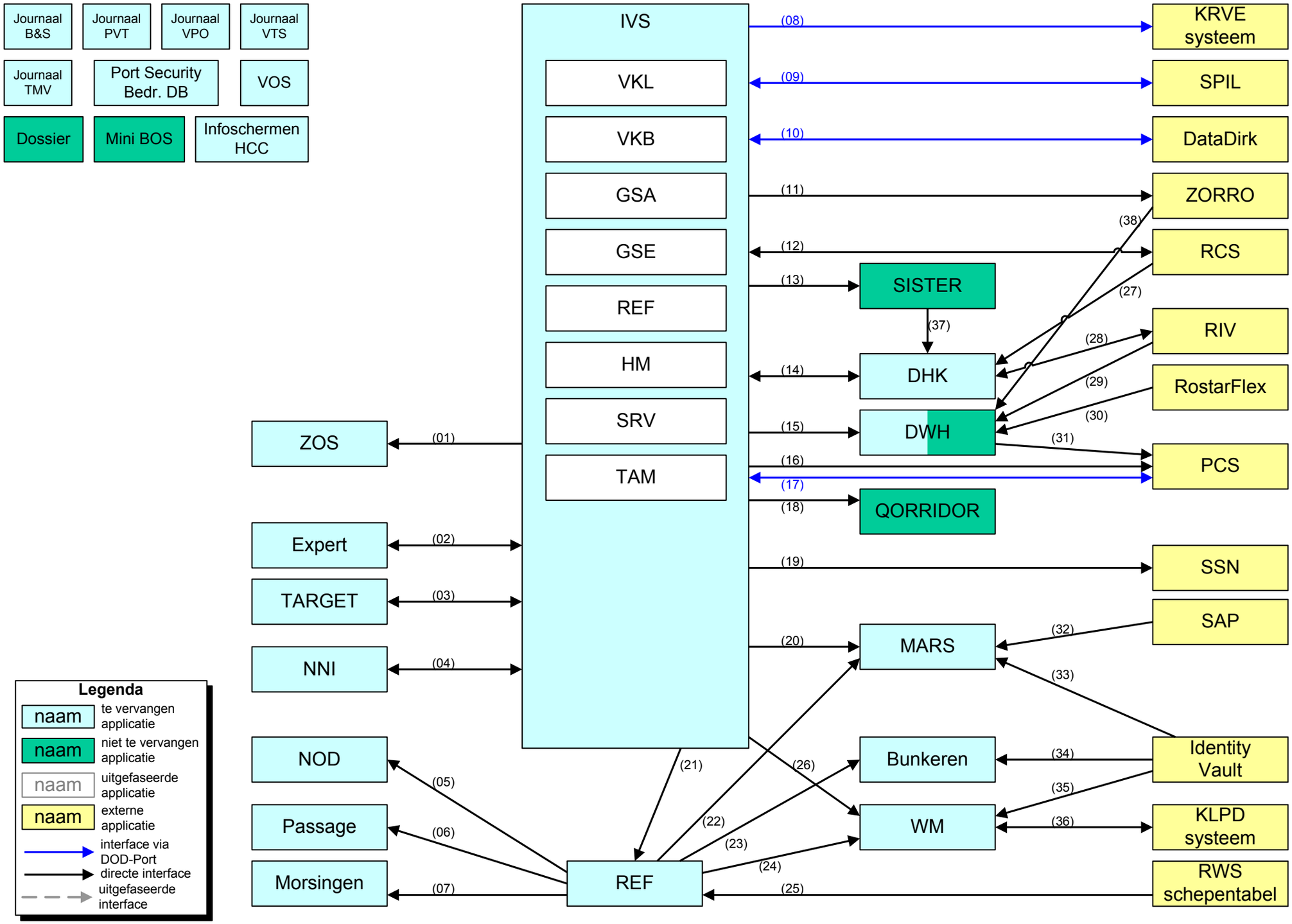
About 6 years ago...

24/7 system for traffic management, transport, environmental safety and crisis management.

