RDF Recipes for Context-Aware Interoperability in Pervasive Systems

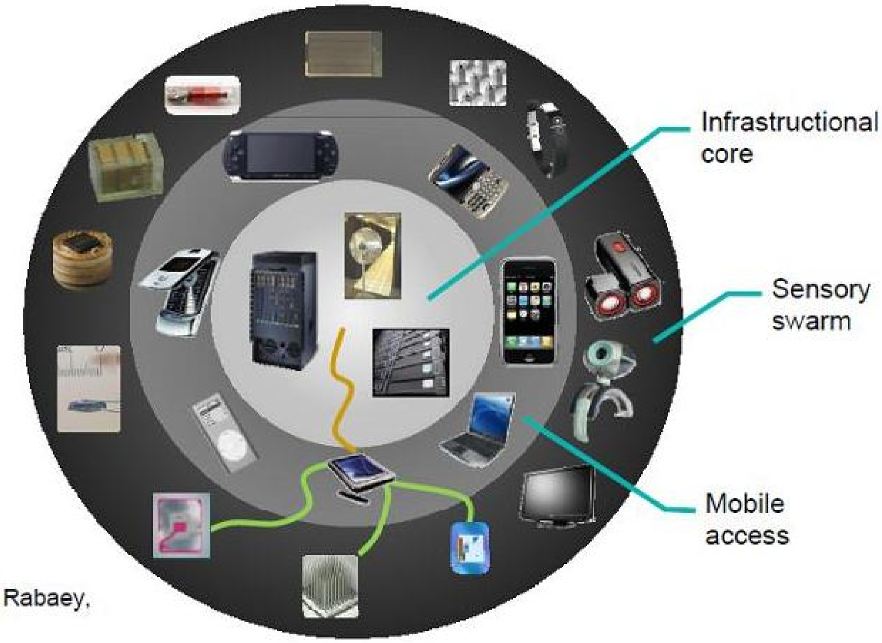
Anna Kosek, NXP
Aly Syed, NXP
Jon Kerridge, Edinburgh Napier Univesity

Pervasive computing/ambient intelligence, ….

Sensors and actuators everywhere

- Everyday objects will have sensing, processing and wireless networking capabilities
- Physical world will start reacting to sensory data aided by computers and actuators

Swarm of Sensors & Actuators



Courtesy Jan M. Rabaey, UC Berkeley

A Comparison

	Self contained functionality	Ad hoc net-working	Computing know-how of manufacturers	Low energy usage	# of device makers in the world	Installation of devices by
Infra-structure devices	Yes	No	Very high	Important	Very limited	Experts
Mobile access devices	Yes	Maybe yes	Usually high	Important	Limited	Users with assistance
Sensory swarm devices	No, Usually Distributed	Always	Can be low to very low	Very important	Very large	Usually non experts

Sensory swarm requires very distributed functionality embedded in a large number of devices made by a large number of manufacturers who often are not computing experts, devices are installed by non experts

Challenge 1:

- How to achieve useful application in a sensory swarm
- How to support:
  - Large variety of devices
  - Large number of devices

Ontology and Knowledge Representation

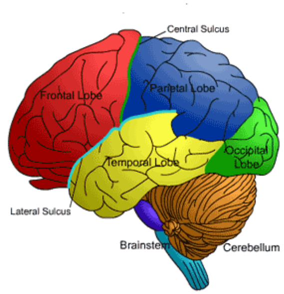
- AI systems uses knowledge of component's behavior and environment influence on it to achieve intelligent behavior
  - Most knowledgebases are designed from the scratch, even when building a similar system
- Need for a flexible standard for knowledge representation, that describes concepts and relations between concepts → ontology

How to make such devices interoperable

Premise:

Electronic devices have become so capable that they can also reason with semantic information drawn from ontology

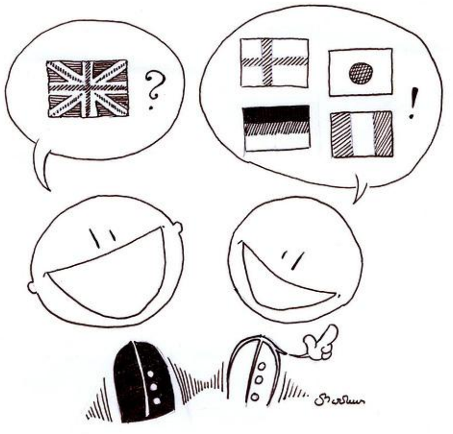
- Our solution: Achieve interoperability giving devices knowledge and reasoning capabilities



