Atom Free IT

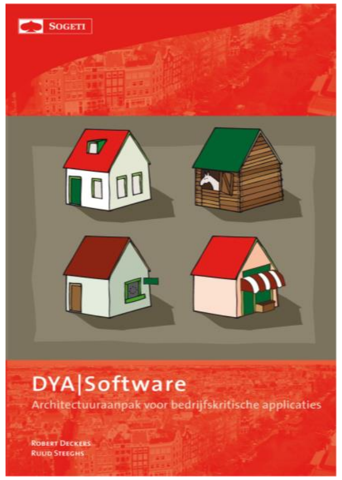
Quality time

An exploration in the land of non-functionals

Robert Deckers V3 201406012

A little about me

- Education:
  TUE informatica, PDEng SW technology
- Jobs:
  Conscription, KISS b.v.,
  Philips Research, Philips Medical,
  Sogeti, Atom Free IT
- Current focus
  Model driven development
  Architecture
  Analysis/Requirements
- Expertise:
  Modelling, Model Driven Development,
  Architecture:
    Research and innovation
    System, software, information, enterprise
    Consultancy, coaching
  Requirements engineering & management
  Method engineering



SOGETI

DYA|Software

Architectuuraanpak voor bedrijfskritische applicaties

Robert Deckers
Ruud Steeghs

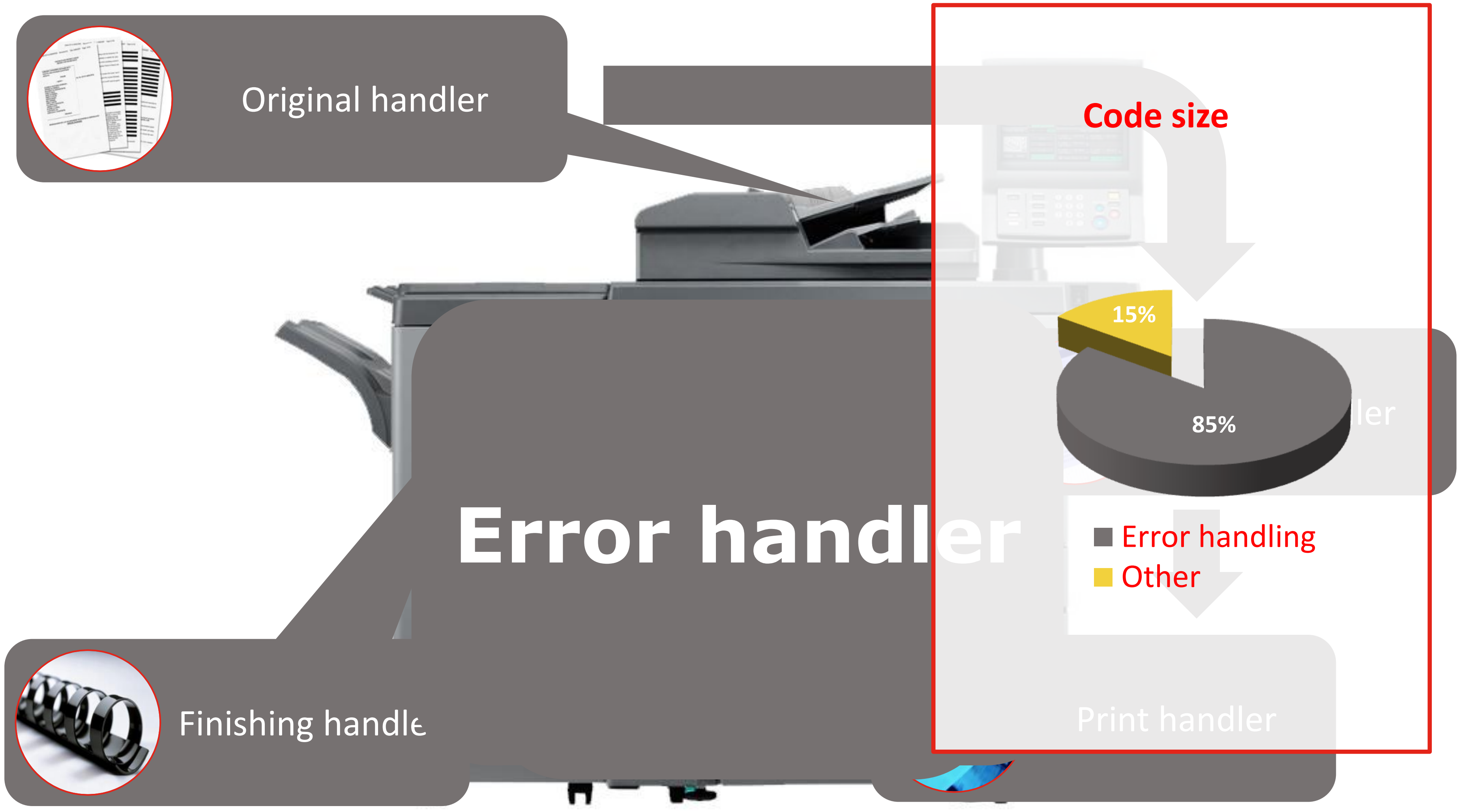
Ufep.webs.com
www.evengoeievrienden.nl
Whatever… Just create

robert.deckers@AtomFreeIT.com
www.AtomFreeIT.com

Agenda

- An example: copier
- Finding aspects
- Quality attributes
- Sorry, but we need a language
- Towards a shared and extendible framework

An example: a copier



Will it copy… right?

Copier software

Test environment

→Design for the dominant aspect

But,

HOW?

