Complementing goal models for adaptive systems

João Pimentel
{ jhcp@cin.ufpe.br}

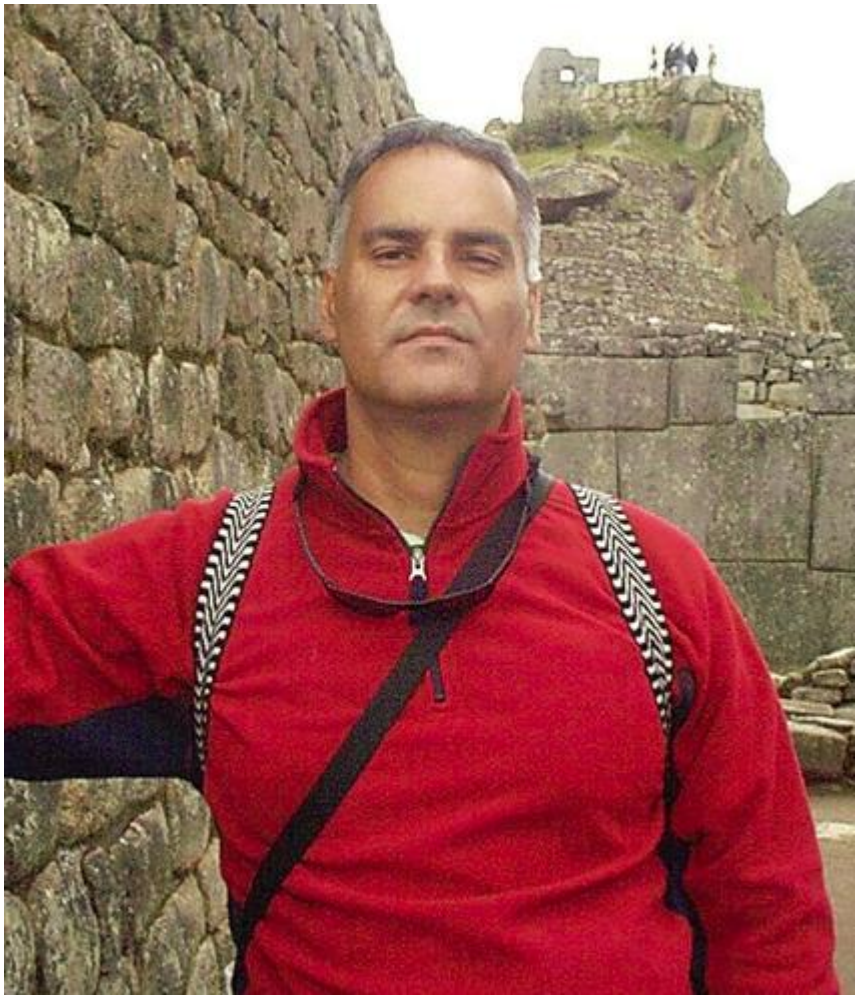
UNIVERSITÀ DEGLI STUDI
DI TRENTO

João Pimentel

- 3rd year Ph.D – UFPE/Brazil
- In trento for 1 year
  - From September 2012

- www.cin.ufpe.br/~jhcp

MsC and PhD with Jaelson



10-months

2010-2011

12-months

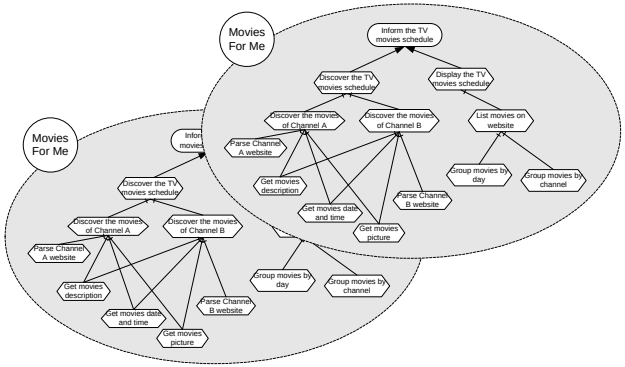
2012-2013

Agenda

- Adaptation metrics
- Futurology
- Failure policy
- From req to arch

I.
ADAPTATION METRICS[1]

Comparing different alternatives



models

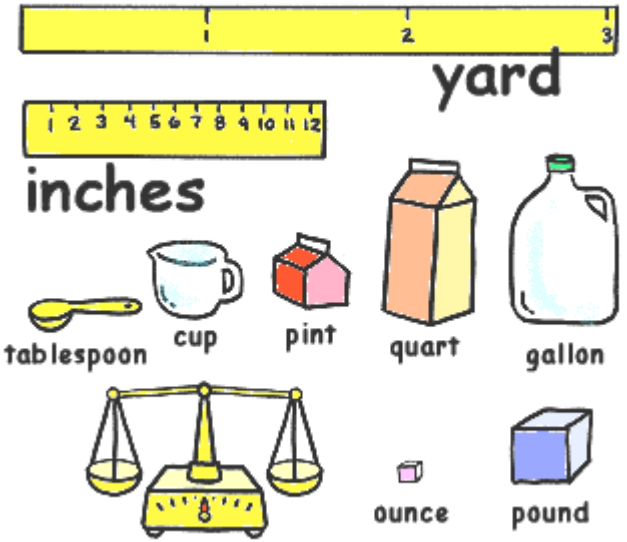
components

decisions

Criteria:

Reusability, reliability, performance, **adaptability**, maintainability...

how to **measure adaptability?**



We build on previous work by Submaranian & Chung:

Architecture Adaptability Index (AAI) =

$$\frac{\text{Sum of the Adaptability Index (EAI) of all architectural elements}}{\text{Total number of elements of the architecture model}}$$

…and…

Software Adaptability Index (SAI) =

$$\frac{\text{Sum of the AAI of all architecture models of the software}}{\text{Total number of architecture models for that software}}$$

But the problem remains: **how to define the adaptability index (EAI) of a particular element?**

We propose to

map the metrics to *i** (iStar) models

and

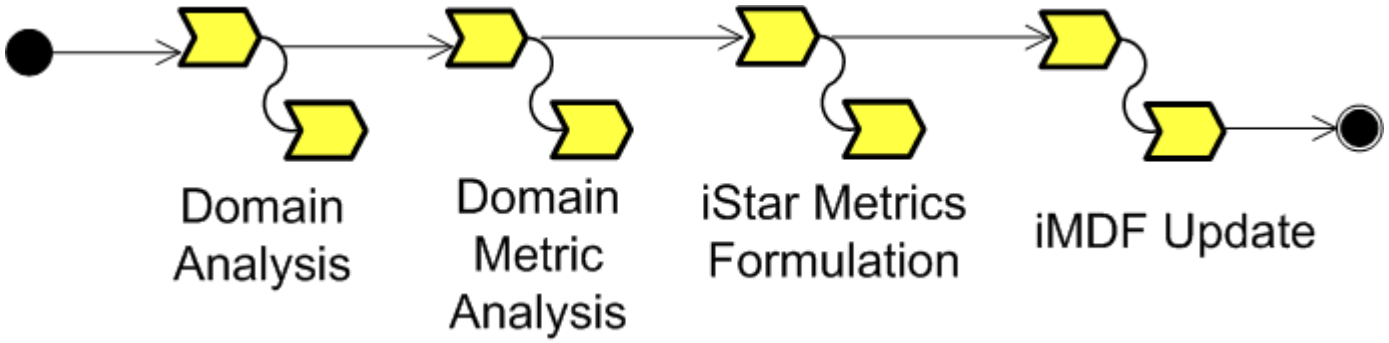
use its richer **expressiveness** to

define EAI

In order to perform the mapping of architectural metrics onto *i**, we use the

*i** Metrics Definition Framework method

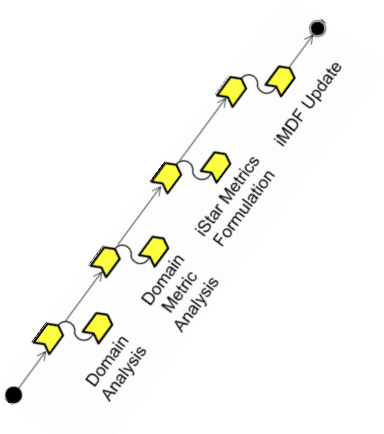
$iMDF_M$



*i** Metrics Definition Framework

*i*MDF

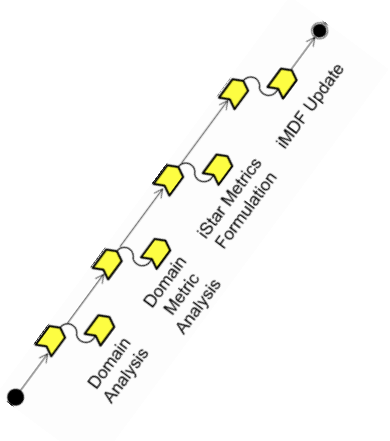
Catalog of patterns



Method

i* metrics in OCL

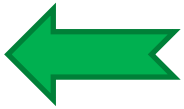
Catalog of patterns



Method

i* metrics in OCL

Modeling tool



*i** Metrics Formulation

Based on the original metrics, the architecture → *i** mapping and the iMDF catalogue of patterns:

Architecture Adaptability Index (AAI) =

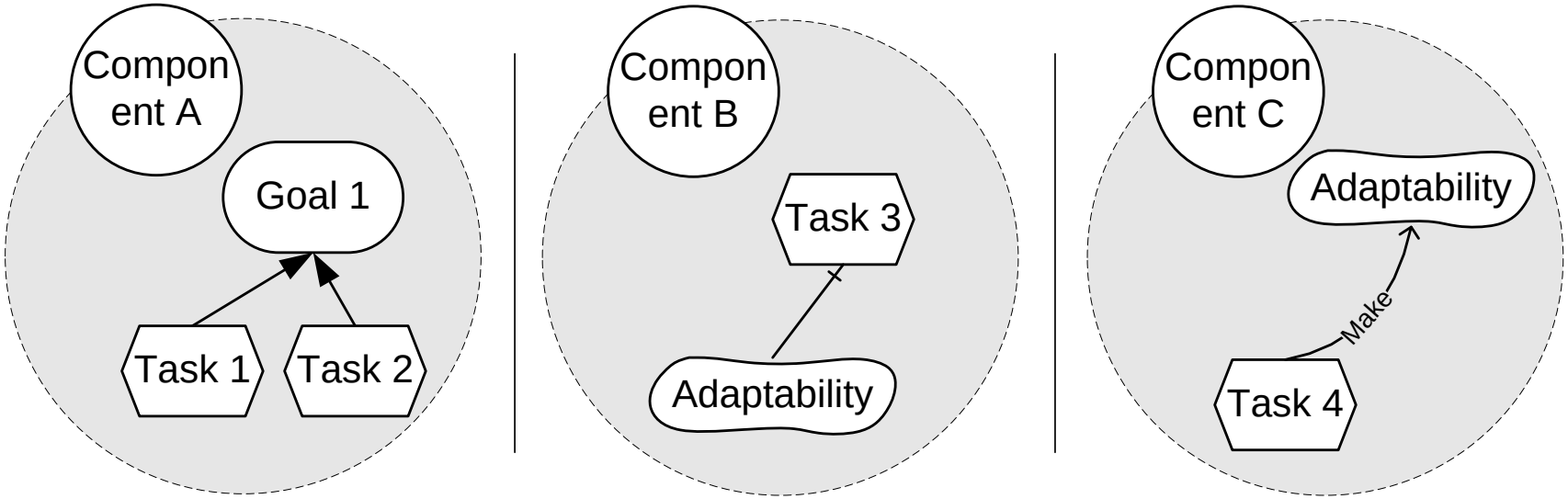
AAI ::= self.allActors().eai()->sum()+self.allDependencies().eai()->sum() / self.allActors()->size() + self.allDependencies()->size()

Software Adaptability Index (SAI) =

SAI ::= allModels().aai()->sum() / allModels()->size()

EAI of individual components (SR)

The SR decomposition may help:



