Semantic Reasoning and Verification for Ambient Assisted Living

Thibaut Tiberghien, Mounir Mokhtari, Liu Yan & Dong Jin Song

FSFMA 2014
2nd French Singaporean Workshop on Formal Methods and Applications
13 May 2014, Singapore

Outline

I. Introduction

II. Context reasoning

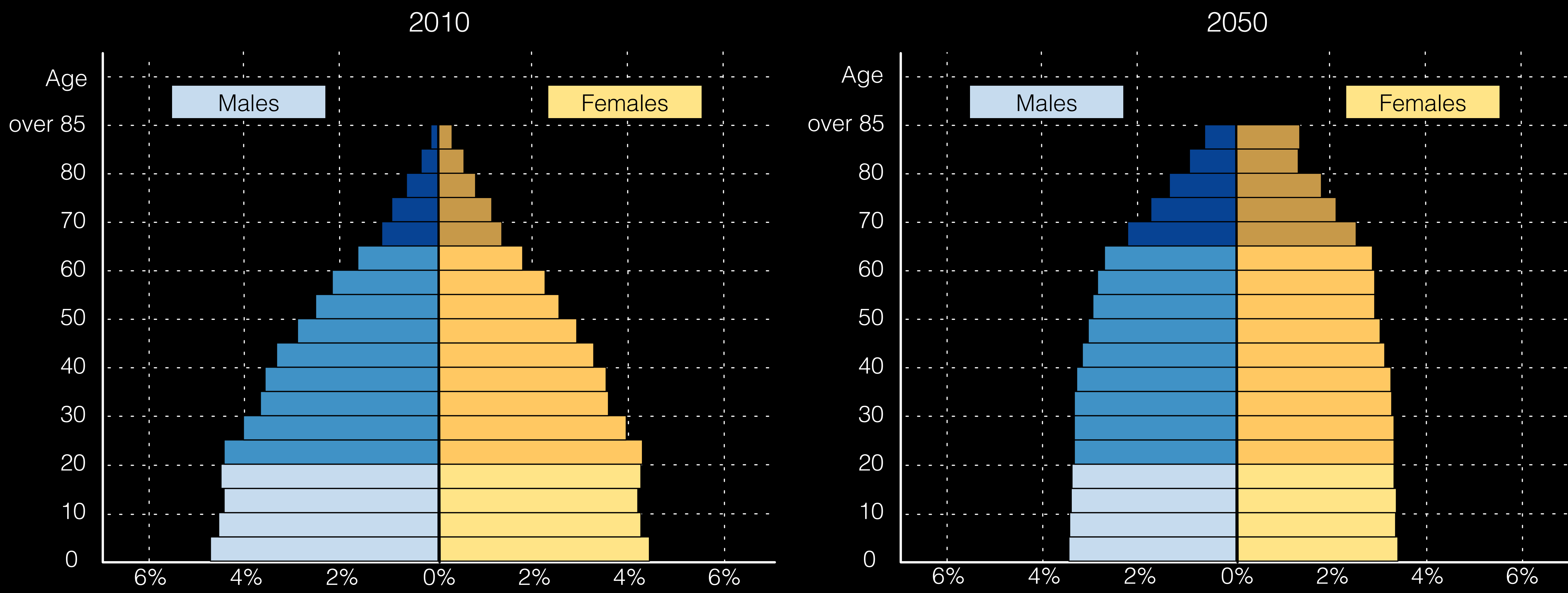
- Reasoning strategies
- Reasoning architecture

III. Validation & Verification

IV. Conclusion

Population ageing

– Major demographic changes *



* United Nations, World Population Prospects: The 2010 Revision. http://esa.un.org/unpd/ wpp/index.htm

Population ageing

- Elderly support ratio is shrinking
  - e.g. in Singapore *

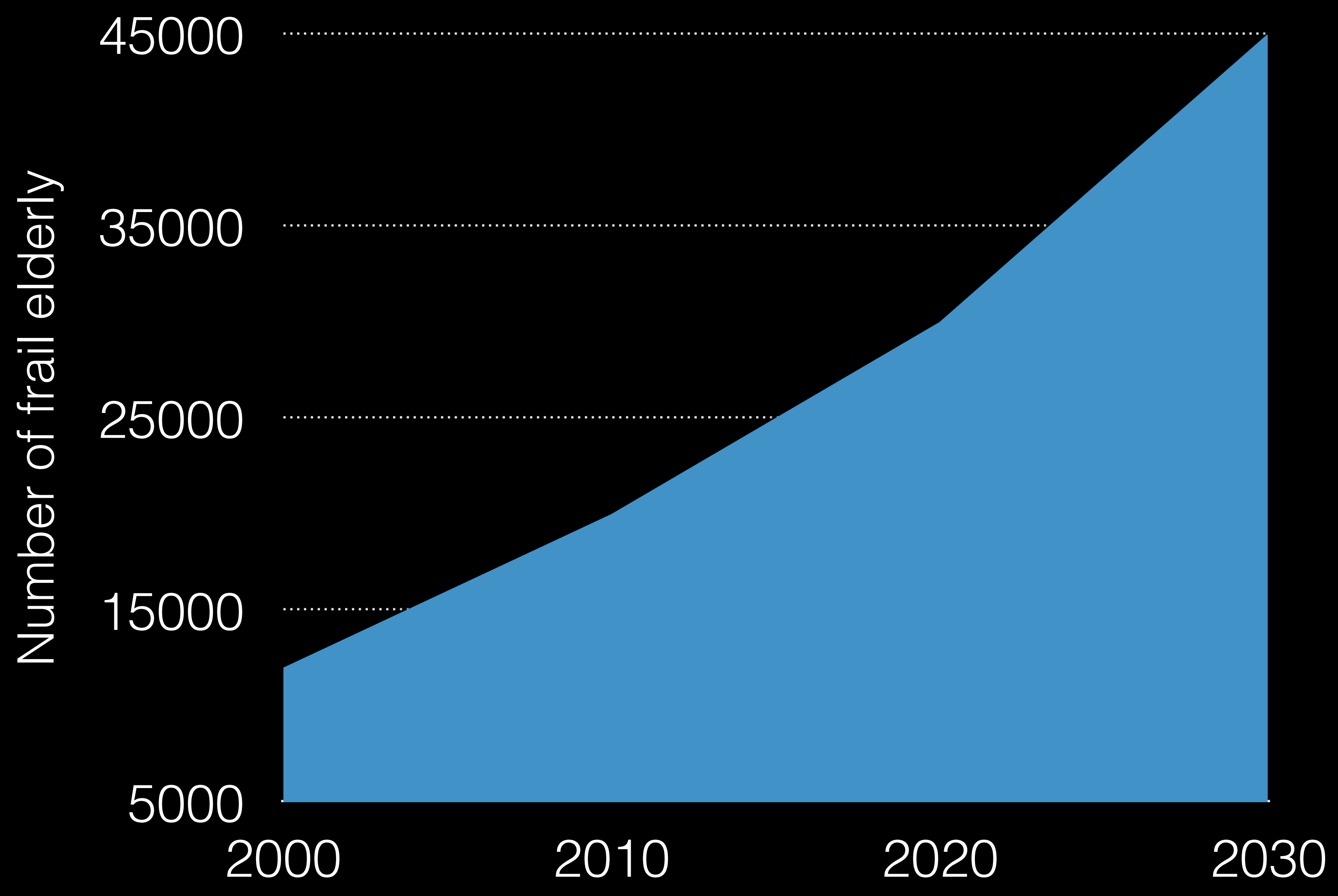
From 13.5 in 1970

to 2.6 in 2015!

* Resident Old-Age Support Ratio. Department of Statistics Singapore. http://www.singstat.gov.sg

Progression of dementia

– Projections of frail elderly population in Singapore *



7 000 beds available in nursing homes

<u>Need in 2015:</u> 22 000 beds

* Dr. Philip Yap, Khoo Teck Puat Hospital, Singapore

What is AAL?

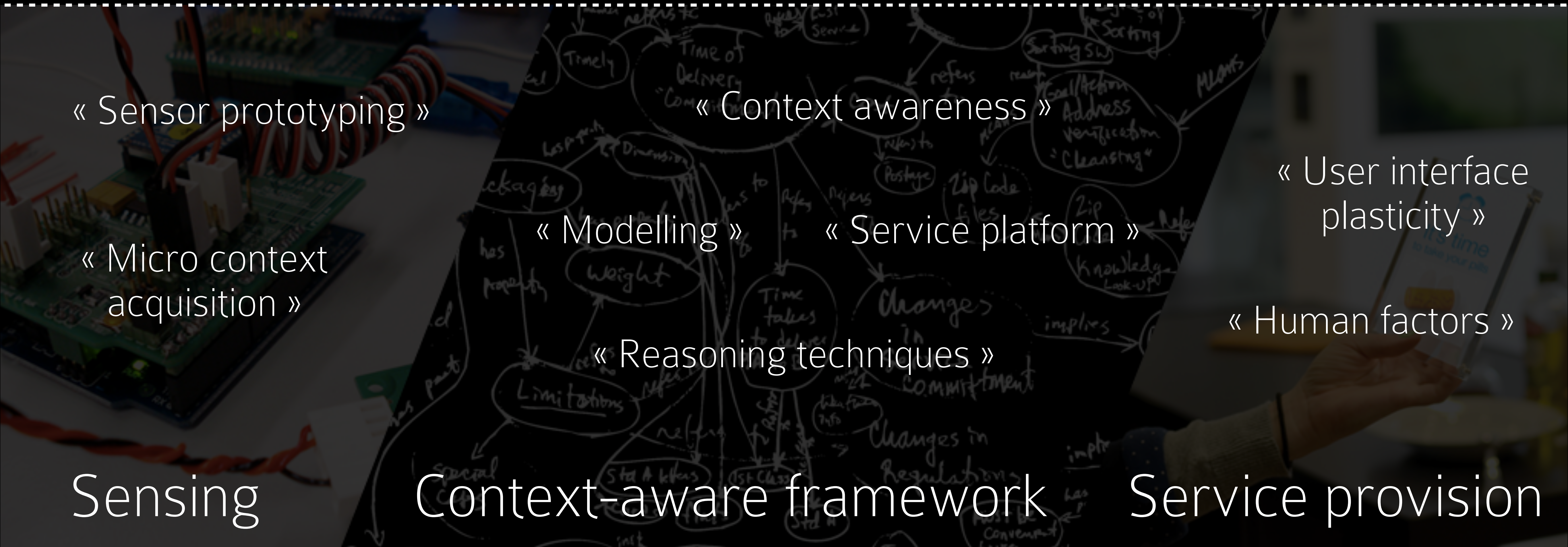
Sensing | Context-aware framework | Service provision

Scientific challenges

« Rules verification »

« Information consistency »

System verification



Communication layer

« Privacy »

« Discovery protocols »

« Wireless communication »

Scientific challenges

« Rules verification »

« Information consistency »

System verification

« Sensor prototyping »

« Micro context acquisition »

« Context awareness »

« Modelling »

« Service platform »

« Reasoning techniques »

« User interface plasticity »

« Human factors »

Sensing

Context-aware framework

Service provision

Communication layer

« Privacy »

« Discovery protocols »

« Wireless communication »

our research work

collaboration with

Liu Yan, Dong Jin Song, NUS (PAT)

Outline

I. Introduction

II. Context reasoning

- Reasoning strategies
- Reasoning architecture

III. Validation & Verification

IV. Conclusion

Related work

Rule-based (Semantic Web)

Sensor fusion
and
binding to context

+
leverage prior knowledge
no learning

-
no implicit patterns
sensitive to noise

Pattern matching (HMM, DBN...)

