MIDLAB Middleware Laboratory

Università di Roma "La Sapienza"
Dipartimento di Ingegneria Informatica,
Automatica e Gestionale

Introduction to home automation systems

Adriano Cerocchi

cerocchi@dis.uniroma1.it

www.dis.uniroma1.it/~cerocchi

Sapienza University of Rome

- Seminar Schedule:
  - Lecture 1 - concurrency in home automation systems
  - Lecture 2 - Introduction to home automation system:
    - what is an home automation system
    - challenge and future direction
    - an example of home automation system: SM4All
    - layers of an home automation system
    - the actuation layer
    - the context awareness
    - the communication layer
      - I2C, CAN, Ethernet, KNX, EDS
  - Lecture 3a - Control and Interfaces
  - Lecture 3b - A real case-study

- What is an home automation system

- for home automation system we mean each technology voted to the automation of the house
- in this sense...
  - ...the garden irrigation clock...
  - ...the thermostat...
  - ...the electrical shutters...
- ...are all example of easy home automation

- **question: are all of the houses equipped with home automation system?**

- What is an home automation system

- **answer: no**
- in general for ***home automation system*** we mean and environment in which the actuation and the control are decoupled ***at least*** by a firmware layer

- a classic electric circuit has a wired (***hardware***) connection between control and actuation, then it ***cannot*** be classified as "home automation"

- in the following we will use the term "home automation system" or "domotic" in compliance to the definition above

- Challenge and future direction

- Yesterday:
  - interfaces: buttons and switches
  - integrated systems: ---
- Today:
  - interfaces: buttons, switches, touch panel (wired and wireless)
  - integrated systems: audio/video (sometimes), thermal control, windows and shutters, ad-hoc interconnection
- Tomorrow (challenges):
  - interfaces: reduce as possible the needs of buttons, switches and touch interfaces, the house must predict what the user want
  - integrated systems: potentially everything

- Challenge and future direction

- research interests:
  - location component
    - follow-me function (open issue...)
  - find the right trade-off between the buttons and the touch interfaces
  - open source solution for house interconnection
  - efficient energy-sharing and energy-management
  - space management for office-oriented house

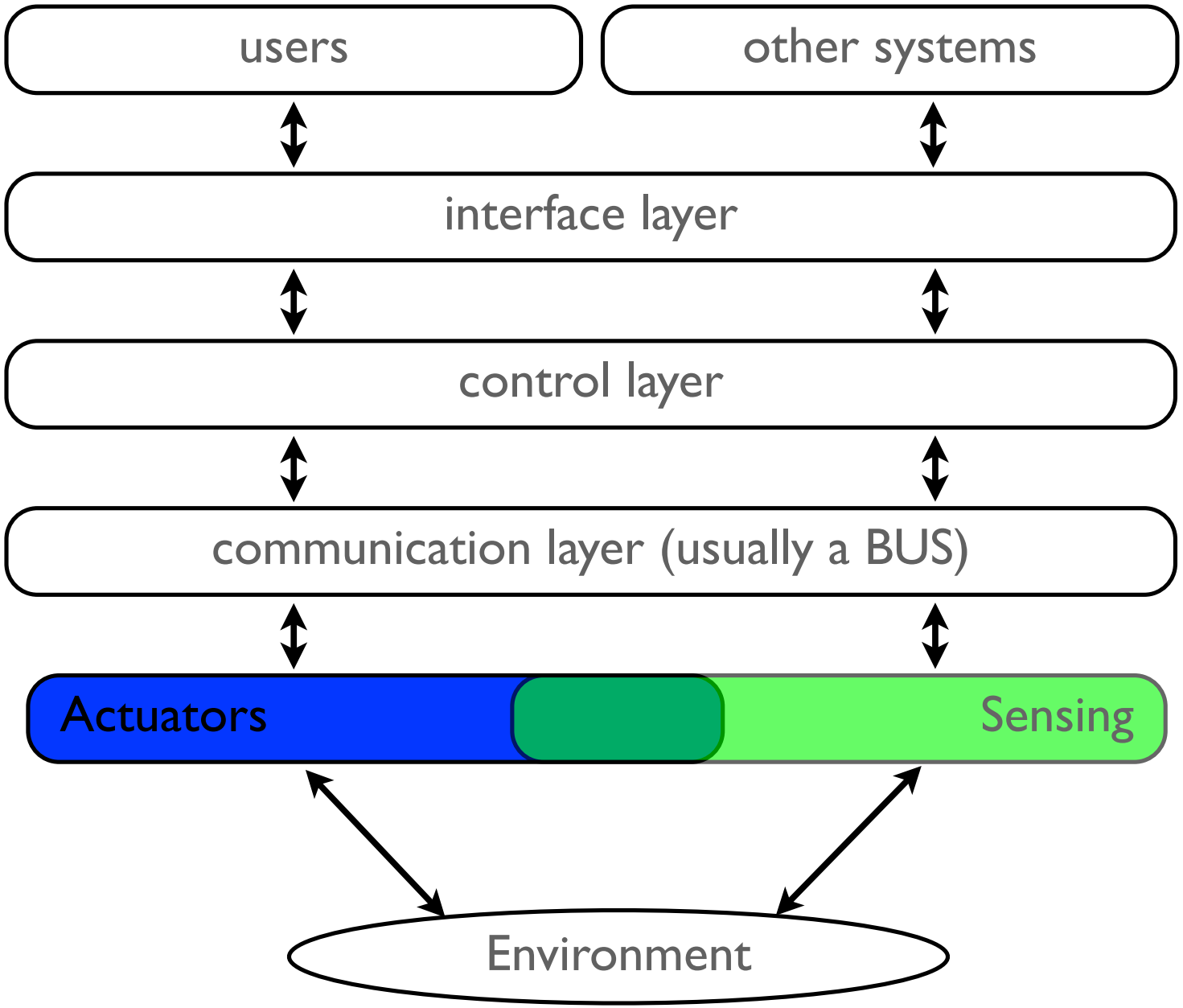
- Drivers:
  - the users are scared about the technology: they don't want to be controlled by a computer
  - the users don't want "something more", they prefer "something known" with some "intelligence"

- Challenge and future direction

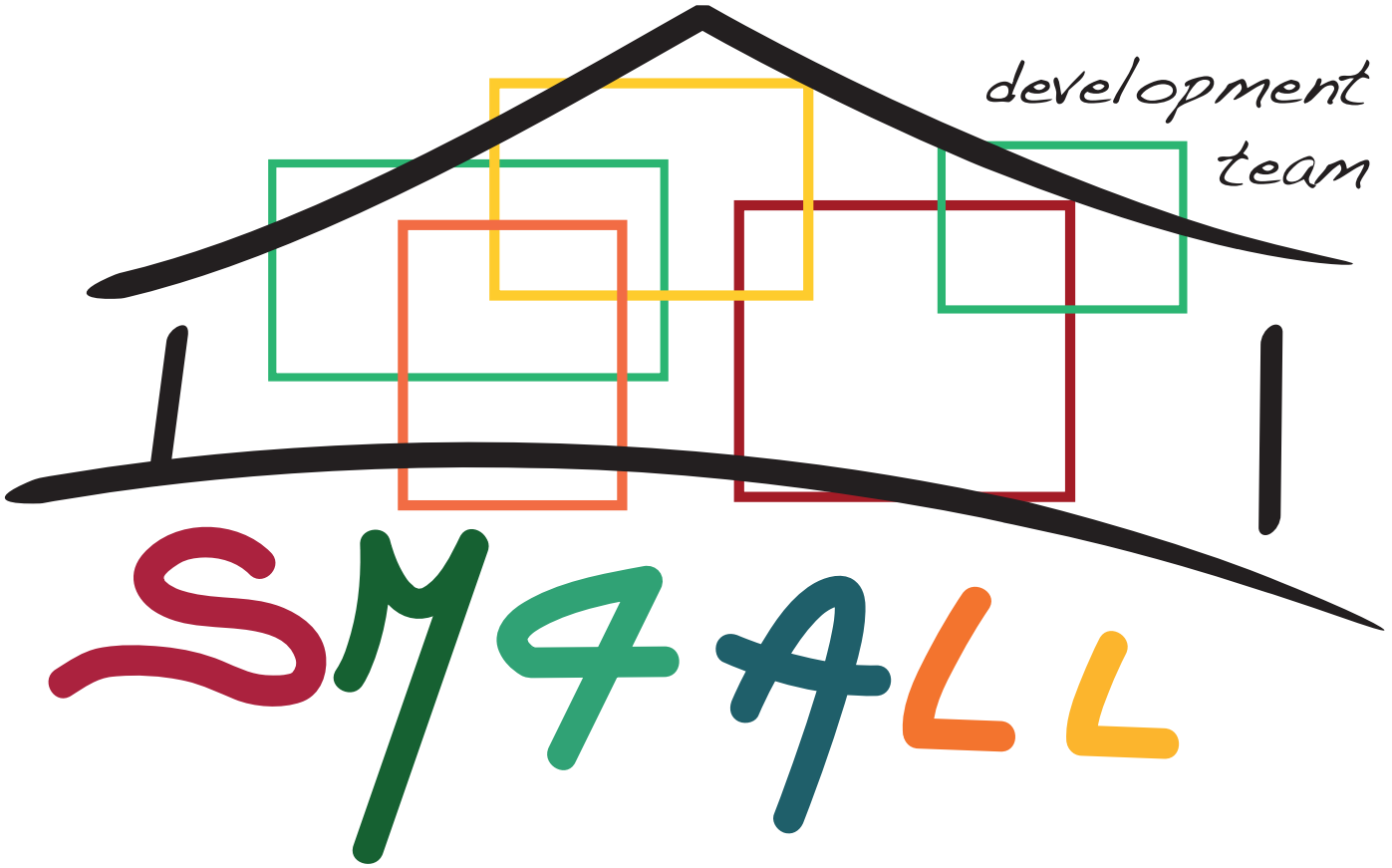
- Nice fast test: which is your favourite functionality?
  - a. open/close with a single button all the shutters
  - b. switch-off all the lights
  - c. control the house status from outside

- The 70% of the users answered "b"...so: the more the functionalities are simple and the more the users will appreciate them

- Layers of an home automation system



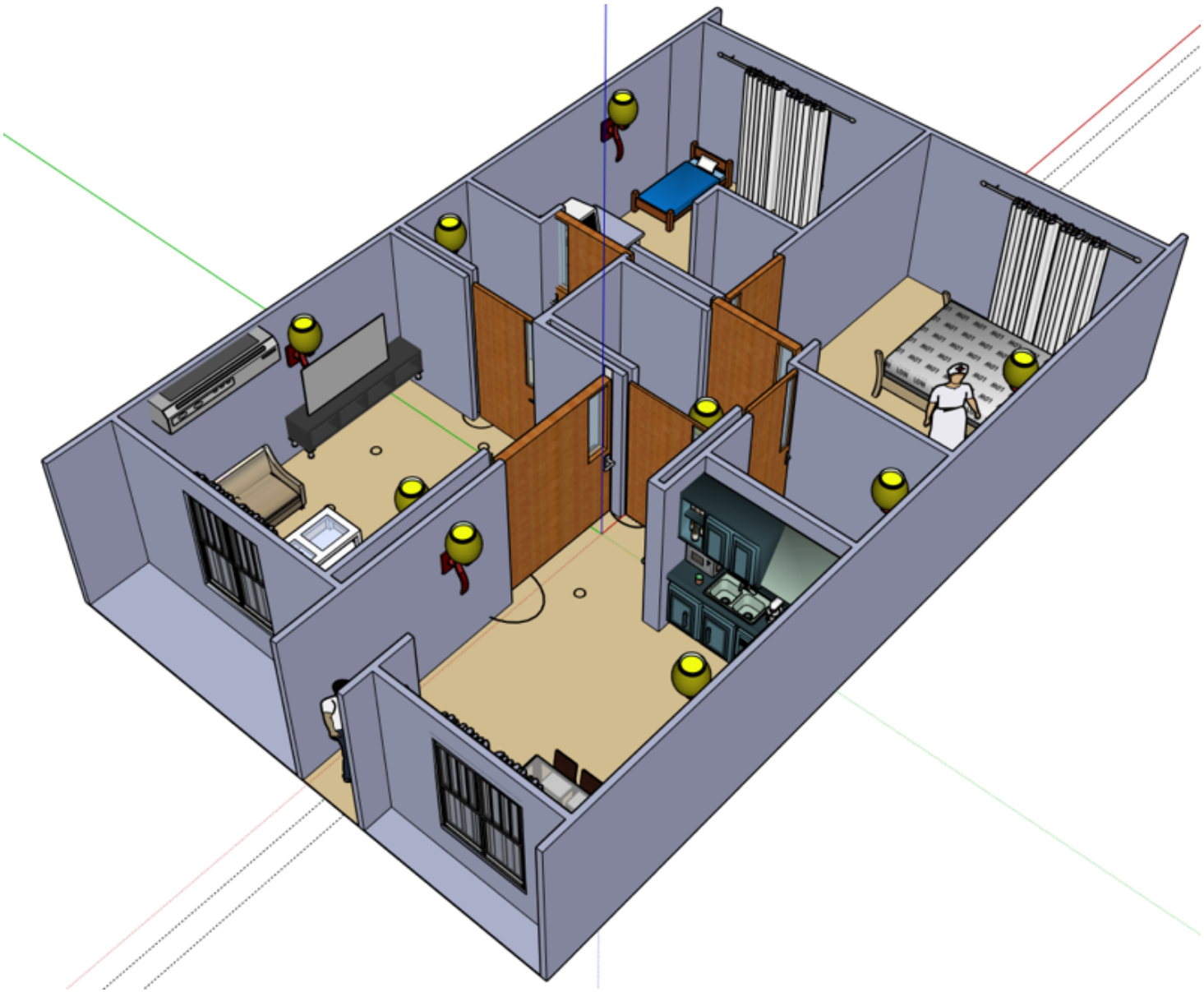
- an example of an home automation system: SM4All