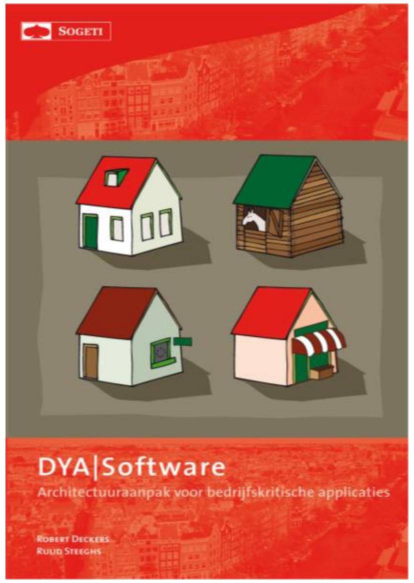
→ Original desig coding effort

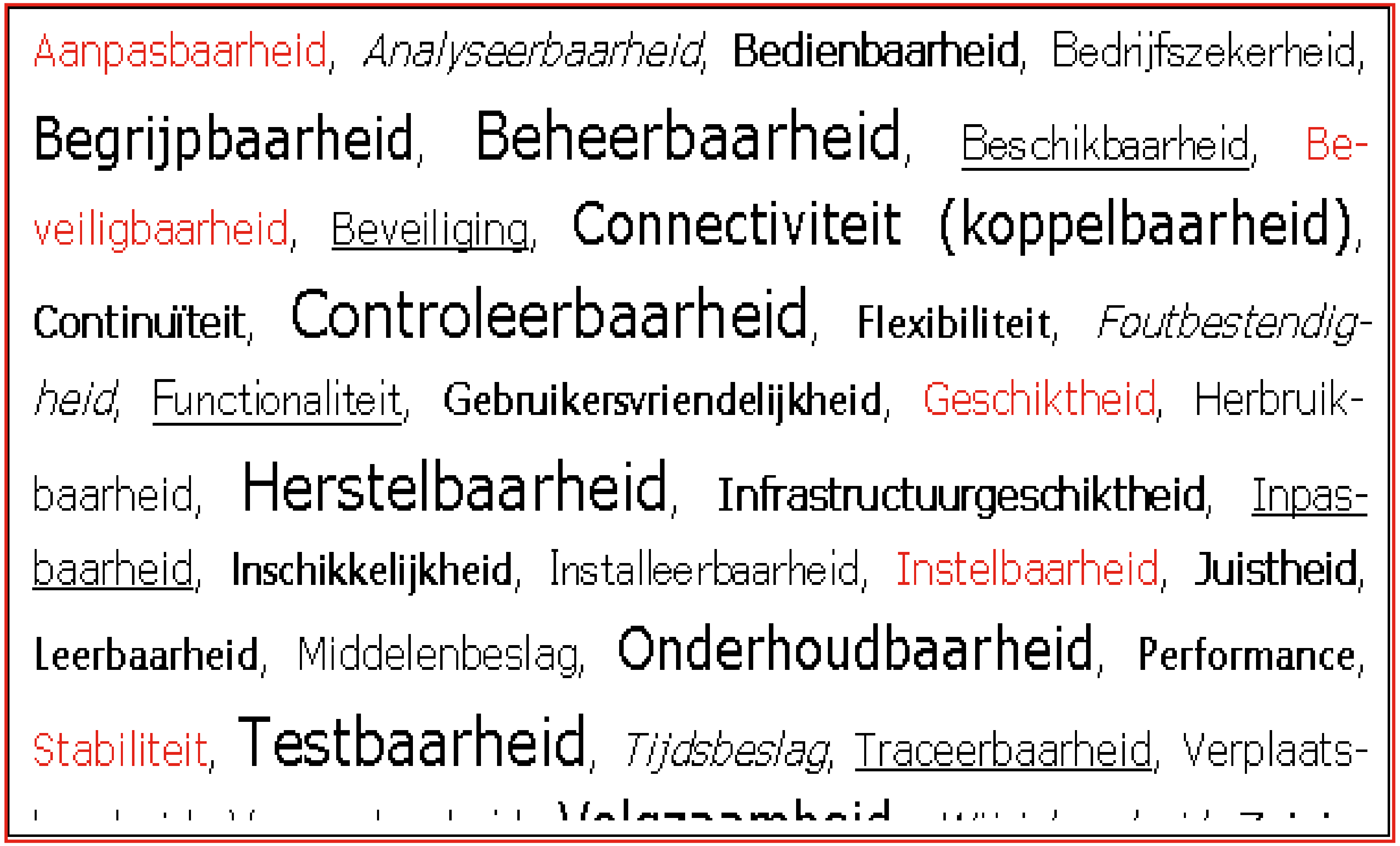
→ $Reliability > $copying

It begins with finding aspects

- **From experience:**
  - A standard list/framework

Look into an existing list…





Aanpasbaarheid, *Analyseerbaarheid*, **Bedienbaarheid**, Bedrijfszekerheid, Begrijpbaarheid, Beheerbaarheid, Beschikbaarheid, Beveiligbaarheid, Beveiliging, Connectiviteit (koppelbaarheid), **Continuïteit**, Controleerbaarheid, **Flexibiliteit**, *Foutbestendigheid*, Functionaliteit, **Gebruikersvriendelijkheid**, Geschiktheid, Herbruikbaarheid, Herstelbaarheid, **Infrastructuurgeschiktheid**, Inpasbaarheid, **Inschikkelijkheid**, Installeerbaarheid, Instelbaarheid, **Juistheid**, **Leerbaarheid**, Middelenbeslag, Onderhoudbaarheid, **Performance**, Stabiliteit, Testbaarheid, *Tijdsbeslag*, Traceerbaarheid, Verplaats-

ISO25010 (f.k.a. ISO9126)

(Sub)Characteristic
Functional suitability
Functional completeness
Functional correctness
Functional ap
Perform
Ir
Usability
Appropriateness re
Learnability
Operability
User error protection
User interface aesthetics
Accessibility

Reliability
Maturity
Availability
Fault tolerance
Testability
Portability
Adaptability
Installability
Replaceability

But,

How to select?

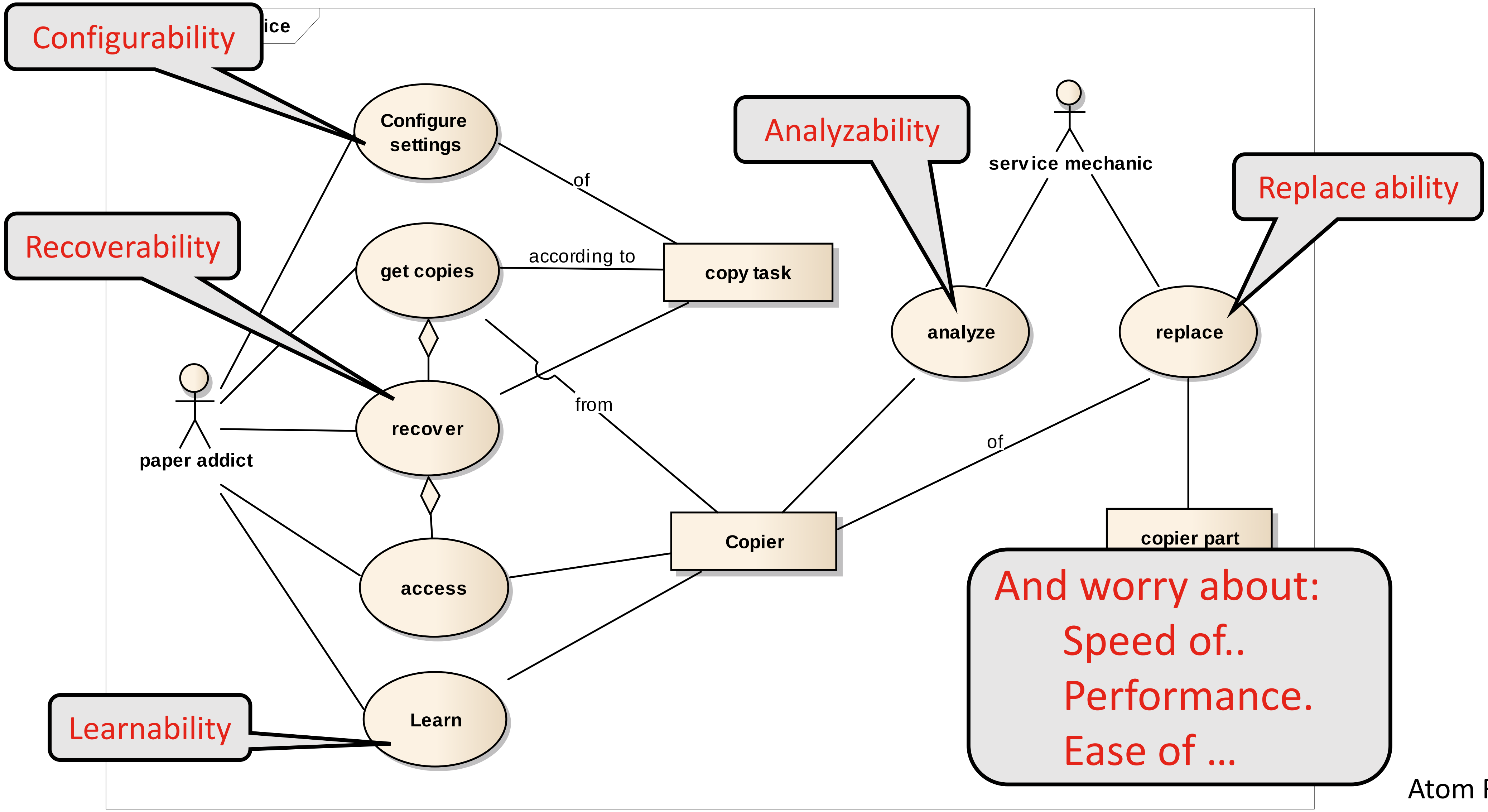
What do I miss?

Where is the coherence?