Port of Rotterdam, Project HaMIS

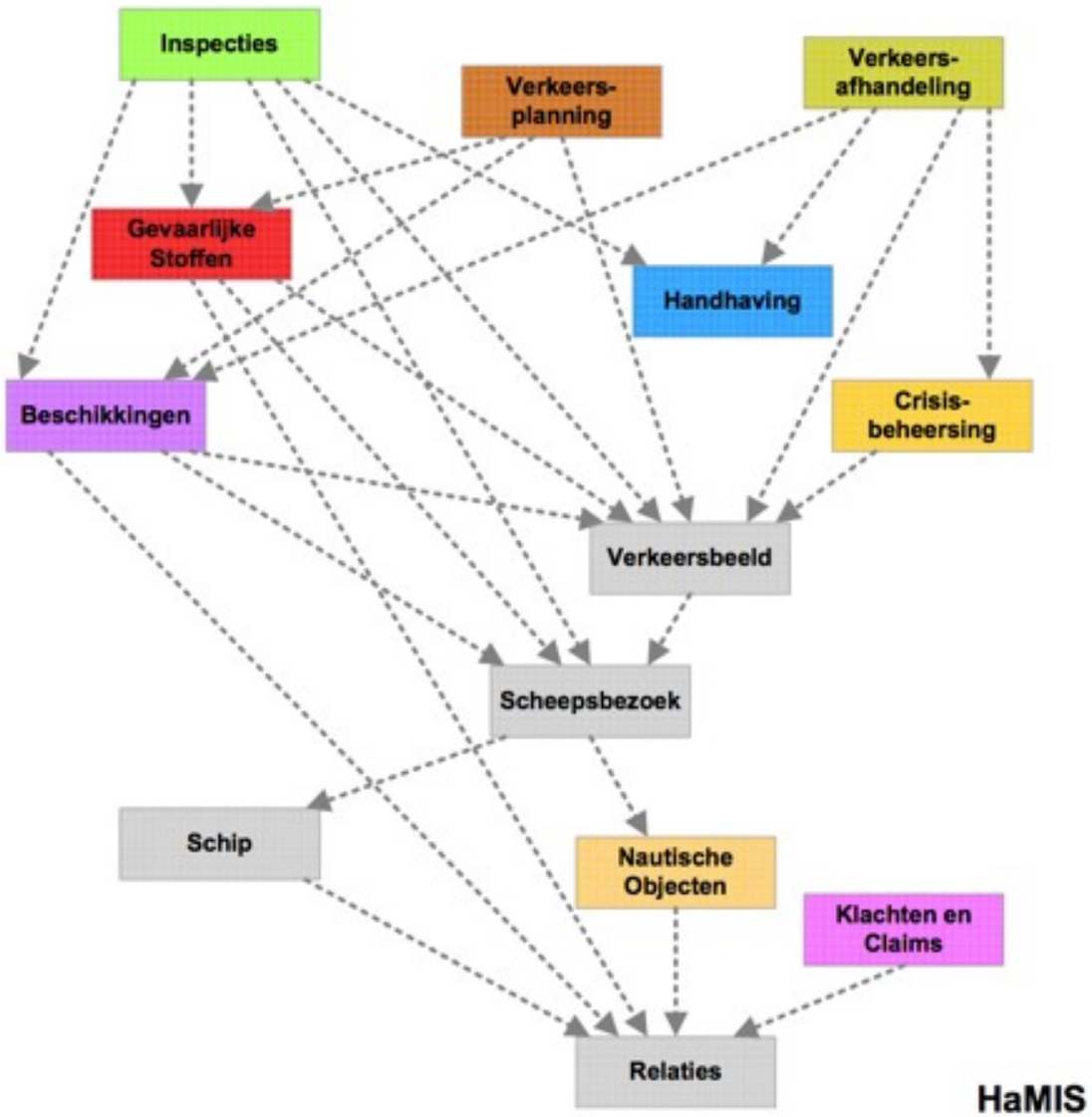
1. Replace outdated traffic management system
2. Support the port of Rotterdam expansion