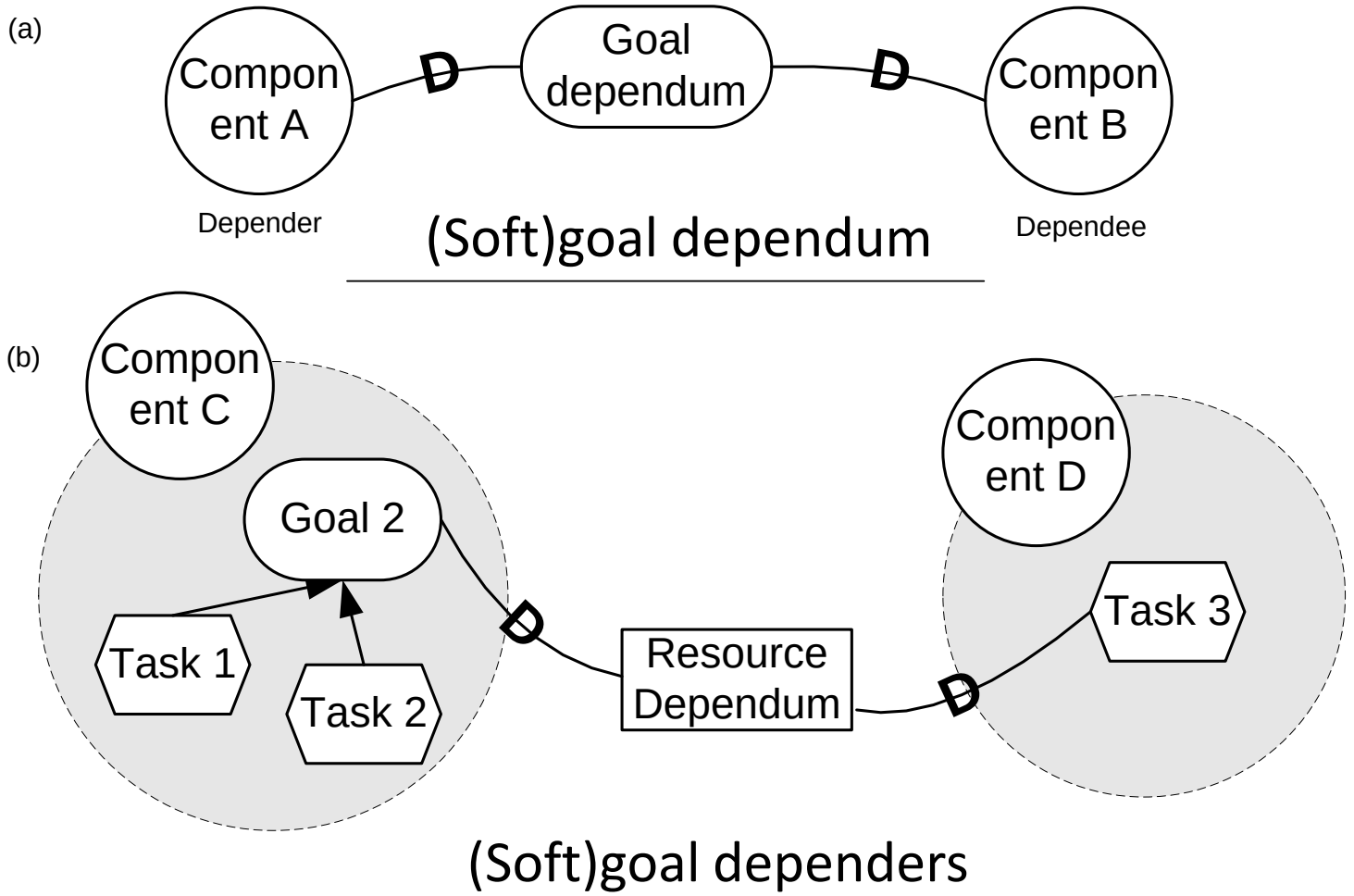
Means-end decomposition

Contribution to softgoal

Softgoal-to-task decomposition

EAI of individual components: (SD)

Dependencies may identify adaptability too:



EAI of individual components (extensions)

Approach	Base notation	Extension towards adaptability	Architecture
Lapouchnian	Tropos	Context annotations	Not defined
Ali	Tropos	Context annotations	Not defined
Dalpiaz	Tropos	Context annotations Recovery activities	Self-reconfiguring component
Morandini	Tropos	Context annotations Recovery activities Fault modeling	Multi-agent
Jian	*i**	Context annotations Runtime adaptation	Not defined
Qureshi	Techne	Runtime adaptation	Service-based
Bencomo	KAOS	Runtime adaptation Flexibility language	Not defined
Baresi	KAOS	Recovery activities	Service-based

II.

ANTICIPATING CHANGES WITH FUTUROLOGY[2,3]

Motivation

Adaptation and evolution is great

Motivation

However, it is ***very hard*** to develop systems that

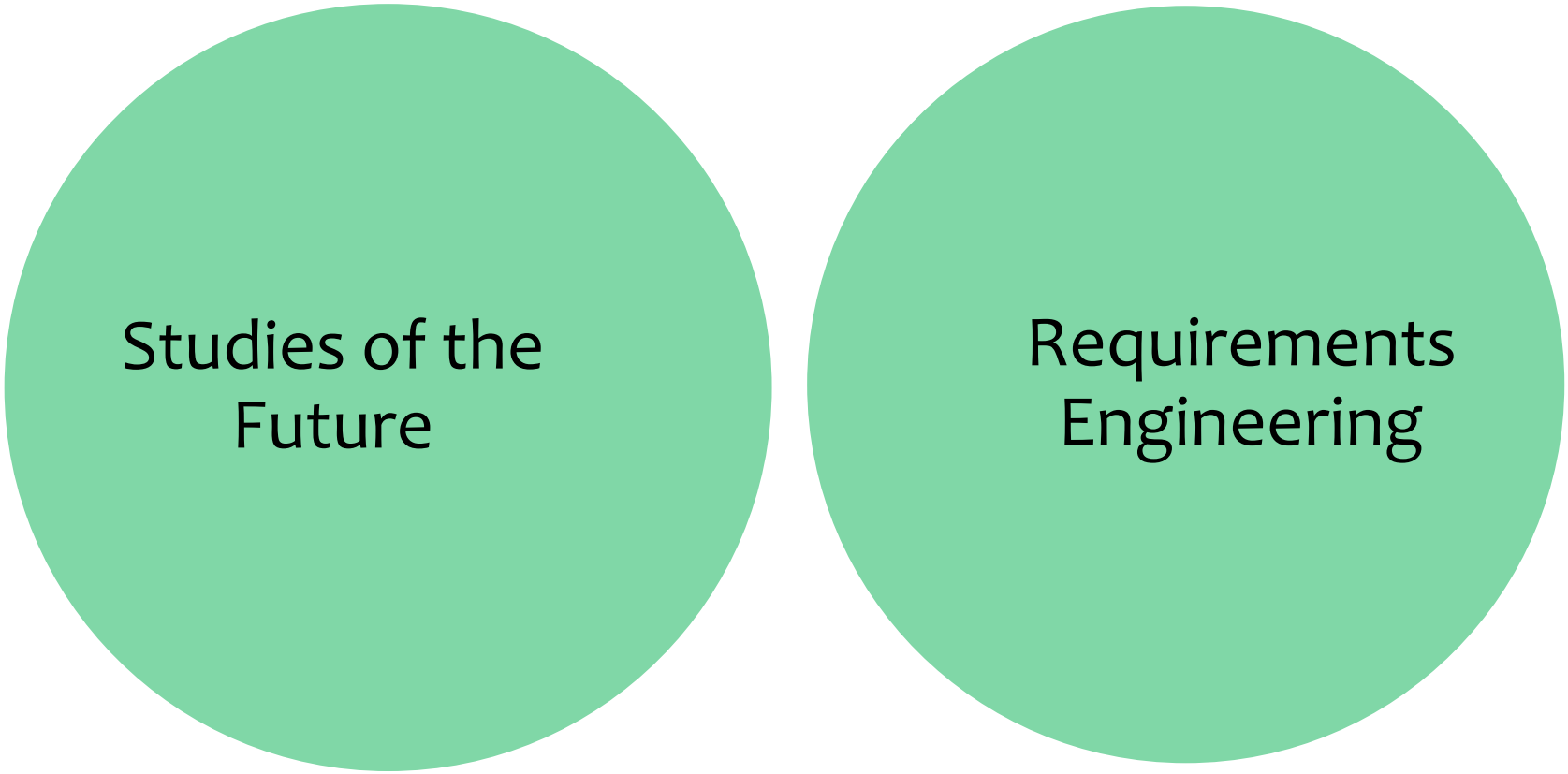
- Adapt to ***any*** scenario
- Evolve in ***any*** way

Motivation

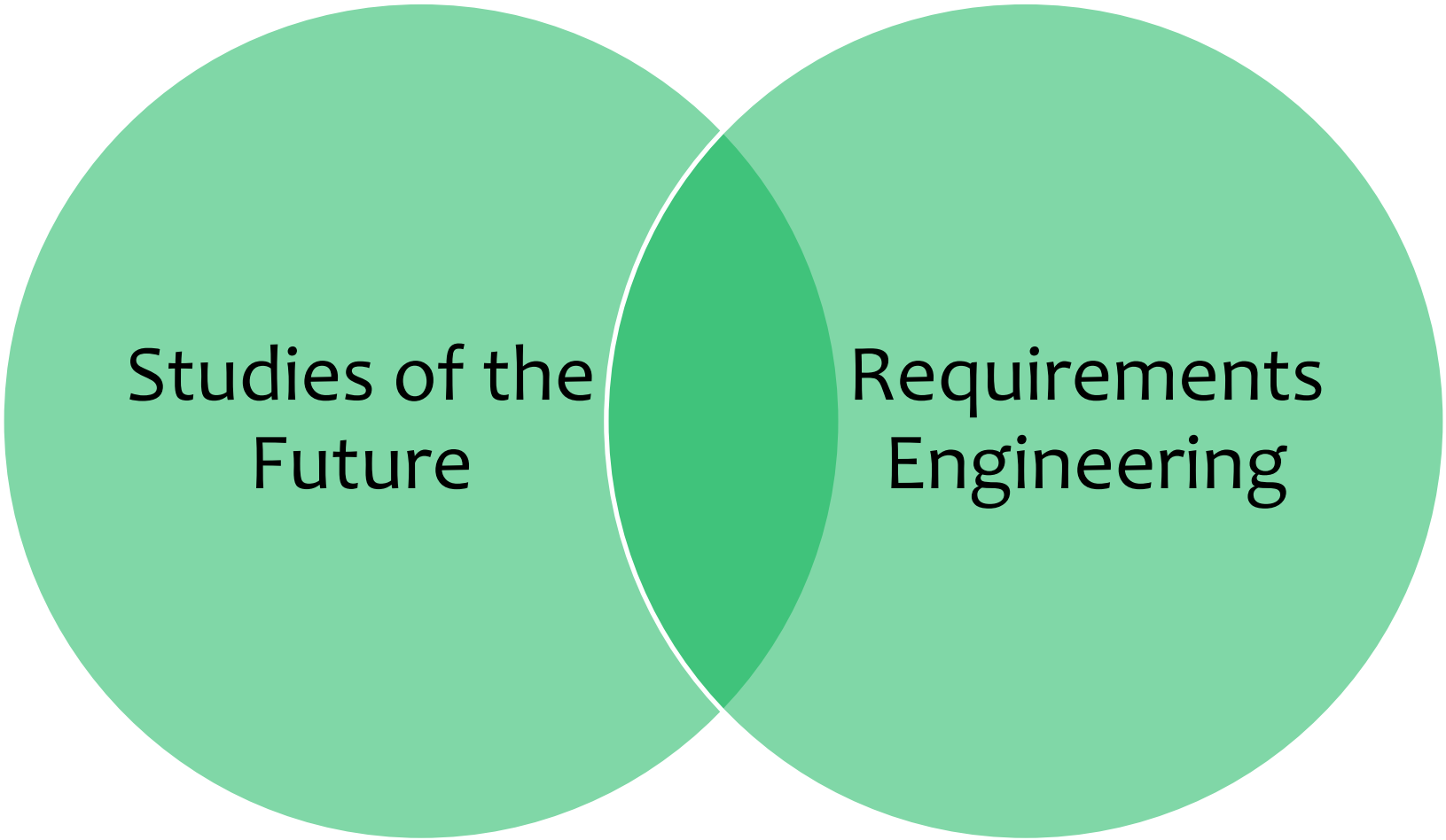
If we could see the future...

(sort of)

Reduce the gap



Reduce the gap



Definitions

- **Future event:** *a future event is an occurrence that is expected to take place in the future.*
- **Representation of the future:** *a representation of the future is a model that describes a set of future events.*
- **Foresight method:** *a foresight method is a means of creating a representation of the future.*

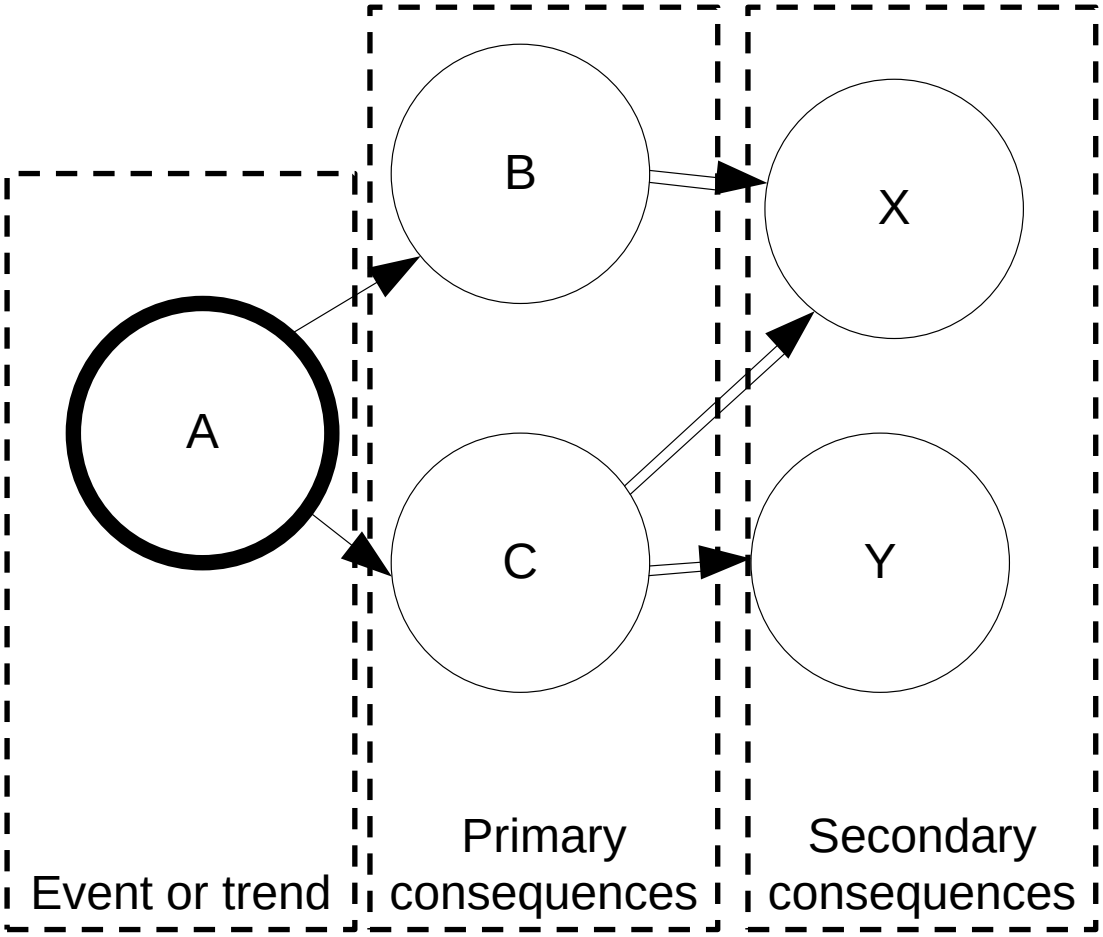
Classification of foresight methods

Category	Method
Collect judgments from Experts	Delphi Futures Wheel Participatory methods
Forecast time series and other quantitative measures	Econometrics forecast Regression Analysis Trend Impact Analysis Structural Analysis
Understand the linkages between events, trends and actions	System Dynamics Agent Modeling Trend Impact Analysis Cross Impact Analysis Relevance Trees Futures Wheel Simulation Modeling Multiple perspectives Causal Layered Analysis Field Anomaly Relaxation
Portray alternative plausible futures	Scenarios Futures Wheel Simulation and Gaming Agent Modeling