Finding aspects

- **From stakeholder:**
  - Interaction: Things you do with the system
  - Tasks: Activities that involve the system
  - Concerns: Things you worry about

From use case to quality attributes



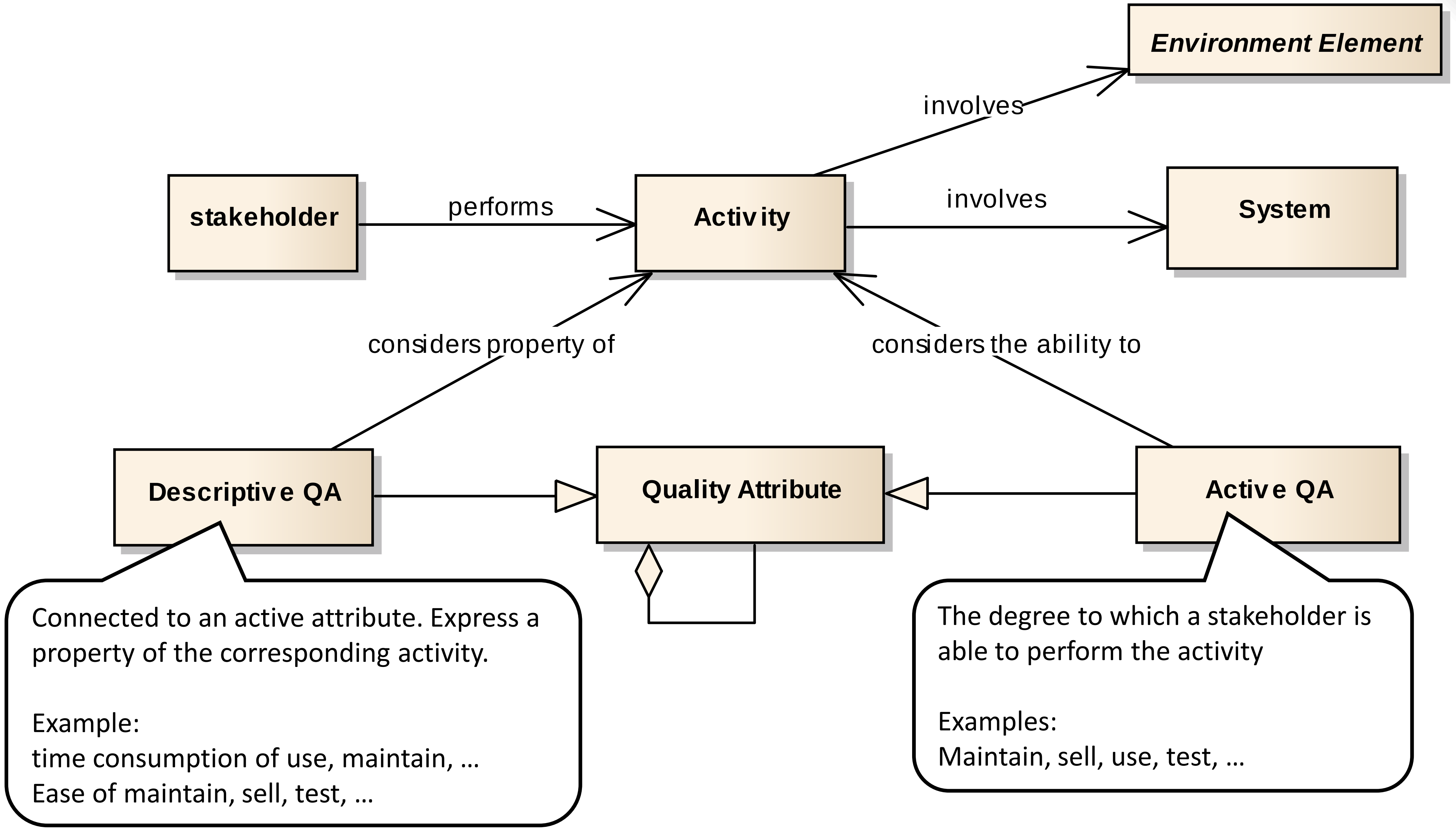
And worry about:

Speed of..

Performance.

Ease of …

Active and descriptive quality attributes



Connected to an active attribute. Express a property of the corresponding activity.

Example:
time consumption of use, maintain, …
Ease of maintain, sell, test, …

The degree to which a stakeholder is able to perform the activity

Examples:
Maintain, sell, use, test, …

Finding aspects

- **From experience:**
  - According to a system expert or a domain expert
  - Things the architect thinks and talks about in his work
  - *A standard list/framework*
- **From stakeholder:**
  - *Interaction: Things you do with the system*
  - *Tasks: Activities that involve the system*
  - Concerns: Things you worry about
- **The system:**
  - Things the system does
  - Things the system guards

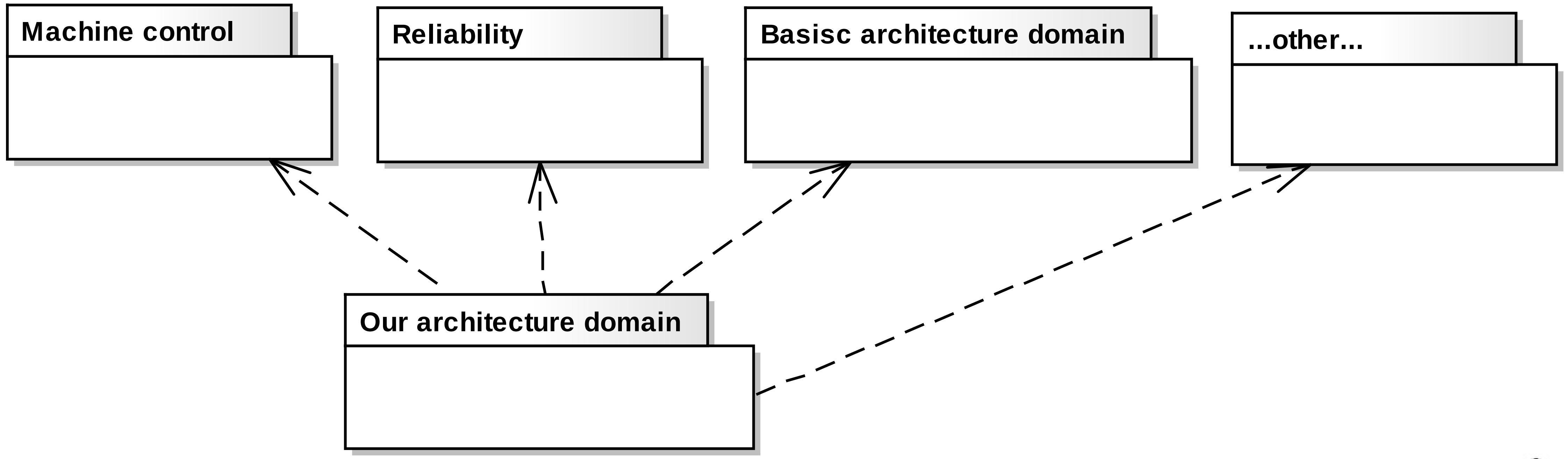
Every answer to HOW? needs a language

- With concepts to:
  - Express requirements
  - Reason with
  - Express solutions
  - Implement
- In order to:
  - Make trade-offs between aspects
  - Integrate solutions
  - Verify solutions
  - Predict an prove quality
  - …