Detection of patterns
in sequence of
preprocessed sensor data

+
no prior knowledge
data driven

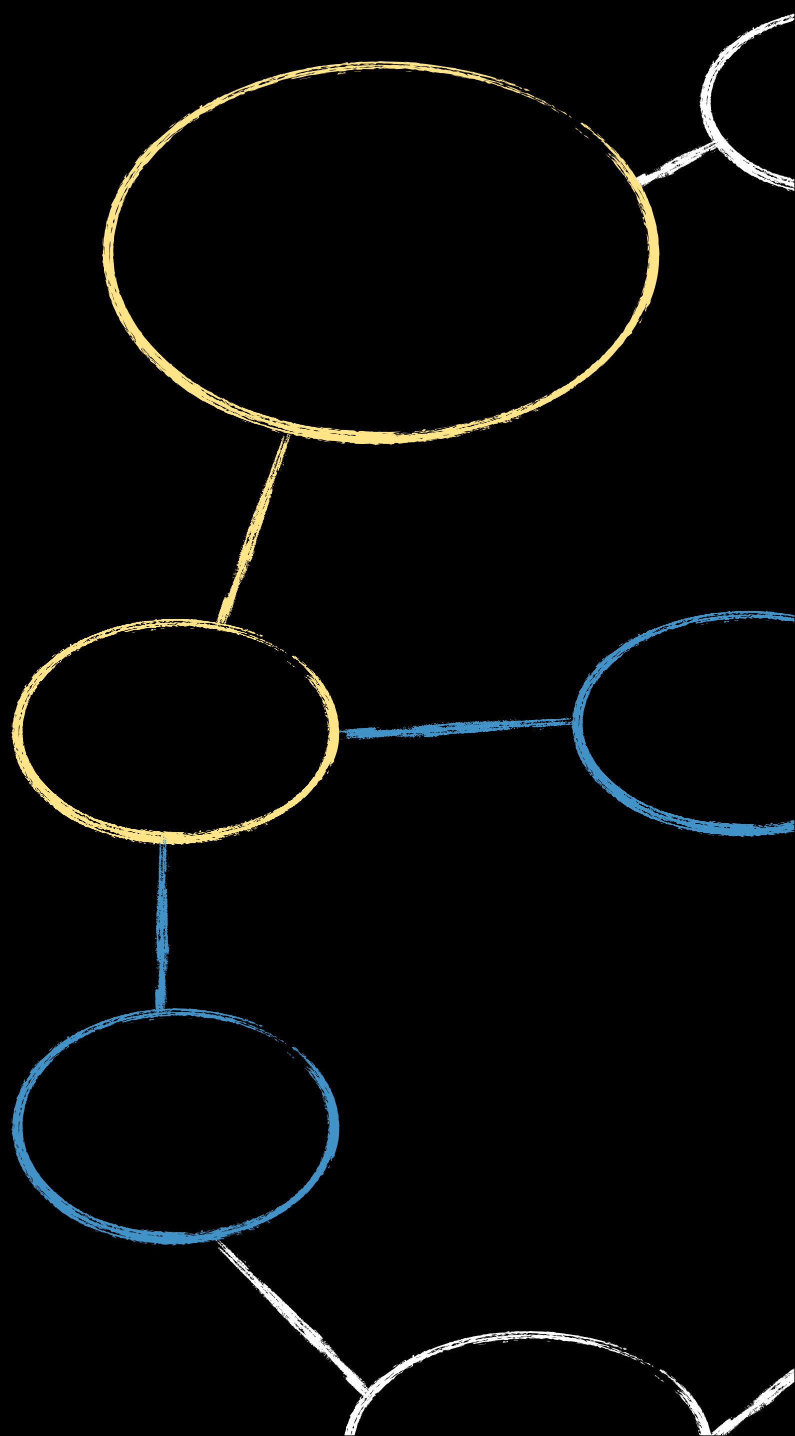
-
labelled dataset for learning
fine grain sensing

Semantic Web Technologies

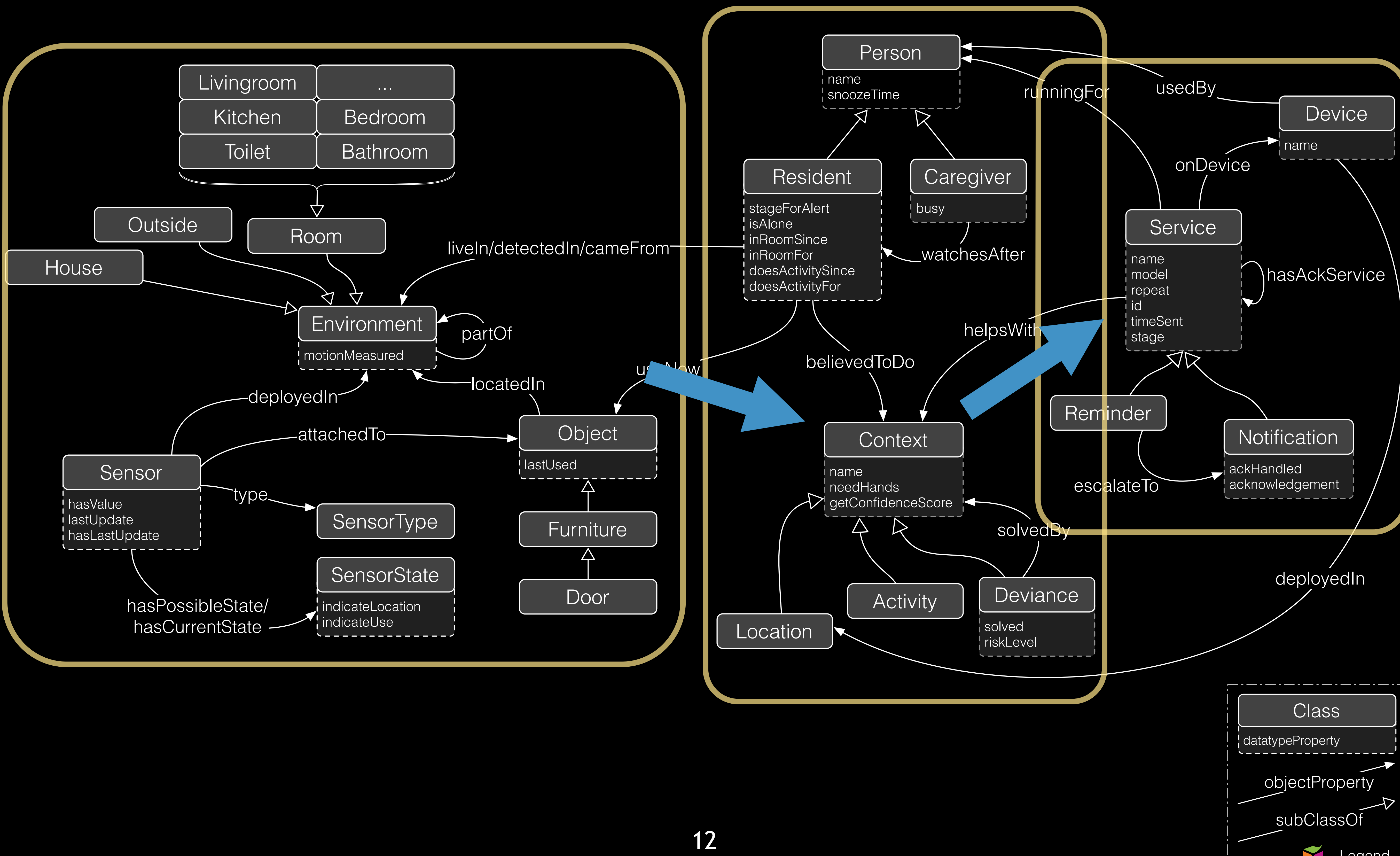
- Metadata-data graph model
- Make statements about resources
- Triple-based representation

  « subject predicate object »

- URI and namespaces
  - RDF, RDF-S, OWL (TBox)
  - Own namespaces (TBox + Abox)
- Serializations: XML, N3, ...



Functional context model



Rule design: epistemology

- Type I: Rules of reasoning
    - Irrevocable knowledge
    - Scientific models, common sense
- Type II: Rules of fact
    - Domain knowledge
    - Experts' experience, intuition and insight
    - Possible contradiction (scoring)
- Type III: Arbitration rules

Type I - Rules of reasoning

∀ Sensor se; SensorState st; Room r, r2; House h; Resident u; Timestamp t;

{se hasCurrentState st} ∧ {se hasLastUpdate true} ∧ {st indicateLocation true} ∧ {se deployedIn r} ∧ {r partOf h} ∧ {u liveIn h} ∧ {u detectedIn r2} ∧ {r2 ≠ r} ∧ {se lastUpdate t} ⟹ {u detectedIn r} ∧ {u cameFrom r2} ∧ {u inRoomSince t}

```
@prefix log:  <http://www.w3.org/2000/10/swap/log#>.
@prefix math: <http://www.w3.org/2000/10/swap/math#>.
@prefix ts:   <triplestore#>.

## track resident location [live + persistent]
{?se qol:hasCurrentState ?st. ?se qol:hasLastUpdate true. ?st qol:indicateLocation true
    . ?se qol:deployedIn ?r. ?r qol:partOf ?h. ?u qol:liveIn ?h. ?u qol:detectedIn ?r2.
     ?r log:notEqualTo ?r2. ?se qol:lastUpdate ?t} => {?u qol:detectedIn ?r. ?u qol:
    cameFrom ?r2. ?u qol:inRoomSince ?t. ts:n3store ts:update {?u qol:detectedIn ?r. ?u
     qol:cameFrom ?r2. ?u qol:inRoomSince ?t}}.

## infer durations [live only]
{?since qol:hasDurationEquivalent ?for. ?x ?since ?start. hom:clock qol:hasValue ?now.
    (?now ?start) math:difference ?duration} => {?x ?for ?duration}.
```

Type II - Rules of fact

```
## go to the toilet (+ max duration)
{?u qol:detectedIn ?r. ?r a qol:Toilet. ?u qol:inRoomFor ?d. ?d math:lessThan ?u!prof:
    maxToiletDuration} => {hom:goToilet :getScore 7}.

## occupied (+ max duration)
{?u qol:liveIn ?h. ?h qol:motionMeasured ?m. ?m math:notLessThan ?u!prof:
    minOccupiedMotion} => {hom:occupied :getScore 2}.
{?u qol:believedToDo hom:occupied. ?u qol:doesActivityFor ?d. ?d math:notLessThan ?u!
    prof:maxOccupiedDuration} => {hom:occupied :getScore -1}.

## sleep (+ max duration)
{?u qol:detectedIn ?r. ?r a qol:Bedroom. ?u qol:inRoomFor ?d. ?d math:notLessThan ?u!
    prof:minSleepInitiation. ?r qol:motionMeasured ?m. ?m math:lessThan ?u!prof:
    maxSleepMotion} => {hom:sleep :getScore 6}.
{?u qol:believedToDo hom:sleep. ?u qol:doesActivityFor ?d. ?d math:notLessThan ?u!prof:
    maxSleepDuration} => {hom:sleep :getScore -5}.

## run away
{?u qol:useNow ?d. ?d a qol:Door. hom:clock qol:hasValue ?t. ?t func:hours-from-
    dateTime ?h. ?h math:notLessThan ?u!prof:outTooLate. ?h math:lessThan ?u!prof:
    outTooEarly} => {hom:runAway :getScore 9}.
{?u qol:detectedIn ?o. ?o a qol:Outside. ?u qol:inRoomFor ?d. ?d math:notLessThan ?u!
    prof:outTooLong} => {hom:runAway :getScore 9}.

## fall
{?u qol:believedToDo hom:nothing. ?u qol:doesActivityFor ?d. ?d math:notLessThan ?u!
    prof:maxInactiveDuration} => {hom:fall :getScore 8}.

## meet people at home
{?u qol:isAlone false} => {hom:socialize :getScore 8}.
```

Type III - Arbitration rules

Compute the Rule-Based Confidence Score

```
## sum scores and compute confidence [live only]
## if no activity stand out then give score to hom:nothing [live only]
{?ac :getScore ?x. ?SCOPE e:findall (?sc {?ac :getScore ?sc} ?list). ?list math:sum ?
    total} => {?ac :getFinalScore ?total}.
{?SCOPE e:findall (?sc {?ac :getScore ?sc} ?list). ?list math:sum ?grandtotal. ?
    grandtotal math:lessThan 2} => {hom:nothing :getScore 10}.
{?SCOPE e:findall (?sc {?ac :getFinalScore ?sc} ?list). ?list math:sum ?grandtotal. ?
    grandtotal math:notLessThan 0.1. ?ac0 :getFinalScore ?fs. (?fs ?grandtotal) math:
    quotient ?cs} => {?ac0 qol:getRBConfidenceScore ?cs}.

## infer most probable activity [persistent only]
{?SCOPE e:findall (?rbc {?ac qol:getRBConfidenceScore ?rbc} ?list). ?list e:max ?maxrbc
    . ?ac qol:getRBConfidenceScore ?maxrbc. ?u qol:liveIn ?h. hom:clock qol:hasValue ?
    now} => {ts:n3store ts:update {?u qol:believedToDo ?ac. ?u qol:doesActivitySince ?
    now}}.
```