- But how much knowledge does a device need?
  - A device gets knowledge about itself, what functions it can perform
  - A device gets knowledge about its surroundings, what other functions it needs to perform an application

Semantic Interoperability

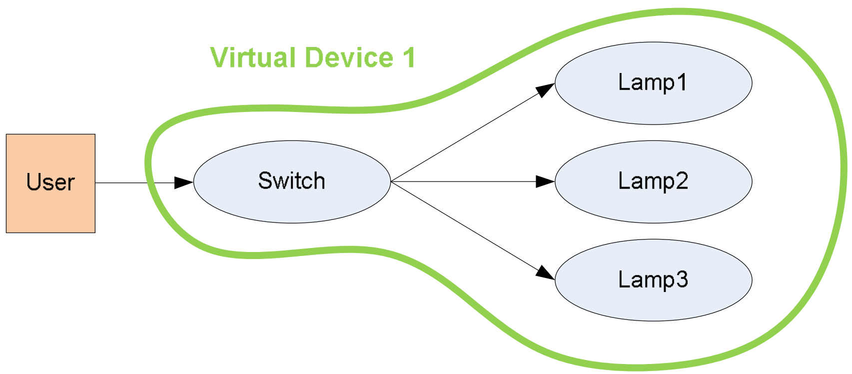
- Proper interpretation of information transmitted between two independent components



- Can be achieved using ontology (following the W3C approach)
- Necessary to share understanding of concepts and relations between concepts

Architecture

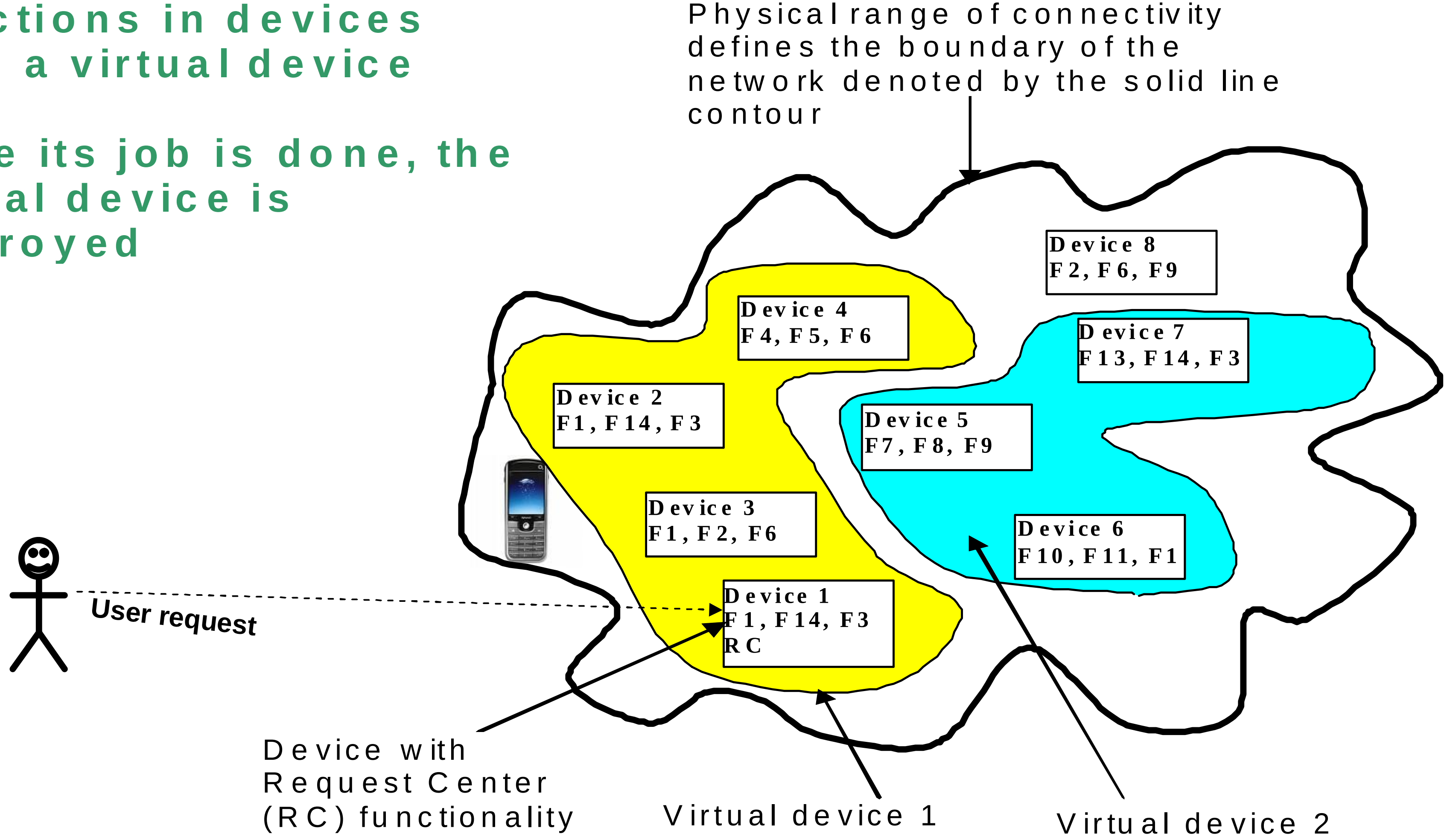
- The presented architecture is designed to control a smart space with many small, energy frugal devices
- No central control is present, functions and services distributed, ad-hoc network organization, communication over simple broadcast-based protocol
- Devices perform tasks by organizing into sub-networks and cooperating to deliver a required action