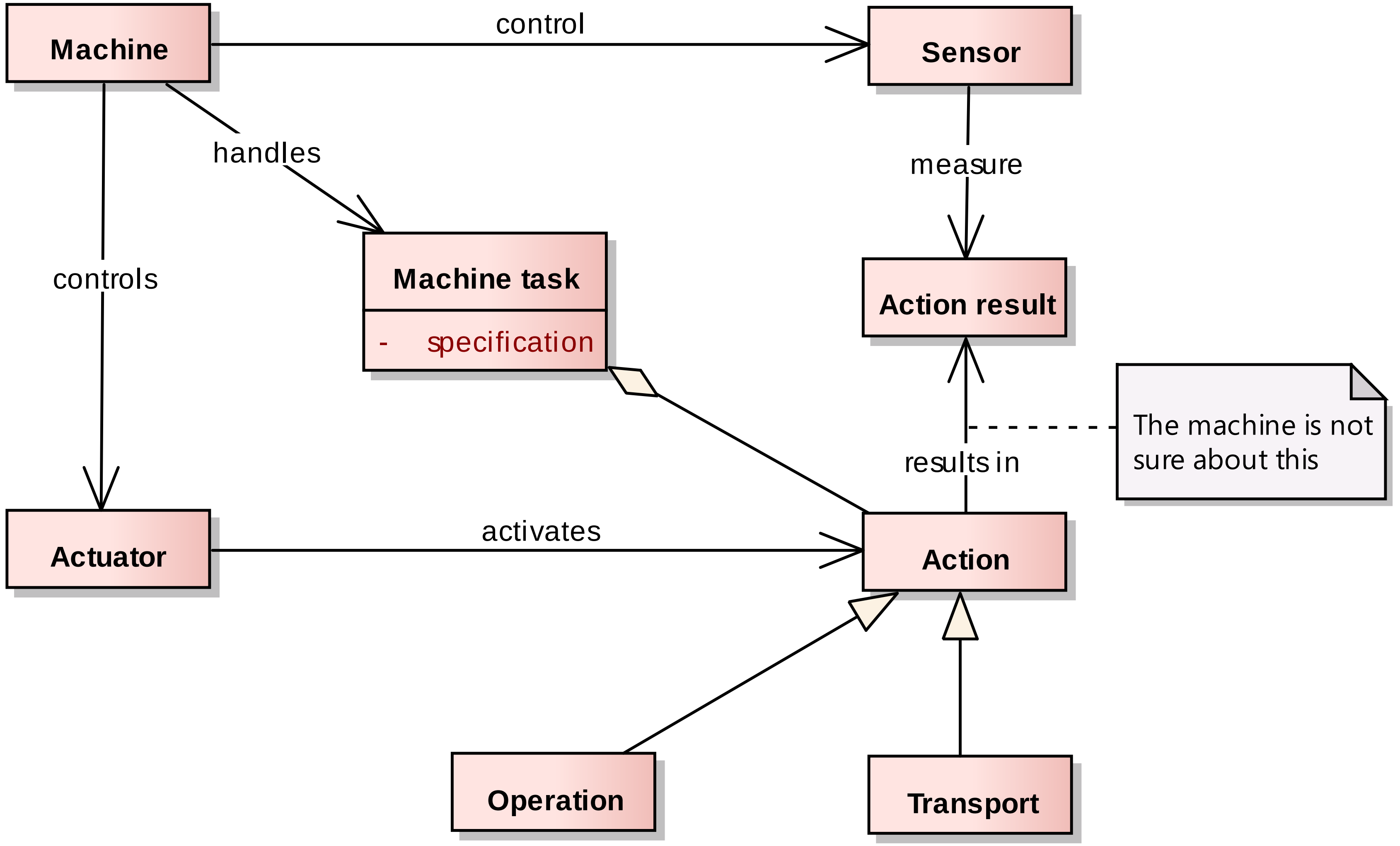
Method:

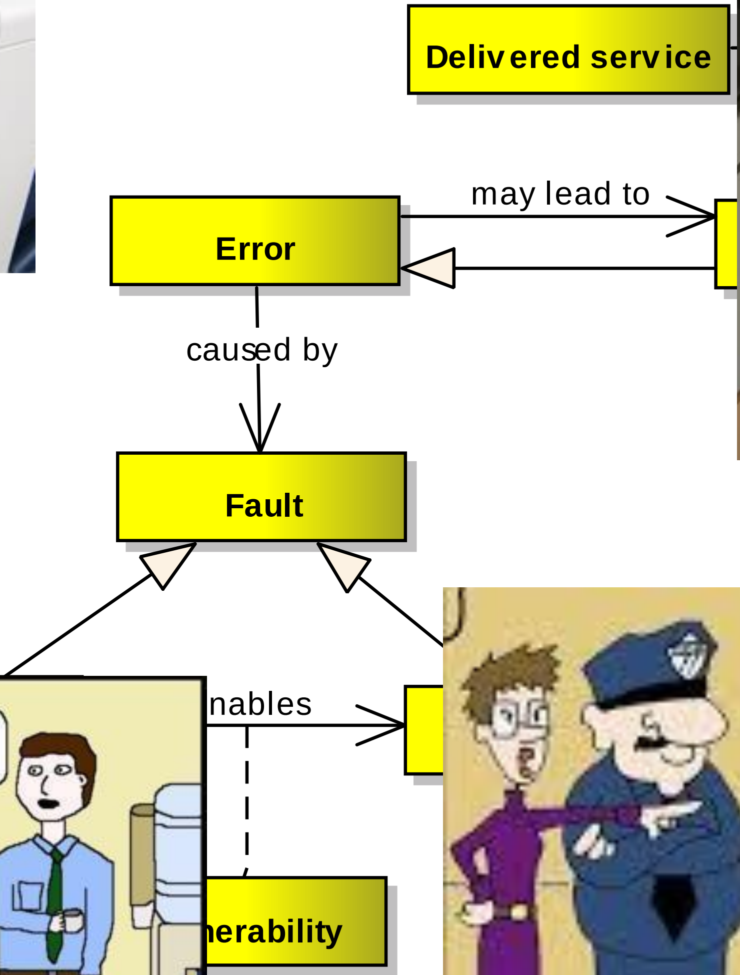
- **Concepts**
- **Notation**
- **Grammar**
- **Guidance**