Outline

I. Introduction

II. Context reasoning

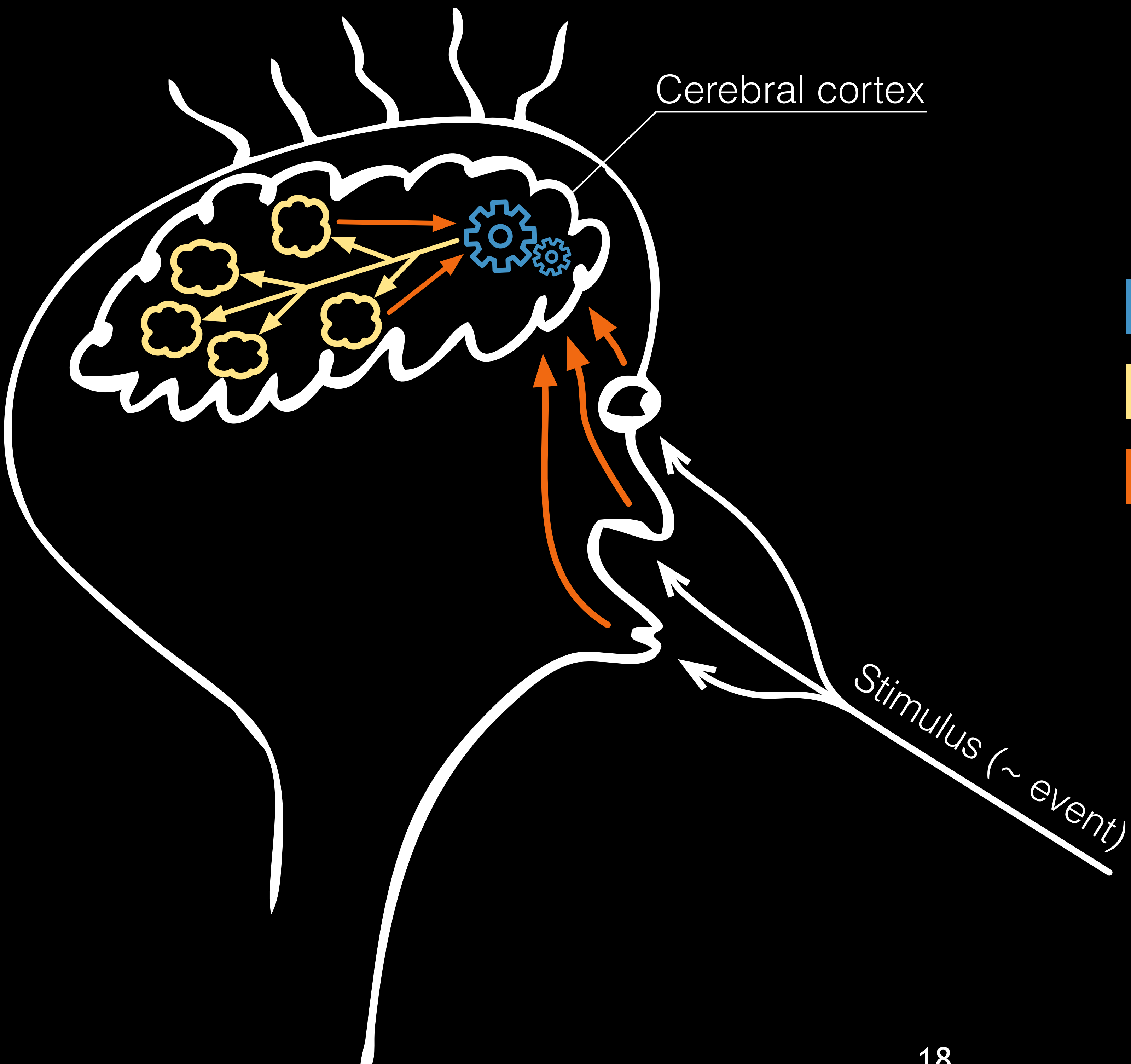
- Reasoning strategies
- Reasoning architecture

III. Validation & Verification

IV. Conclusion

« La Pensée »

A Cognitively Inspired Hybrid Reasoning Architecture

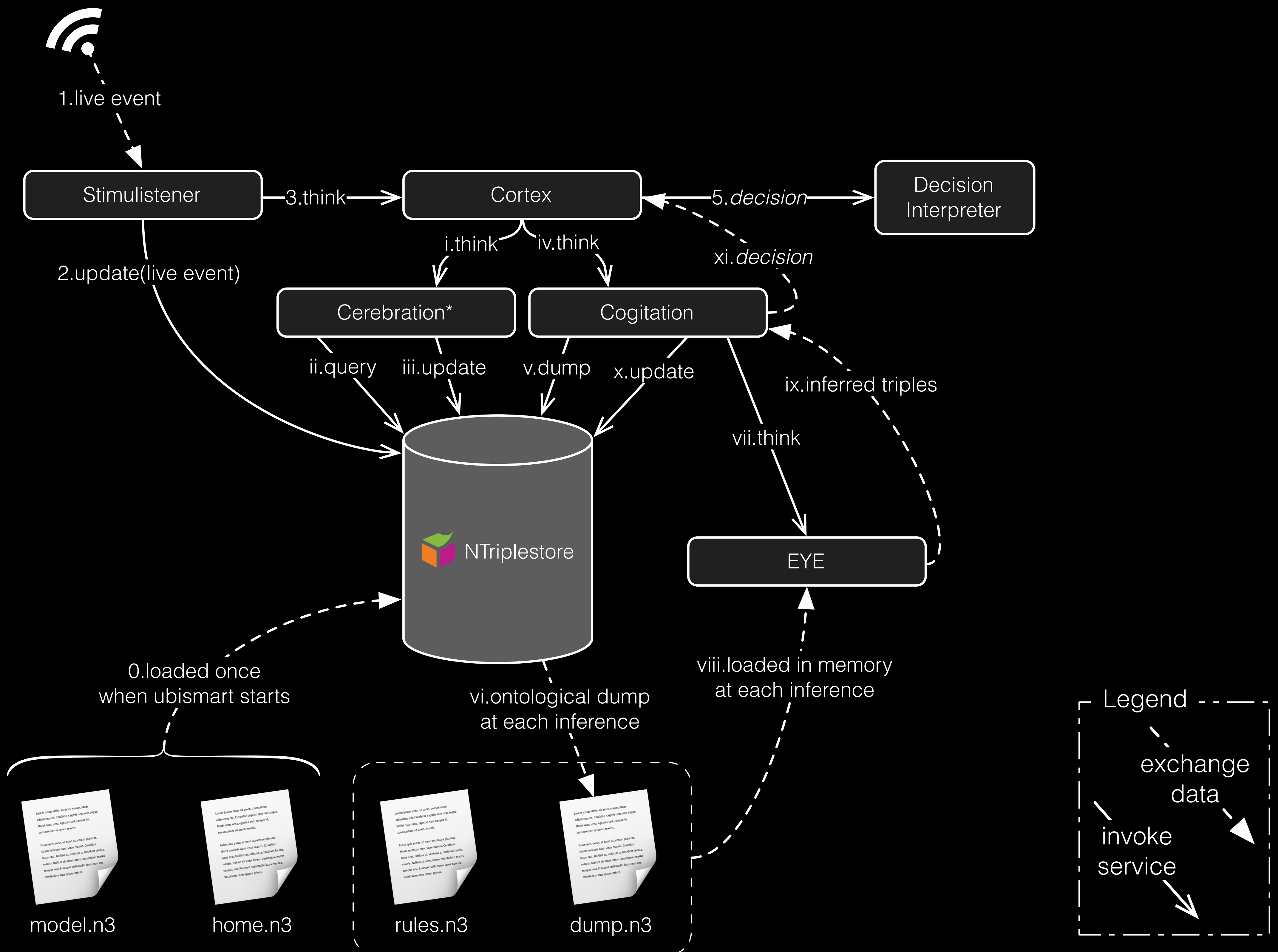


Cogitation (conscious mind)

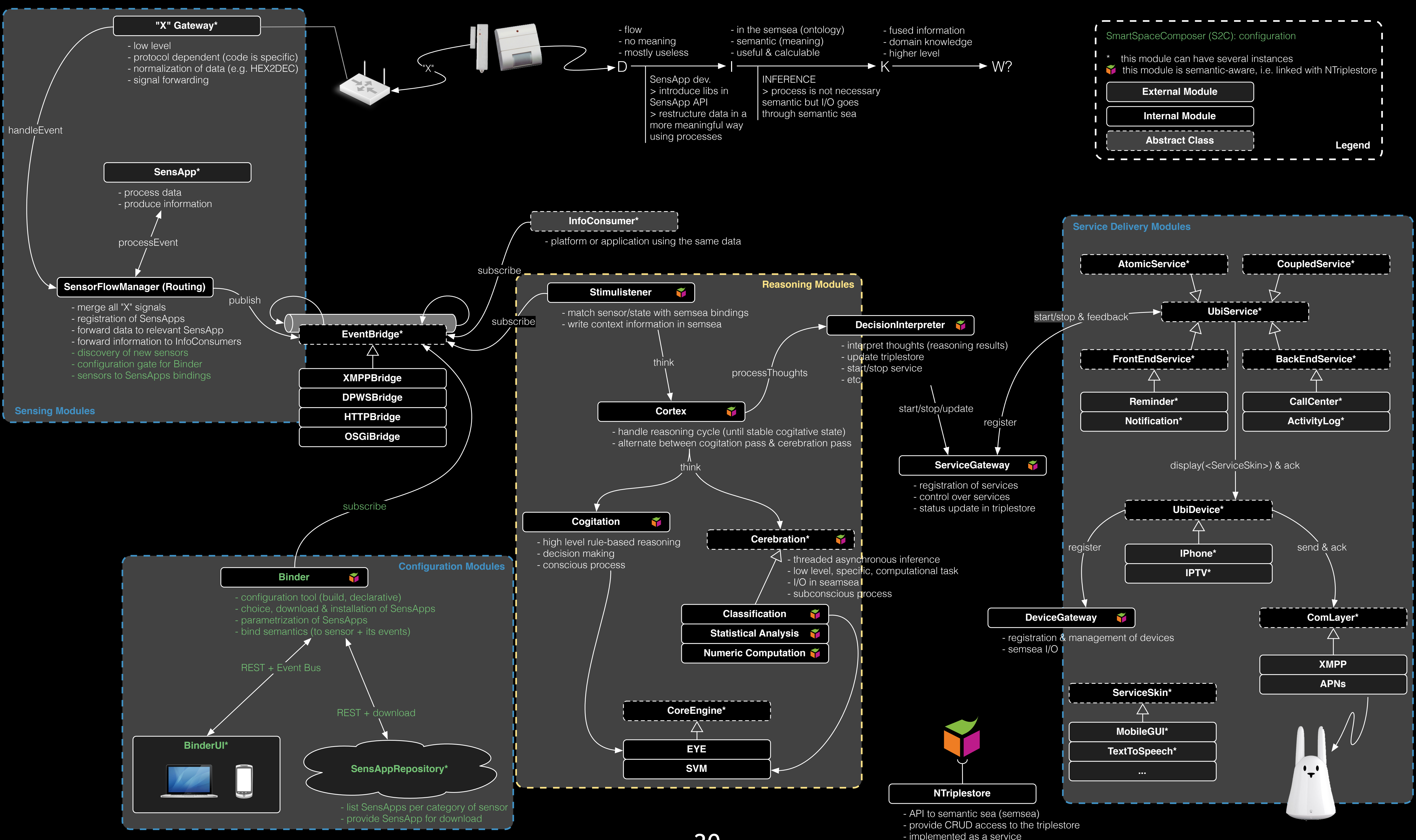
Cerebration (unconscious mind)