old

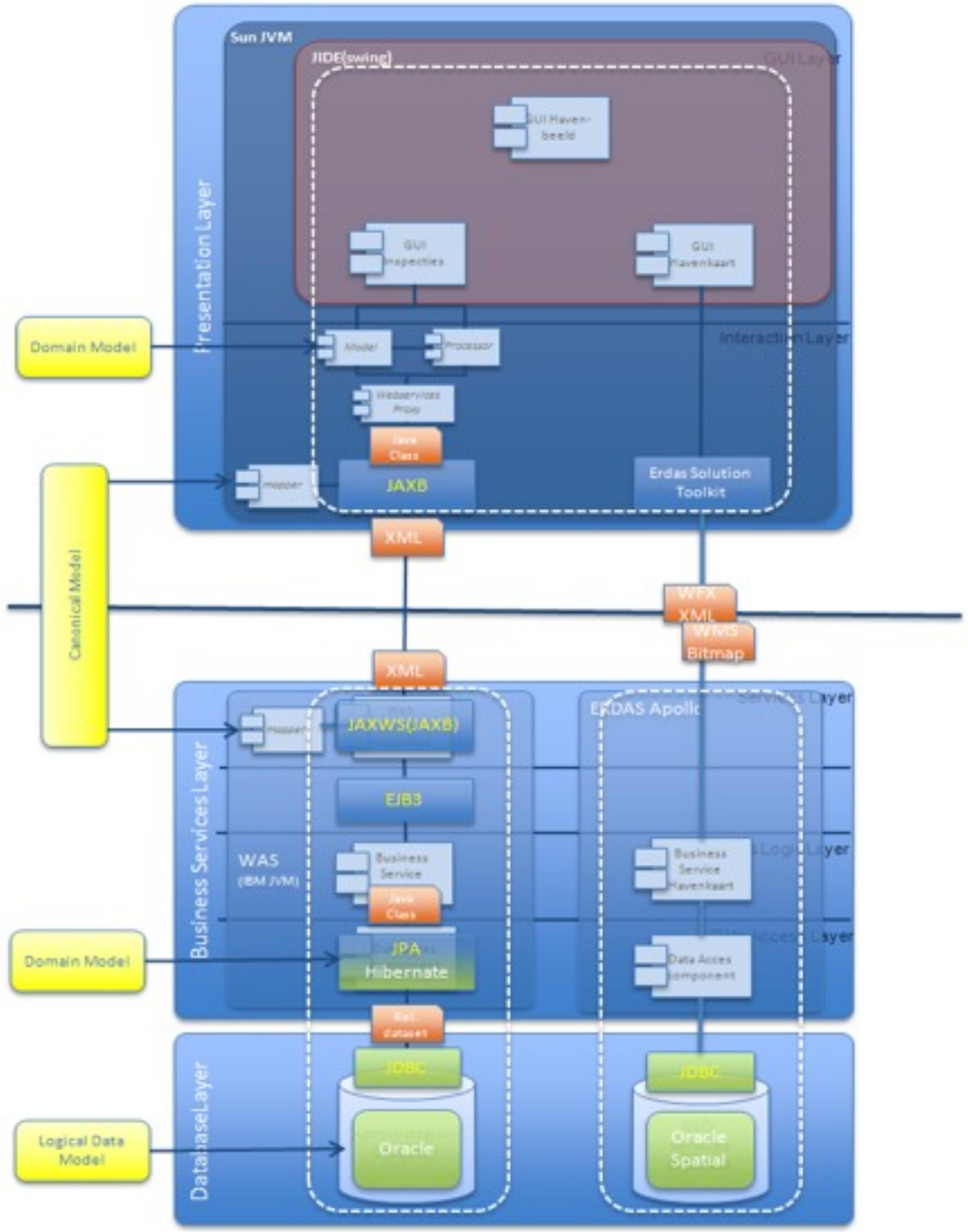


Viktor Grgić @vgrgic, LeanArch.eu

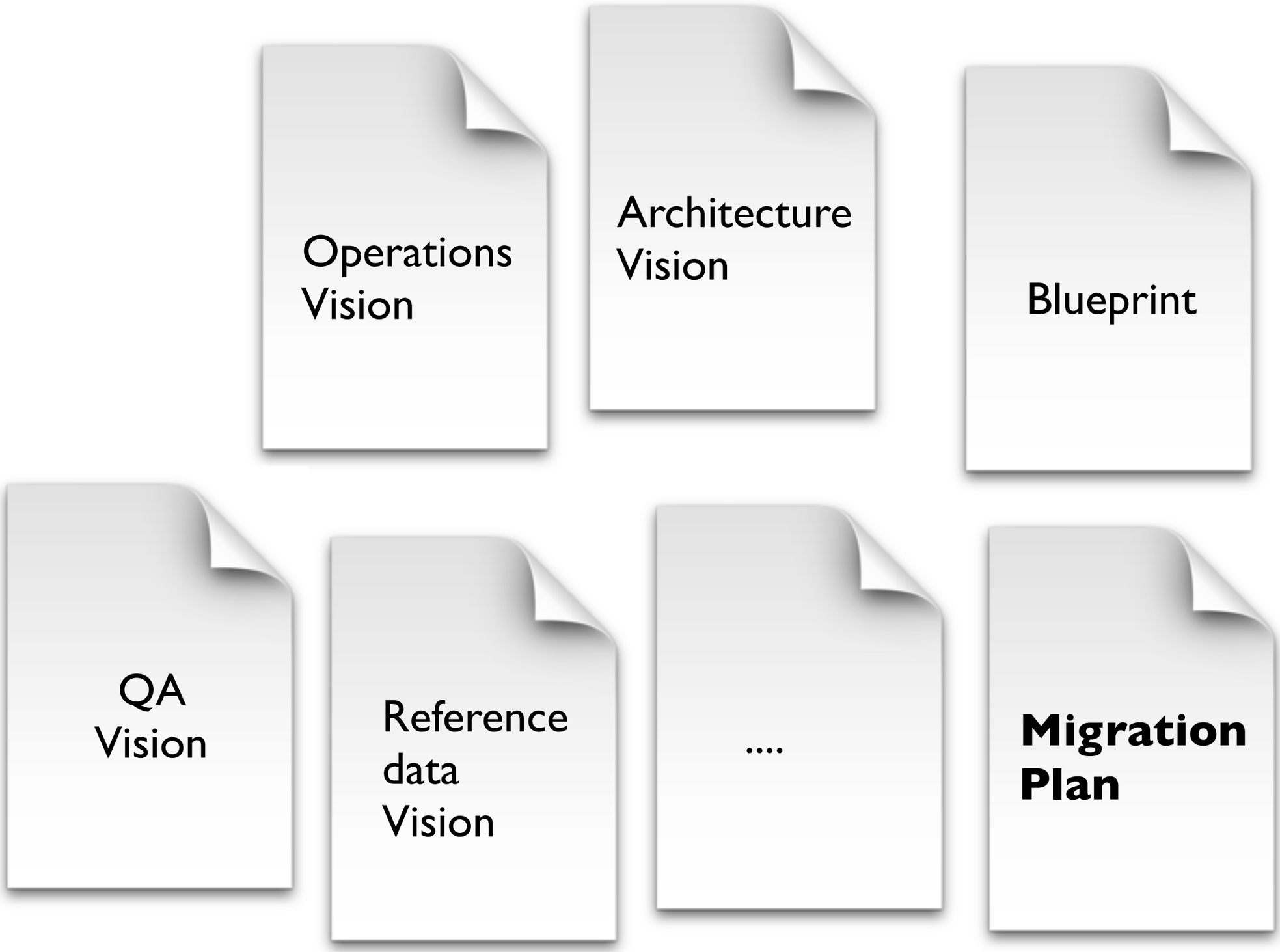
New (as defined about 5 years ago)