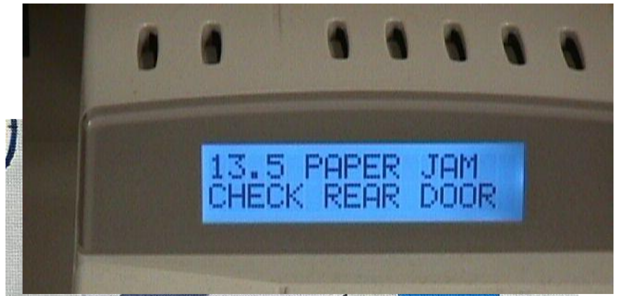
Domains intertwine

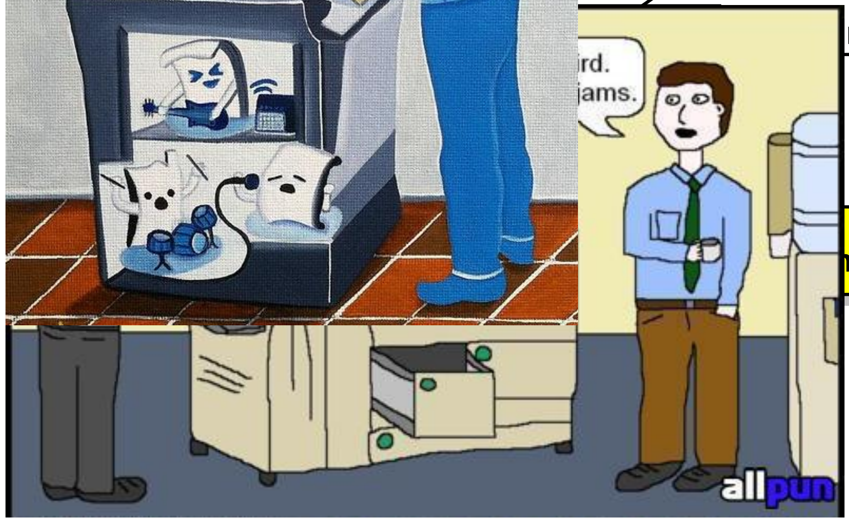


Domain model for machine control

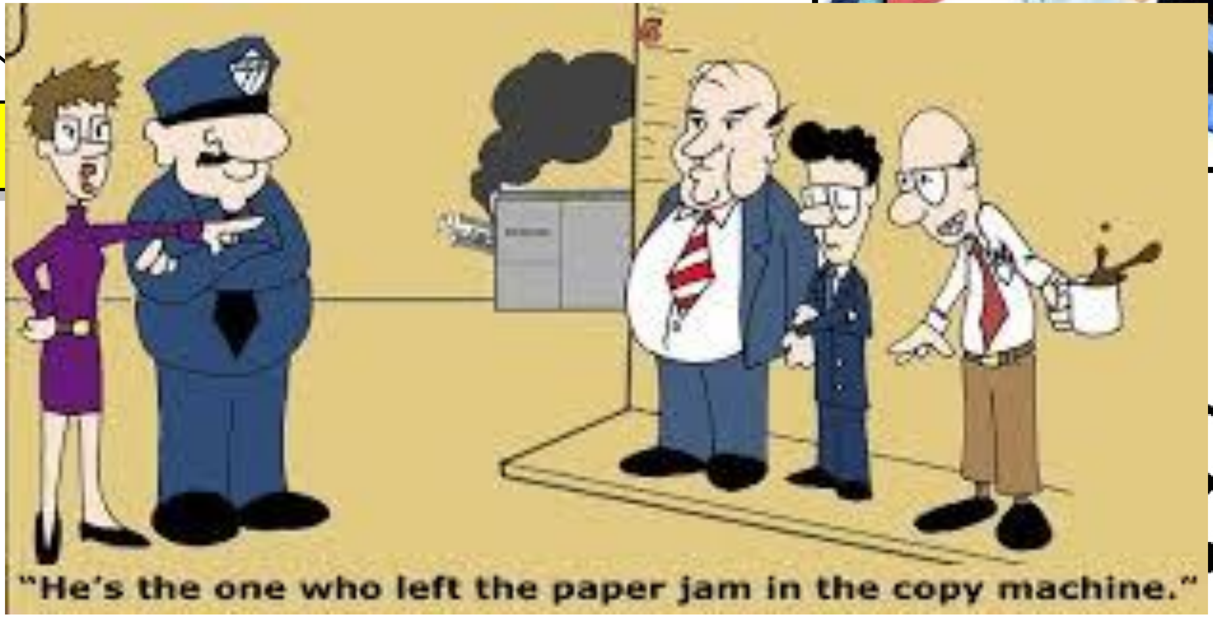



Delivered service
Error
may lead to
caused by
Fault
nables
erability

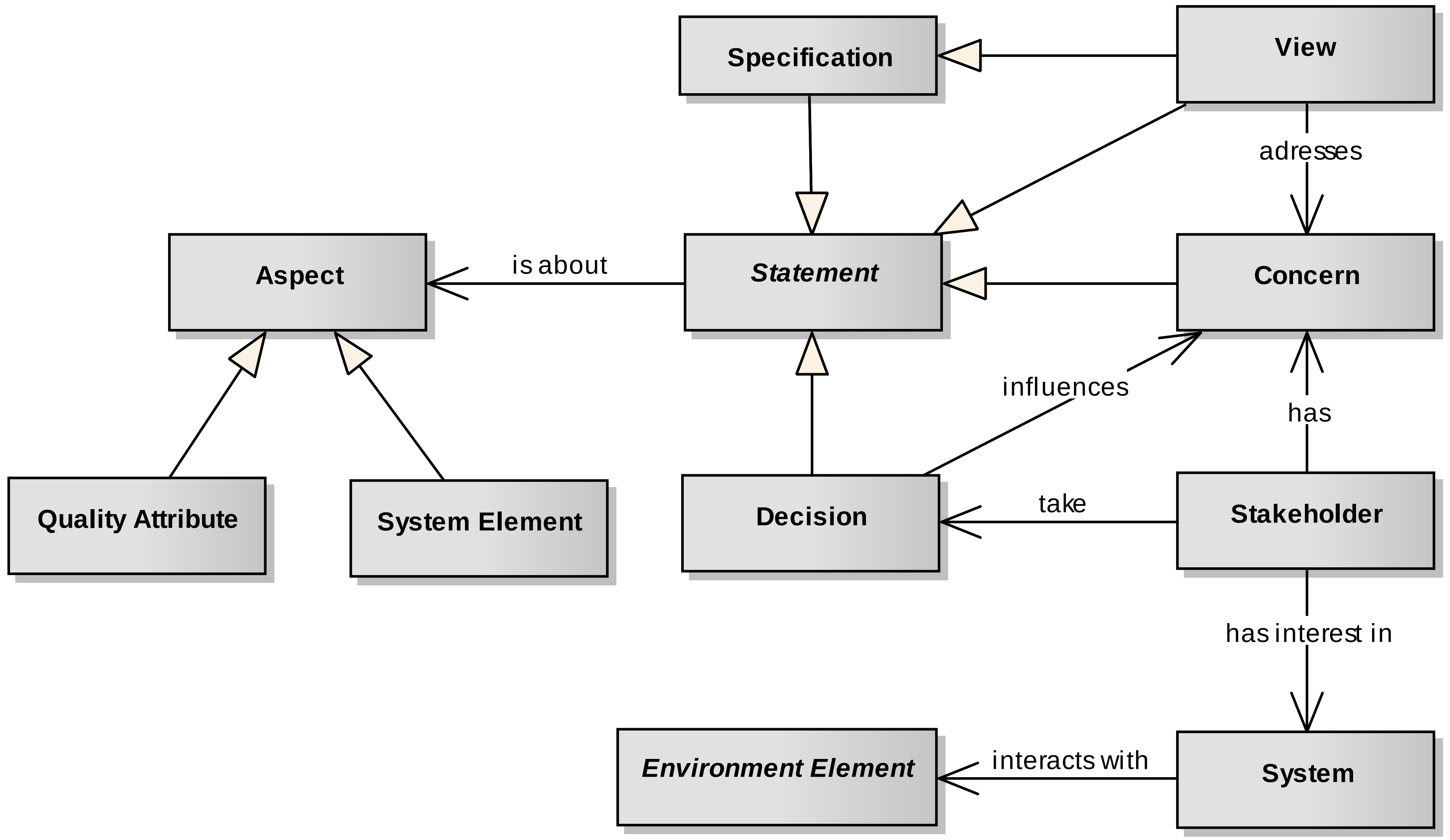

13.5 PAPER JAM
CHECK REAR DOOR


rd.
jams.
allpun

paper jam

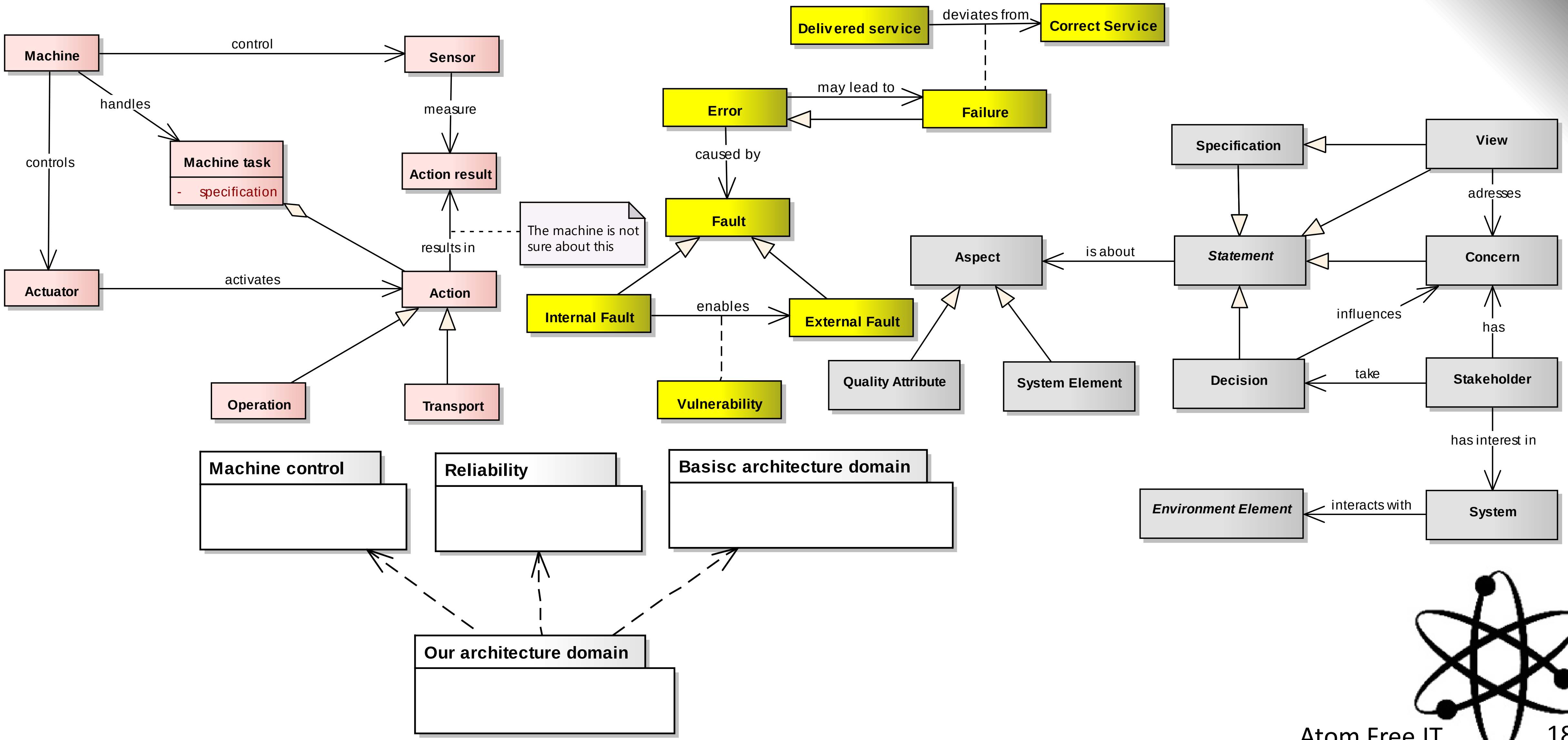