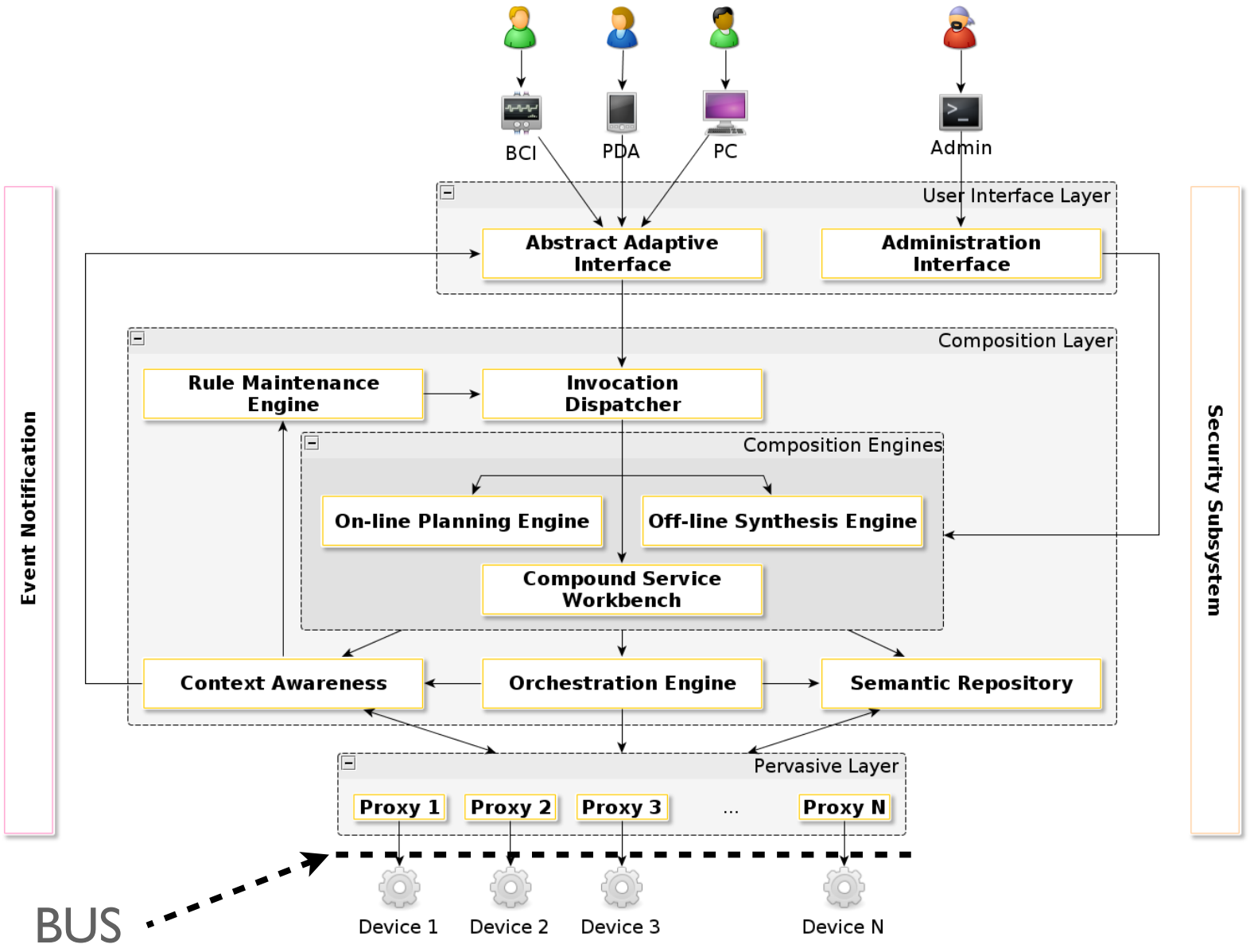
- an example of an home automation system: SM4All

- an example of an home automation system: SM4All

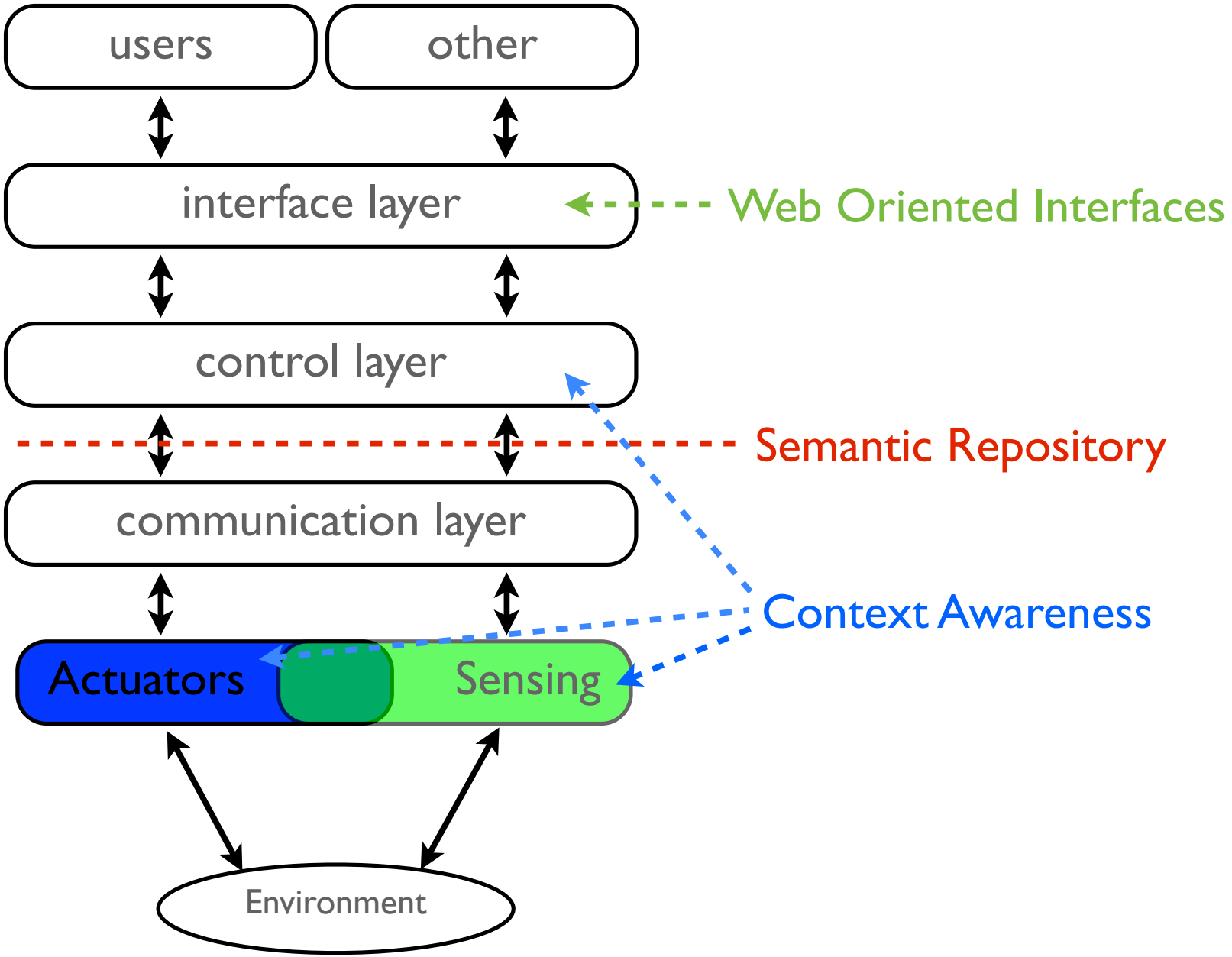
- an example of an home automation system: SM4All



- an example of an home automation system: SM4All



- fundamental concepts of SM4All



- commercial brands for home automation

KNX brands

- commercial brands for home automation

Stand-Alone brands

- The actuation layer

- an actuator is a device that is **able to modify the environment**

- some examples:
  - a bulb light connected to a switch is an actuator
  - an automatic door is an actuator
  - an electrical shutter connected to a switch is an actuator

- question: **a relay is an actuator ?**

- The actuation layer

- answer: potentially yes...
  - if the relay is connected to a device (ex. a bulb light) it can be intended as an actuator
  - if the relay is not connected to a device...it is not an actuator

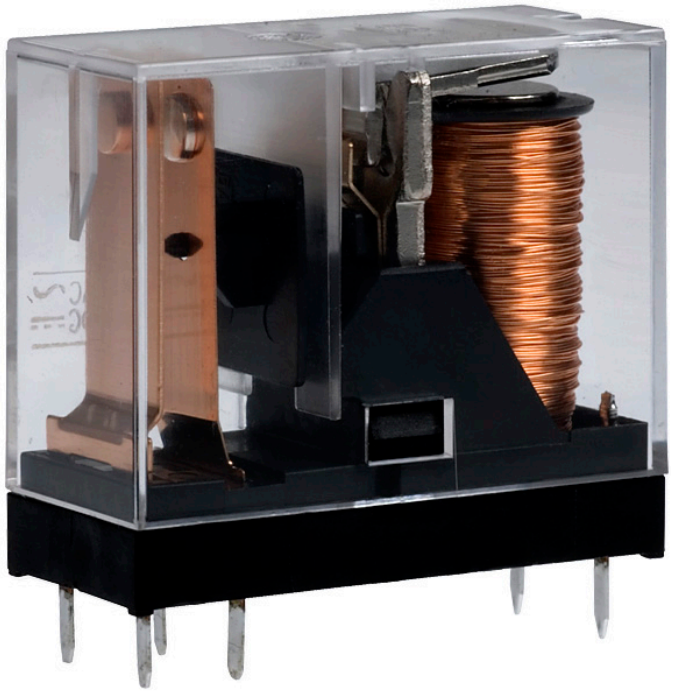
- in general when we talk about relay we intend them connected to devices

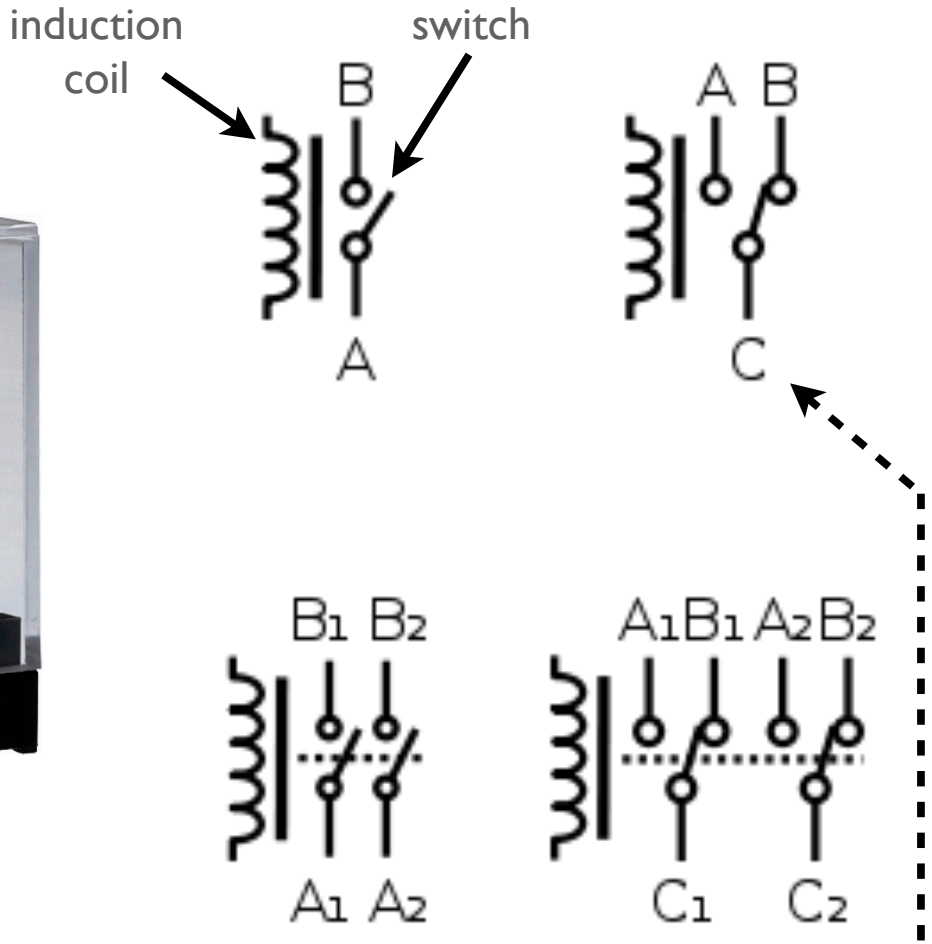
- The actuation layer: devices

- from a low-level point of view an actuator always contains:
  - one or more relays
  - one or more dimmers
  - a serial port