Knowledge formalisation

Live event processing

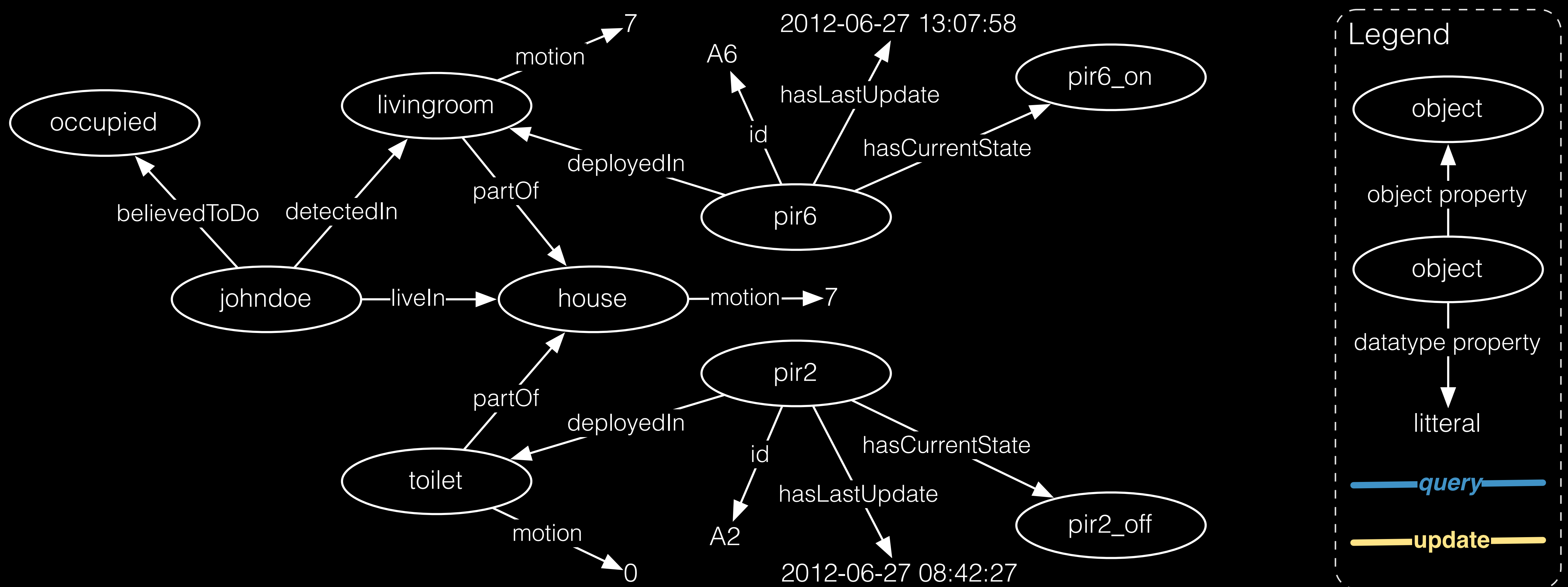


Hybrid reasoning



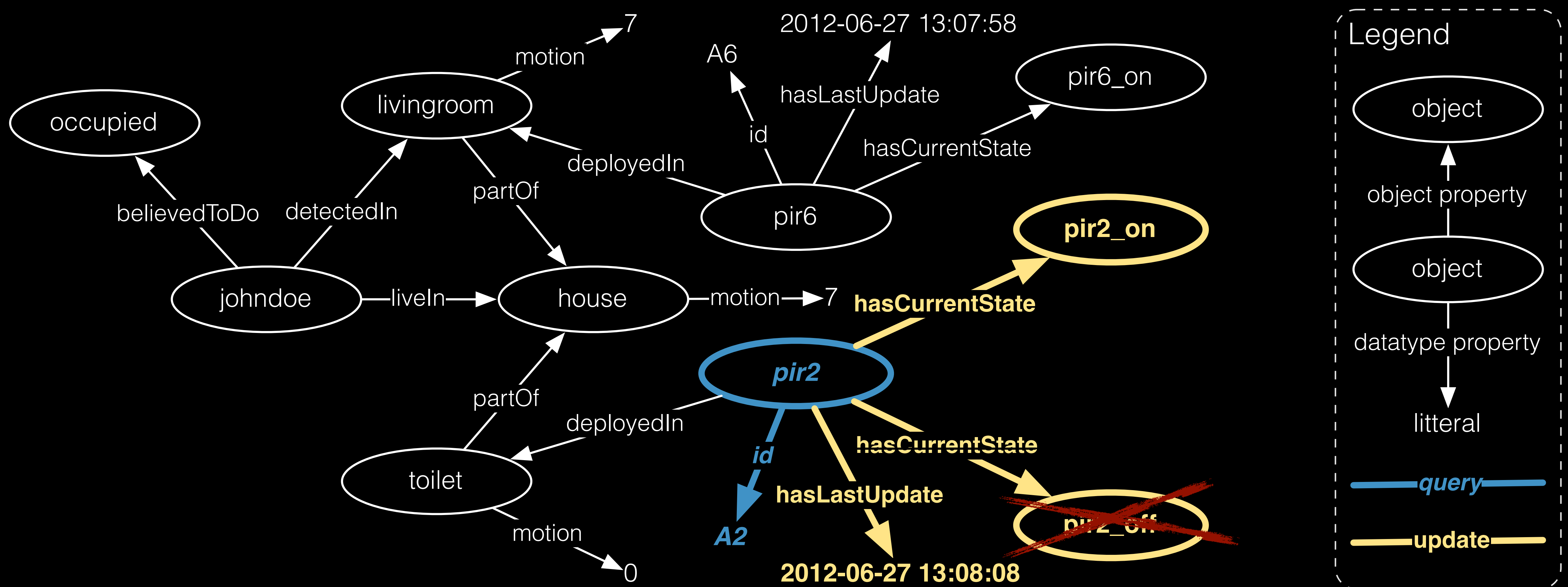
Ontology evolution breakdown

– initial ontology



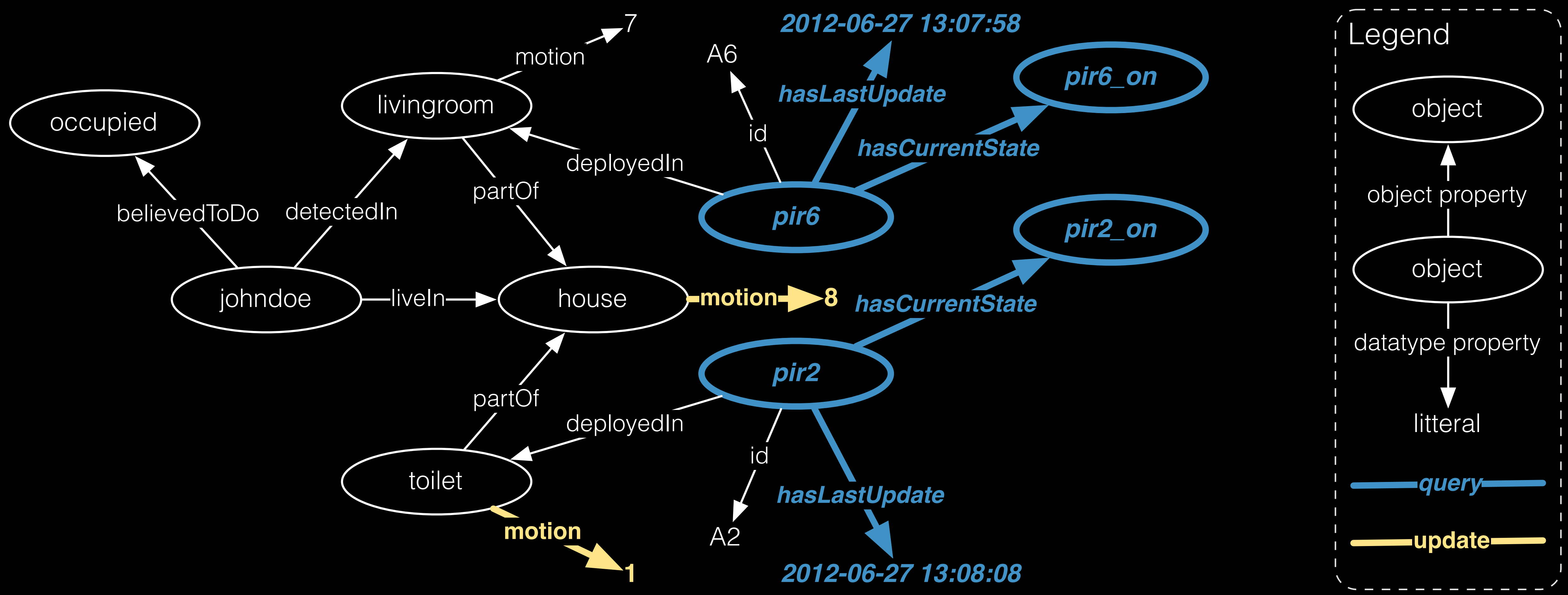
Ontology evolution breakdown

- Stimulistener: new event update



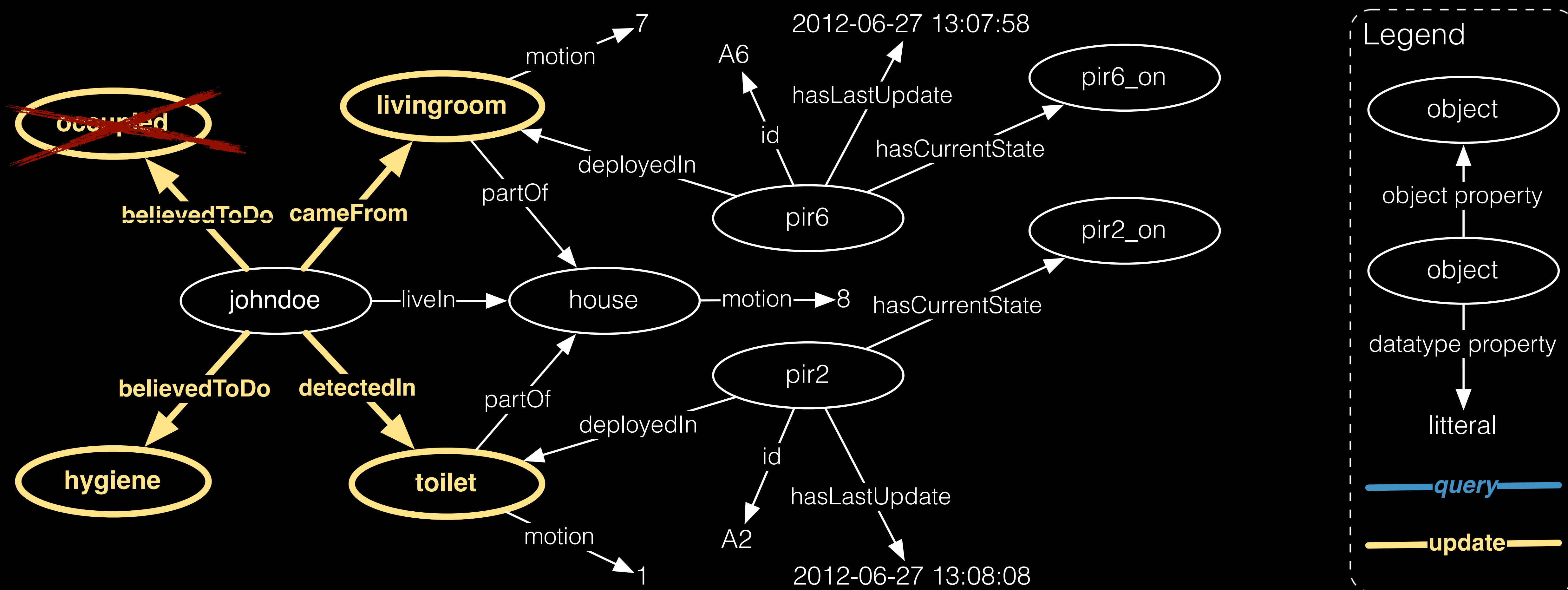
Ontology evolution breakdown

– Cerebration: motion estimation



Ontology evolution breakdown

– Cogitation: activity inference



{?u qol:detectedIn ?r. ?r a qol:Bathroom. ?u qol:inRoomFor ?d. ?d math:lessThan 1800. ?a1 a qol:Hygiene. ?u qol:canDo ?a1} => {?a1 qol:getScore 9}.