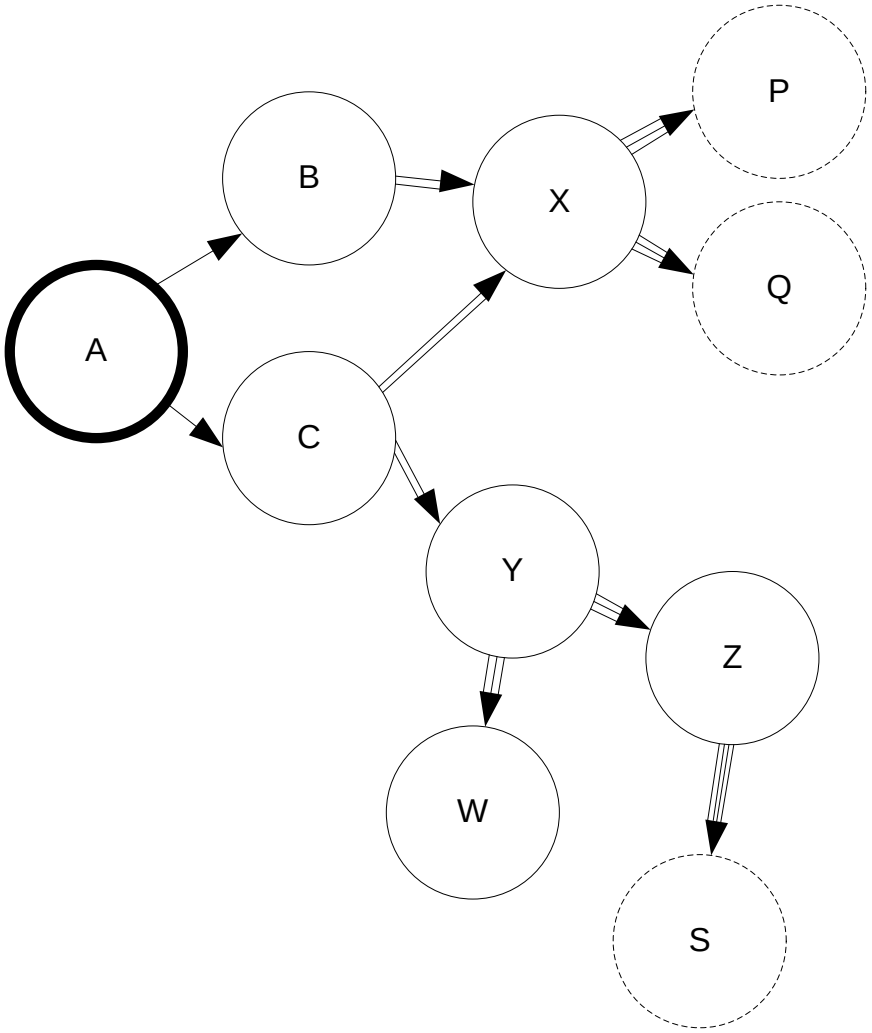
Our proposal

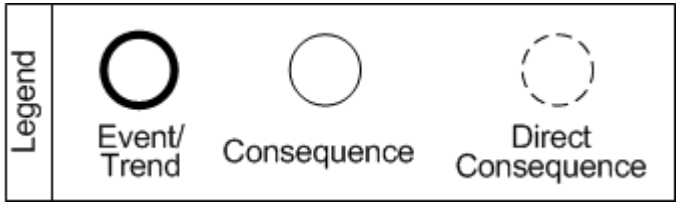
- Futures wheel
  - Provides a clear picture of the future events that may impact the system;
  - Easy to be understood and used by stakeholders
  - Low effort required
- *i**
  - Widespread goal modeling language
  - Provides a suitable mechanism to represent alternative behaviors of a system

Futures wheel



Futures wheel extension

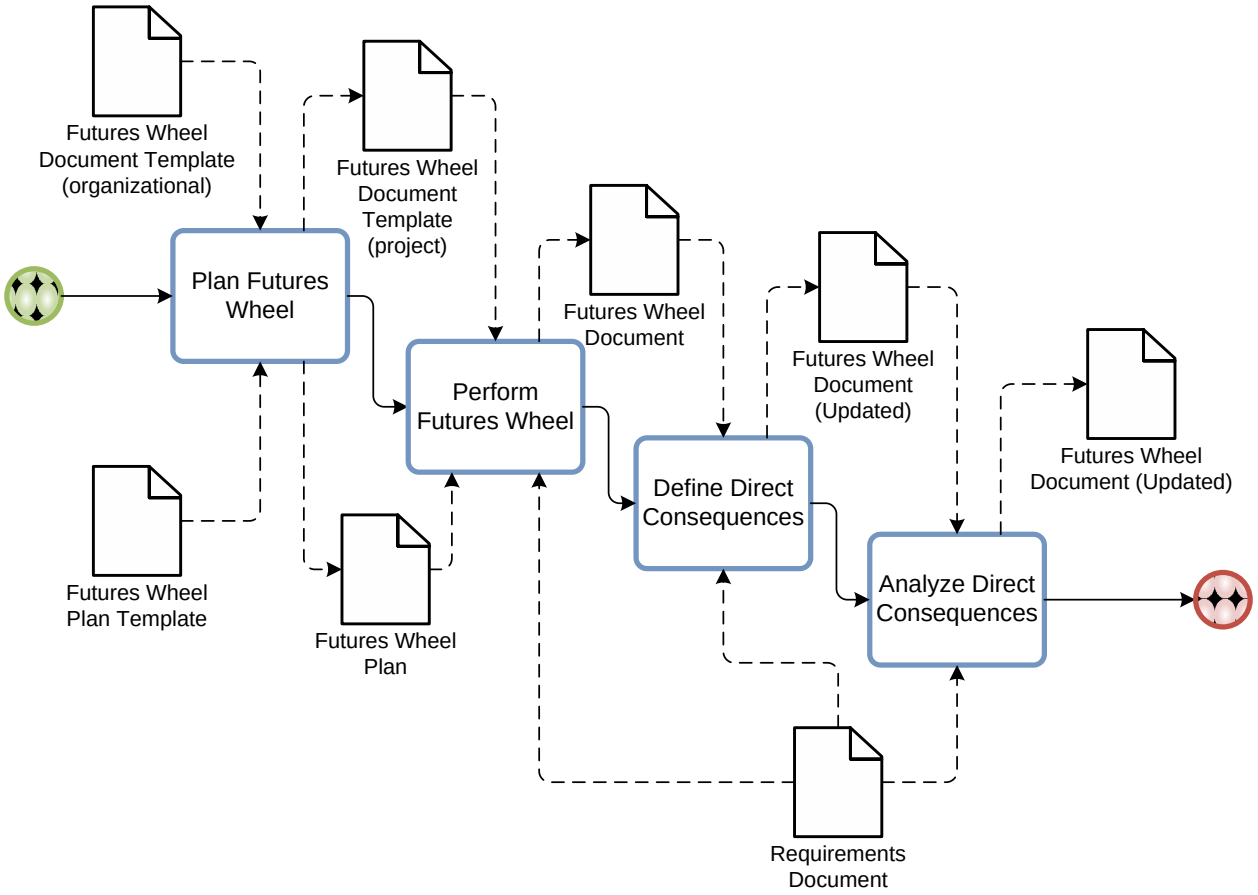




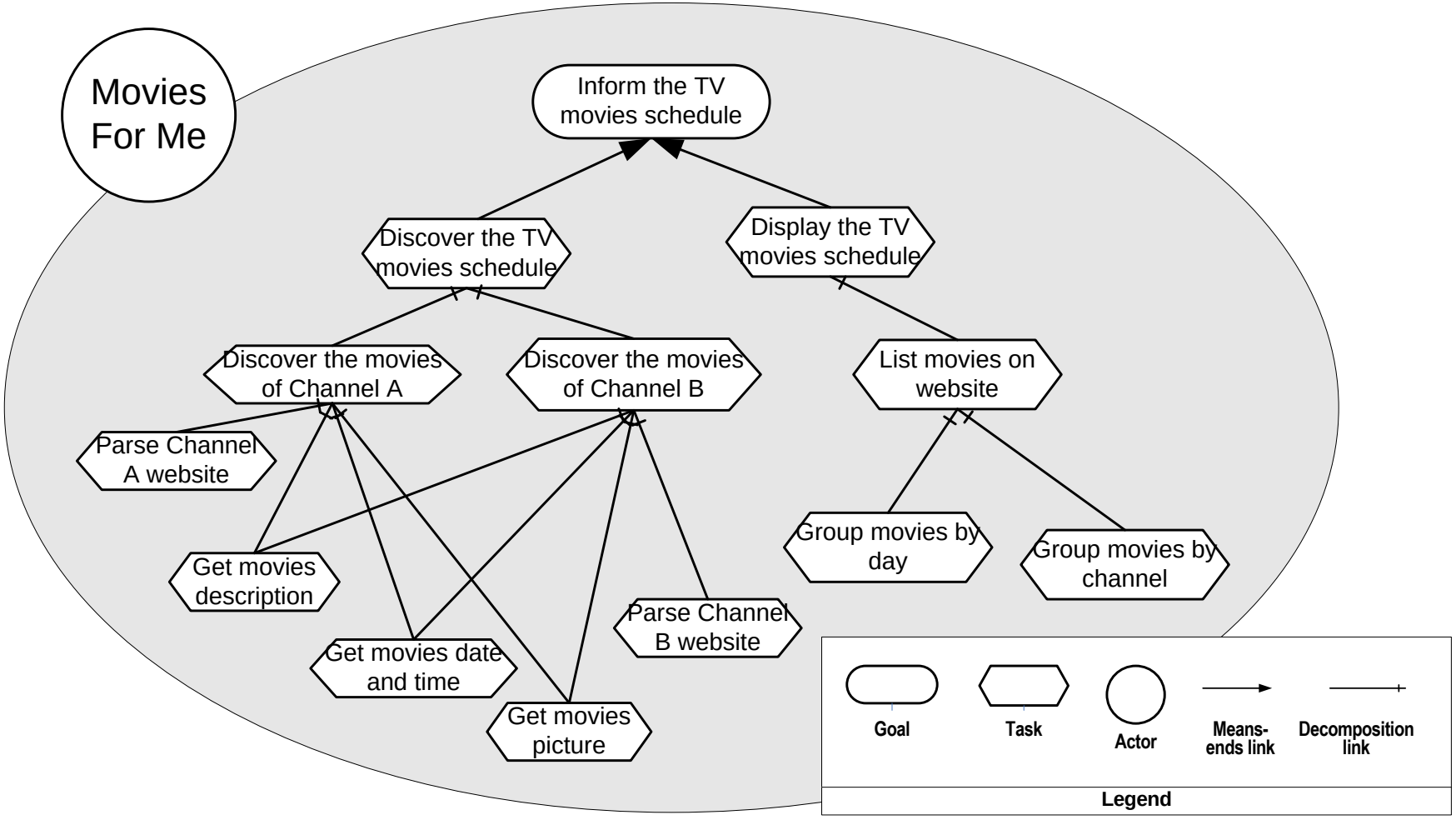
Integrating futures wheel and goal models

- Build extended futures wheel models
  - Build futures wheel model
  - For each consequence, analyze how it affects the system
- Adapt goal model
  - For each direct consequence in the futures wheel model, analyze how the system can be altered in order to deal with the consequence
  - Change the goal model accordingly