- relays, dimmers and serial ports represents all of the devices that can be controlled by an home automation system in order to implement actuators

The actuation layer: relays





relays usually work with a normally-opened/ normally-closed logic

- The actuation layer: relays

- a relay is required in order to switch on/off an electrical line
  - relays are useful because they use a low-voltage controller (tipically 12V) to interrupt an high voltage one (220V or higher)

- relays are the most common device for automation due to several reasons:
  - they are inexpensive
  - they are reliable (large MTTF, the first patent of a relay is of **1840**, they born with the telegraph)
  - they can have a very reduced size
  - the energy consumption of each relay is practically zero

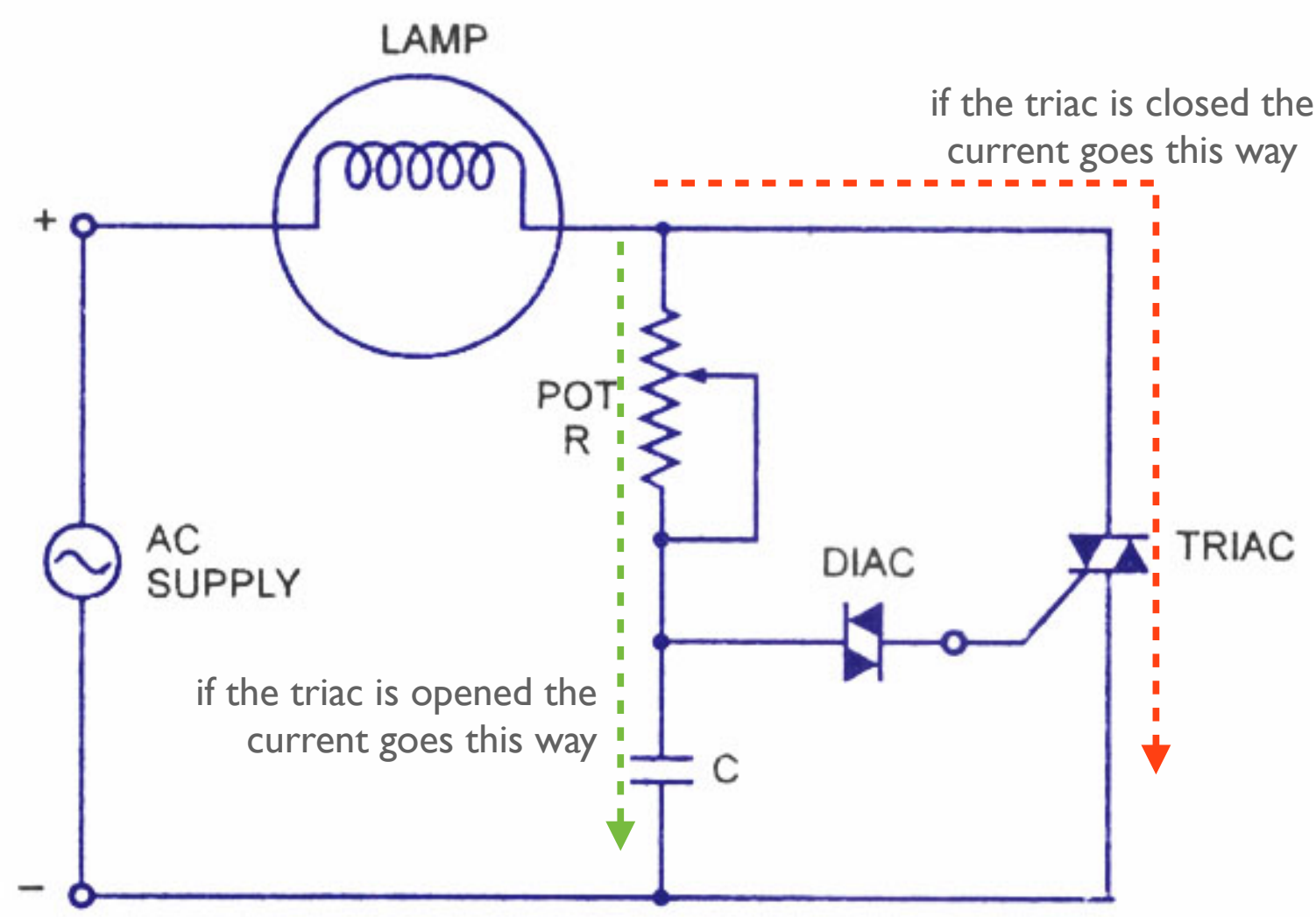
- The actuation layer: dimmers

- a dimmer is an electric regulator

- it is invented in the 1961 by Joel Spira that is the head of the Lutron Company

- a dimmer can be implemented in different ways:
  - resistance-based dimmer
  - current-transformer-based (variac)

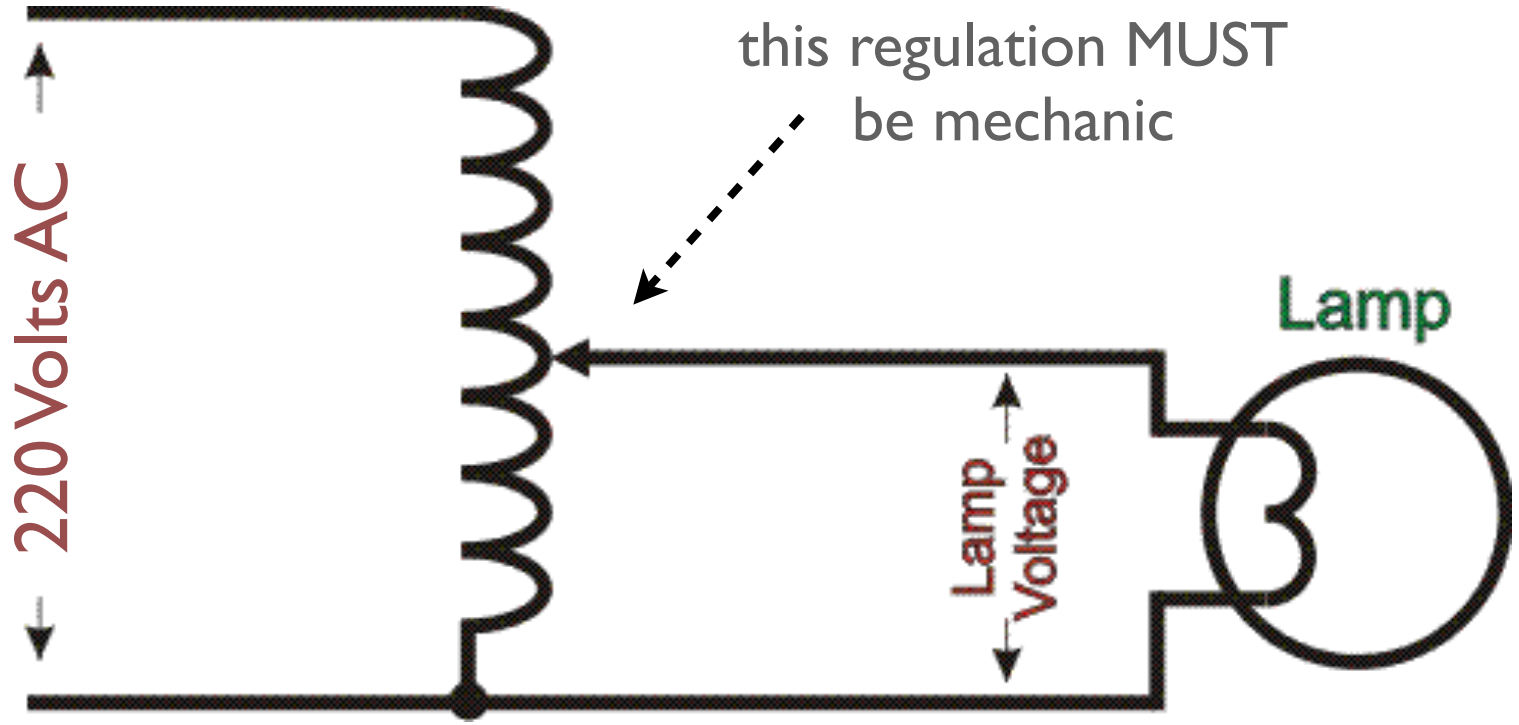
- The actuation layer: a very easy triac-based dimmer



Triac Lamp Dimmer Circuit

- The actuation layer: a very easy triac-based dimmer

- The actuation layer: a very easy variac-based dimmer



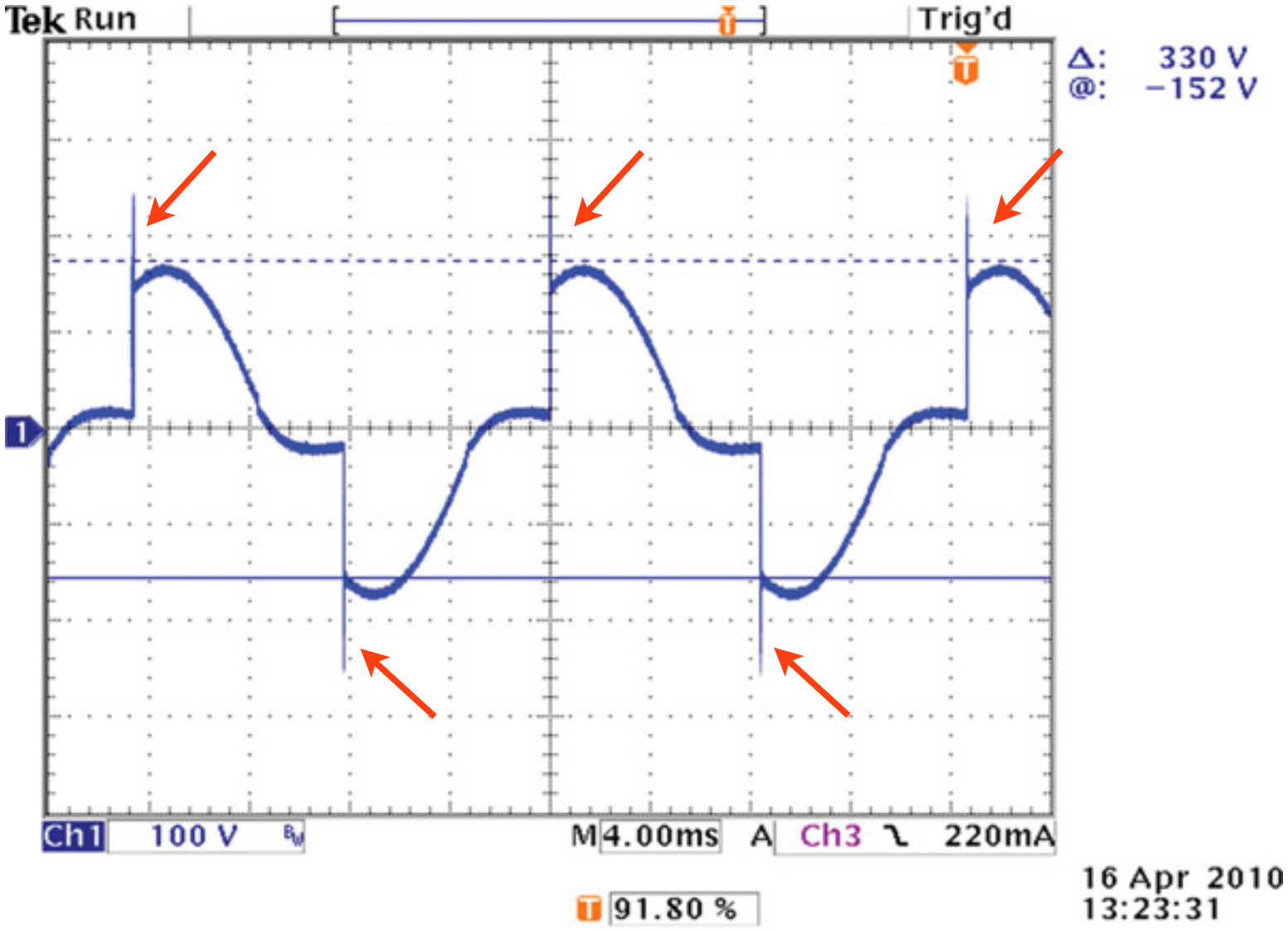
- The actuation layer: a very easy variac-based dimmer

- The actuation layer: dimmer issues

- **variac-based dimmers** offer best performance but unfortunately they cannot be used for home automation due to:
  - the large size (and weight...)
  - the mechanic regulation