HaMIS



Lots of documents

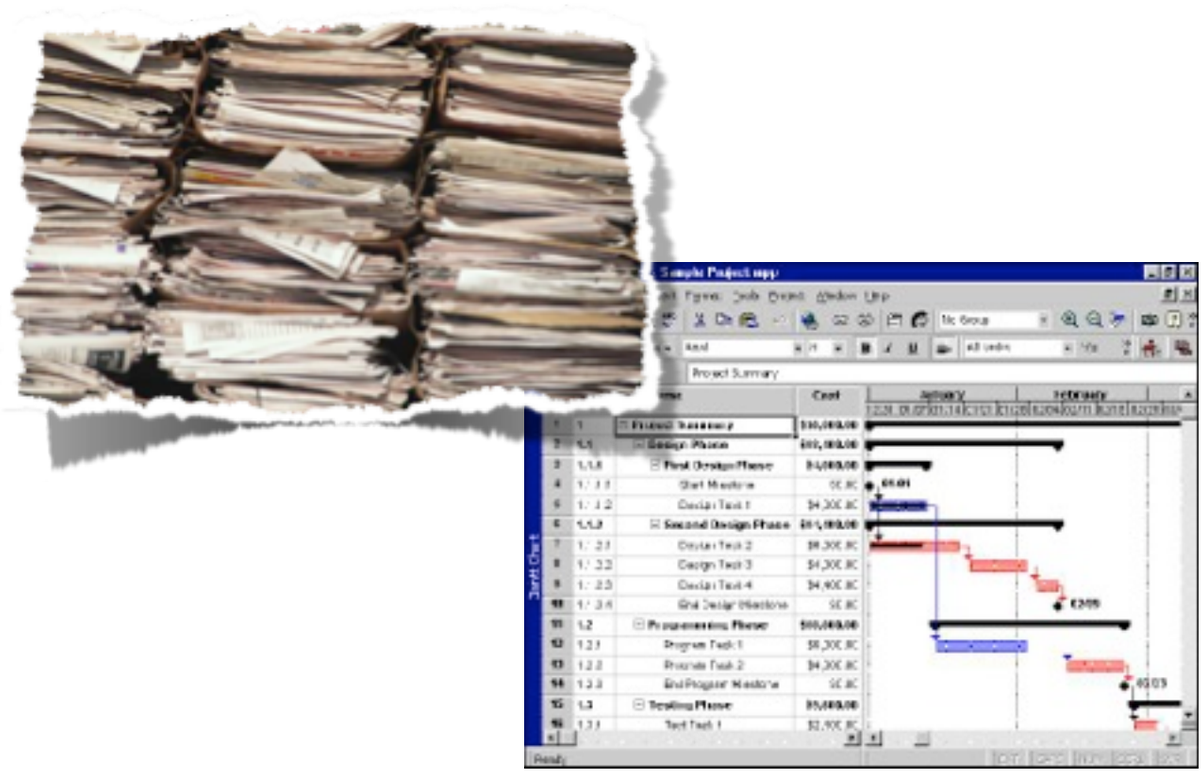


We just need...