Outline

I. Introduction

II. Context reasoning

– Reasoning strategies

– Reasoning architecture

III. Validation & Verification

IV. Conclusion

Traditional AAL validation

- Difficulty to obtain relevant ground truth
  - Synthetic dataset
  - Role-played and labelled dataset (living lab)

- Irrelevant to real settings: is system usable?
- Does not validate economical "performance"
- Does not certify system behaviour

A 3-fold validation process

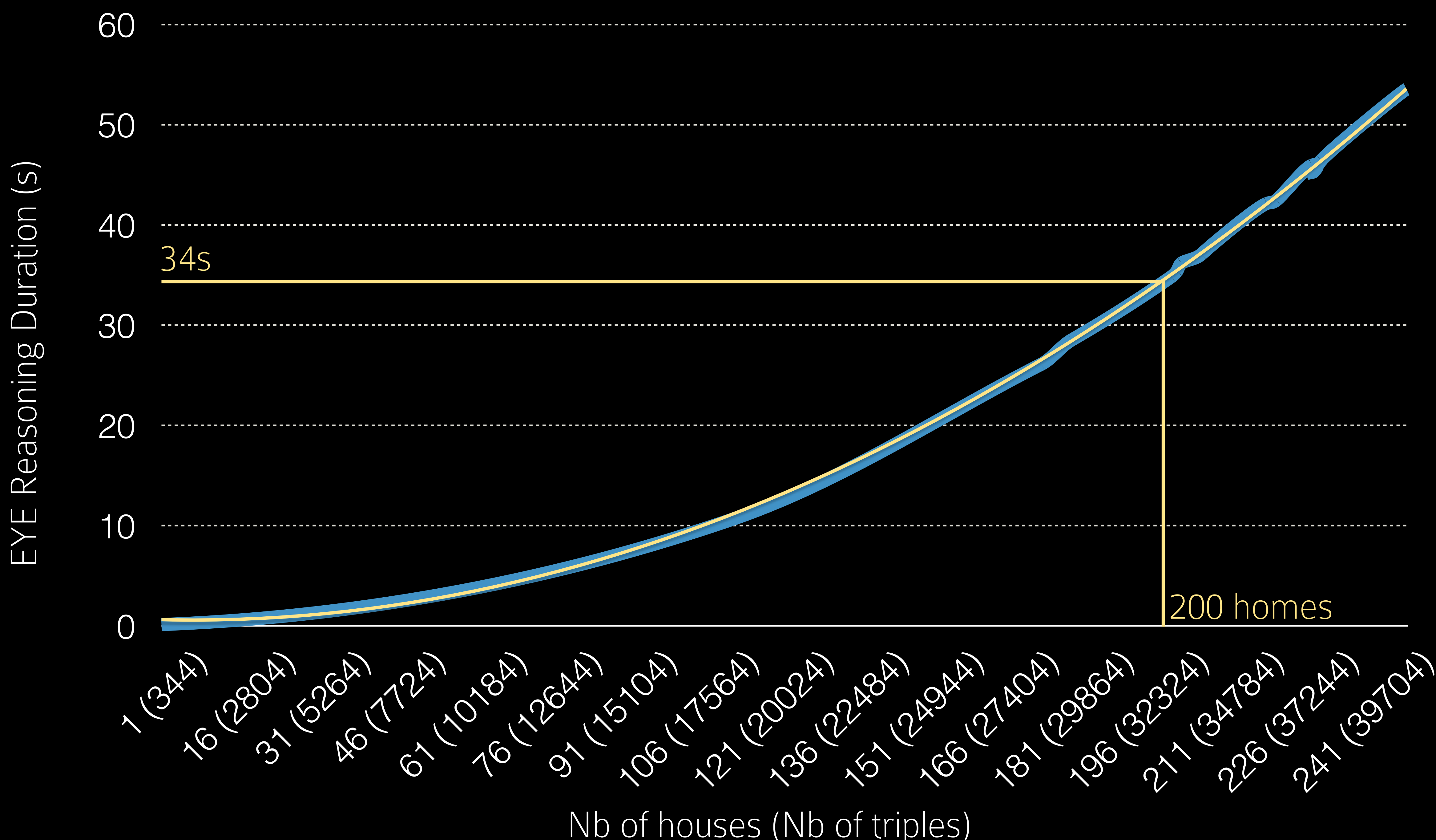
- Performance: Load test

- Usability: In-situ system trial
  - STARhome: controlled (technical validation)
  - Singapore nursing home: semi-controlled (top-down)
  - French individual homes: no control (bottom-up)

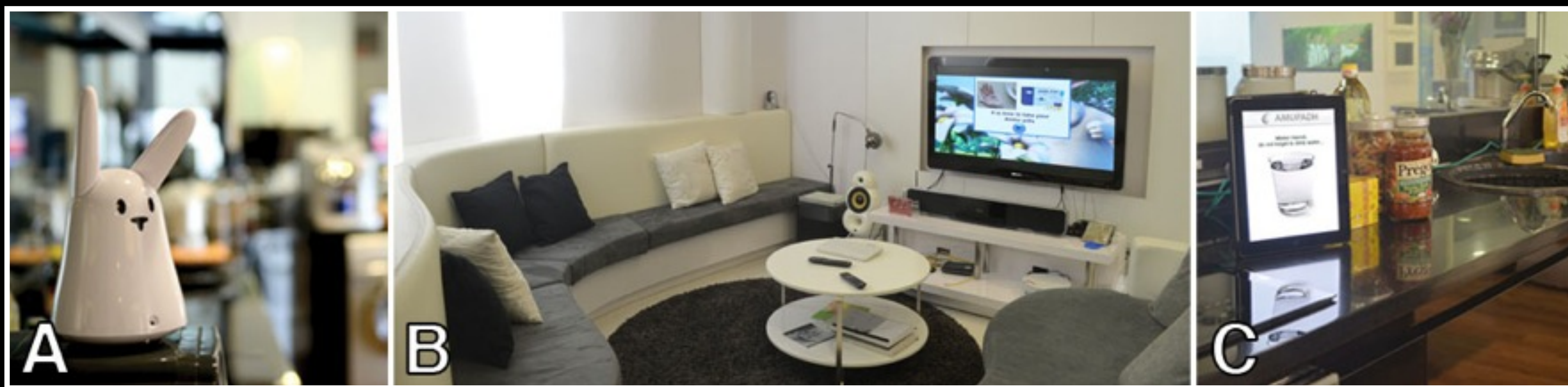
- Certification: Rules verification

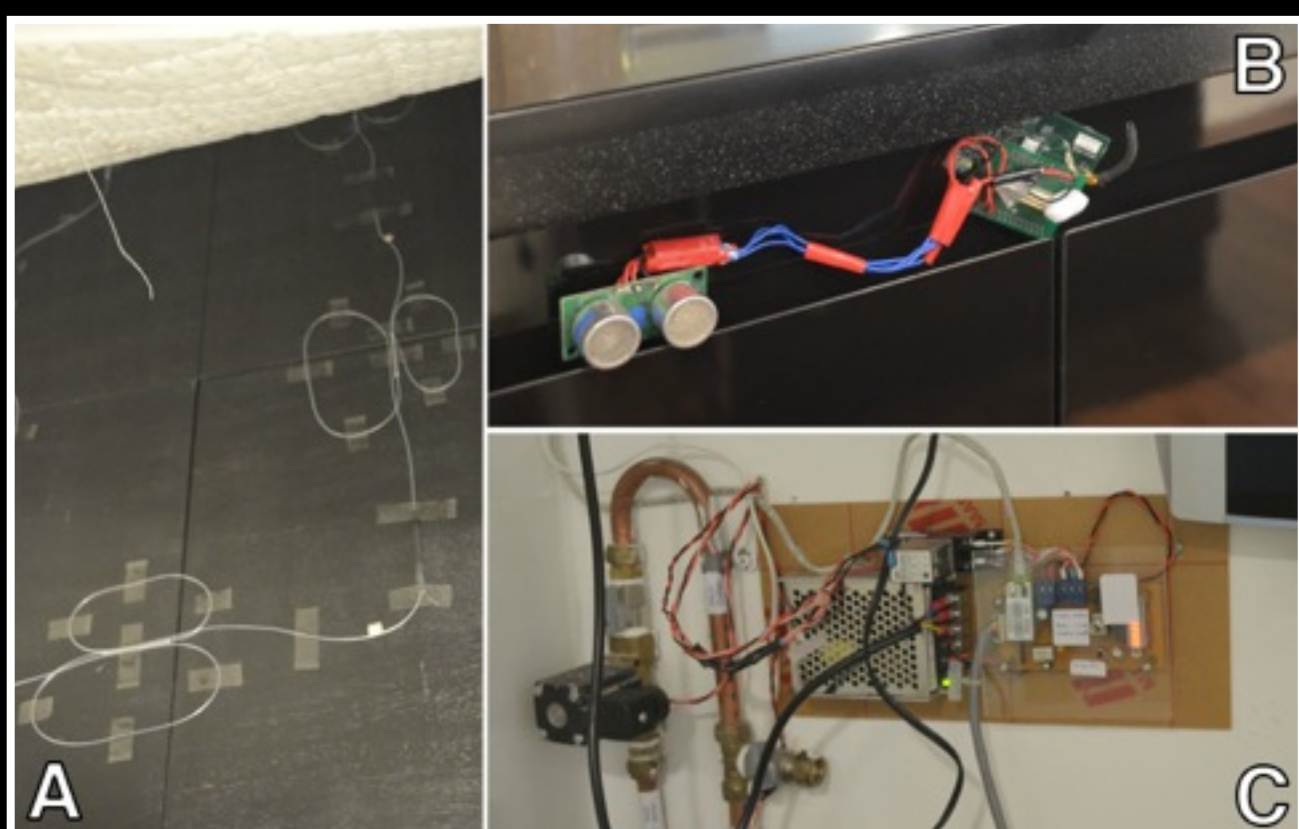
Performance

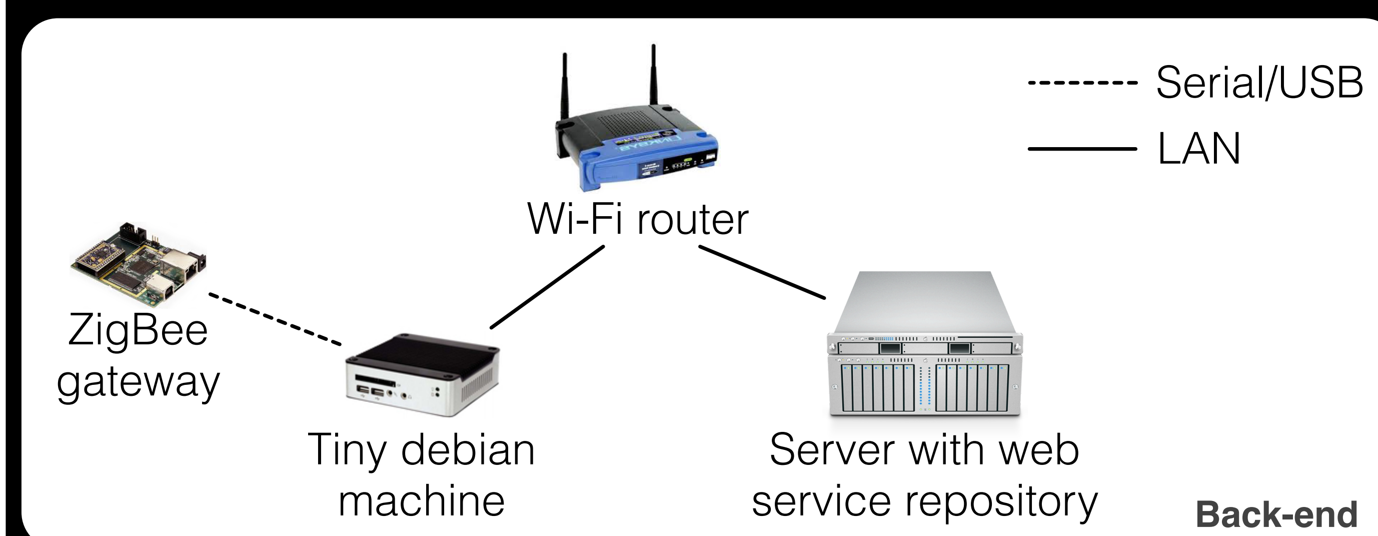
- Reasoning load test to 250 houses (40K+ triples)