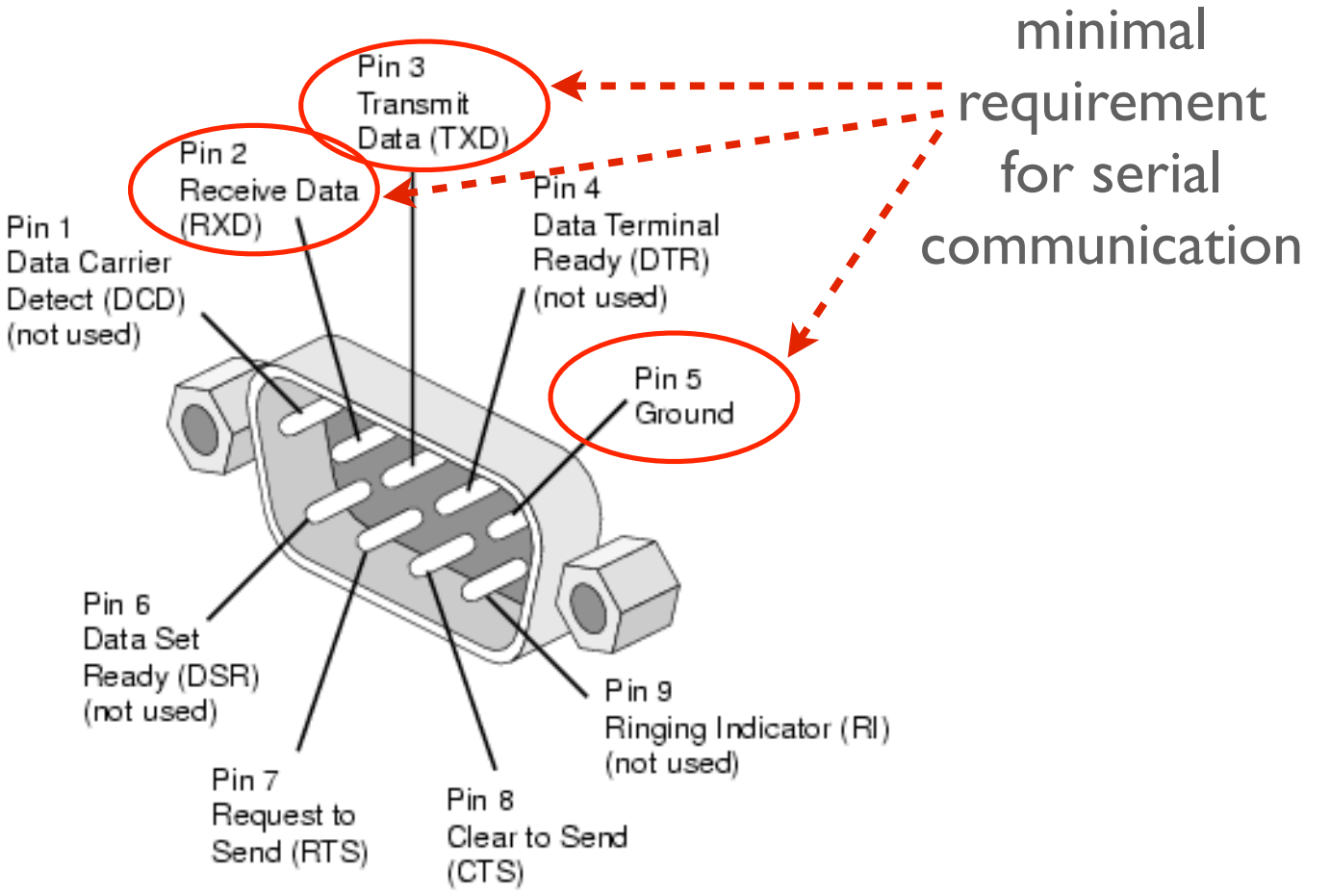
- **triac-based dimmers** can be used for home automation but...
  - they introduce electrical noise (see next slide) - LC filtering circuits are required, they are usually larger part of the dimmer
  - they introduce buzzing due to LC filters
  - they are usually designed for specific load (if the user change it they can have faults)
  - they can introduce problems with the electricity meters (because they introduce some peaks during the triac commutation) - **this is so rare**

- The actuation layer: dimmer electrical noise



- The actuation layer: serial port

  - is a standard
  - widely supported by programming language



- The context awareness

- the context awareness is a fundamental concept of home automation: the more the system is aware of the environment and the more it can be predictive
- the "context awareness" can be build using:
  - sensors
  - action history (that's why actuation and sensing layers are overlapped)
- sensors can measure a large number of physical entity (temperature, humidity, light level, ...)
- action history can provide some hints:
  - ex 1. if the light actuator is switched on and we know that it is located in room X it is possible to state "in the room X there is the light" without the need of a sensor
  - ex 2. if the "switch off all the lights" command is invoked from the bed room, probably the user is sleeping...

- The context awareness

  - a limited knowledge of the context can be obtain just looking at the history of the actuations
  - ...anyway the presence of the sensors enhance so much the capability of context awareness, an example: **deterministic fault detection with action-reaction graph**

failure detection and isolation using action-reaction graph

from theory to practice

definition of action-reaction graph

- An **action-reaction graph** [ARG] is a directed graph that represents the observable effects of each action within a complex system
- An ARG is ruled by a language
- A boolean ARG is ruled by boolean language

fault model

- we are interested in permanent **value-faults:** is a fault that causes a module to respond within the correct time interval but with a constant incorrect value
- ASSUMPTION [A1]: channel are perfect, then the fault can only be caused by the nodes

Faults in action-reaction graph

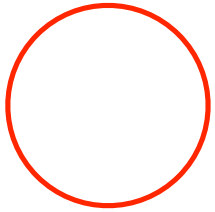
- In real systems it is possible to have faults, in case of faults it is possible to find effects that don't respect the language, we call this situation **inconsistency**
- each inconsistency implies always two nodes
- within a system can occurs different inconsistencies
- ASSUMPTION [A2]: even in case of fault it is possible to access to the internal state of each node
- ASSUMPTION [A3]: if exists **at least** a couple of nodes that respects the language **after a state change** then the two nodes are correctly working

the inconsistency

- An inconsistency is an implication between two nodes that doesn't belong to the language
- in order to solve the inconsistency between two nodes A and B an oracle is required. An oracle is a couple of node X-Y respecting the assumption [A3] such that exactly one between X and Y is A or B.
- using the oracle it is possible to discover if a node implied by the inconsistency is correctly working
- once the oracle is found, it is possible to infer the faulty node

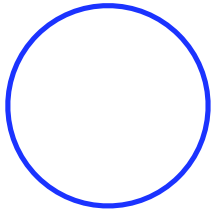
ARG example

Boss



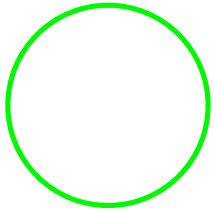
Boss language
- peaceful –> ?
- nervous –> nervous
- angry –> peaceful

Technical Manager



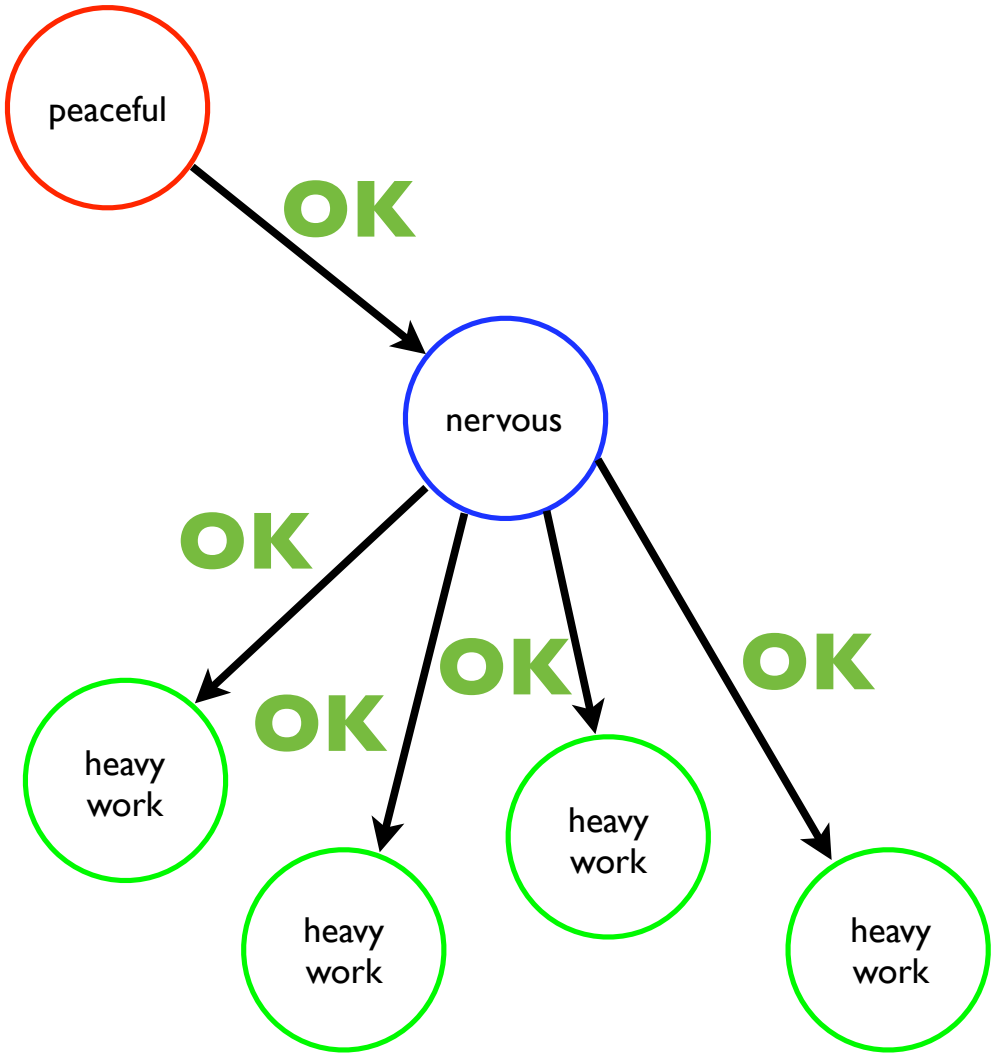
Technical Manager language
- peaceful –> ?
- nervous –> heavy work
- angry –> be quiet

Employee



Employee language
- hard work
- be quiet
- other

ARG example

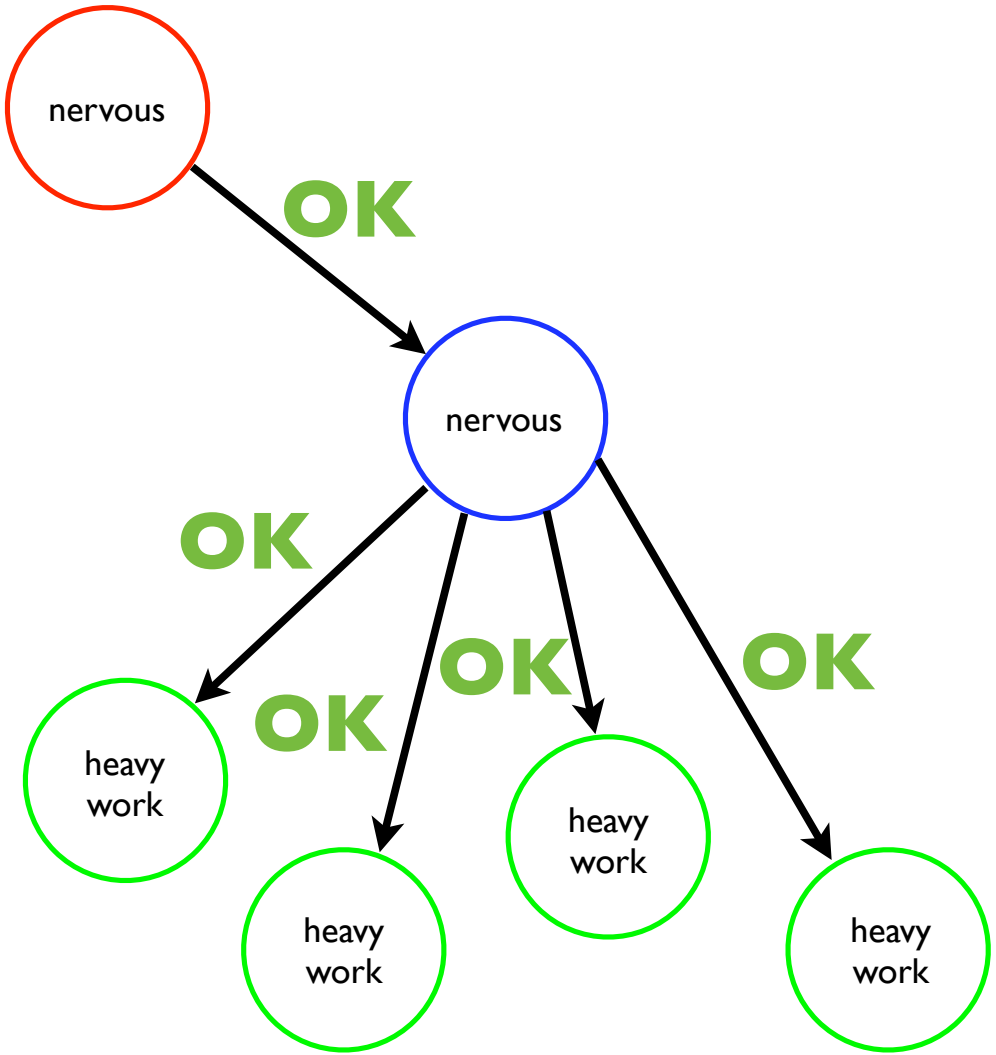


Boss language
- peaceful –> ?
- nervous –> nervous
- angry –> peaceful

Technical Manager language
- peaceful –> ?
- nervous –> heavy work
- angry –> be quiet