"He's the one who left the paper jam in the copy machine."

The architecture domain

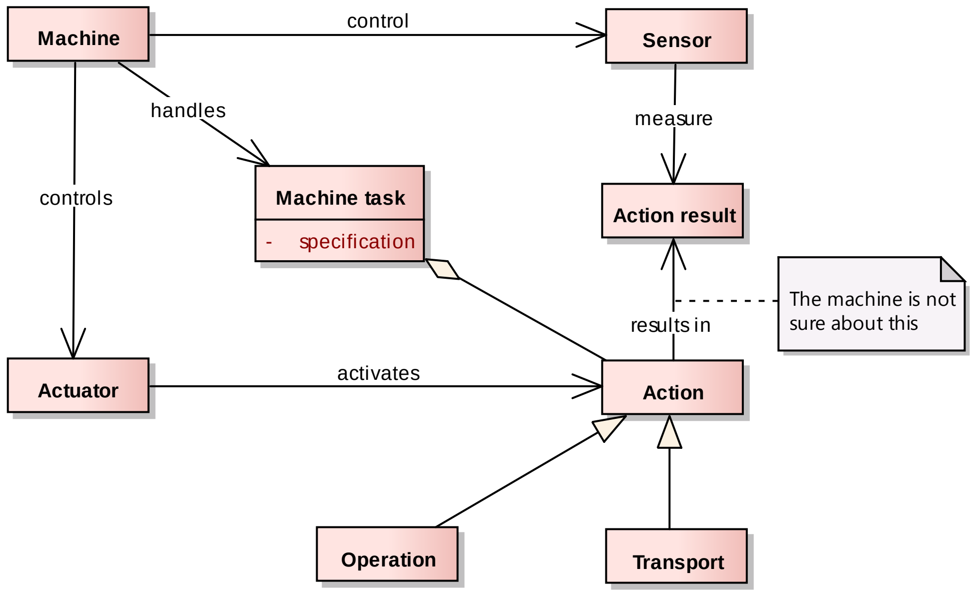


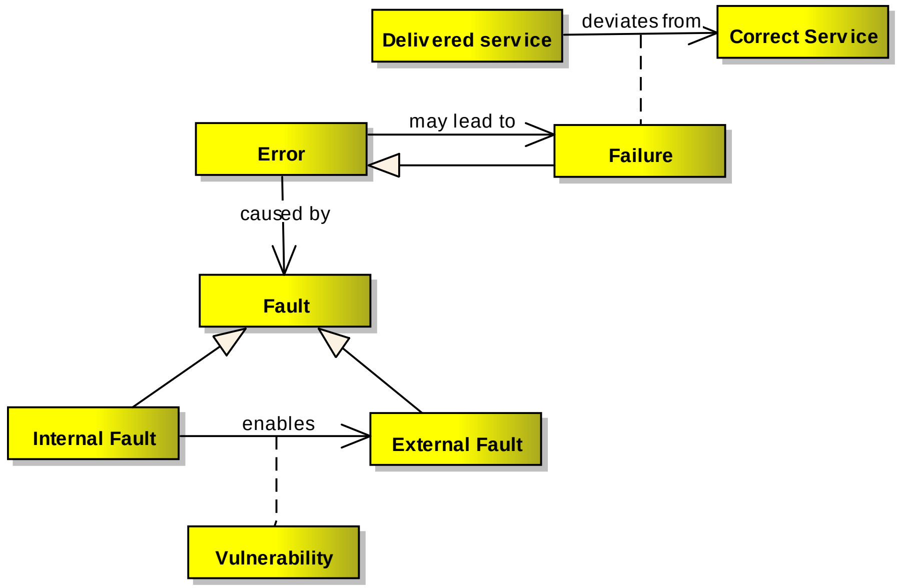
From: DYA|Software, based on ISO/IEC/IEEE 42010/ IEEE1471

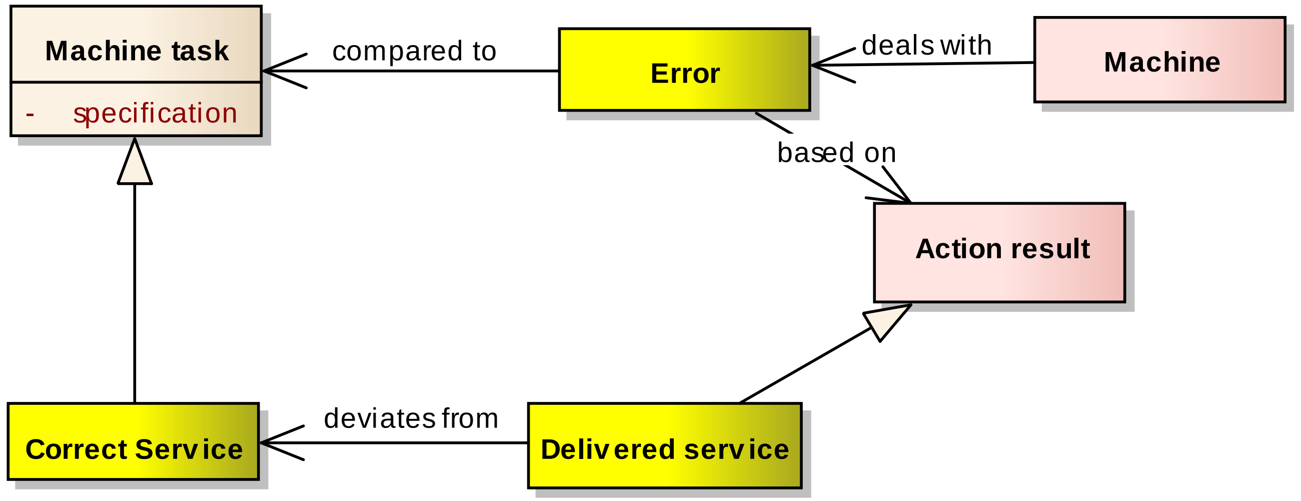
Connect domains



Connect reliability with machine control







Connect architecture with ….