100+ identified use cases

Project Start Architecture

Q&A department



Architecture with 4 layers & 6 subsystems

BPEL process server

6 architects

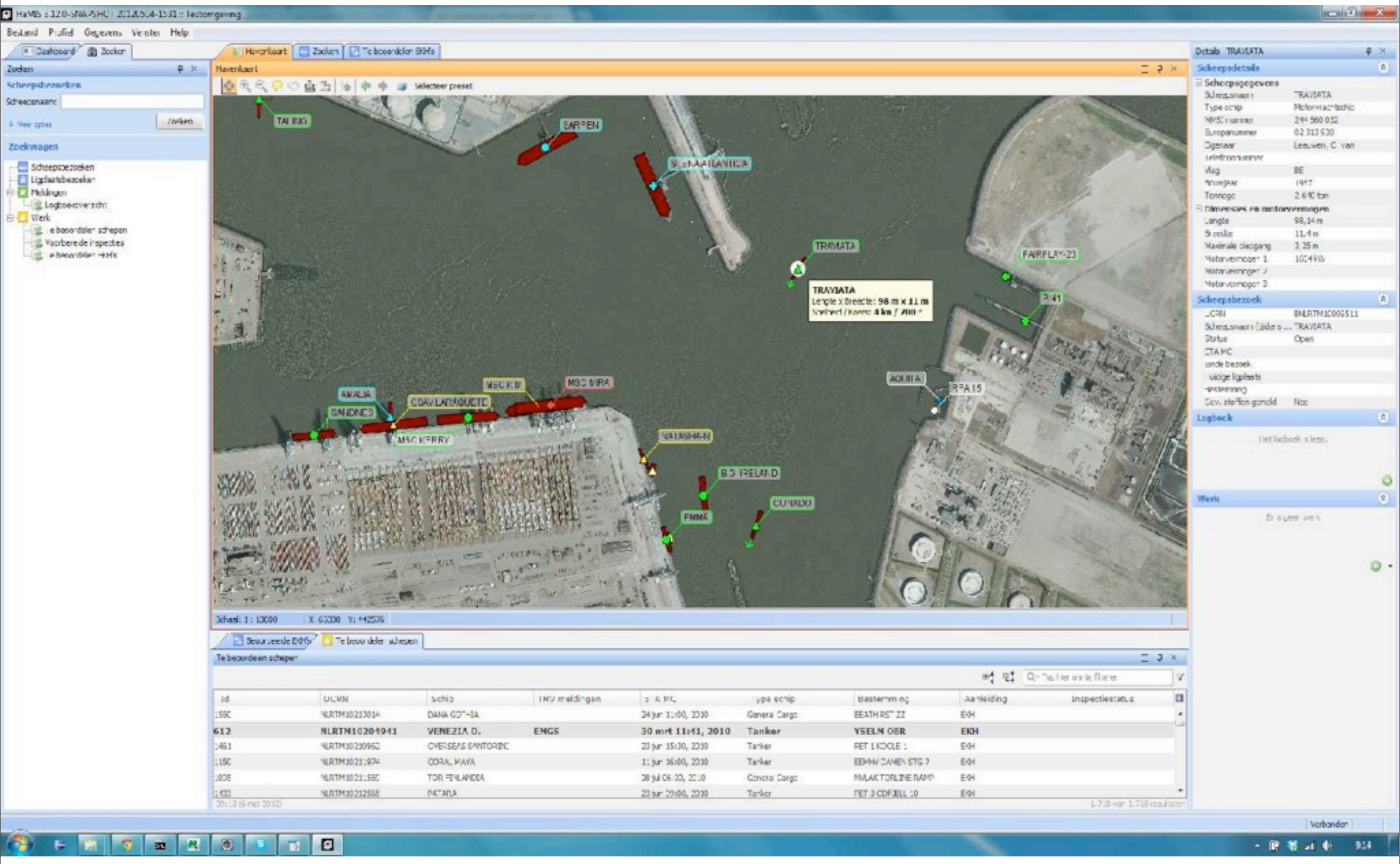
RUP and Scrum but

800+ architectural decisions

1 Use Case in 6 months

(Re)Started

* Ship inspections process
* Little risk
* **Significant business value**
* Migration stuff completely postponed
* Agile / Scrum introduction
* One sprint later, useful functions delivered in **production**



Viktor Grgić @vgrgic, LeanArch.eu

Welcome to double legacy!