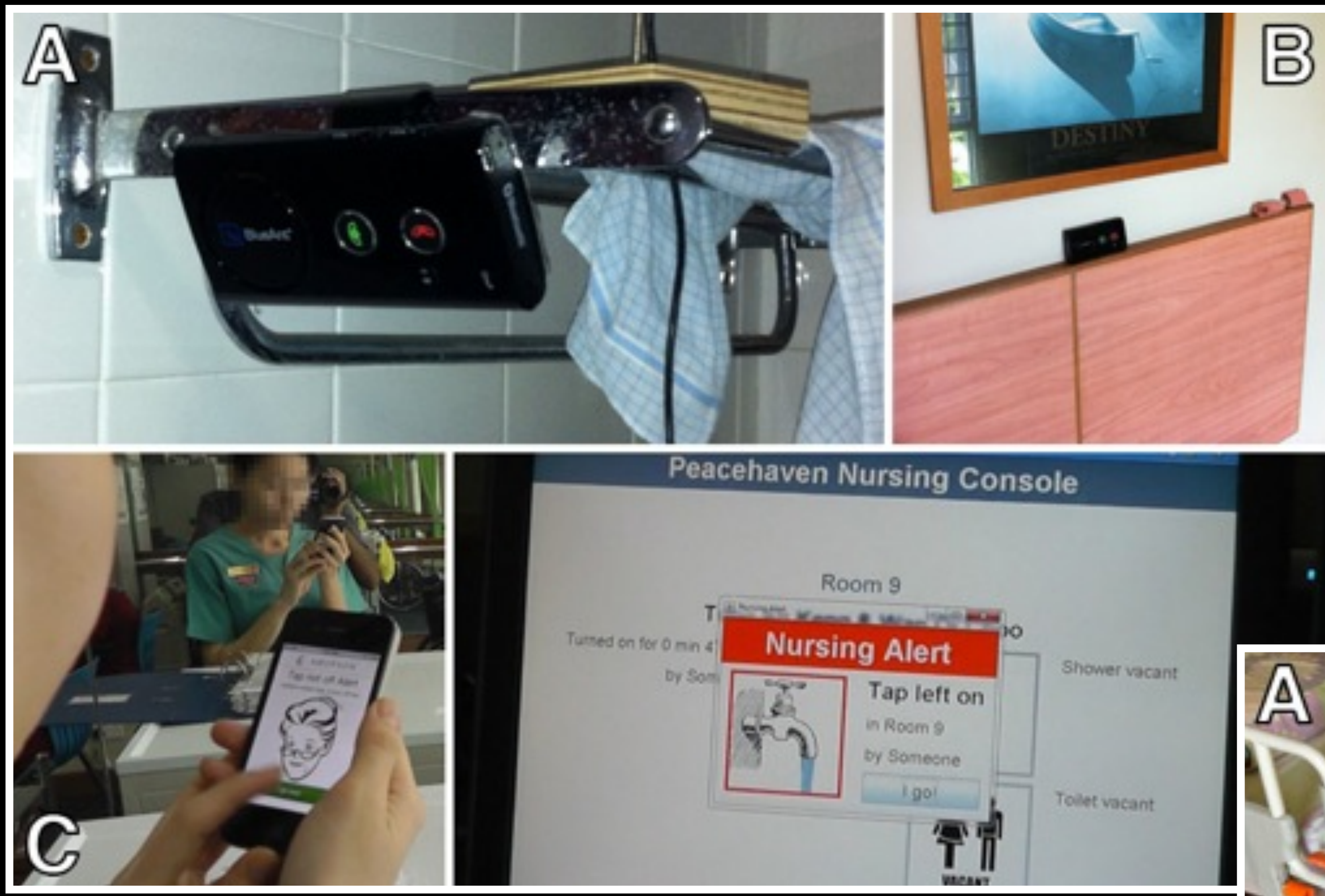
STARhome


A
B
C


A
B
C


Serial/USB
LAN
Wi-Fi router
ZigBee gateway
Tiny debian machine
Server with web service repository
Back-end

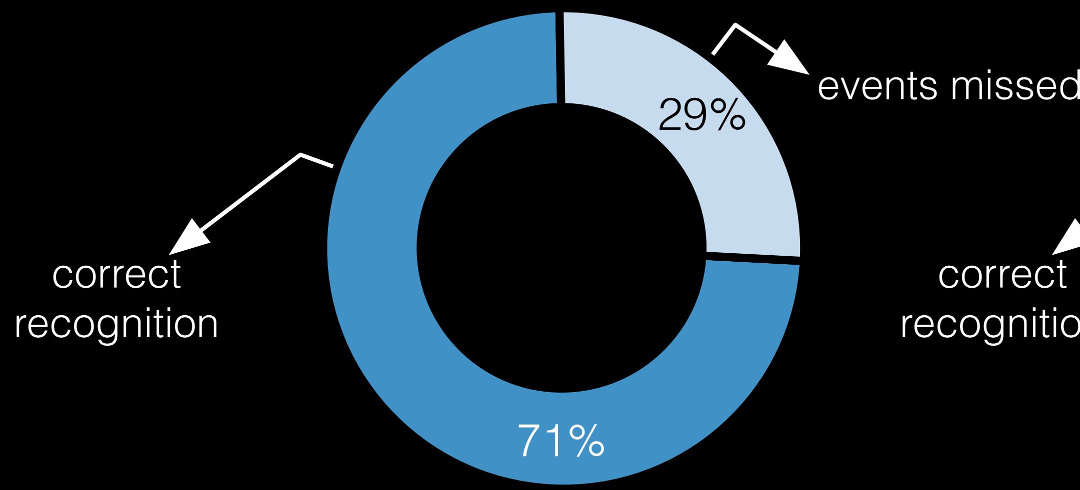
Nursing home in Singapore



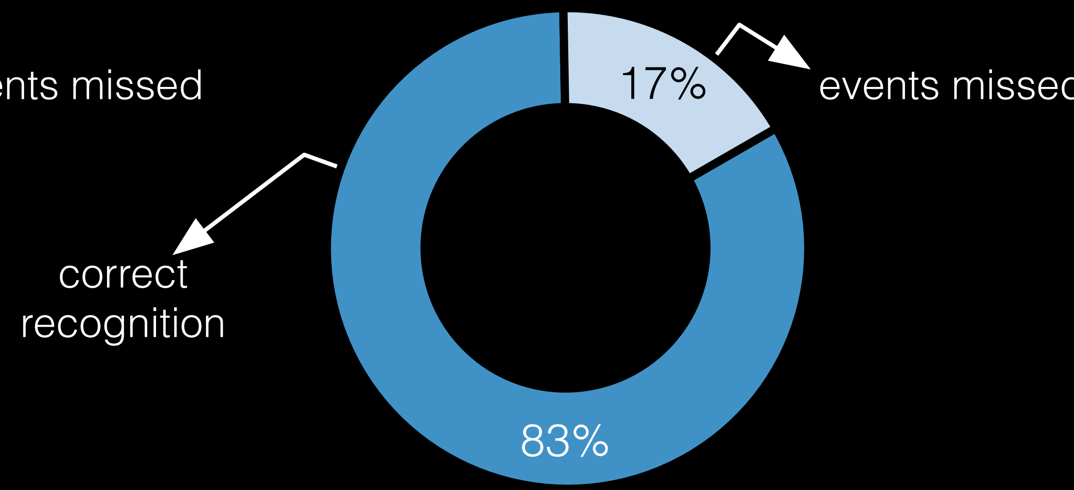
- 14 months trial
- 8 residents with dementia
- 2 caregivers

Results

- Recognition rate of events

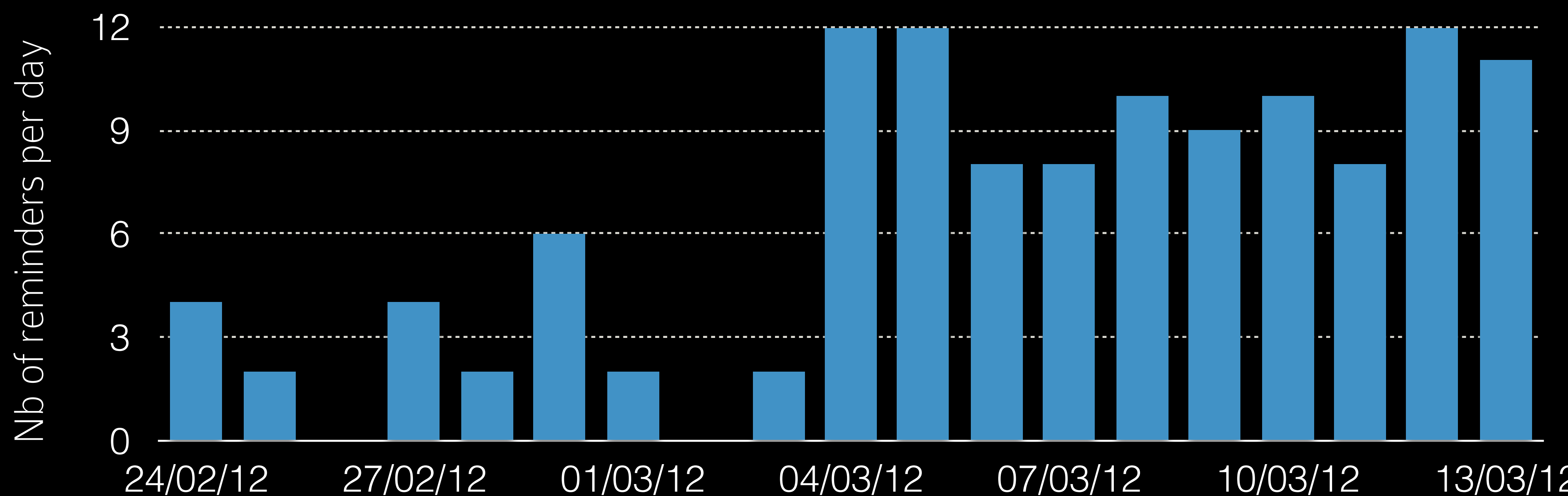


Feb. 2012 (phase 2)