Homework ☺

A shared and extendible framework

- Framework:
  - Terminology
  - Model based
- Requirements:
  - Standard non-functional requirements
  - Templates
- Designs
  - Reference designs (model driven)
  - Related to non-functional requirements

Continued….

Architectural curiosity:

- How do you express and relate NFRs/qualities and aspects?
- How do you reason about, express, reuse designs?
- How do you guard the consistency of your design?

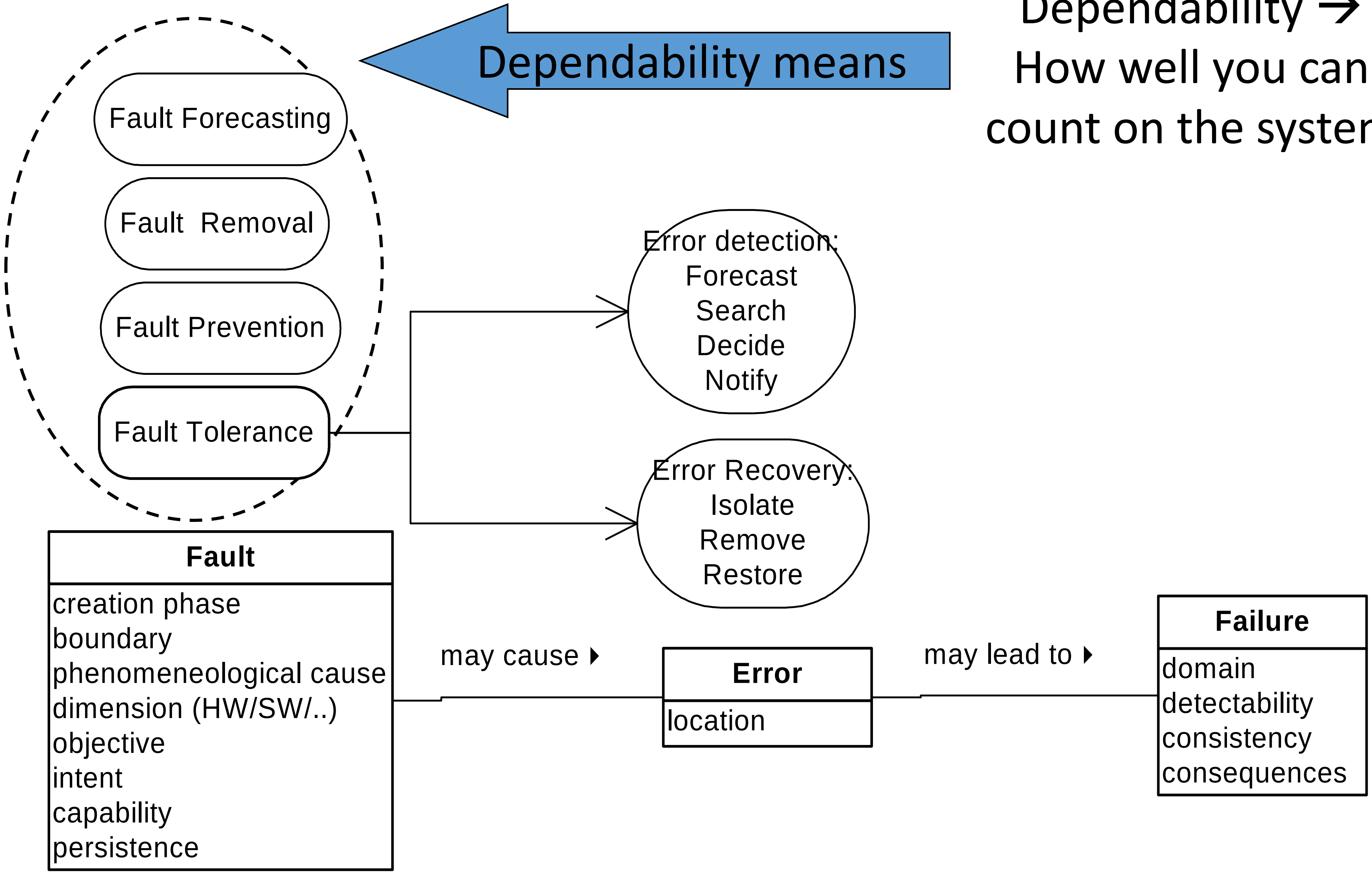
- Join us!
  - Robert.Deckers@AtomFreeIT.com
  - Angelo@delphino-consultancy.nl

Thank you

Atom Free IT automates the automation for true business agility

Dependability concepts

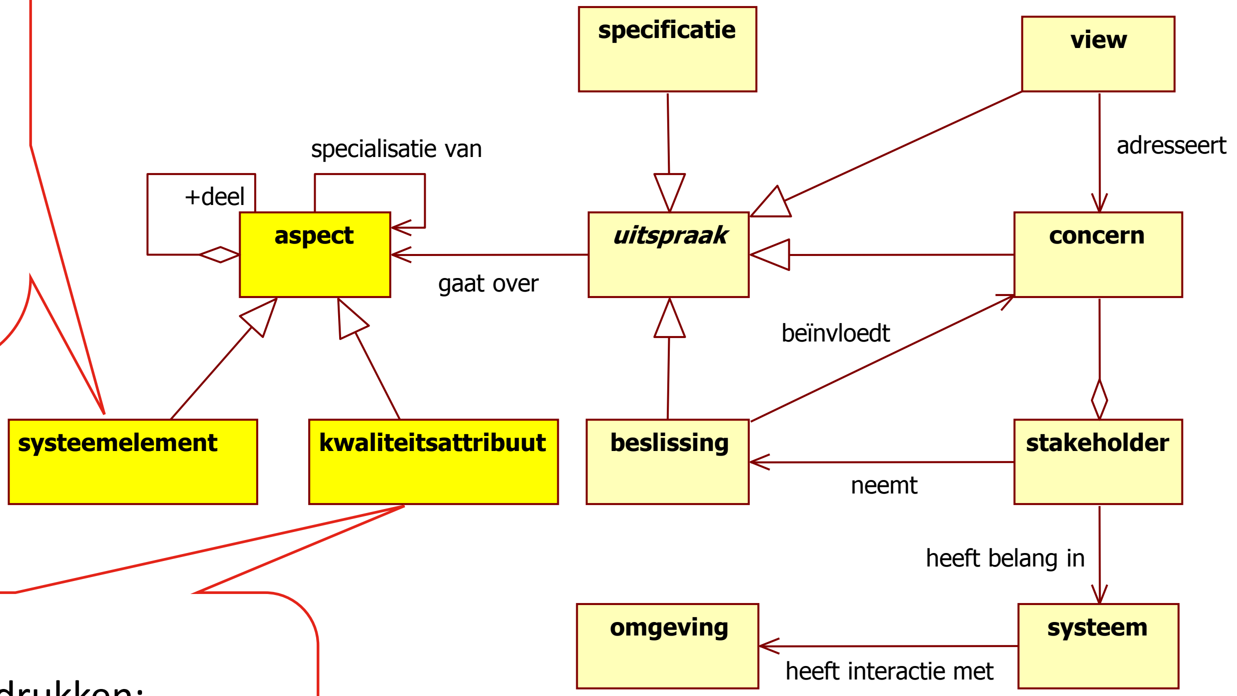
Dependability → How well you can count on the system



Two main types of aspects

Systeemelementen
elementen waaruit systeem bestaat:

- softwarecomponenten
- functies
- interfaces
- ontwerppatronen
- plan van aanpak
- functioneel ontwerp
- projectplan