Employee language
- heavy work
- be quiet
- other

ARG example

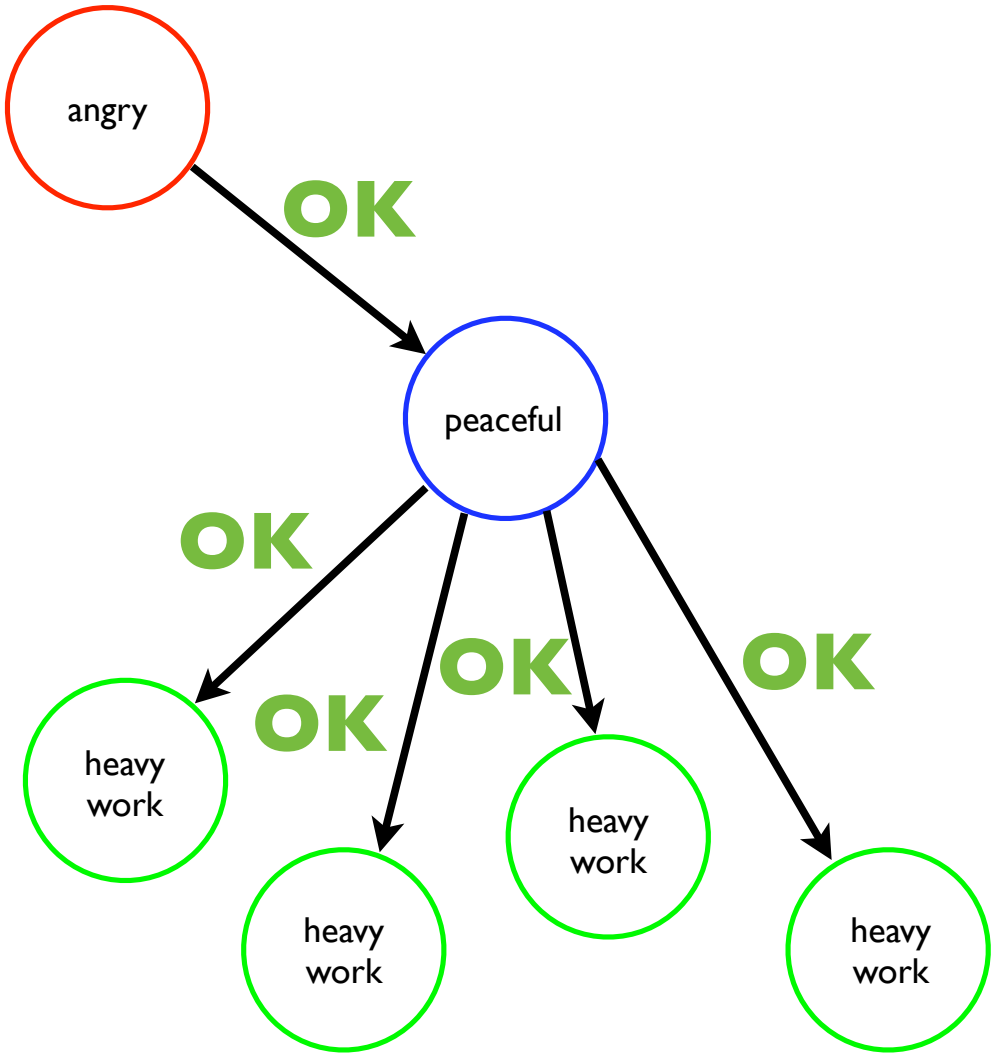


Boss language
- peaceful –> ?
- nervous –> nervous
- angry –> peaceful

Technical Manager language
- peaceful –> ?
- nervous –> heavy work
- angry –> be quiet

Employee language
- heavy work
- be quiet
- other

ARG example

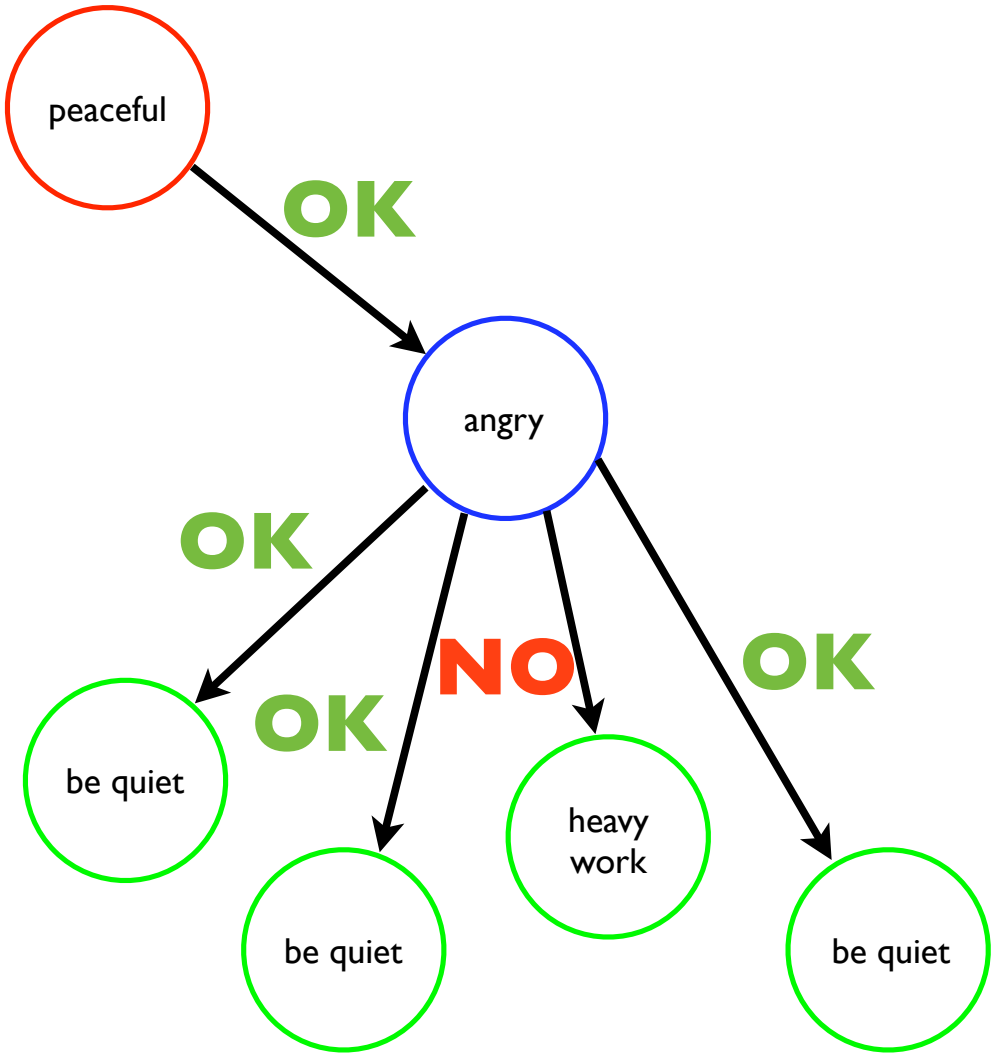


Boss language
- peaceful –> ?
- nervous –> nervous
- angry –> peaceful

Technical Manager language
- peaceful –> ?
- nervous –> heavy work
- angry –> be quiet

Employee language
- heavy work
- be quiet
- other

ARG example

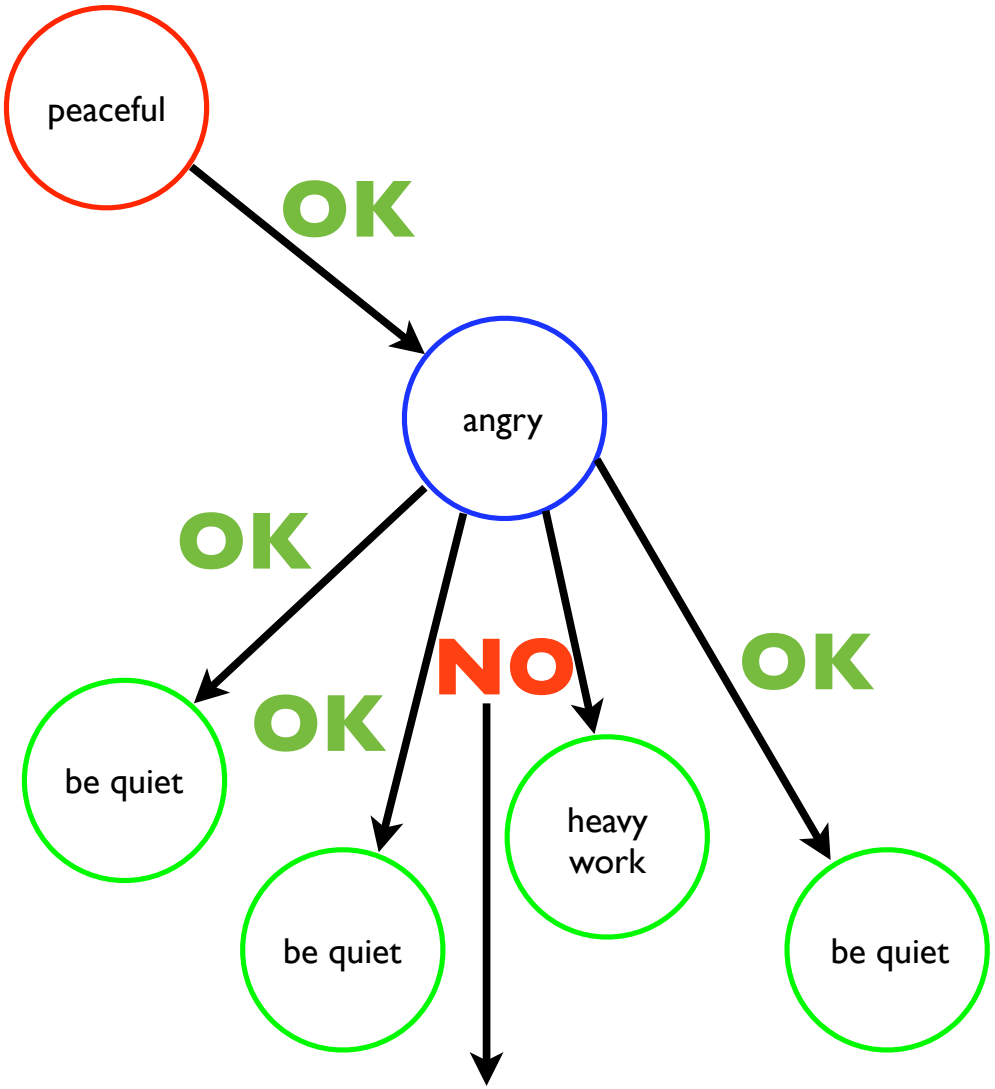


Boss language
- peaceful –> ?
- nervous –> nervous
- angry –> peaceful

Technical Manager language
- peaceful –> ?
- nervous –> heavy work
- angry –> be quiet

Employee language
- heavy work
- be quiet
- other

ARG example



Boss language
- peaceful –> ?
- nervous –> nervous
- angry –> peaceful

Technical Manager language
- peaceful –> ?
- nervous –> heavy work
- angry –> be quiet

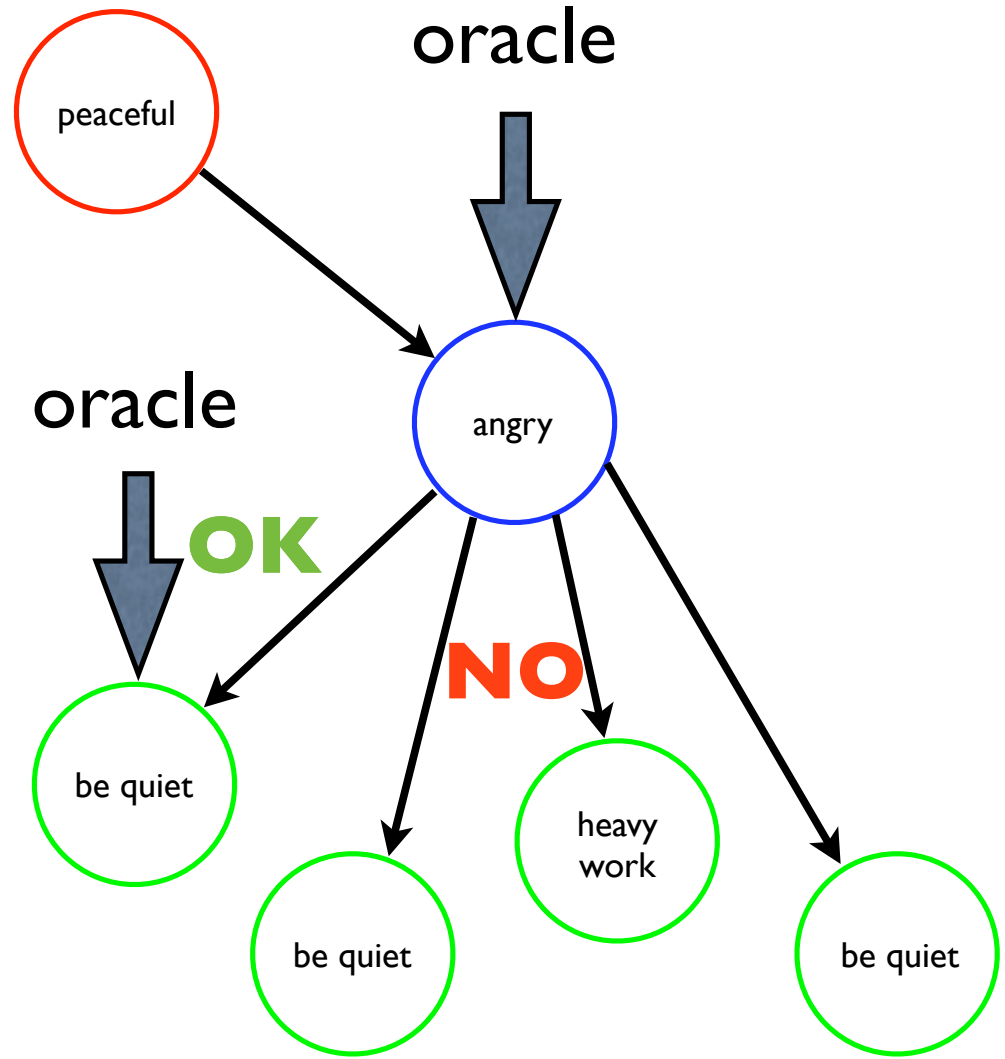
Employee language
- heavy work
- be quiet
- other

inconsistency → Who is faulty?