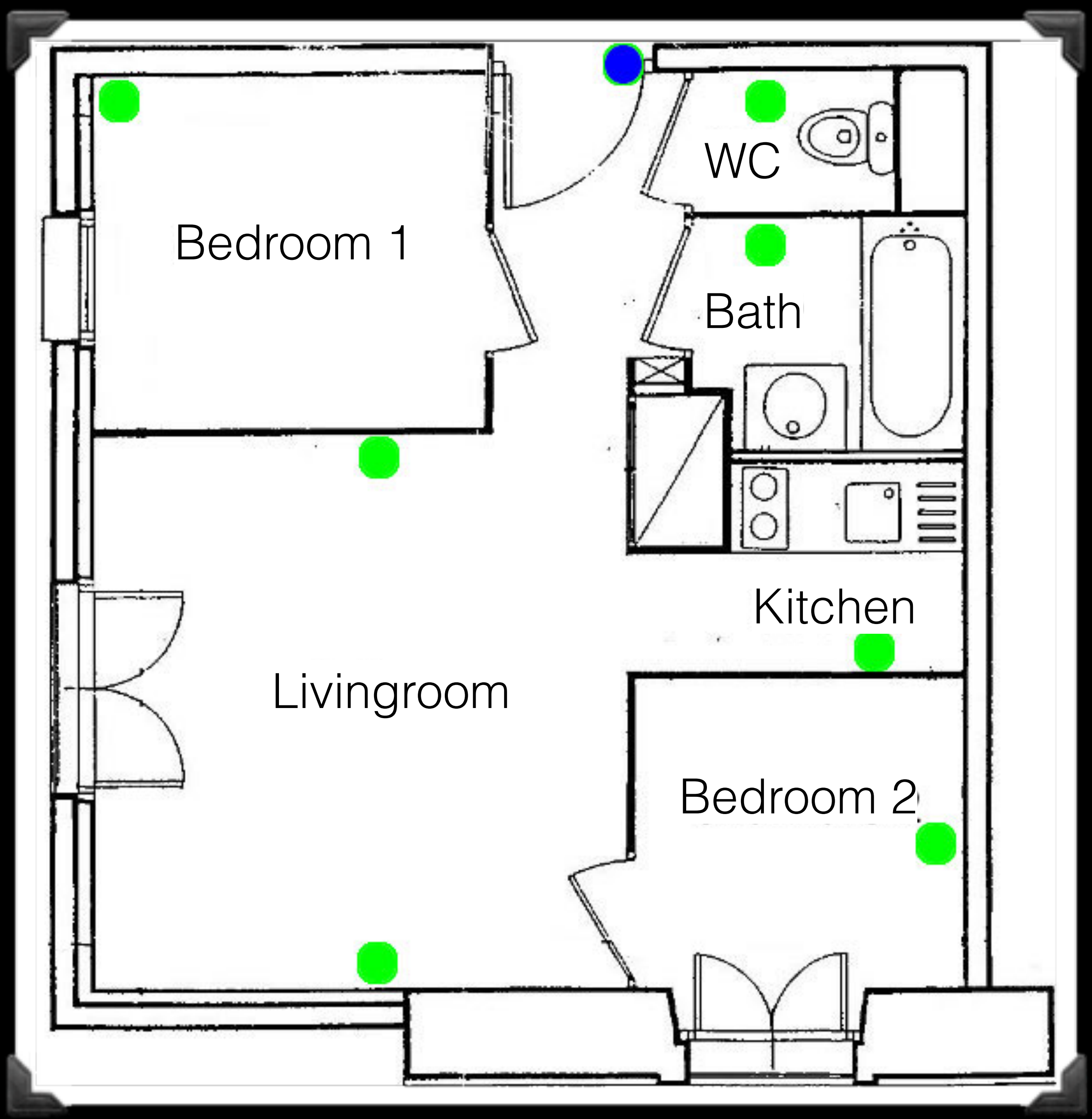


Sep. 2012 (phase 3)

- Early detection of the health issues



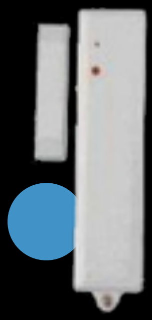
Individual homes in France



- 3 homes
- 2 months trial in average

motion sensor

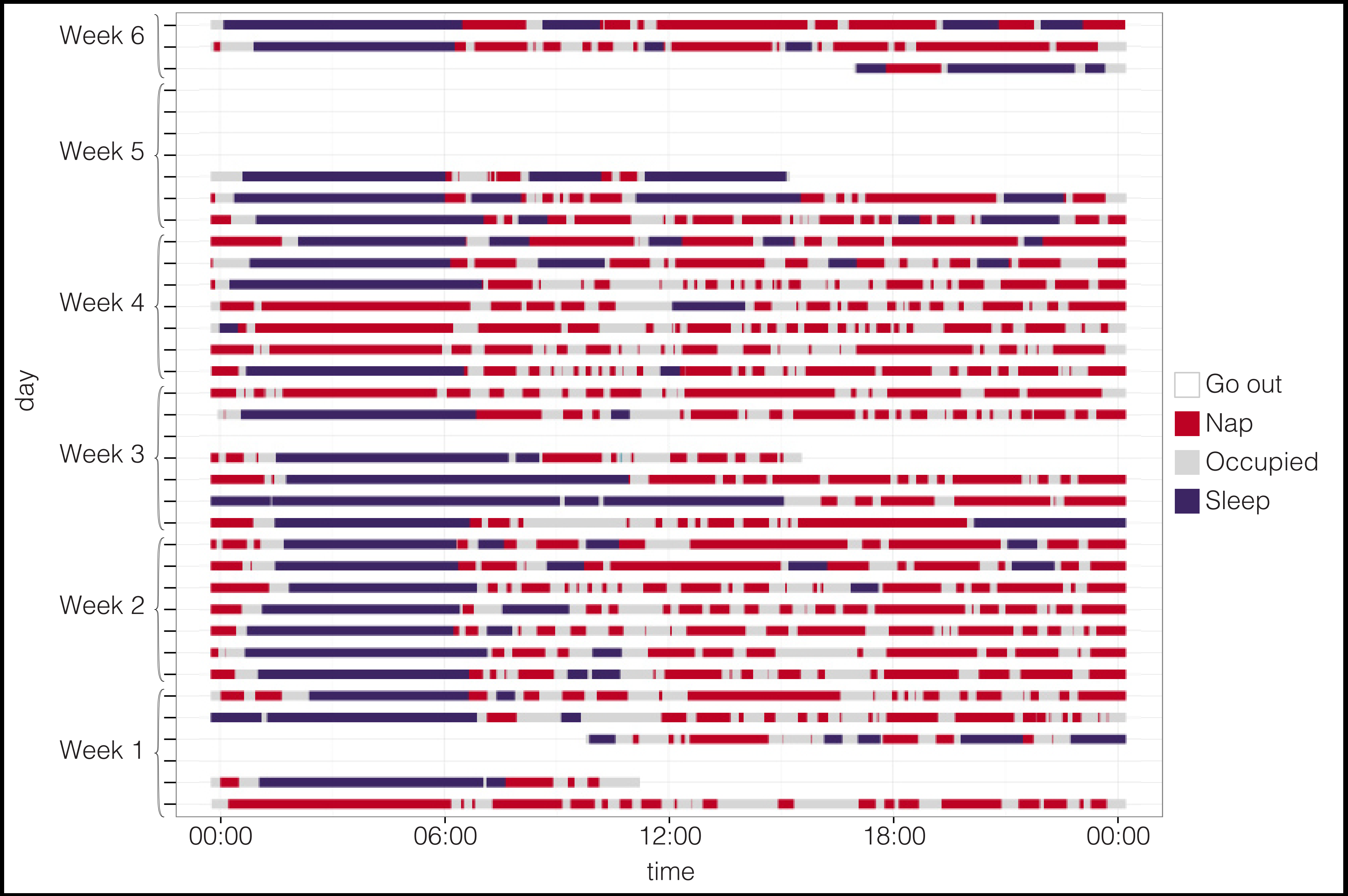
door open/close sensor



internet-connected gateway (BeagleBone)