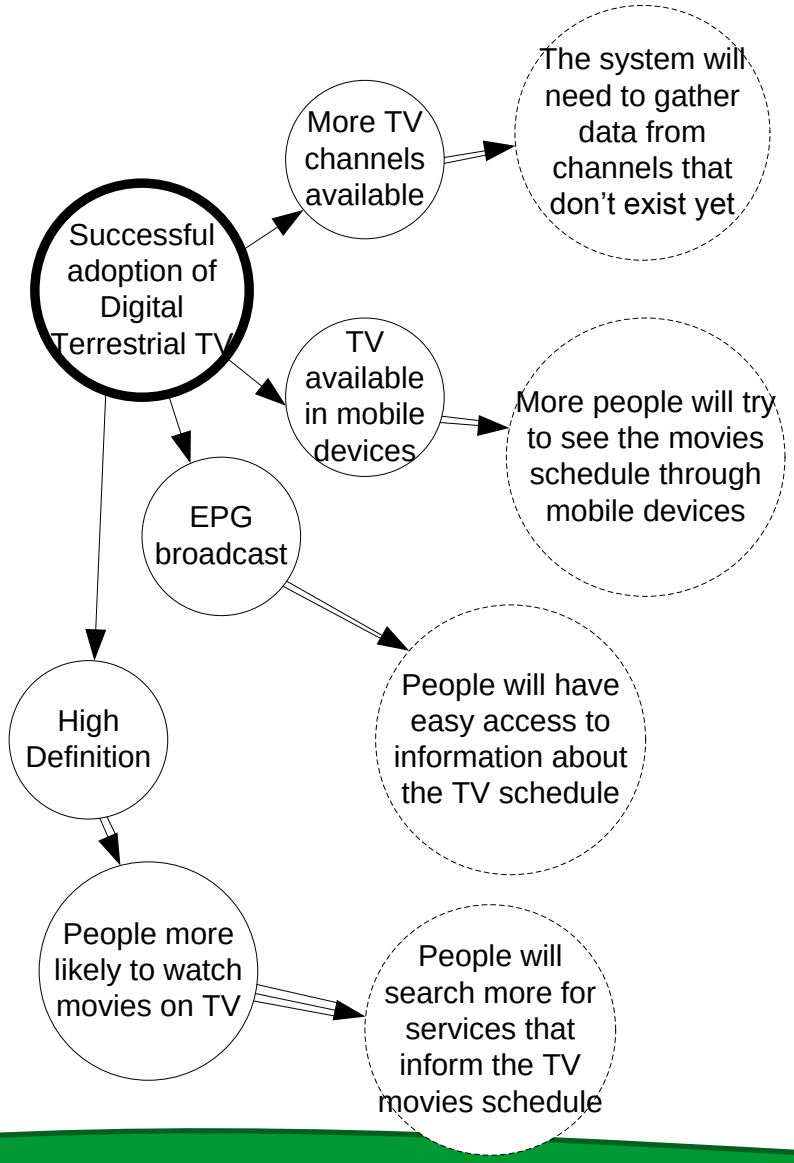
Process


Futures Wheel Document Template (organizational)
Futures Wheel Document Template (project)
Plan Futures Wheel
Futures Wheel Document
Futures Wheel Document (Updated)
Perform Futures Wheel
Futures Wheel Document (Updated)
Define Direct Consequences
Futures Wheel Plan Template
Futures Wheel Plan
Analyze Direct Consequences
Requirements Document

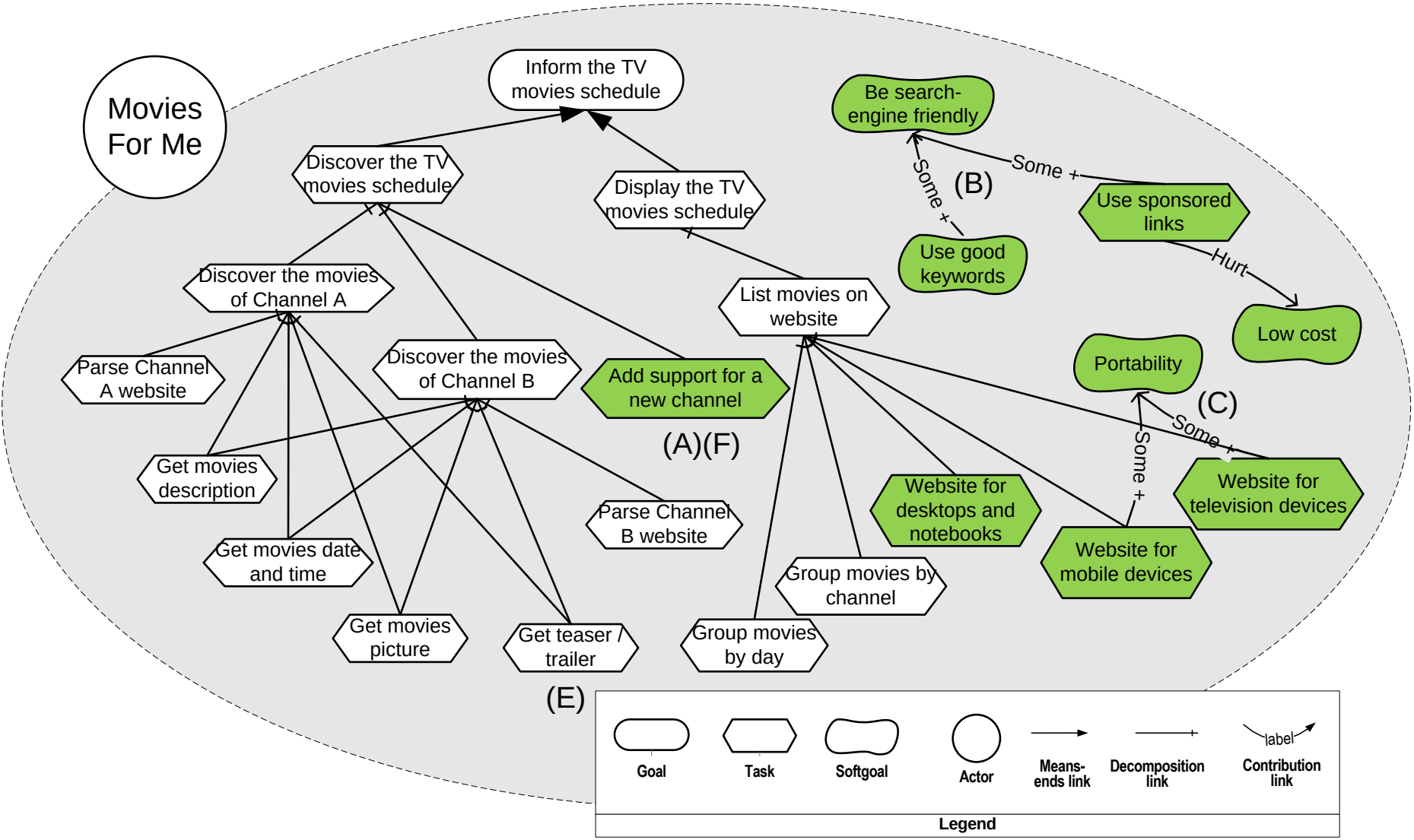
Case study – original goal model



Case study – futures wheel



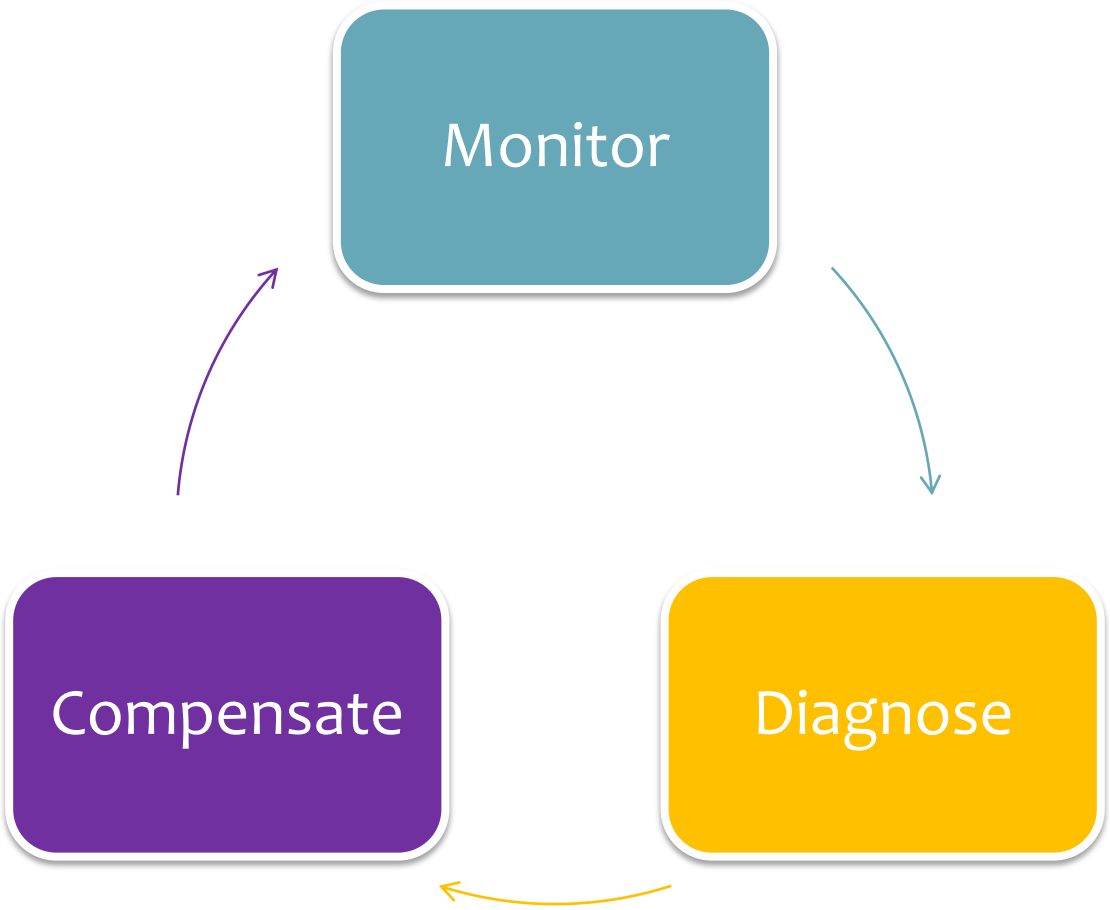
Case study – modified goal model



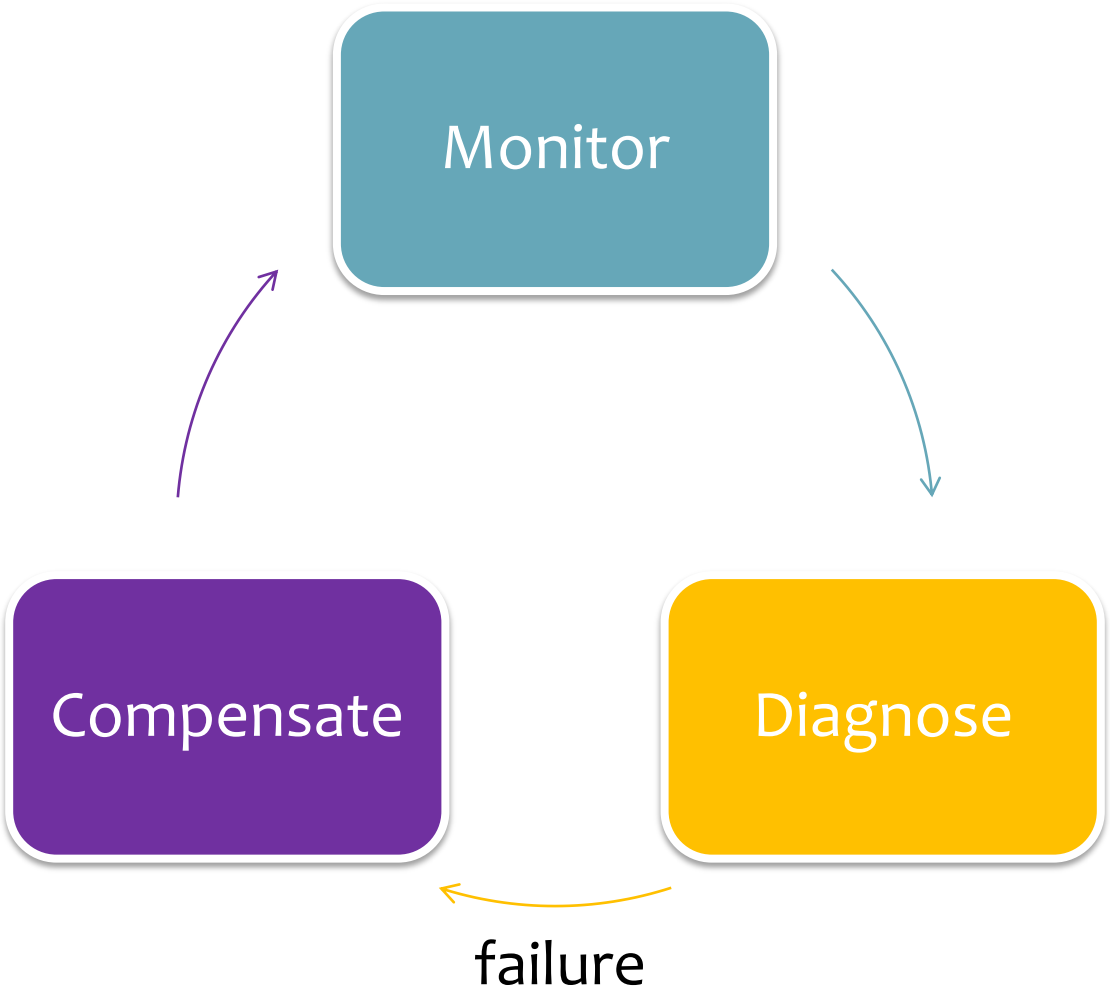
III.

FAILURE POLICY – THE *FAST* APPROACH[4,5]

MDC Cycle



MDC Cycle



Different faults, different reactions

"Some faults lead to the unusability of a device or even to catastrophes; some faults can be ignored or are actually never discovered."