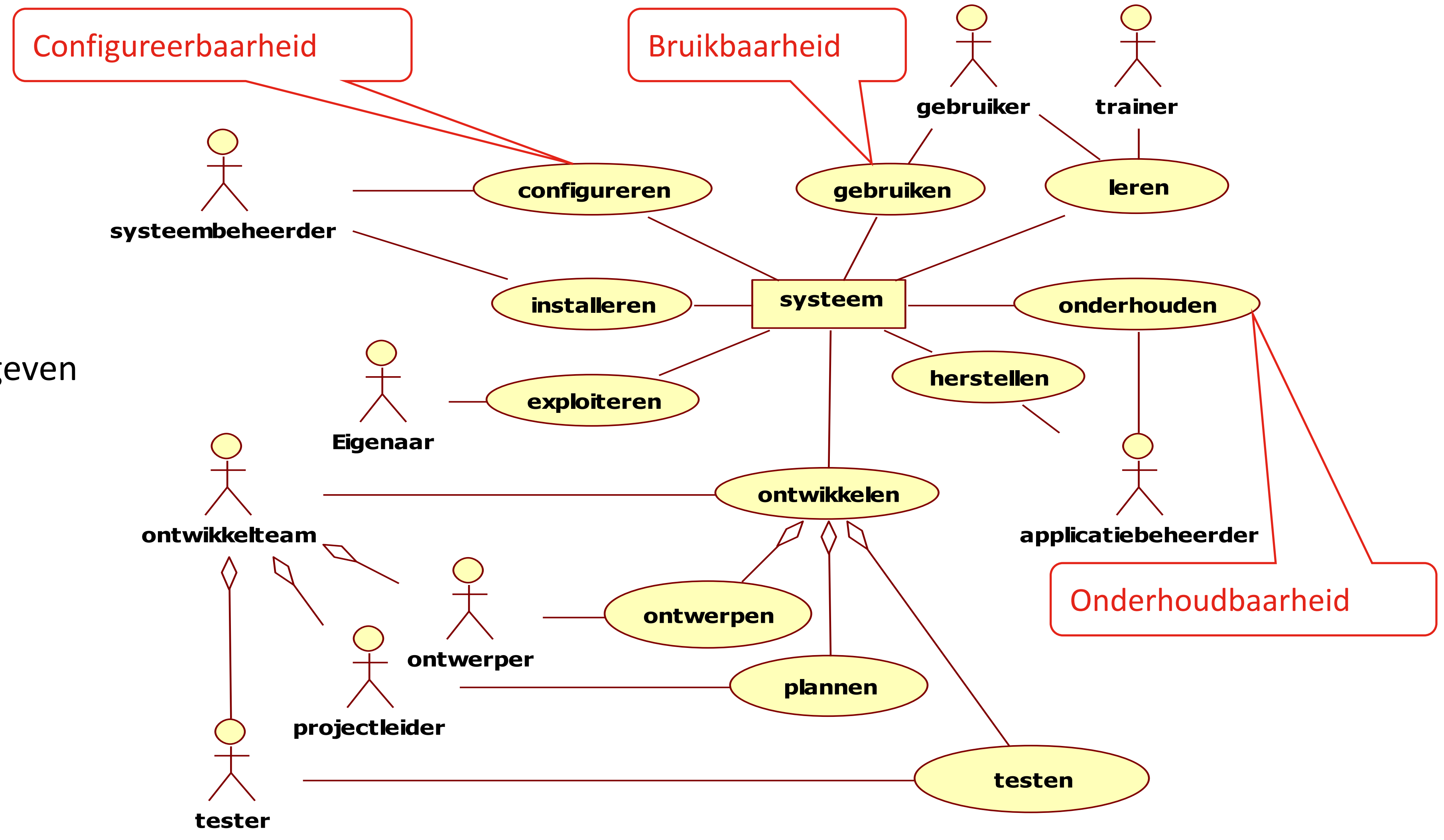


Kwaliteitsattributen
om kwaliteit in uit te drukken:

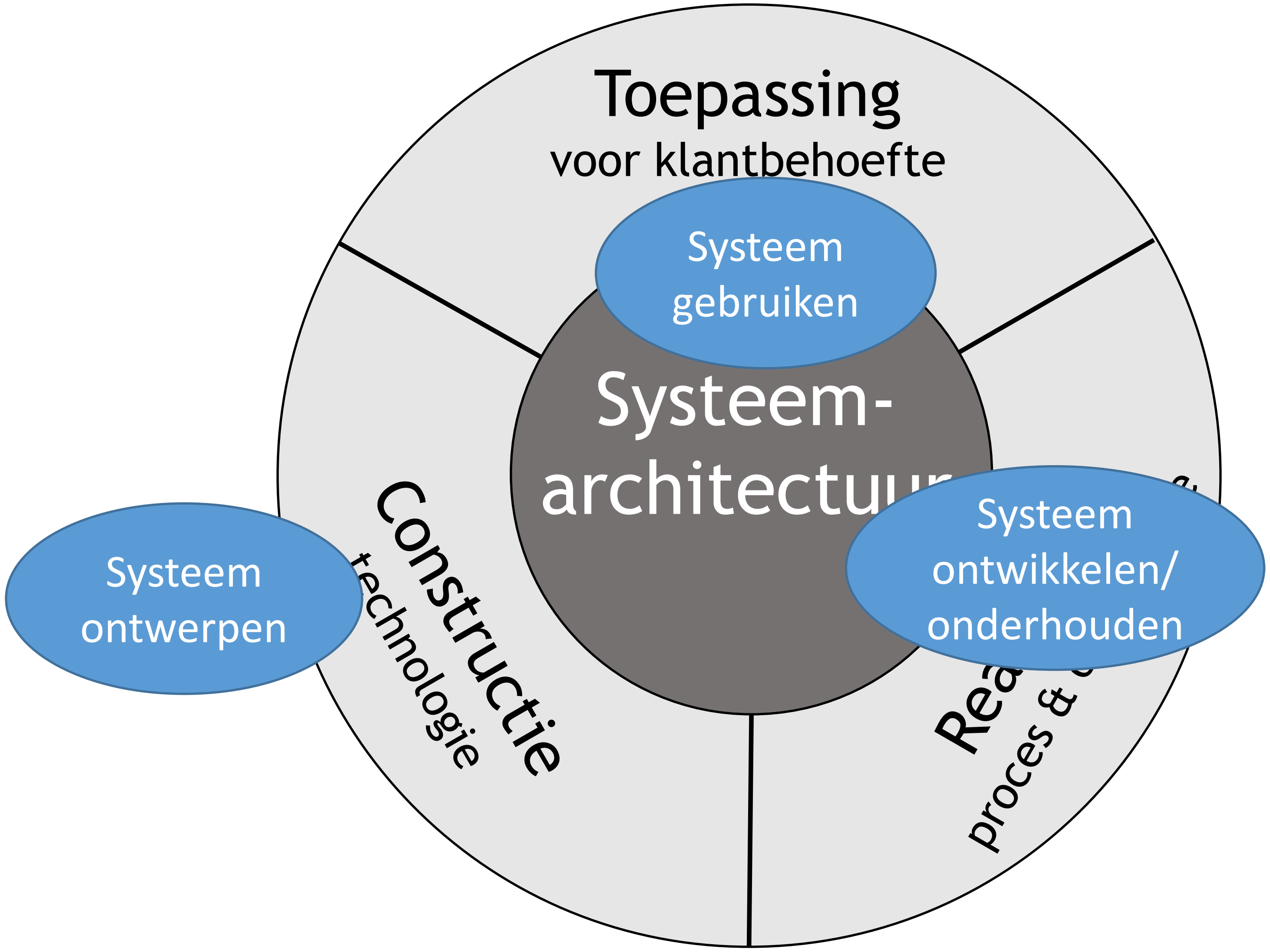
- onderhoudbaarheid
- beschikbaarheid
- stabiliteit
- etc.

Stakeholder activities

Voorbeelden geven



3 hoofdassen


Toepassing
voor klantbehoefte
Systeem gebruiken
Systeem-
architectuu
Systeem ontwerpen
Constructie
technologie
Systeem ontwikkelen/ onderhouden
Rea
proces &