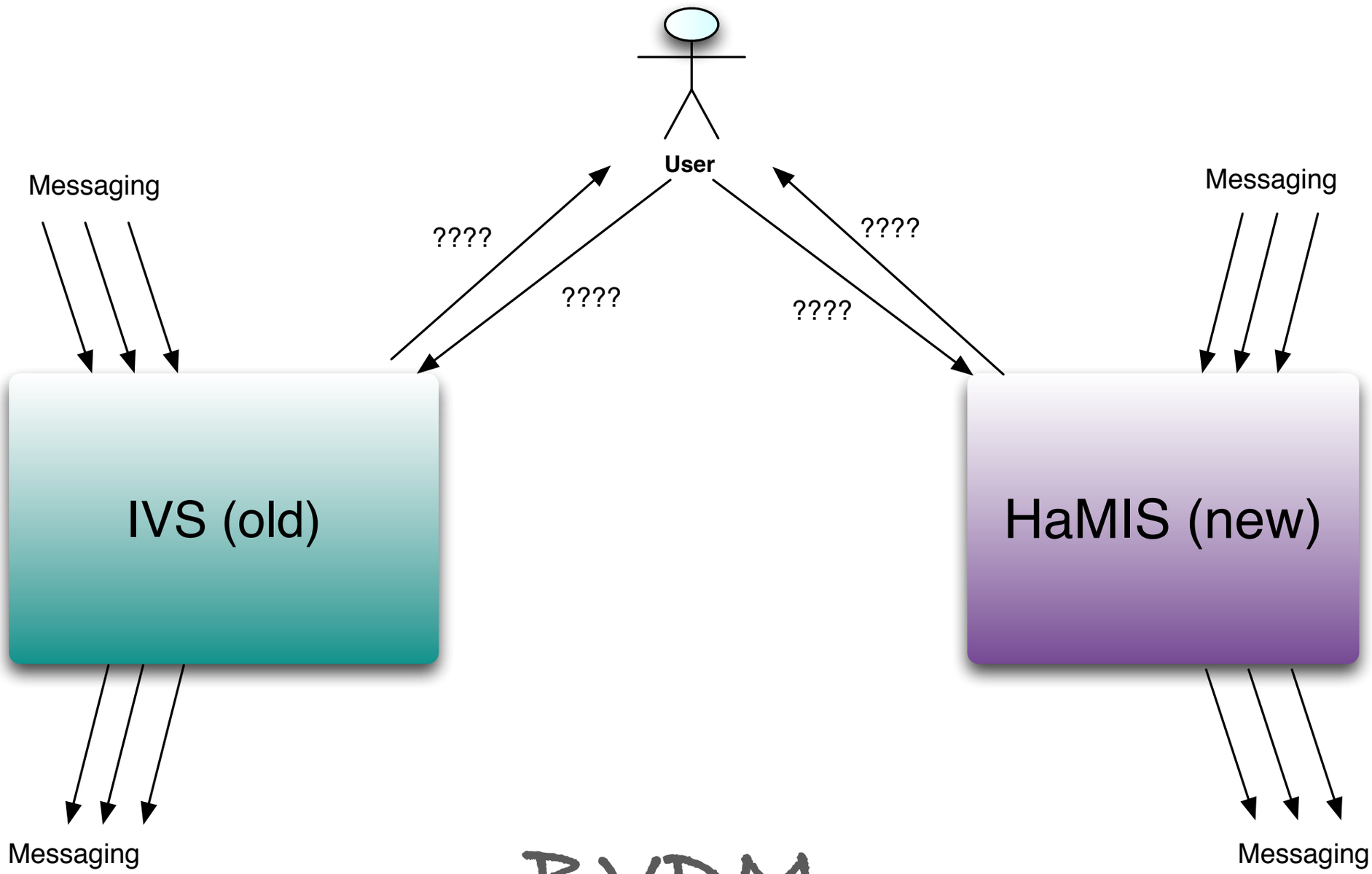
Any decision that can be postponed is postponed

...also the migration path

Which path to choose?