Bernhard Rinner - Detecting and Diagnosing Faults (2002)

A failure is...

... the unsuccessfull execution of ...

a task

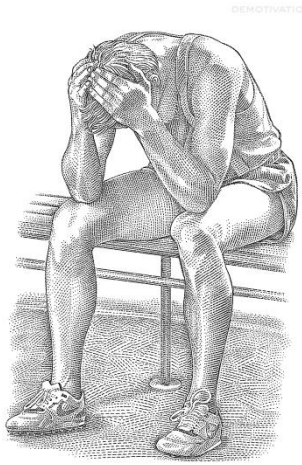
a service operation

a method

a use case

a goal

...



FAST

- This framework allows the **user** to define **conditions** on which failures may be ignored
  - i.e., it won't trigger a compensation
- The conditions are based on the **context** and on the **number** of failures occurrence

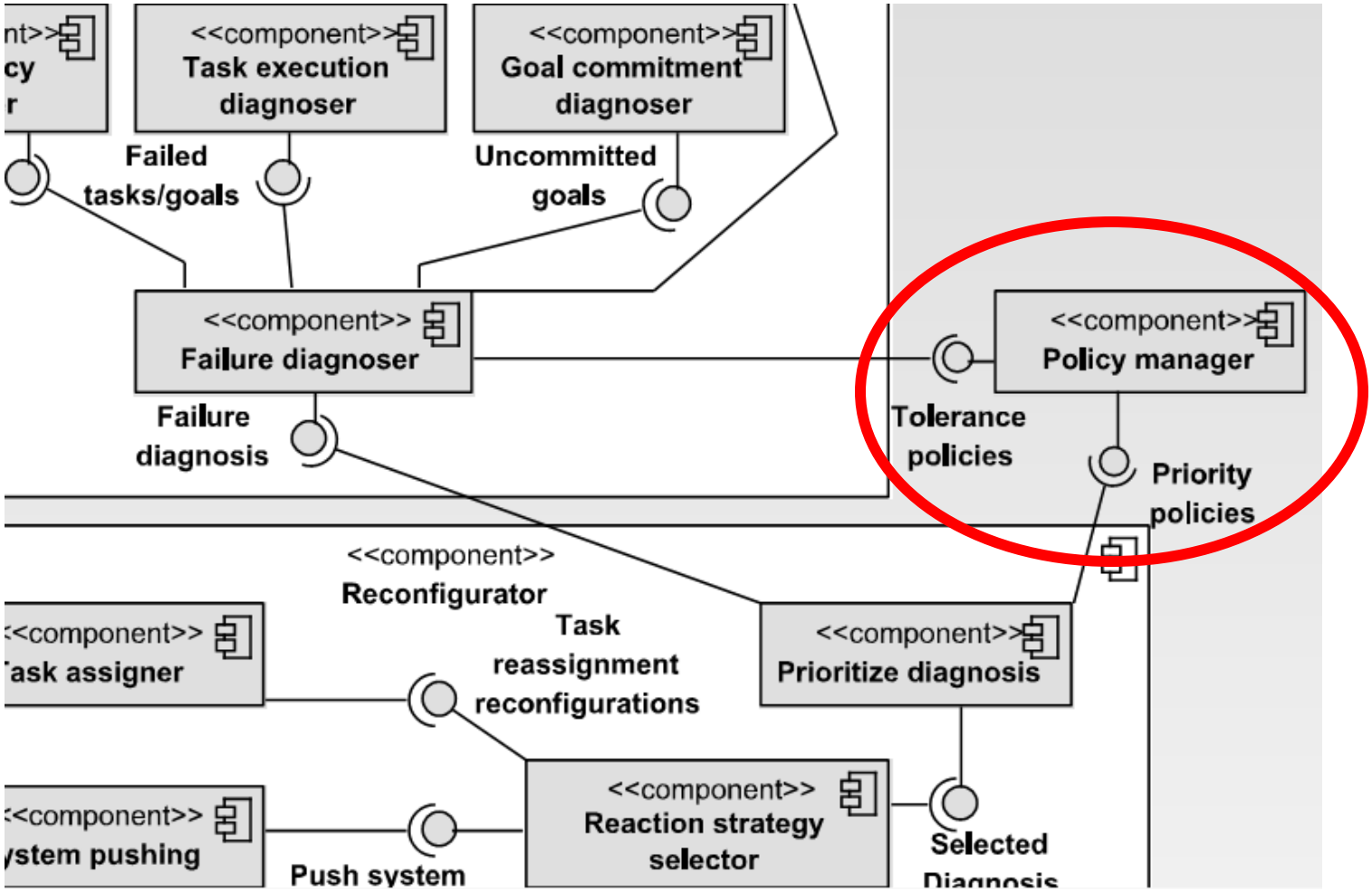
Policies

- Unlike requirements, which are
  - Somewhat static
  - Defined at design time
- This failure handling specification is
  - User / installation dependent
  - Defined at deployment time

Failure policy

- *Context-based*
  - *failuresSet* isAllowedToFailIf *contextExpression*
  - **Ex:** *failureX isAllowedToFailIf calendar.day=Sunday*
- *Limit of consecutive failures*
  - *failuresSet* isAllowedToFailAtMost *limit*
  - **Ex:** *failureX: failureY isAllowedToFailAtMost 4*

Fabiano's component



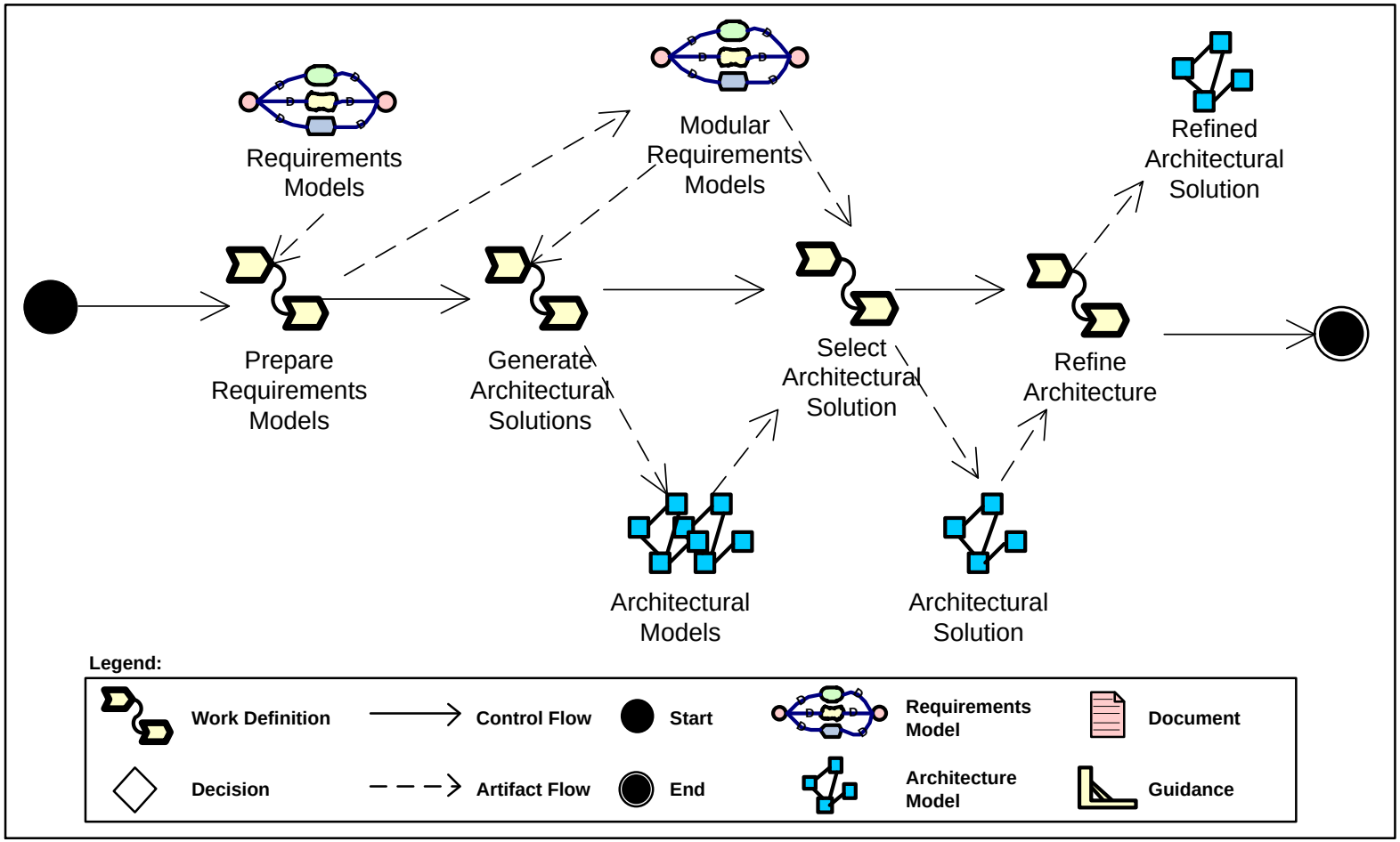
IV.

FROM REQ TO ARCH (ADAPTIVE SYSTEMS)[6]

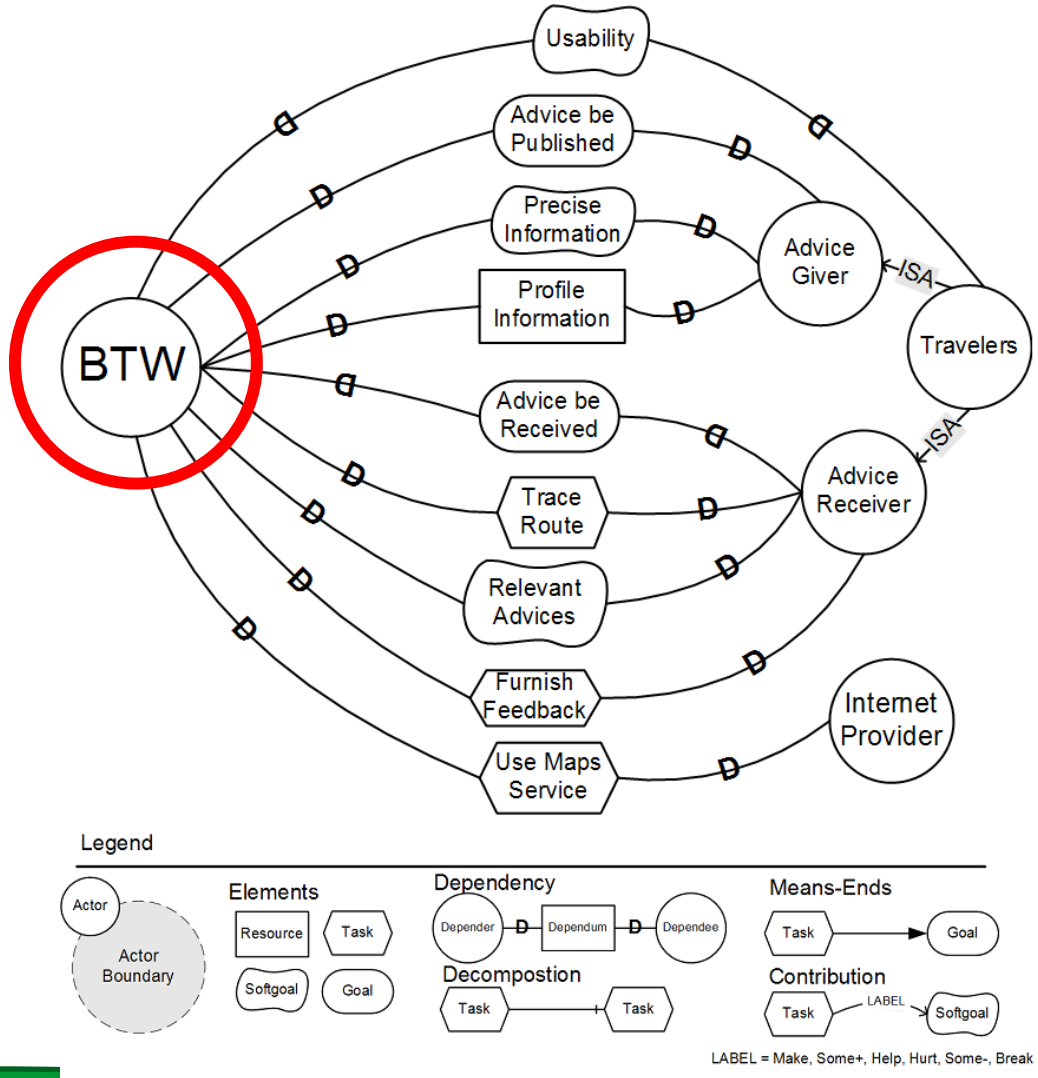
STREAM-A (Adaptive)