Interoperability architecture

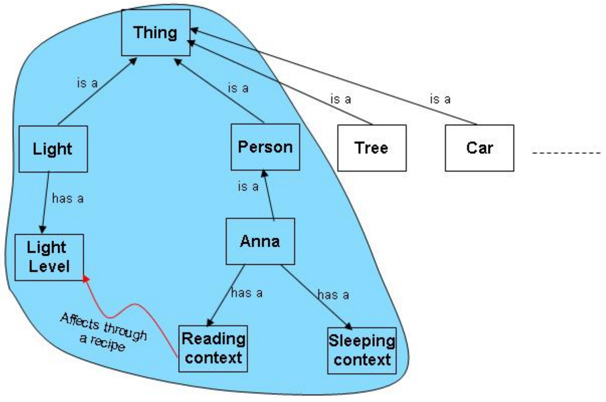
Functions in devices form a virtual device

Once its job is done, the virtual device is destroyed



Device architecture

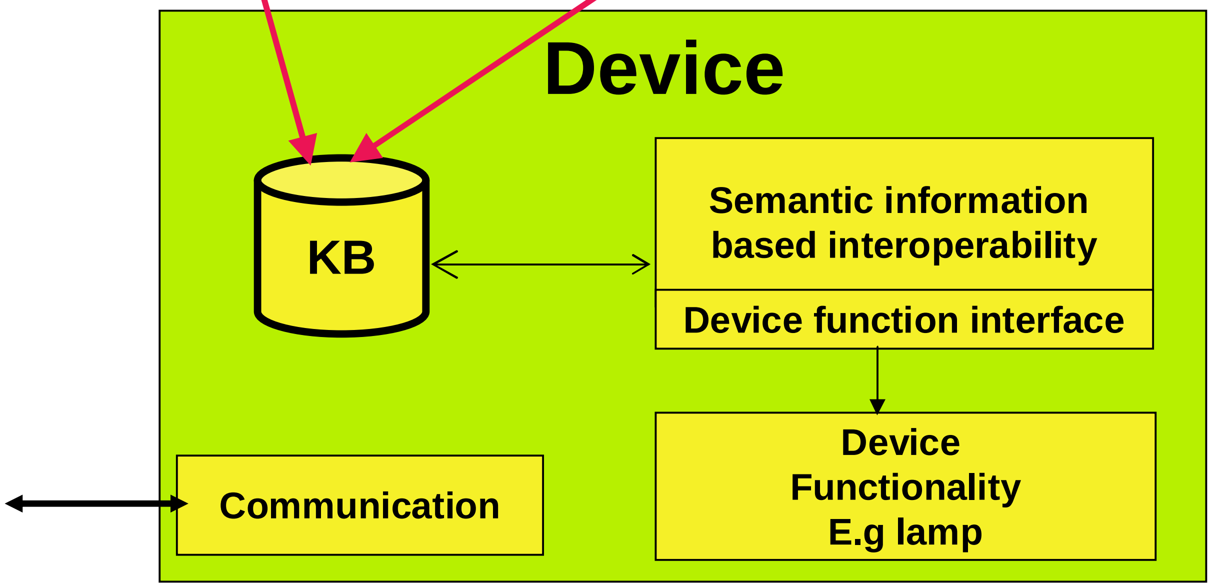
Partial Ontology