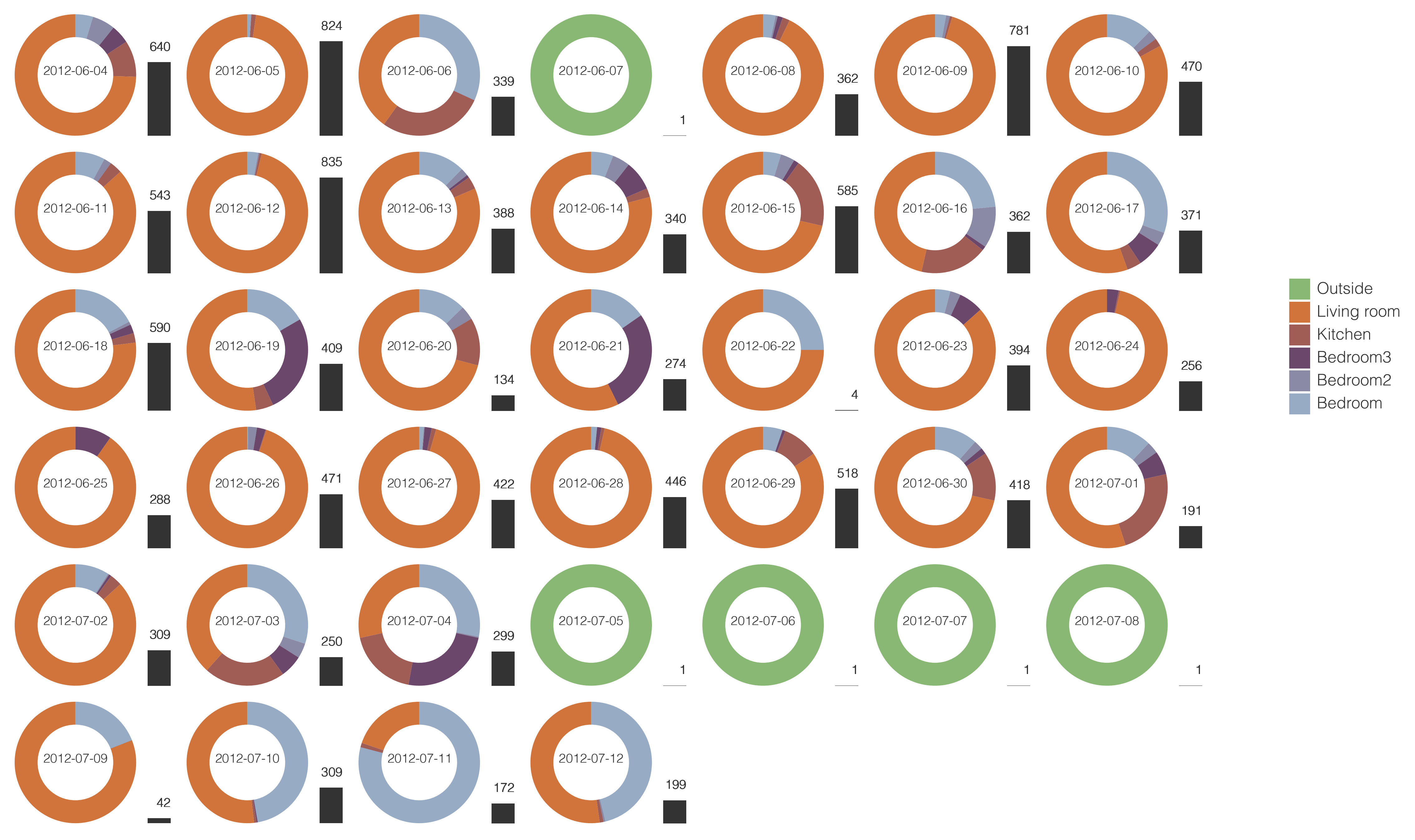
Results

- Activity inference



Results

– Contextual dashboard



Outline

I. Introduction

II. Context reasoning

   – Reasoning strategies

   – Reasoning architecture

III. Validation & Verification

IV. Conclusion

Liu Yan & Dong Jin Song (PAT)
@Peacehaven Nursing Home

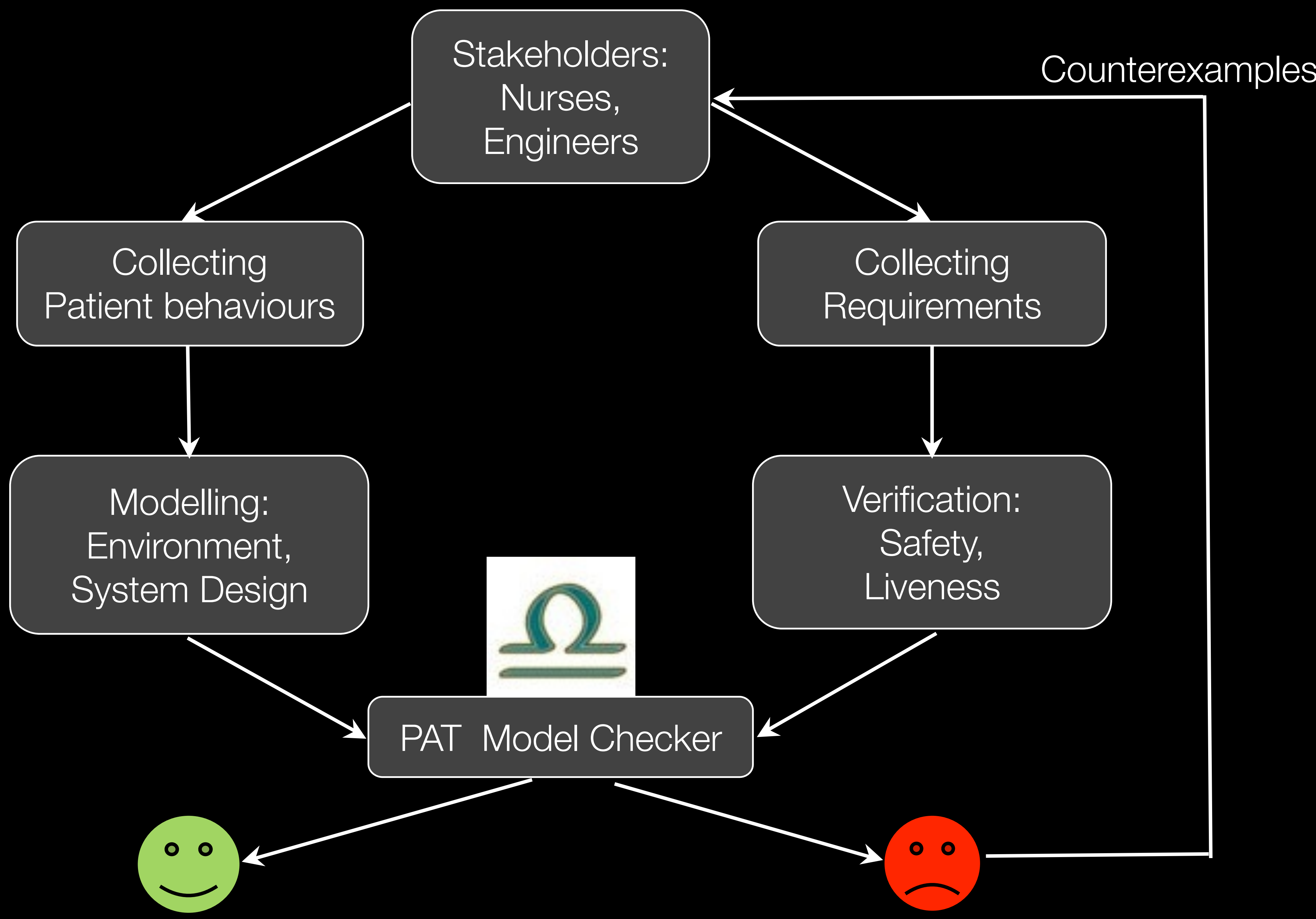
Model checking of AAL system

- People + environment + system modelling
    - Formalisms for concurrent interaction

- Comprehensive verification for certification
    - Automatic verification and exhaustive search

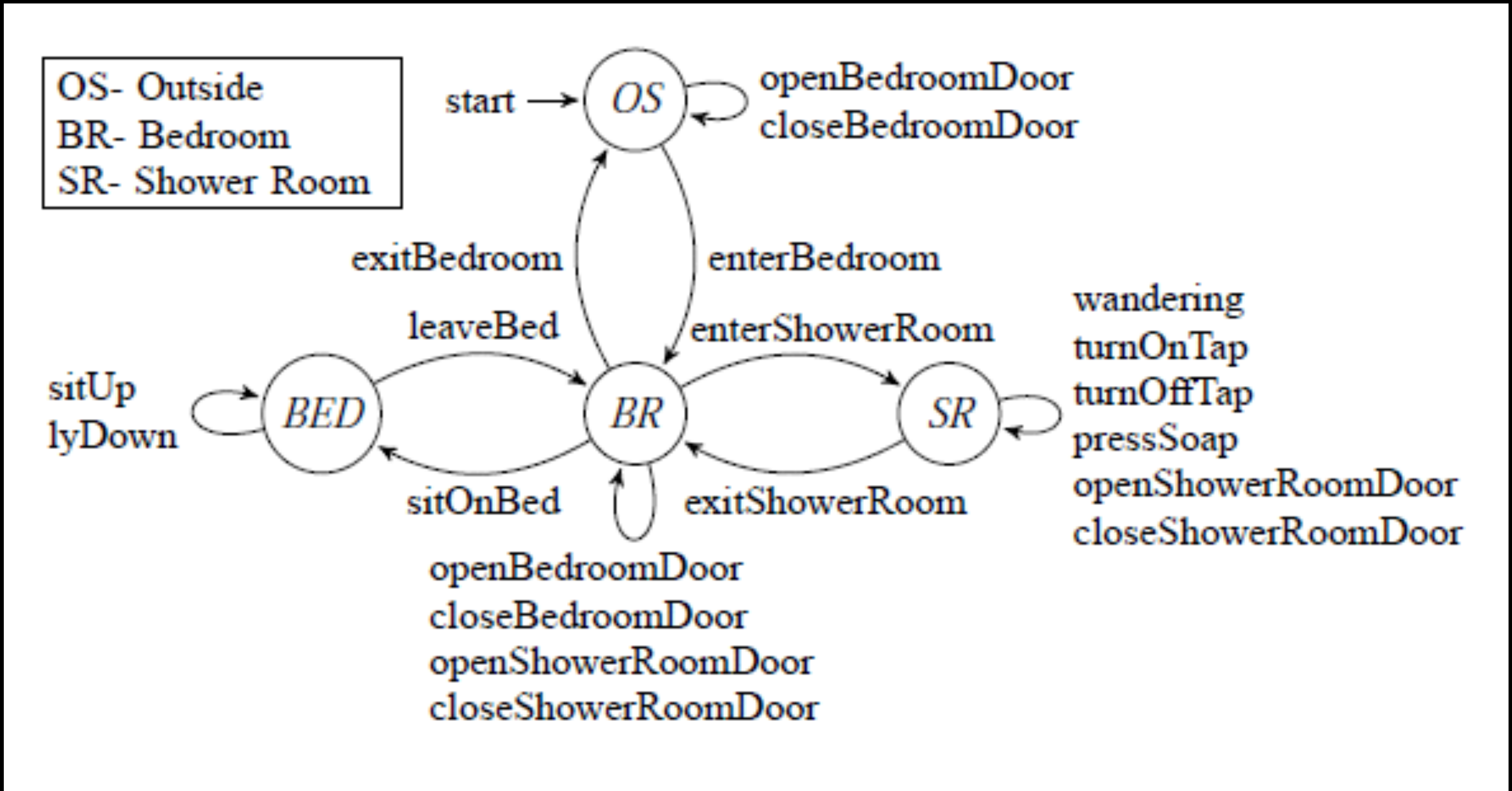
- Counter-examples useful for bug tracking
    - E.g. with rule conflict

Formal analysis approach



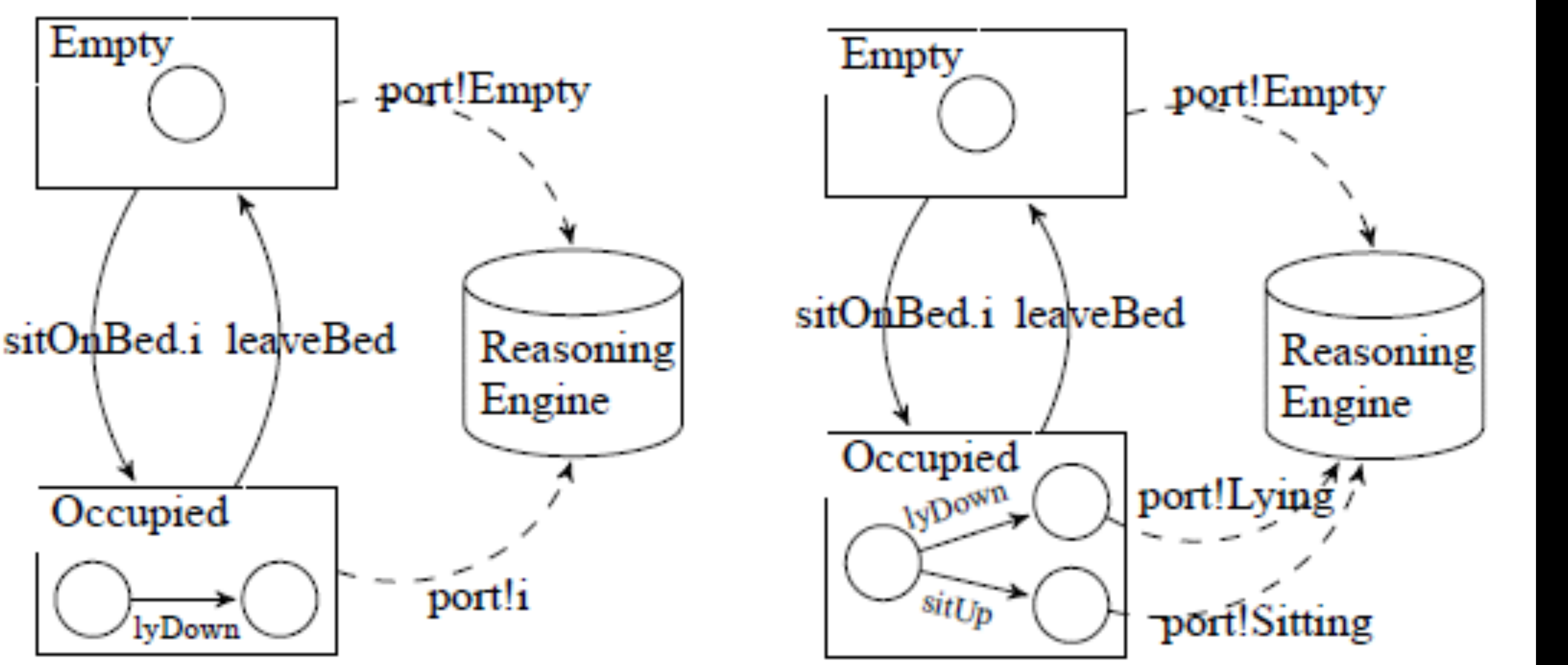
Labelled transition systems

– Modelling of people's behaviours



Labelled transition systems

- Modelling of system's behaviours



(a) Bed RFID Reader (b) Bed Pressure Sensor

Properties specification

- P1: Deadlock freeness
- P2: Guaranteed reminders

```
#assert SmartBedroom() |= [](UsingWrongBed -> <>RemindedWrongBed);
```

- P3: System inconsistency

```
#define InConsistency (Pos_Person[1] == SHOWROOM
    && ShowerFlag && sensors[PIR] == SILENT);
```

- P4: Conflicted or erroneous reminders

```
# define ConflictReminder (ReminderStage[ShowerNoSoap * 2] != 0
    && ReminderStage[WanderingInSR * 2] != 0);
```

Verification result

- Run on Intel Xeon CPU E5506 2.13GHz, 32GB RAM
- Total runtime around 3.5hours

Bugs?	Property	Result	Nb of states	Nb of transitions	Execution time (s)
-	P1.1	-	-	-	OOM
☺	P1.2	True	1,433,654	2,038,064	815
☺	P1.3	True	10,783,353	15,832,370	7,045
☺	P2.1	True	1,599,797	2,430,351	1,945
☹	P2.2	False	68,178	130,734	39
☹	P2.3	False	2,192,251	4,531,005	12,414
☹	P2.4	False	832,144	1,663,779	729
☹	P2.5	False	4,314	5,150	1.6
☺	P2.6	True	1,579,579	2,377,381	1,913
☹	P3	True	572	745	0.3
☹	P4.1	True	14,675	20,615	6.1
☹	P4.2	True	2,446	3,036	1.11
☹	P4.3	True	2,332,744	3,001,756	1,047

Verification result

- System split in two for memory purpose

System part	# Rules	# Non-reachable	# Redundant/ #Duplicated	# Conflict
Bedroom	17	2	8/3	2
Shower Room	22	5	16/2	-
Avg. Time (s)	-	2.05	3.05	-

Bug report with error trace

- System inconsistency
    - enterShowerRoom.1 -> turnOnTap -> exitShowerRoom.1 -> port.PIRShowerRoom.Silent

- False alarm
    - enterBedroom.2 -> sitOnBed.2.1 -> promptReminder.1

Outline

I. Introduction

II. Context reasoning

  - Reasoning strategies
  - Reasoning architecture

III. Validation & Verification

IV. Conclusion

Conclusion

- Contribution of model checking to AAL
  - Formal verification and certification of system
  - Useful bug tracking tool at design level
  - Save time on costly deployments
  - Avoid patient disturbance of early testing

- An excellent verification tool during system design
- To be used in conjunction with real deployments