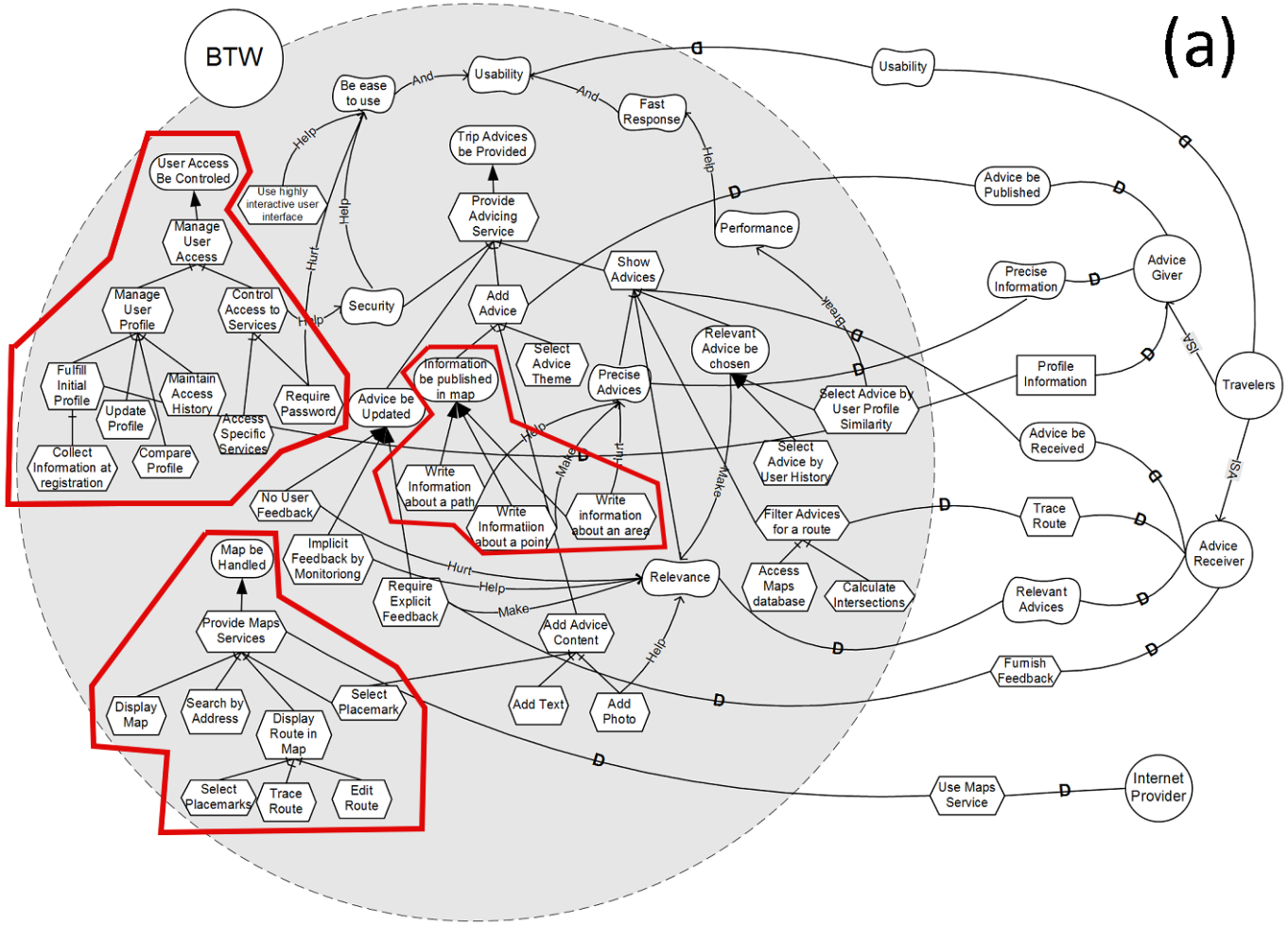
STREAM

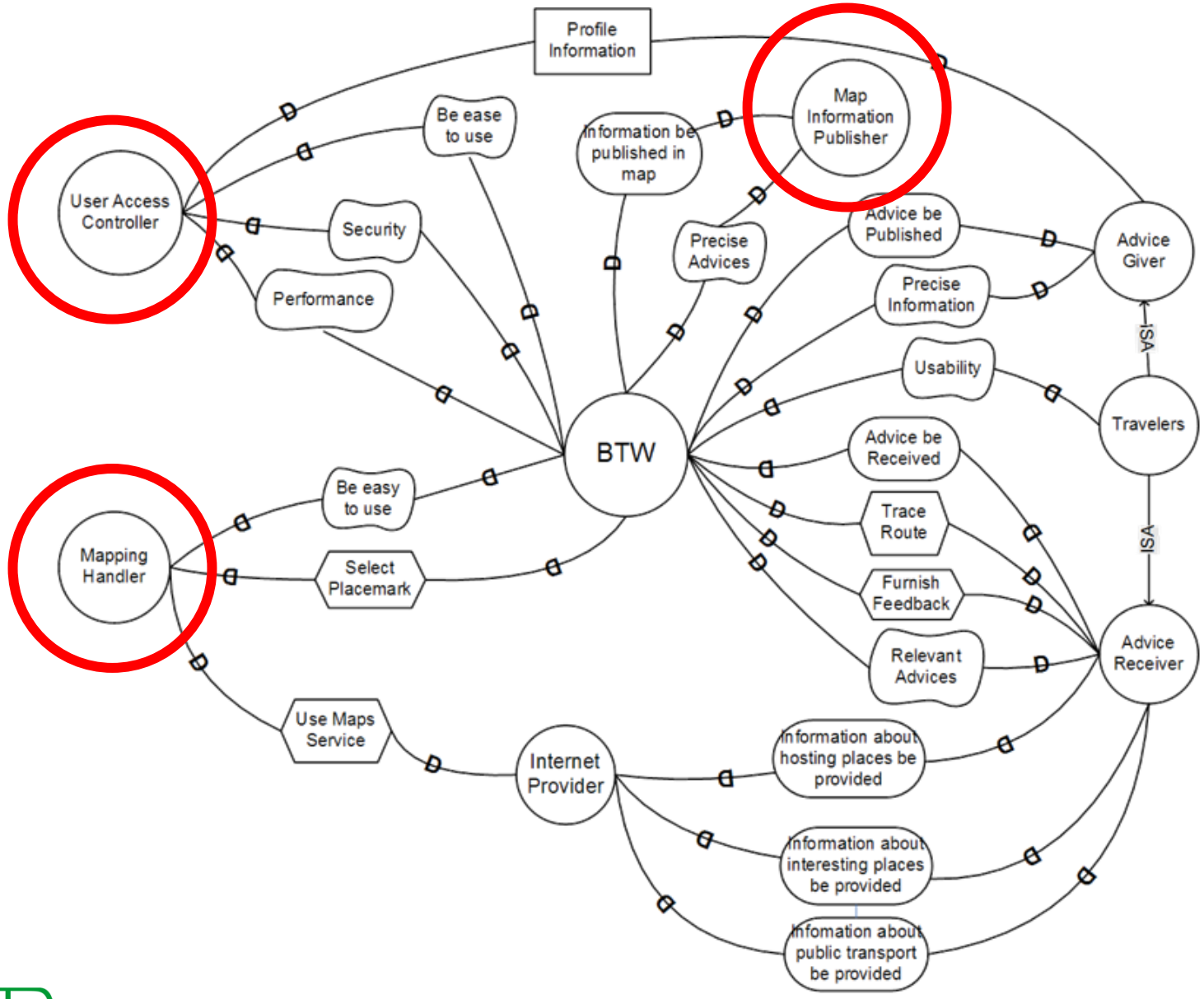
- An approach based on model transformations that generates architectural models from requirements models
  - Source language: i*
  - Target language: Acme


Requirements Models
Modular Requirements Models
Refined Architectural Solution
Prepare Requirements Models
Generate Architectural Solutions
Select Architectural Solution
Refine Architecture
Architectural Models
Architectural Solution
Legend:
Work Definition
Control Flow
Start
Requirements Model
Document
Decision
Artifact Flow
End
Architecture Model
Guidance

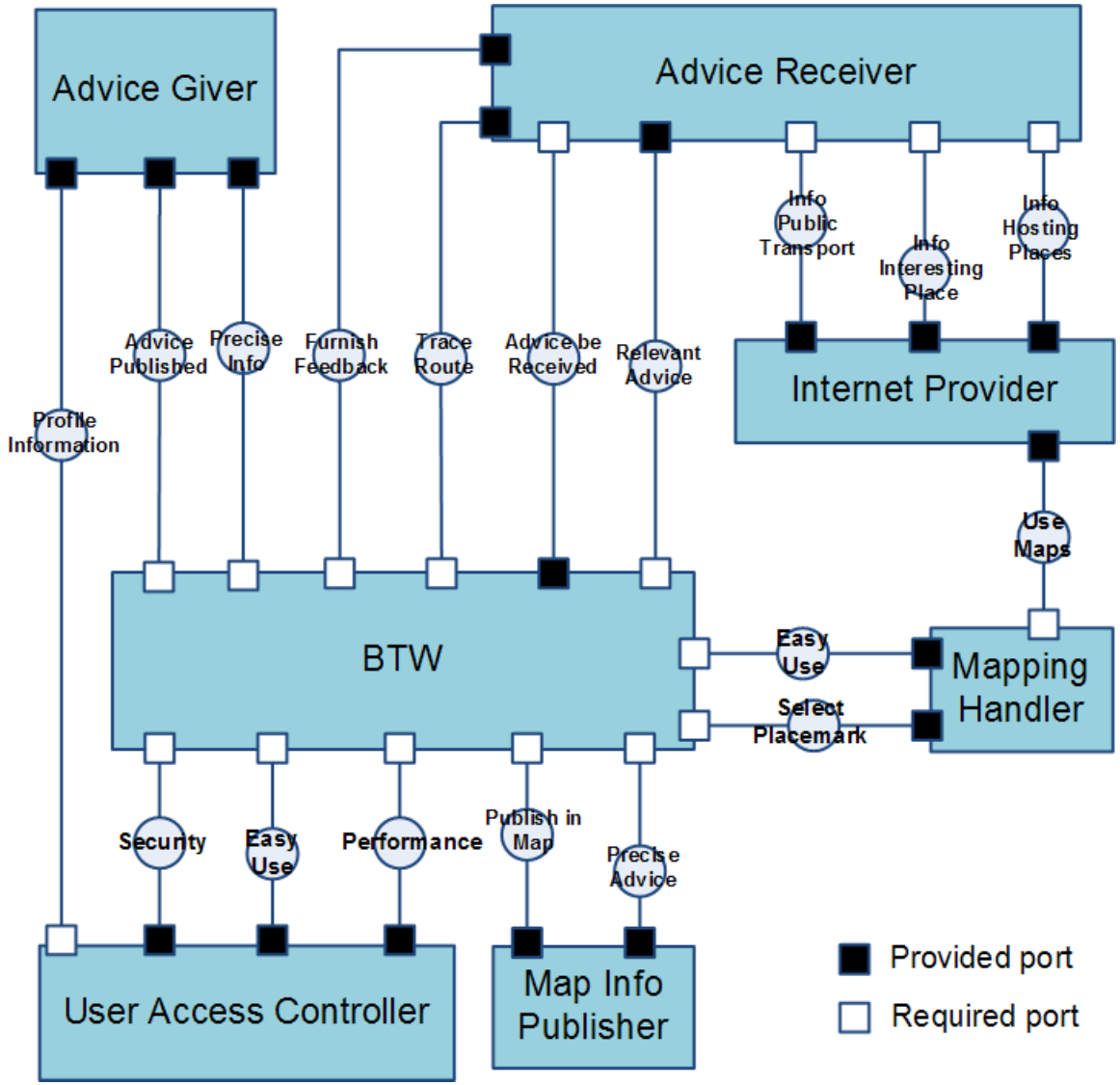
BTW: system-to-be



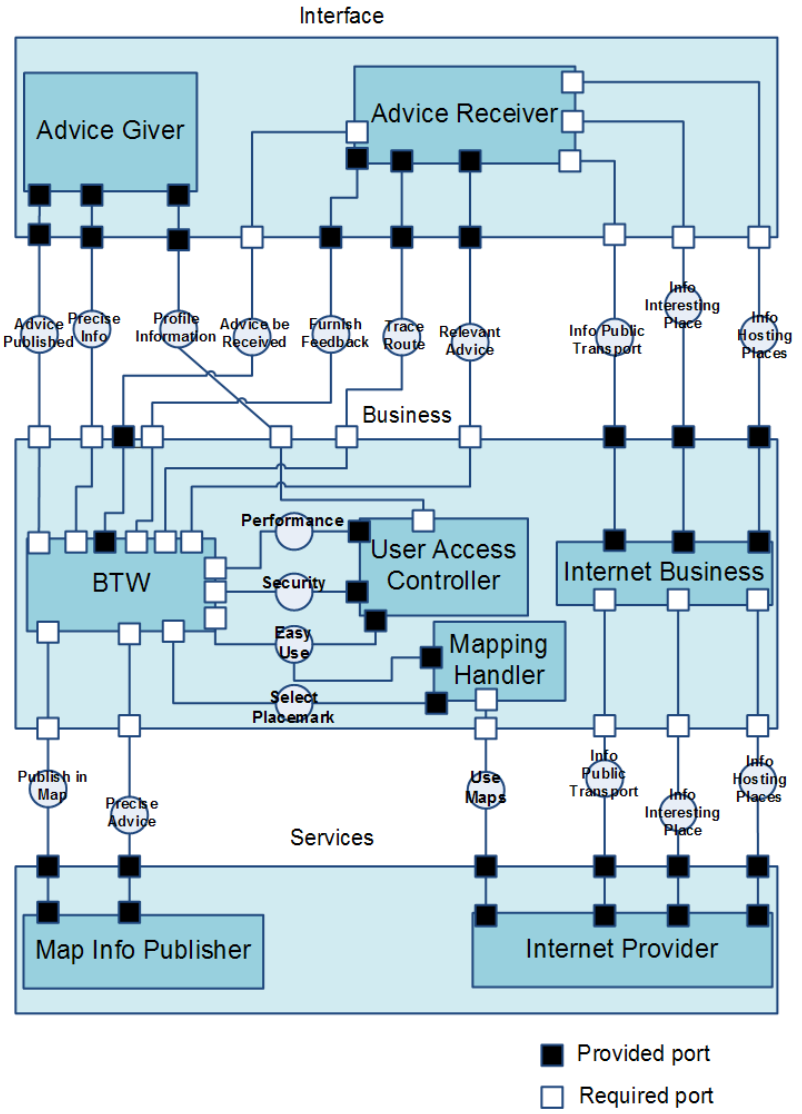

(a)
BTW
Usability
Be ease to use
And
Usability
And
Fast Response
Help
Trip Advices be Provided
User Access Be Controled
Use highly interactive user interface
Help
Provide Advicing Service
Advice be Published
Performance
Manage User Access
Hurt
Show Advices
Precise Information
Advice Giver
Manage User Profile
Control Access to Services
Help
Security
Add Advice
Break
Relevant Advice be chosen
Isa
Fulfill Initial Profile
Maintain Access History
Information be published in map
Select Advice Theme
Precise Advices
Profile Information
Travelers
Update Profile
Require Password
Advice be Updated
Help
Select Advice by User Profile Similarity
Access Specific Services
Advice be Received
Collect Information at registration
Compare Profile
Make
Hurt
Select Advice by User History
Write Information about a path
Isa
No User Feedback
Write Informatiion about a point
Write information about an area
Make
Filter Advices for a route
Trace Route
Map be Handled
Implicit Feedback by Monitoriong
Hurt
Help
Relevance
Access Maps database
Calculate Intersections
Advice Receiver
Require Explicit Feedback
Make
Relevant Advices
Provide Maps Services
Add Advice Content
Help
Furnish Feedback
Display Map
Search by Address
Select Placemark
Add Text
Add Photo
Display Route in Map
Select Placemarks
Trace Route
Edit Route
Use Maps Service
Internet Provider