ARG example

- in order to find the faulty node an oracle is required

ARG example



Boss language
- peaceful –> ?
- nervous –> nervous
- angry –> peaceful

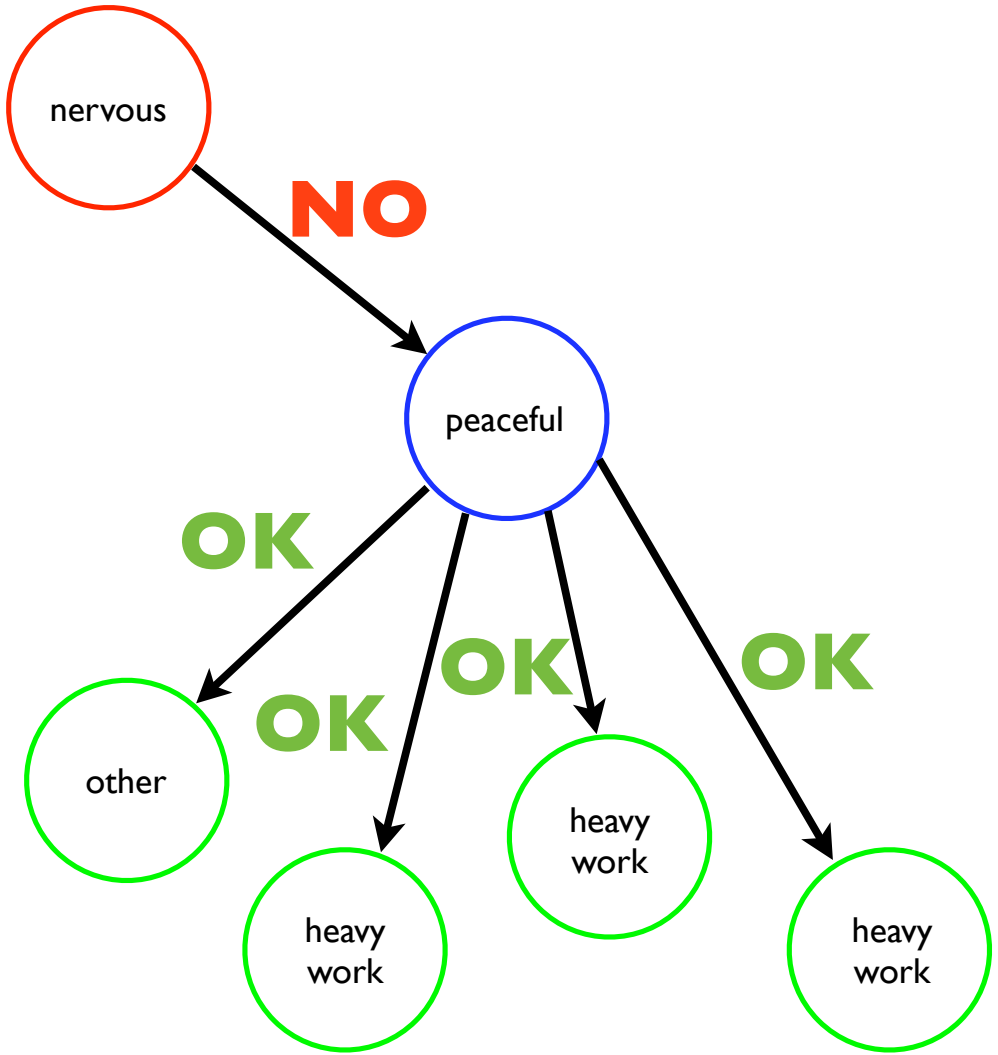
Technical Manager language
- peaceful –> ?
- nervous –> heavy work
- angry –> be quiet

Employee language
- heavy work
- be quiet
- other

passive and active fault discovery

- a **passive fault discovery** is based on the observation of the current state
  - not all of the rules of a language are useful for create oracles: a rule 0–>0|1 in a boolean language does not give information about the correctness
- an **active fault discovery** can force the state of the nodes in order to create oracles (i.e. it solves the problem introduced by useless rules)

ARG example



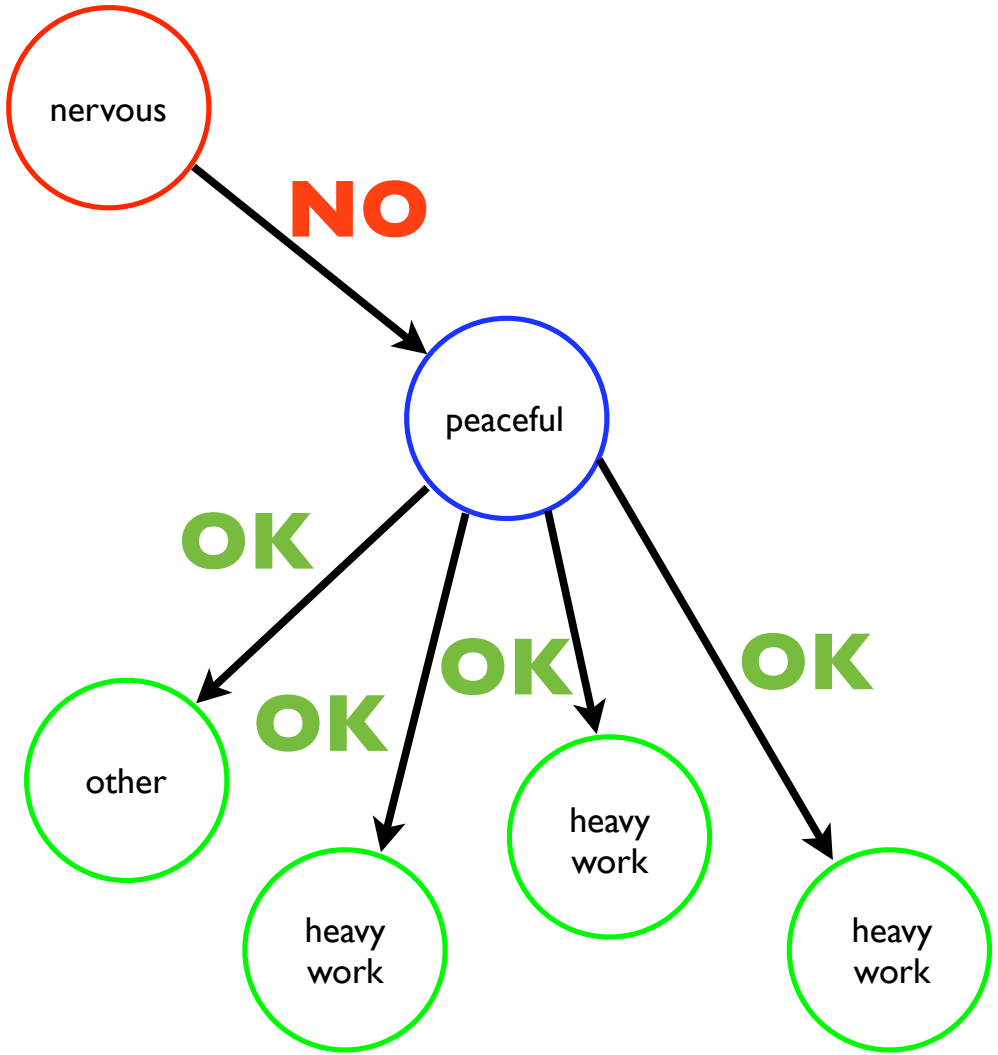
Boss language
- peaceful –> ?
- nervous –> nervous
- angry –> peaceful

Technical Manager language
- peaceful –> ?
- nervous –> heavy work
- angry –> be quiet

Employee language
- heavy work
- be quiet
- other

there are no oracle:
the passive detection will fail

ARG example



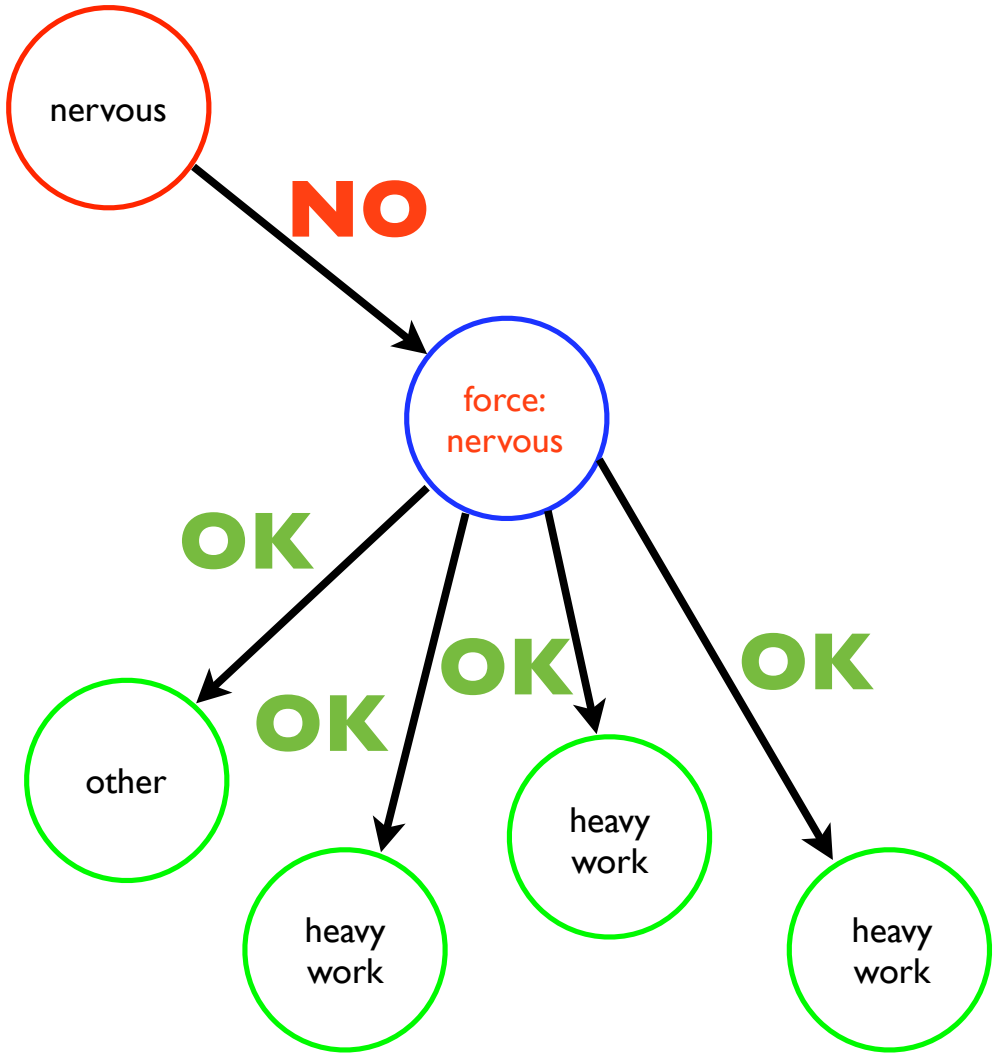
Boss language
- peaceful –> ?
- nervous –> nervous
- angry –> peaceful

Technical Manager language
- peaceful –> ?
- nervous –> heavy work
- angry –> be quiet