Recipes that tell a device what to do in some situation

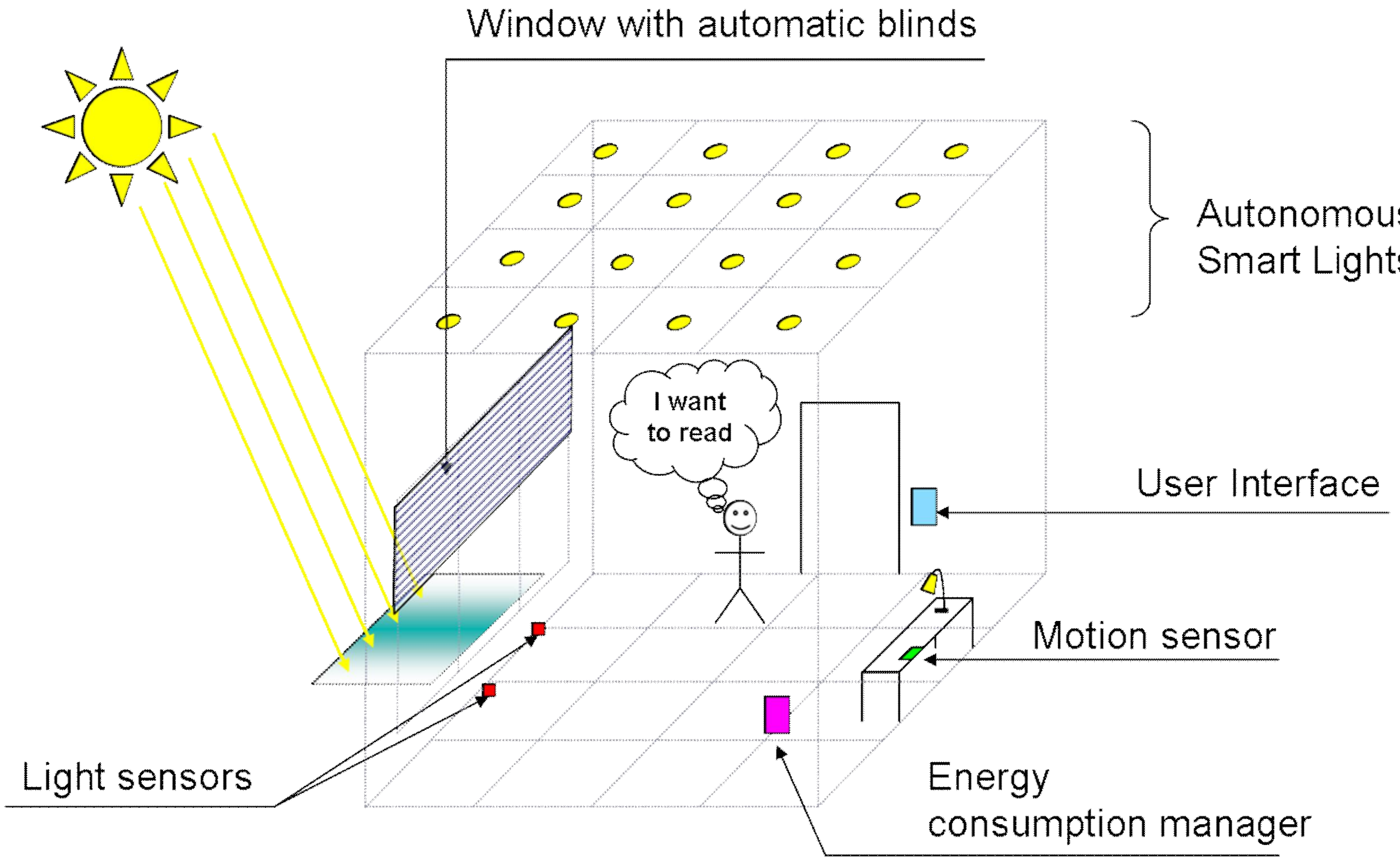
Recipe a: For context Anna reading, turn light level to 500 Lux
Recipe b: For context Anna reading, turn light level to 0 Lux
Recipe c: ……..