* Should we do all reporting first in the new system?
* Which part of the system should we replace first?
* How do we prevent a big bang?
* Where to start?
* **We followed ships in harbour!**

Migration without a plan

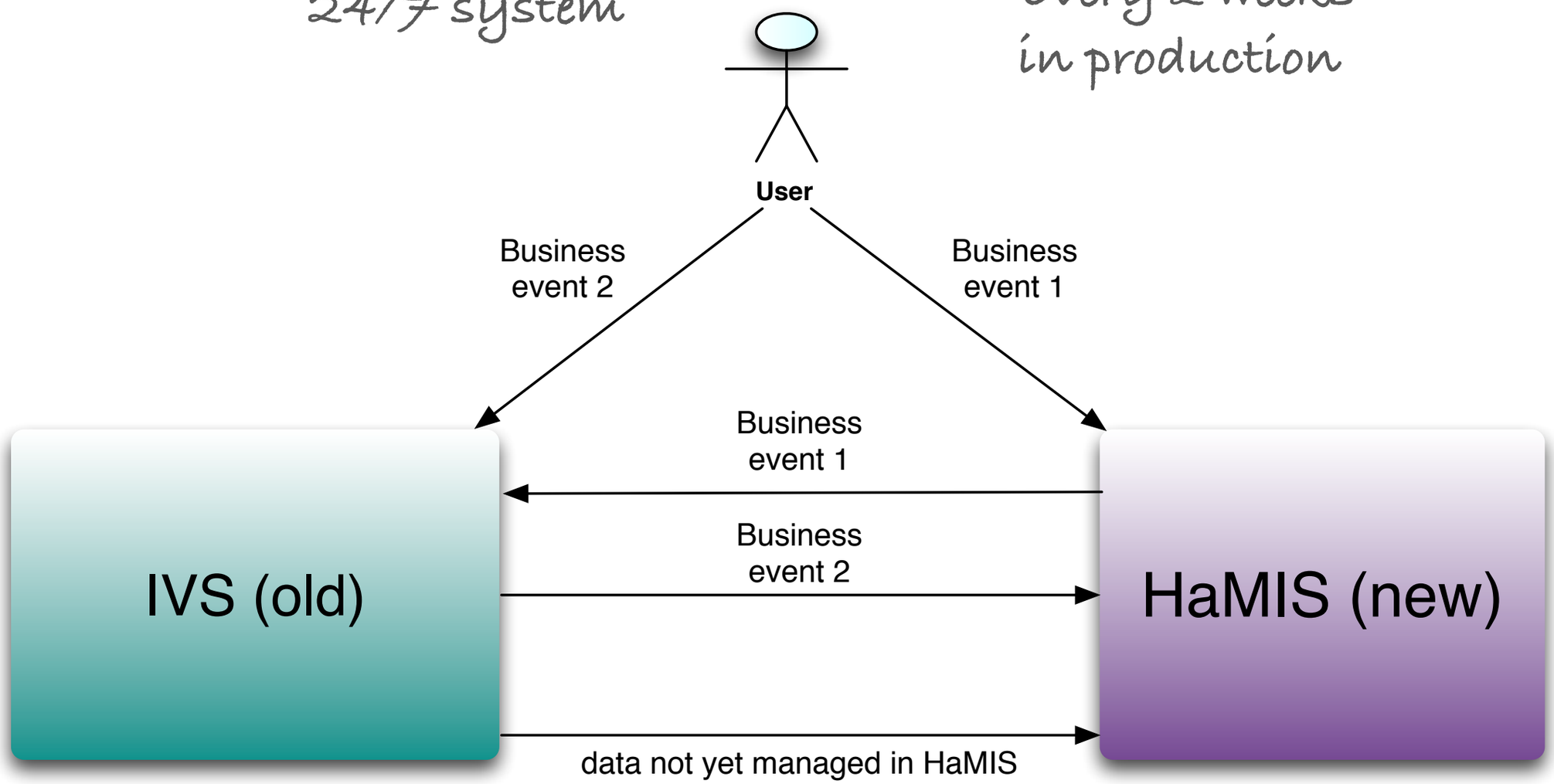


BVDM

Migration = Strangulation

24/7 system

Every 2 weeks
in production



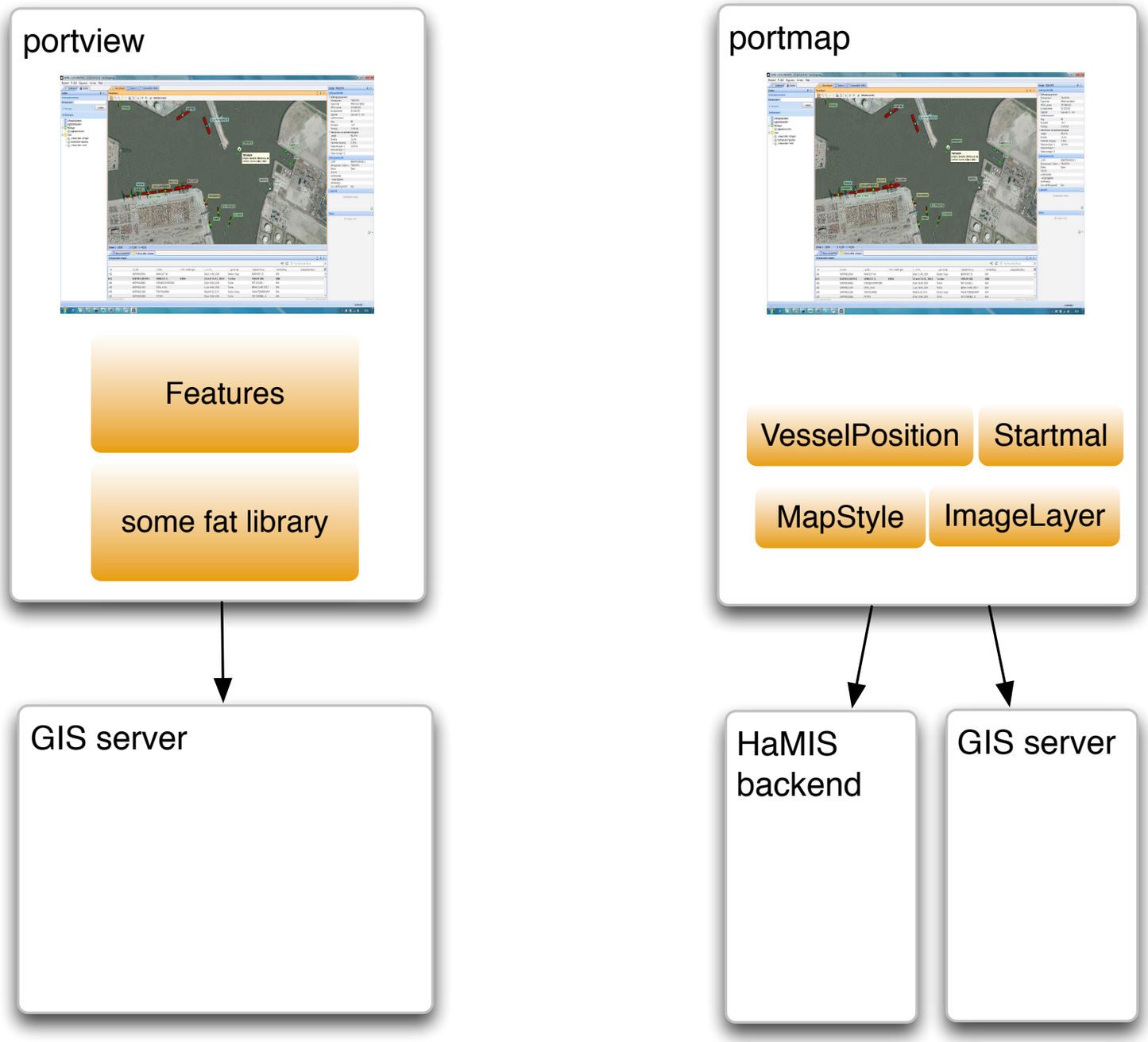
"The most important reason to consider a strangler application over a cut-over rewrite is reduced risk." - Martin Fowler, StranglerApplication

We banned term:
Generic or Reusable
on every level

Aren't you creating a mess?

* One team delivers the simplest possible solution
* Another team delivers another similar even simpler possible solution :-)
* ...and maybe third one.
* Then, we get together and talk about what you learned and which one is really the best in **this context**.

Example


portview
Features
some fat library
GIS server
portmap
VesselPosition
Startmal
MapStyle
ImageLayer
HaMIS backend
GIS server

Delivering value with partial foundation or DVWPF :-)

Examples

* Data model with production data is gradually extended and continually refactored
* IBM Websphere replaced by JBoss
* User interface redesigned 2 or more times
* SOAP replaced by binary Hessian
* Core domain model concepts (gradually) redesigned: inspection checklist, vessel visit vs. vessel

Each team is a cross-functional team and has intensive contact with end-users

Some challenges

* Teams want to understand business first, before accepting epics and stories.
* Continuous discussion about priorities
* Making business value SMART.
* Technical complexity is **NOT** a real challenge.
* Should we spend (or waste) money on synchronization vs. postpone going live.

65 sprints later;
Almost all of them
delivered business
value (functions
in production)

Software Quality ★★★★★

Agile (Fowler) ?? ★★★

Lessons learned

* Never stop delivering business value.
* Deal with migration gradually / strangulation
* Everything in software can be soft even data / domain model!
* Trust in business / context driven approach above standard / generic solutions
* Involve everyone to solve complex problems

Software quality

Agile Scrum

Business events

DevOps

Liquid architecture

User domain

Context

Thank You !

You work in Hong Kong?
Please contact me!

@vgrgic