Profile Information
Map Information Publisher
Be ease to use
Information be published in map
User Access Controller
Security
Performance
Precise Advices
Advice be Published
Advice Giver
Precise Information
ISA
Usability
Travelers
BTW
Advice be Received
Be easy to use
Trace Route
Mapping Handler
Select Placemark
Furnish Feedback
VSI
Advice Receiver
Relevant Advices
Use Maps Service
Internet Provider
Information about hosting places be provided
Information about interesting places be provided
Infomation about public transport be provided

(a)

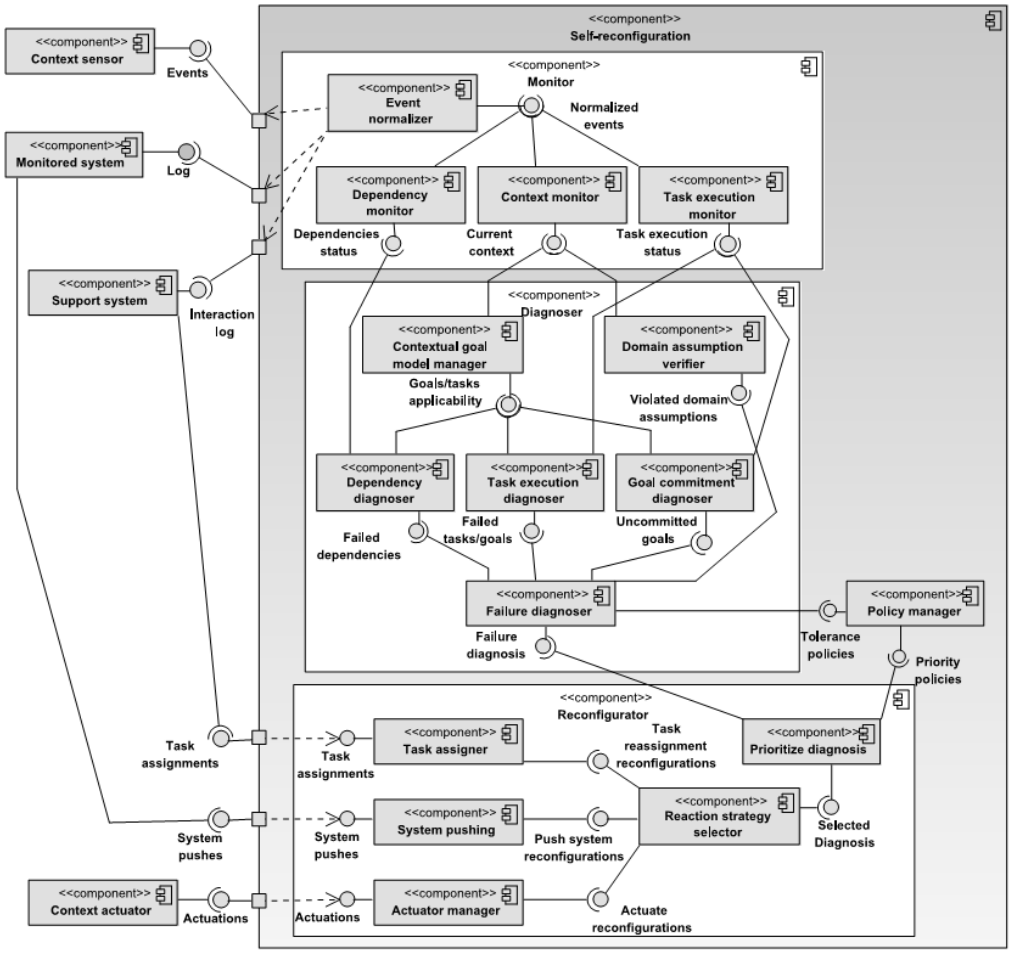

Advice Giver
Advice Receiver
Info Public Transport
Info Interesting Place
Info Hosting Places
Advice Published
Precise Info
Furnish Feedback
Trace Route
Advice be Received
Relevant Advice
Internet Provider
Profile Information
Use Maps
BTW
Easy Use
Mapping Handler
Select Placemark
Security
Easy Use
Performance
Publish in Map
Precise Advice
User Access Controller
Map Info Publisher
Provided port
Required port

Architectural style

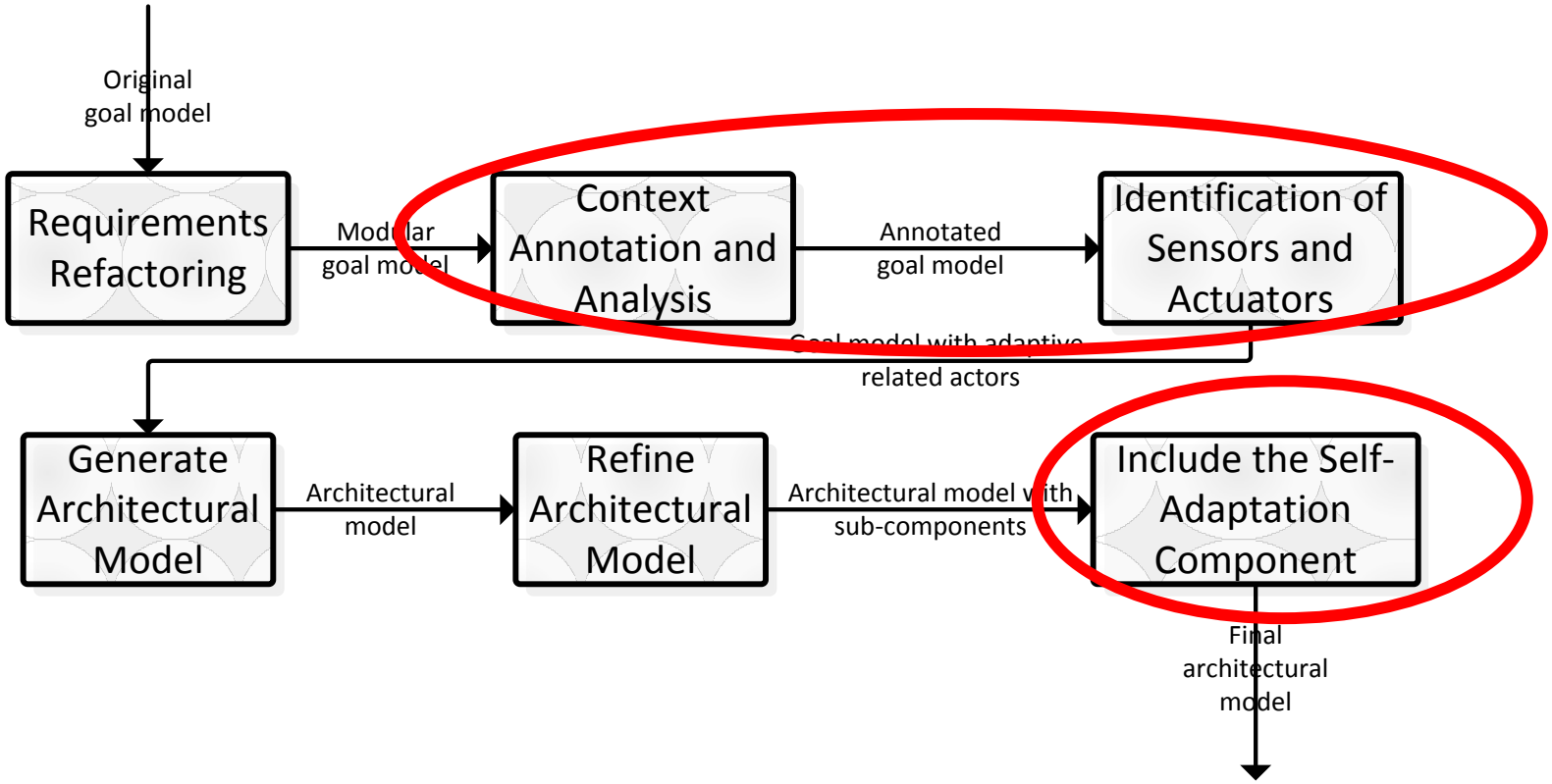


But what about **adaptation**?

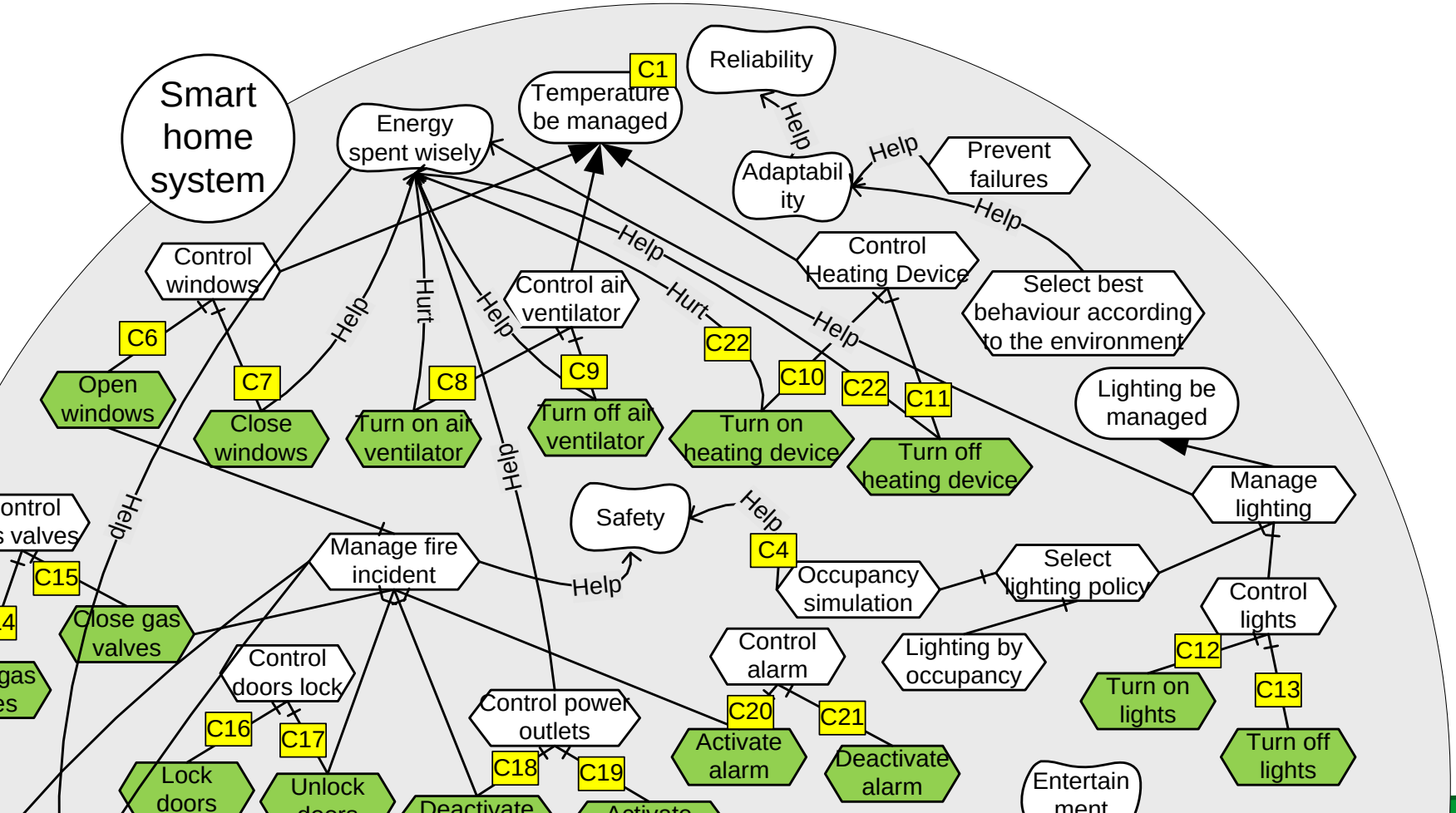
Fabiano's component (again)