Devices communicate using a set of defined messages



n1

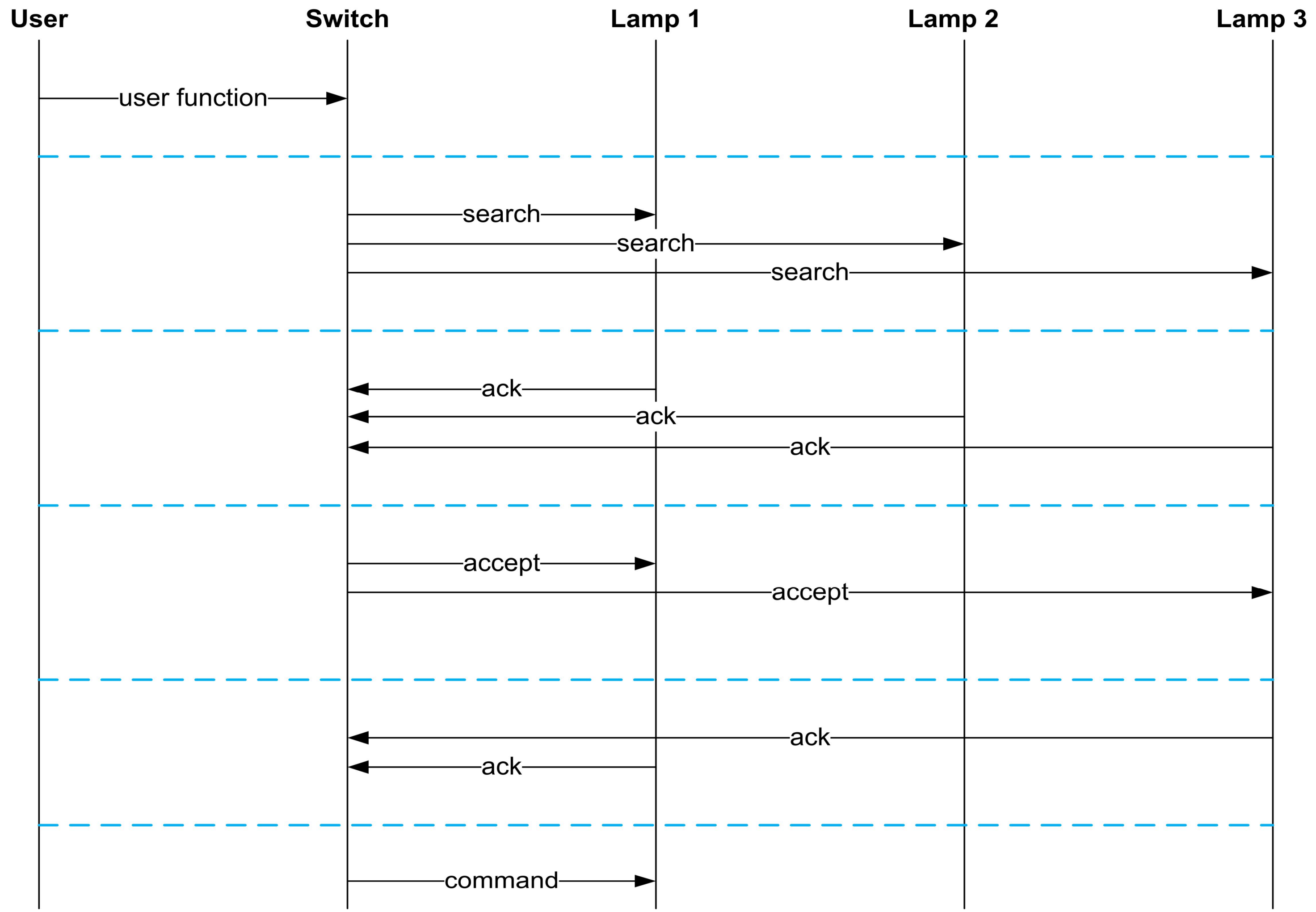
Scenario



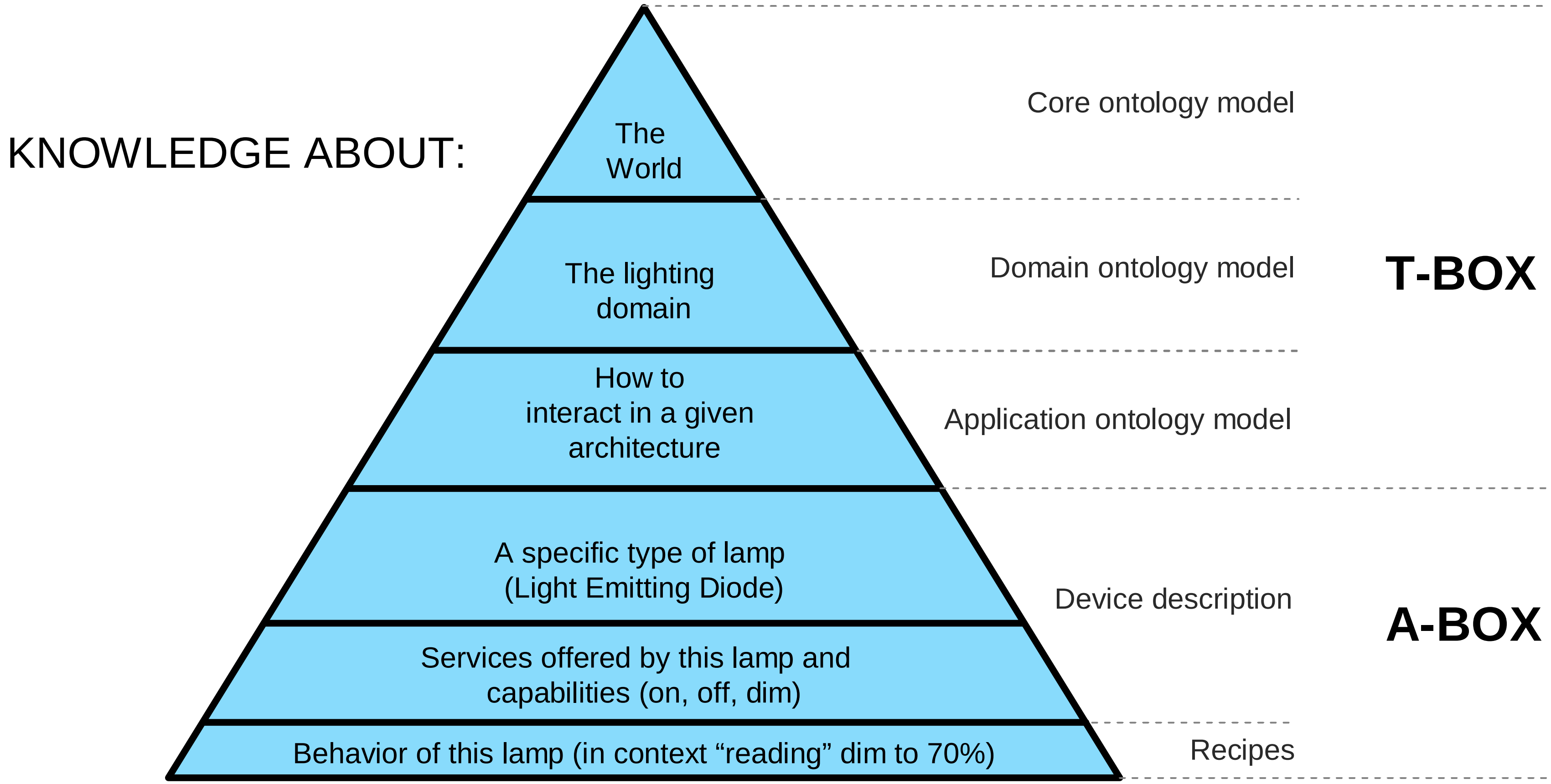
Window with automatic blinds
Autonomous Smart Lights
I want to read
User Interface
Motion sensor
Light sensors
Energy consumption manager

n1 it is not clear that this simulation has benn performed.
nlv10662; 18.6.2010

Discovery and self-organization protocol



Knowledge required for a lamp



Edinburgh Napier UNIVERSITY

Knowledgebase content

- Core, domain and application ontology models
- Device description
- Device capabilities
- Context and users
- Configuration and state
- Recipes

Knowledgebase structure

- KB format is fixed
- Entries are represented in RDF (Resource Description Language) triples:

  (subject, predicate, object)
- Example: To express information about Anna:

  (Anna, is-a, person)

  (Anna, hasAge,27)

  (Anna, hasAddress, Address1)

  (Address1, hasStreetName, Leenderweg)

  (Address1, hasHouseNumber, 116)

  …
- Order is not important

Recipes

Recipe:

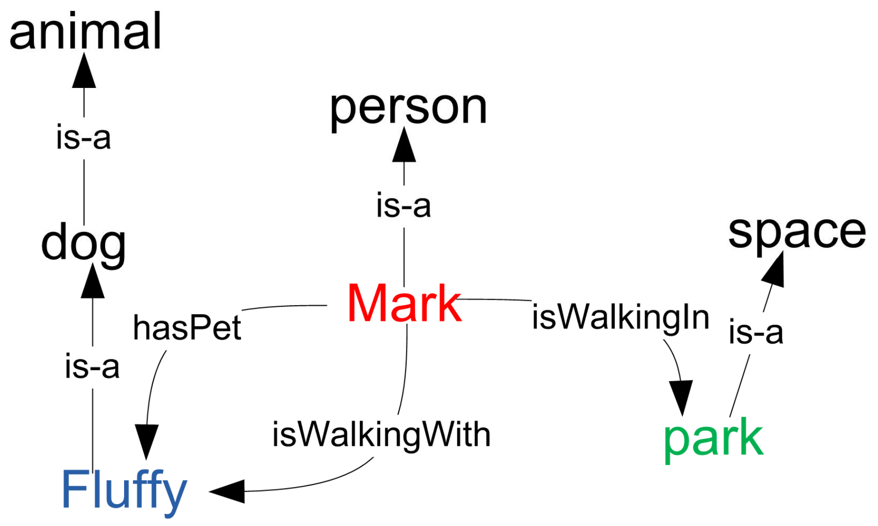
- Header
- Step 1
- Step 2
- Step 3
- …
- Step 4

- Recipe consist of header and a sequence of steps to perform
- Header consist of recipe type, service to be performed, context and other conditions guarding access to a recipe
  - E.g. Perform RecipeX when Anna is reading → context is Anna-reading
- Steps are designed to perform actions and use commands that the device reacts on
  - E.g. Step X   Turn on the light
    Step X+1   Dim to level Y

Describing a process with RDF

Mark is walking his dog Fluffy in the park.

Mark	is-a	person
Mark	hasPet	Fluffy
Dog	is-a	animal
Fluffy	is-a	Dog
Mark	isWalkingWith	Fluffy
Mark	isWalkingIn	park
Park	is-a	space



Recipe example

subject	predicate	object
recipe1	is-a	recipe
recipe1	hasContext	Reading
recipe1	hasPerson	Anna
recipe1	hasRecipeType	configureVD
recipe1	hasStart	step1
step1	is-a	step
step1	usesCapability	startConfiguration
step1	hasNext	step2
step2	is-a	step
step2	usesCapability	output
message1	is-a	message
step2	hasElement	message1
message1	hasMessageType	2
message1	hasAckResponse	0
step2	hasNext	step3
step3	is-a	step
step3	usesCapability	deviceFunctionality
step3	hasCommand	turnOnOff
step3	hasValue	1
step3	hasNext	step4
step4	is-a	step
step4	usesCapability	deviceFunctionality
step4	hasCommand	dim
step4	hasValue	70
step4	hasNext	null
null	is-a	step

Recipe's **harder**. This recipe is associated with context reading and person Anna.

Step 1 saves all data needed for a particular configuration, therefore creates entries in local memory about the virtual device that the lamp is about to join.

Step 2 uses capability output to send an acknowledgment to a device that requested joining a new virtual device, in this case this message is sent to light switch.

Step 3 sends command to turn on the light.

In **step 4** lamp dims to 70 percent. A step called null indicates end of recipe. If a device reach an end of a recipe, a task is finished.

Using *if than else* statements

- Recipe is designed to be a sequence of steps to perform, the choice over performing actions can be done in two ways:
  - Guarding the recipe with different entries in header (separating different behavior for different contexts, people, services)
  - Skipping steps using simple evaluating action
    - If(condition) go to step A
      else go to step B

Conclusions

- Use ontology and RDF to represent knowledge in pervasive system built of small, energy frugal devices.
- A simulation was developed to present functional behavior of a distributed system using RDF based recipes.

NXP Edinburgh Napier UNIVERSITY