Employee language
- heavy work
- be quiet
- other

there are no oracle:
the active detection will discover the fault

ARG example



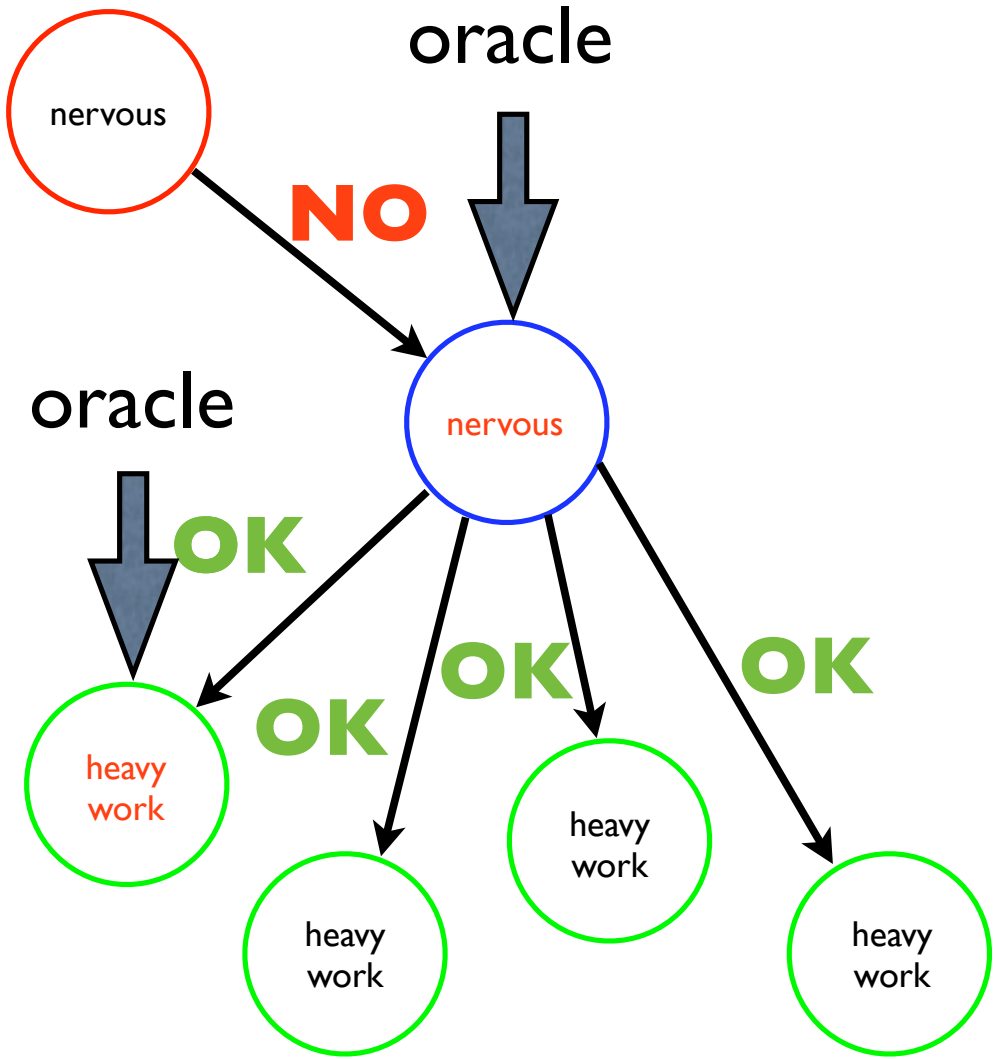
Boss language
- peaceful –> ?
- nervous –> nervous
- angry –> peaceful

Technical Manager language
- peaceful –> ?
- nervous –> heavy work
- angry –> be quiet

Employee language
- heavy work
- be quiet
- other

there are no oracle:
the active detection will
discover the fault

ARG example



Boss language
- peaceful –> ?
- nervous –> nervous
- angry –> peaceful

Technical Manager language
- peaceful –> ?
- nervous –> heavy work
- angry –> be quiet

Employee language
- heavy work
- be quiet
- other

there are no oracle:
the active detection will discover the fault

conclusion

- in case of **passive discovery** the number of detectable fault depends from the language, from the current state and from the graph topology. The current state cannot be predictable, then the capability of **fault discovery is not known a priori**
- in case of **active detection** the number of detectable fault depends from the language and from the graph topology, then **it is known a priori**

- The communication layer

- the communication layer is one of the most important part of the automation system because it decides the bandwidth for the data exchange:
  - a **low-frequency** (i.e. limited bandwidth) communication **allows** (i) to use simple wires and (ii) to interconnect the device with a non-fixed architecture **but** it makes impossible to transmit audio/video and complex events
  - a **high-frequency** (i.e. large bandwidth) communication **allows** to transmit whatever (also full-hd streams) **but** it imposes (i) to use a cat5/cat6 cable (€++) and (ii) a fixed interconnection schema

- the potential of each home automation system is always related to the communication layer

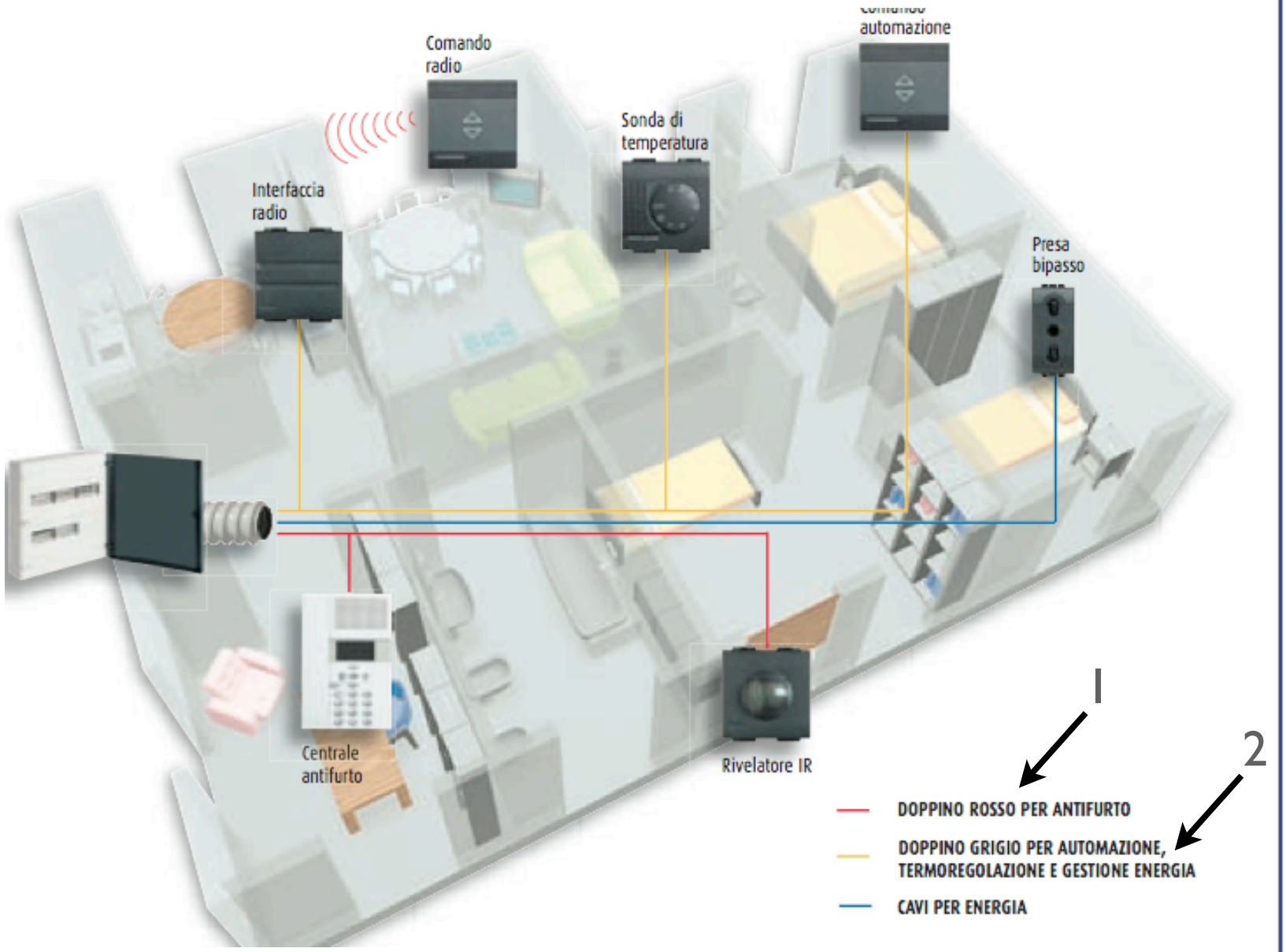
- The communication layer

- in home automation systems the communication layer is always (often) represented by a BUS

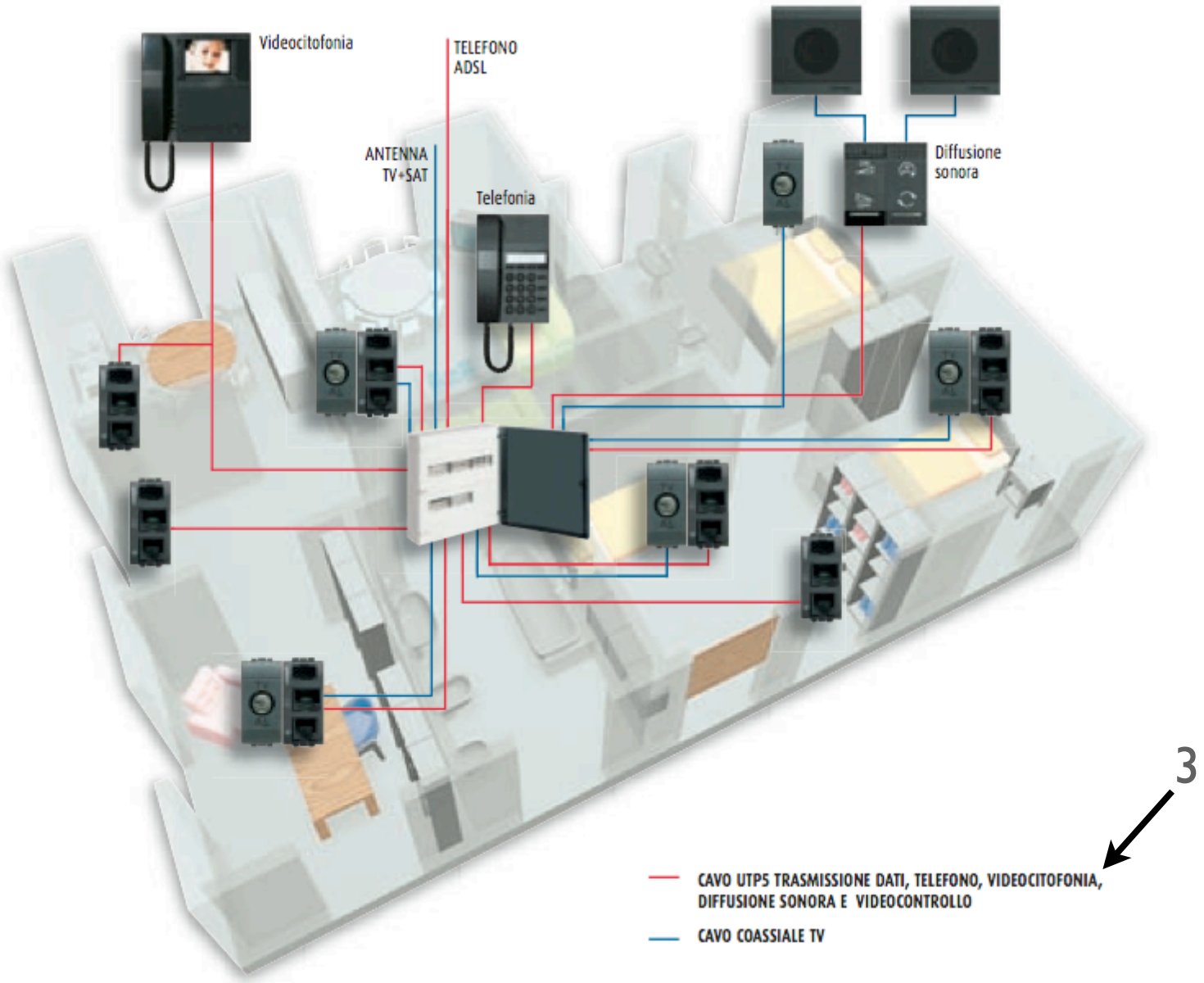
- a BUS is the preferred solution because it dramatically reduces the amount of cable
  - ...even if a large amount of companies do not exploit this advantage..

- a bad bus design example: 3 independent busses for a single home automation system... (see next slide for details)

The communication layer



The communication layer



- The communication layer: a brief overview on some famous bus standards:

  - I^2C
  - CAN
  - Ethernet
  - KNX
  - EDS

  - question: we presents five "wired" protocols...why don't we care about wi-fi protocols?

  - possible answer**S**: reliability? ...security? ...or both?