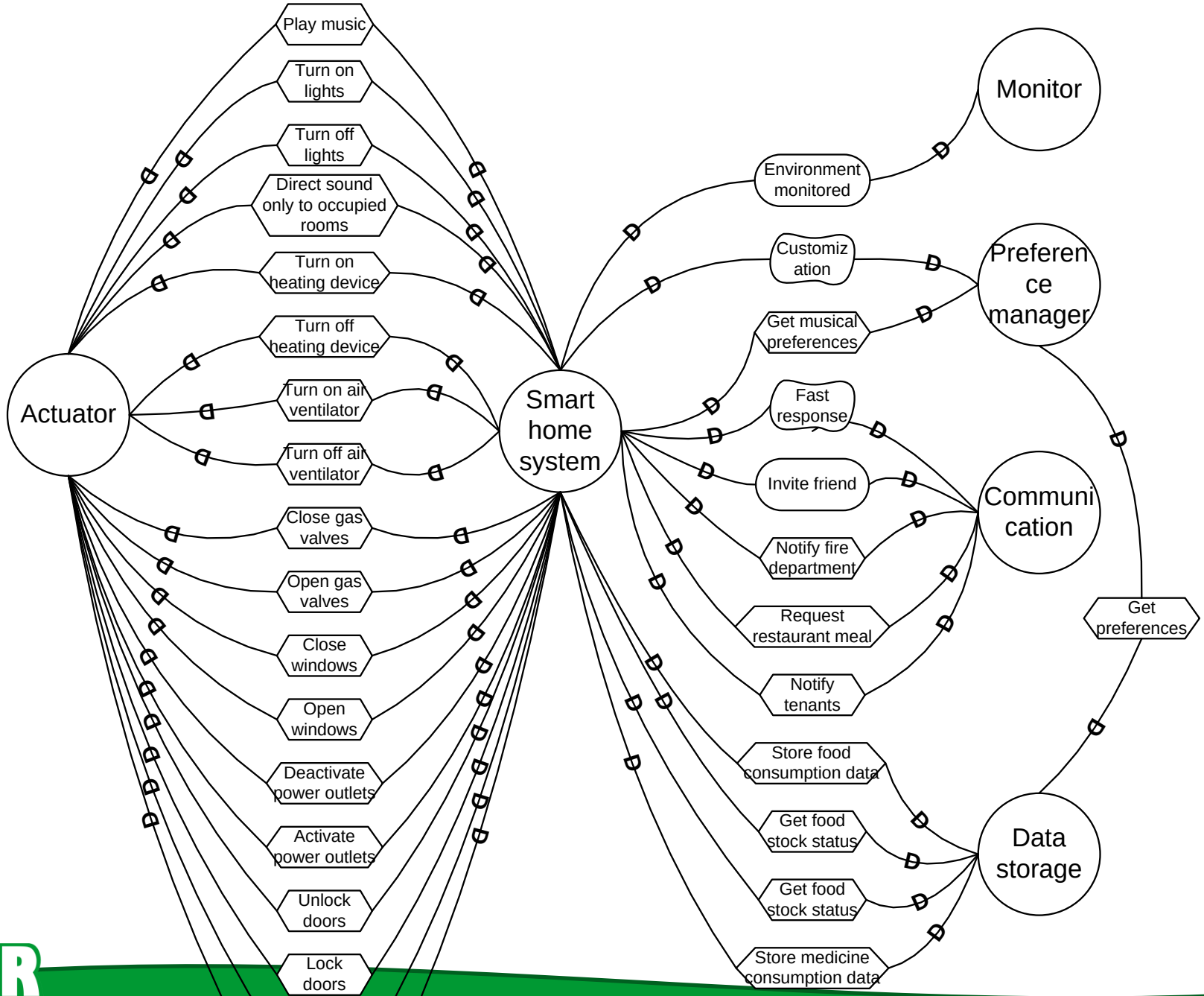


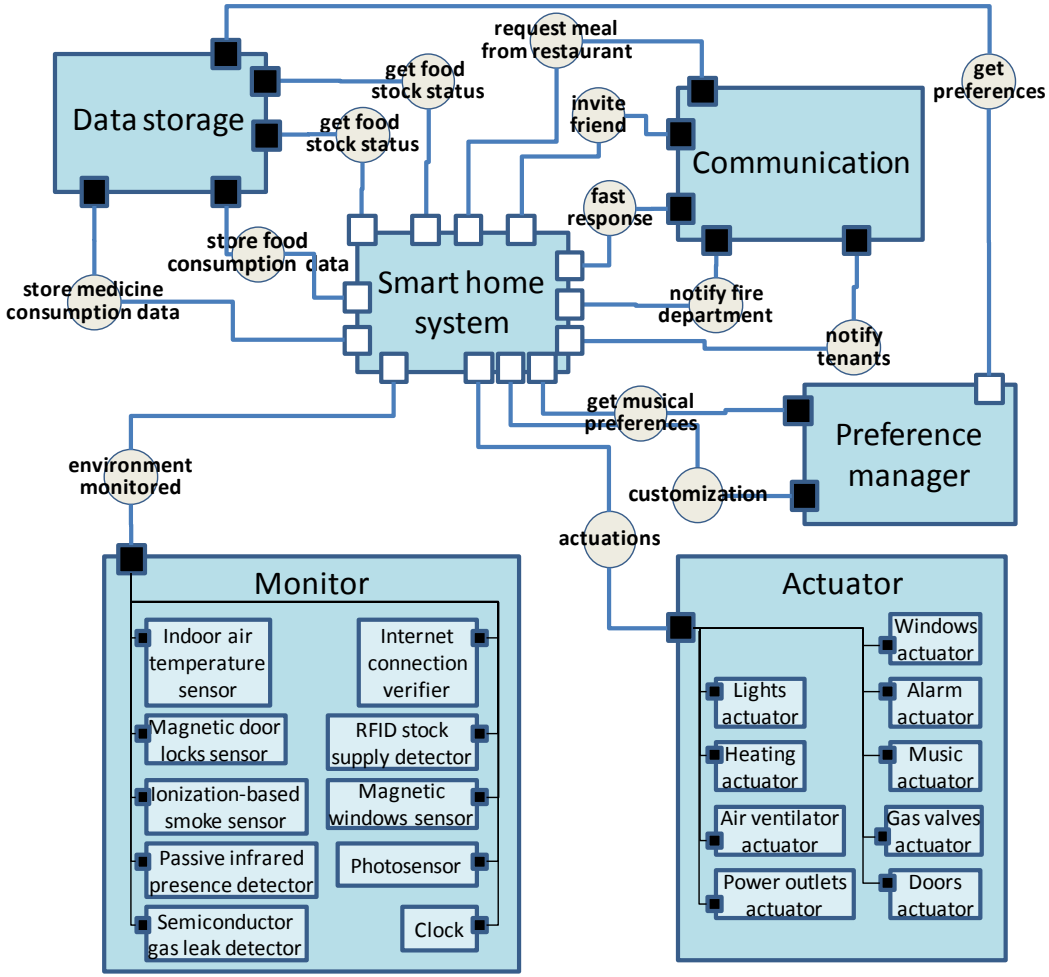
STREAM-Adaptive

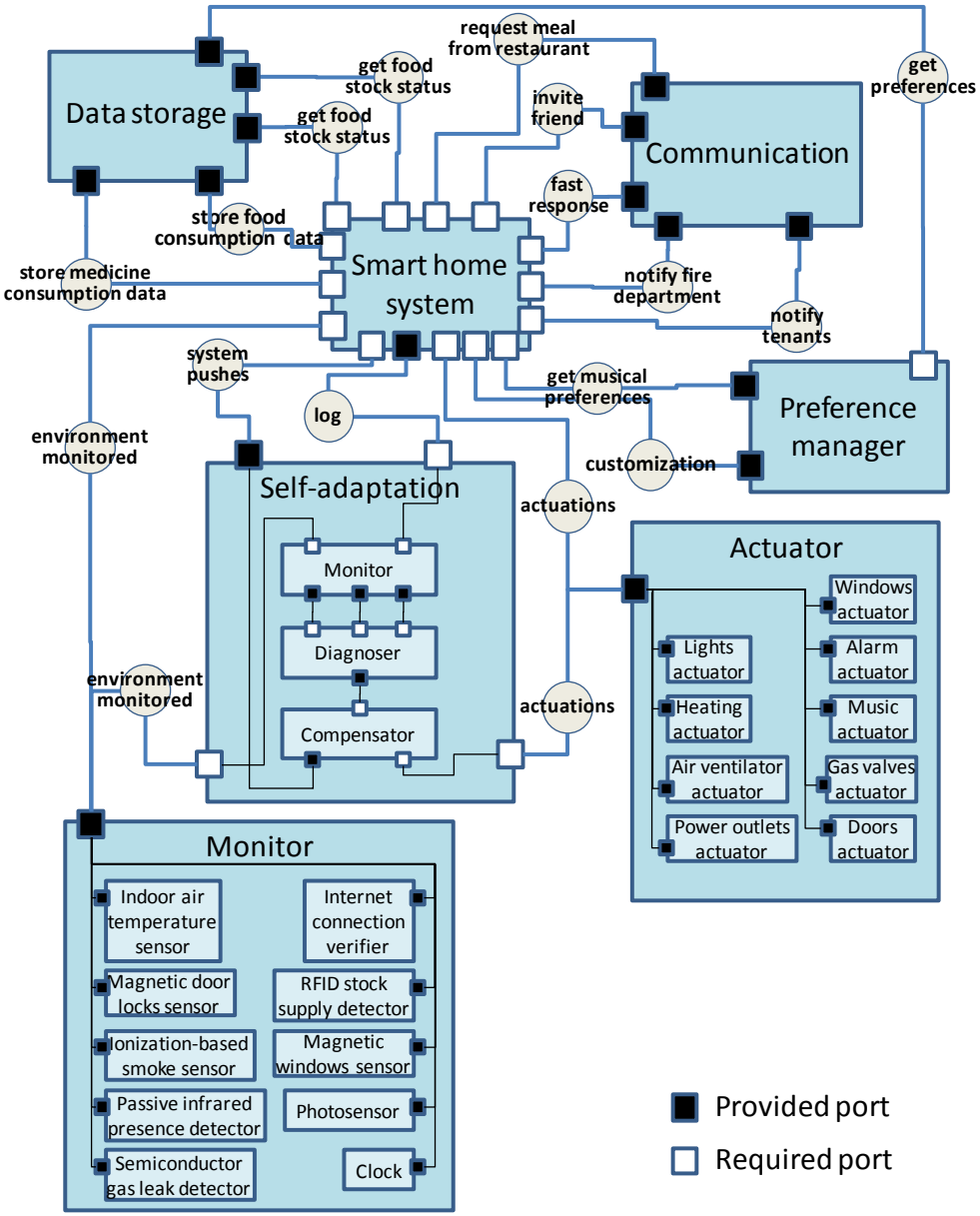


C6: The temperature at the room is hotter than what would be pleasant for the people within it, the temperature outside is colder than the temperature inside the smart-home and the windows are closed.




Play music
Turn on lights
Turn off lights
Direct sound only to occupied rooms
Turn on heating device
Turn off heating device
Turn on air ventilator
Turn off air ventilator
Close gas valves
Open gas valves
Close windows
Open windows
Deactivate power outlets
Activate power outlets
Unlock doors
Lock doors
Actuator
Smart home system
Monitor
Environment monitored
Customiz ation
Preferen ce manager
Get musical preferences
Fast response
Invite friend
Communi cation
Notify fire department
Request restaurant meal
Notify tenants
Get preferences
Store food consumption data
Get food stock status
Get food stock status
Data storage
Store medicine consumption data


request meal from restaurant
get food stock status
get food stock status
invite friend
get preferences
Data storage
Communication
fast response
store food consumption data
store medicine consumption data
Smart home system
notify fire department
notify tenants
get musical preferences
Preference manager
environment monitored
customization
actuations
Monitor
Indoor air temperature sensor
Internet connection verifier
Magnetic door locks sensor
RFID stock supply detector
Ionization-based smoke sensor
Magnetic windows sensor
Passive infrared presence detector
Photosensor
Semiconductor gas leak detector
Clock
Actuator
Windows actuator
Lights actuator
Alarm actuator
Heating actuator
Music actuator
Air ventilator actuator
Gas valves actuator
Power outlets actuator
Doors actuator



- MDC cycle
  - Monitor
  - Diagnose
  - Compesate

Thanks!

References

1. Pimentel, J., Franch, X., Castro, J. **Measuring Architectural Adaptability in i* Models.** CIbSE 2011
2. Pimentel, J., Santos, E. Castro, J., Franch, X. **Anticipating Requirements Changes - Using Futurology in Requirements Elicitation.** International Journal of Information System Modeling and Design
3. Pimentel, J., Castro, J., Perrelli, H., Santos, E., Franch, X. **Towards Anticipating Requirements Changes through Studies of the Future.** RCIS 2011
4. Pimentel, J., Castro, J., Franch, X. **Specification of Failure-Handling Requirements as Policy Rules on Self-Adaptive Systems.** WER 2011
5. Pimentel, J., Santos, E., Castro, J. **Conditions for ignoring failures based on a requirements model.** SEKE 2010
6. Pimentel, J., Lucena, M, Castro, J., Silva, C., Alencar, F., Santos, E.**Deriving Adaptable Software Architectures from Requirements Models: The STREAM-A approach.** Requirements Engineering Journal