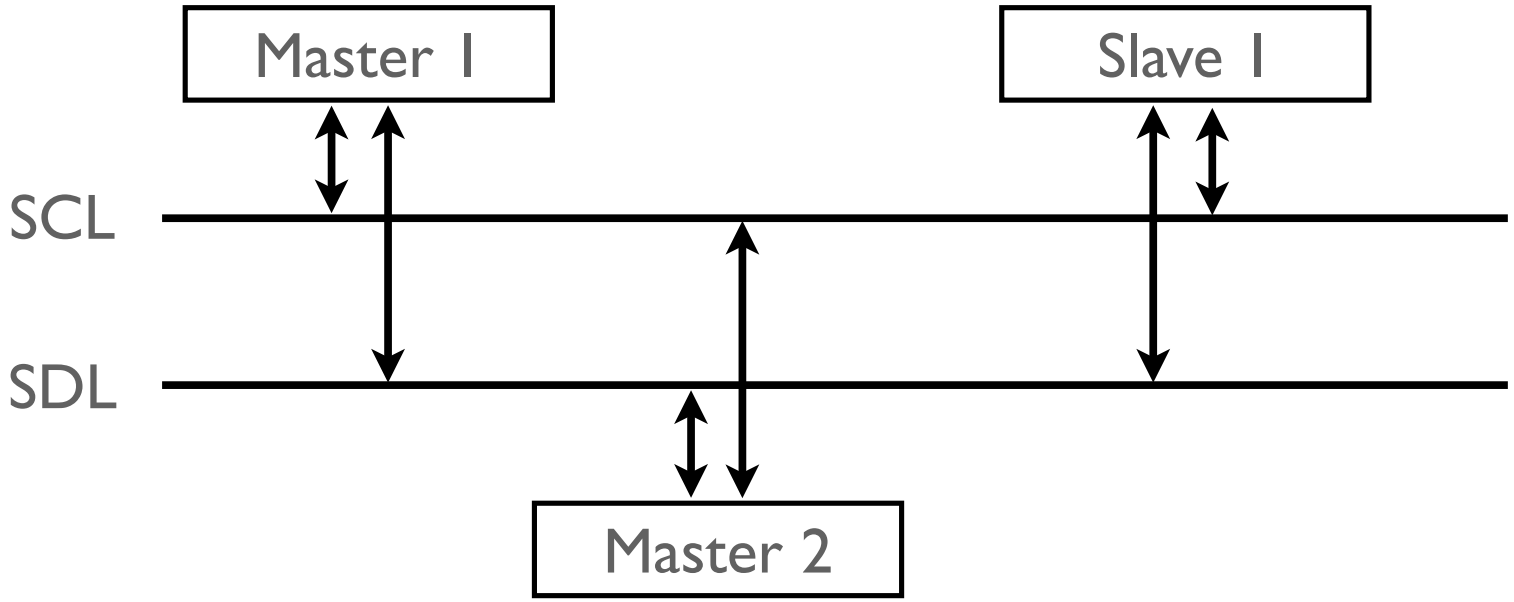
- The communication layer: I^2C

- I^2C was developed by Philips in 1982 (it becomes free from 2006)
- it was designed for micro-controllers interconnection, so it works for very **limited extensions** (less than a meter)
- it was originally developed for low-cost application
- speed: from **100 Kb/s to 400 Kb/s**
- I^2C is characterised by two lines:
  - **SDL**: serial data line
  - **SCL**: serial clock line
- Nodes are **connected both** to the **SDL** and both to the **SCL**
- Nodes are divided in **masters** and **slaves**
  - masters nodes can write on the Serial Clock Line (SCL) and aldo on SDL
  - slaves can only write on SDL only after a master invocation

- The communication layer: I^2C

- the protocol:
  - **transmission**: once the master hear the SCL clear it can start to write the clock signal (on SCL) and the data (on SDL)
  - **reception**: each device has an address, once a slave reads its own address it can read its message from the slave
  - it is possible to use the broadcast address: 0000000
  - **arbitration**: I^2C is a **CSMA/CA** bus, then each node master reads and writes the bus at the same time, once it writes 1 and read 0 (***the opposite is impossible***) it immediately stops to transmit, then the device that is concurrently writing a zero obtain the precedence
  - if two masters are writing exactly the same message to the same address the collision will be not discovered...but it doesn't represent a problem

- The communication layer: I^2C example schema



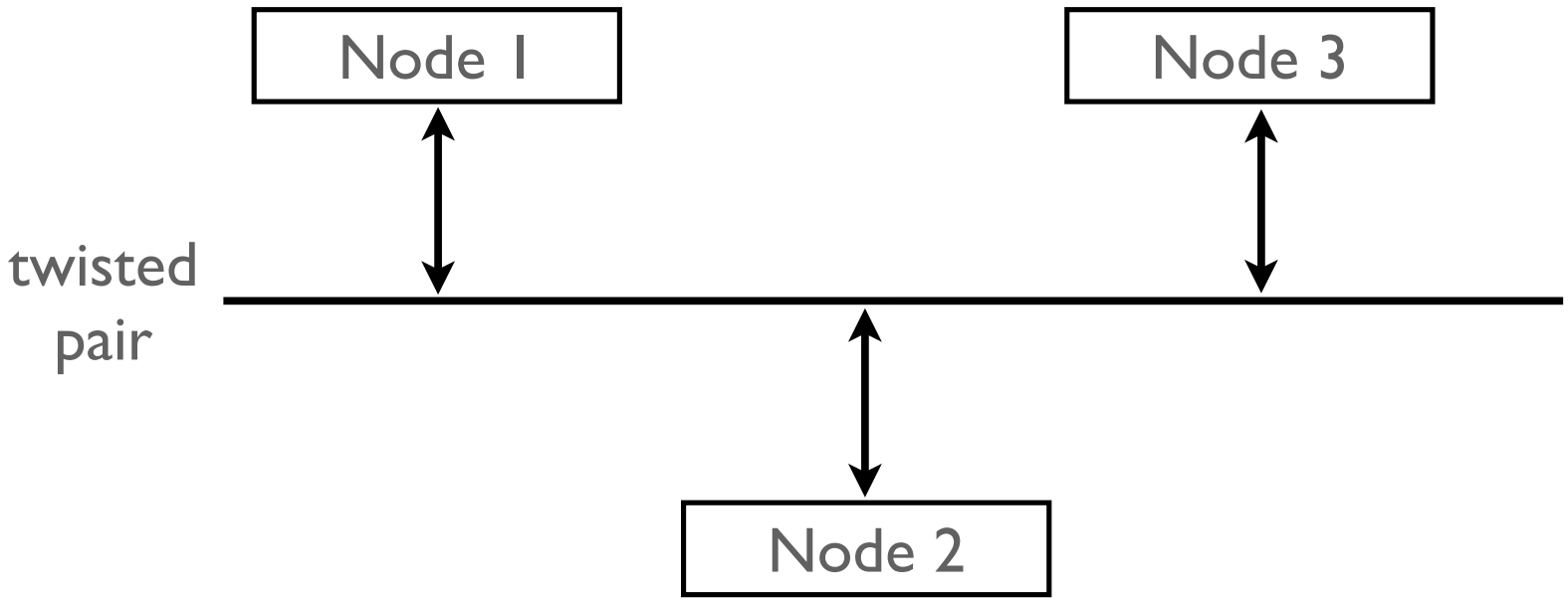
- The communication layer: CAN

- CAN bus was originally developed by **Bosh** in **1983**
- CAN bus was designed for **automotive** electronics in order to allows micro-controllers to communicate without an host computer
- CAN bus runs on a **single twisted pair** cable (up to 40 meters)
- speed: 1Mb/s
- CAN is data-frame oriented, if a message is divided in multiple data-frames the devices must arbitrate the BUS for each data-frame
- CAN BUS **implements priority**: high-priority devices has an address close to zero, low-priority devices the opposite
  - there is a technical reason...see next slide...

- The communication layer: CAN

- the protocol:
  - **transmission**: once a device hears the bus clear it can start to transmit the message.
  - **reception**: once a device hears the start of a transmission it starts to receive it if the message contains its address, the **clock** is "**extracted**" from the message. All the message always starts with a logic 1 and the address of the recipient device
  - **arbitration**: CAN - like I^2C - is a CSMA/CA protocol, each node reads and writes the bus at the same time, once it writes 1 and reads 0 (***the opposite is impossible***) it immediately stops to transmit. **This is sufficient to ensure that the device with a lower address obtains the precedence**.
    - in CAN we talk about 0 as "**dominant**" and 1 as "**recessive**" states

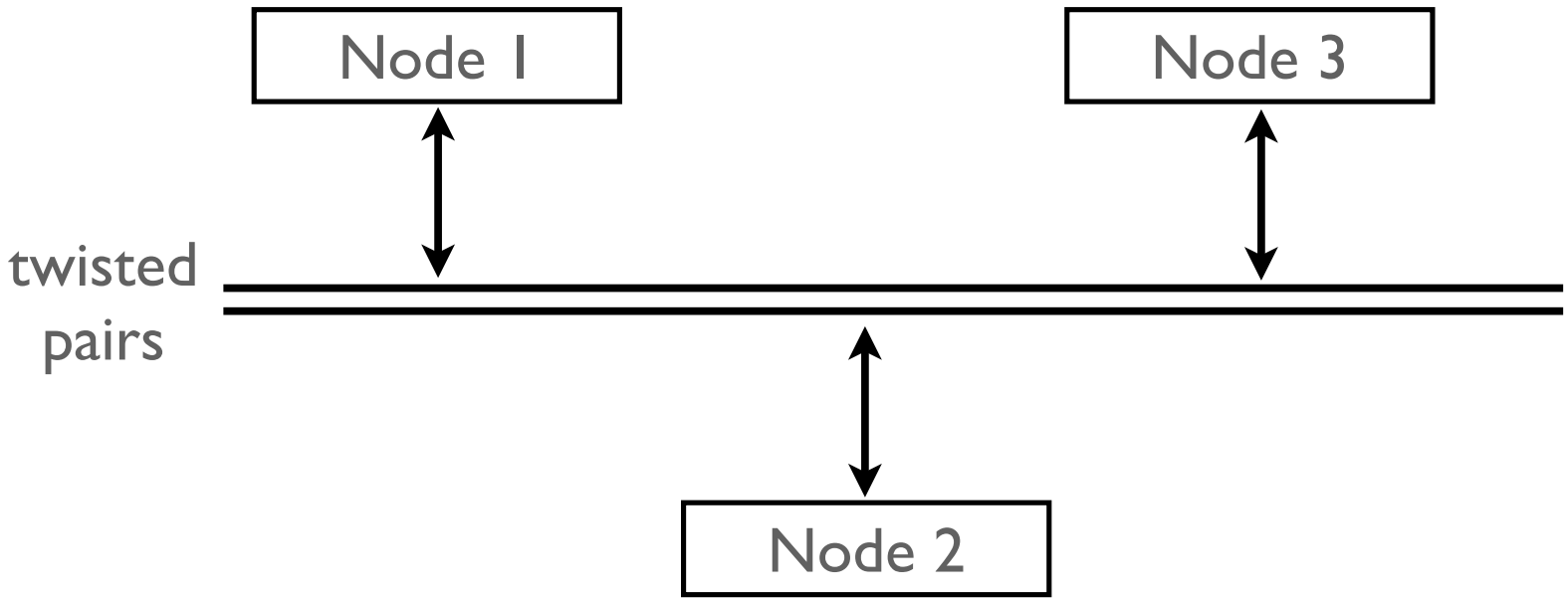
- The communication layer: CAN example schema



- The communication layer: Ethernet [ETH]

- ETH bus was developed by Xerox Parc between the 1973 and the 1974 and it becomes a IEEE standard (802.11) in **1980**
- ETH bus was designed for Local Area Network
- ETH was originally developed for coaxial cable, then the project moved to twisted pairs (tx+/- and rx+/-). A CAT5/6 cable with ETH can cover a distance of **about ~100 meters**
- speed: 10/100 Mb/s, usually **10Mb/s**
- ETH is a **CSMA/CD** protocol: if a collision is detected both the devices stop and retransmit after a random timeout.
- In order to avoid starvation the random timeout is set up with an exponential incrementation criteria

- The communication layer: ETH example schema



- The communication layer: Ethernet [ETH]

- **question**: why does ETH become so popular?
- **answers**: it was presented two years before I^2C and three years before CAN, moreover it is an IEEE standard and it can run up to 100 meters

- **question**: why does the **automotive prefer CAN** even if ETH is faster?
- **answers**: because CAN is CSMA/C**A** and it implements priority
  - what-if the air-bags are connected using ETH? [...]

- The communication layer: Konnex [KNX]

- KNX bus is the evolution of EHS, BatiBUS and EIB, it starts to be developed in **1990**
- KNX is designed to run on a single twisted pair, even if are available extensions for:
  - powerline
  - radio (KNX-RF)
  - infrared
  - ETH (known as KNXnet/IP)
- KNX bus wire can be up to **1.000 meters** (using repeater it can reach 4.000 meters)
- KNX is a **CSMA/CA** bus
- speed: **9.600 bit/s**, **packets** have a **variable length**

- The communication layer: Konnex [KNX]
- in order to develop KNX devices the KNX consortium requires a fee
- KNX consortium is composed by:
  - GIRA
  - **ABB**
  - **AMX** LLC
  - Berker GmbH Co. KG
  - **Bosch** Thermotechnik
  - **Cisco Systems**
  - Control4 EMEA
  - **Creston** International
  - **Daikin** Industries
  - Embedded Automation
  - Jung
  - **Legrand (BTicino)**
  - **Miele** & Cie KG
  - ON Semiconductor
  - Hager
  - **Schneider** Electric Industries S.A.
  - **Somfy**
  - Radiocrafts
  - **Bosch**
  - Russound/FMP Inc.
  - **Siemens**
  - **Toshiba**
  - Uponor corporation

- The communication layer: EDS

- EDS was originally developed in **1999** and it is owned by World Data Bus from 2004
- EDS is a **9.600** bit/s **CSMA/CD** protocol
- in EDS the collision detection is implemented via **ACK** messages: the transmitter element does not listen to the bus during the transmission phase
- EDS is designed to work using 2+1 wires such that:
  - 2 wires are dedicated to **Vcc** and **Data**
  - 1 wire is the **ground** reference
    - the house ground system can be used...usually it is better to use a dedicated line
  - **twisted pair is not required**, it is possible to use standard - ***inexpensive*** - wires up to **1.200 meters** (experimental uses show that the protocol works till 1.600 meters without repeaters)

The communication layer: global recap

	I²C	CAN	ETH	KNX	EDS
max length	1 m	40 m	100 m	1.000 m	1.200 m
speed	up to 400 kb/s	1.000 Kb/s	10.000 Kb/s	9,6 Kb/s	9,6 Kb/s
wire type	on chip	twisted pair	twisted pairs	twisted pair	single wires
arbitration	CSMA/CA	CSMA/CA	CSMA/CD	CSMA/CA	CSMA/CD
costs	medium	medium	medium/low	medium/low	very low