The DomoBus System

Prof. Renato Nunes

Home Automation
Current situation

- There are many different Home Automation technologies and solutions (standard and proprietary)
  - X10, EIB/KNX, LonWorks, CEBus, HomePlug, HAVI, UPnP, ZigBee, Z-Wave, etc, etc...
- These technologies are incompatible and cannot be interconnected directly
- New technologies and solutions are always being developed

Home Automation
Current situation

- Typical systems are quite complex to install and configure (especially the ones that offer good levels of functionality)
  - It is common to require the involvement of technical personnel and the usage of specific (and costly) tools (e.g.: EIB/KNX, Lonworks)
- Commercially available solutions cannot, easily, be adapted to changes in the home nor to new user requirements or user preferences
  - Technical personnel must modify the original project, install the new features and reconfigure the system

DomoBus Objectives

- Offer new ways to monitor and command a home automation system
- Allow a flexible and simple way to change the behavior of a system
  - A common user should be able to change how a system performs, adapting it to new needs or preferences
  - Changes can be made at any time, without any system disturbance, and become immediately operational
- Offer a generic approach to home automation, independent of any specific technology
- Support interoperation with different technologies

The DomoBus project

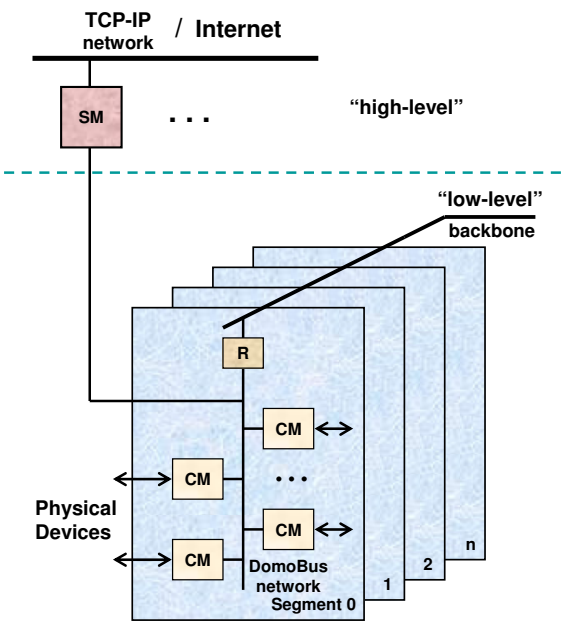
- It is an academic project
- It offers a learning platform and promotes the development and test of new ideas
- It is always open to evolution
- It aims at developing buildings blocks to be used as the basis for future features, more complex, more user friendly and more "intelligent"

- See www.domobus.net

The DomoBus "low-level"

- In the beginning, DomoBus targeted the "low-level":
  - Control Modules (CM) were developed, which interface with sensors and actuators (current prototypes use, typically, *Arduino*-like boards)
  - A specific software architecture was developed for the Control Modules (which is still being improved)
  - A communication protocol was developed to support interaction between them and other system components (also under improvement)

DomoBus Architecture (initial approach)



"High-level" components:

- SM - Supervision Module (PCs or Raspberry-Pi like boards; performs supervision tasks, interface with users, offer remote access through an internet access point)

"Low-level" components:

- CM - Control Module (Arduino-like boards plus interface electronics or power electronics to interconnect with physical devices - sensors and actuators
- R - Router module (optional; interconnects different domobus "low-level" network segments)

Example of an old CM Prototype

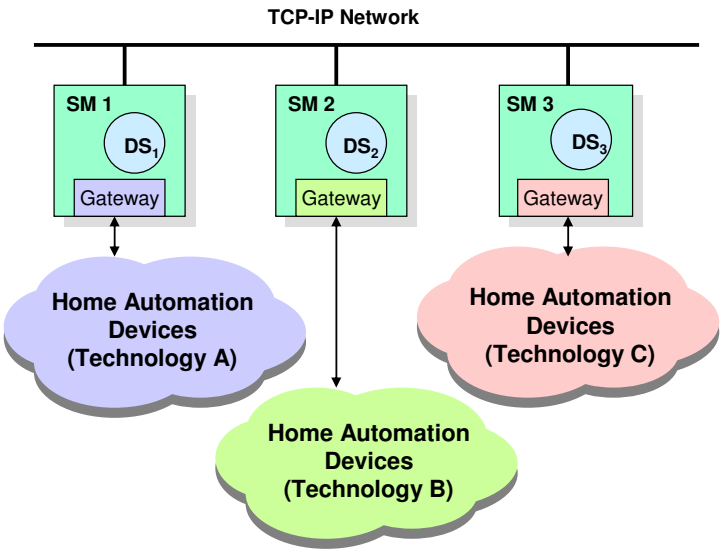
- Microcontroller AT90S8515 from ATMEL
  - 8 KBytes of memory code (FLASH)
  - 512 Bytes of RAM
  - Built-in UART
  - 2 timers
- EIA-485 transceiver for communication
- 20 Input/Output lines

Current prototypes are based mainly on *Arduino*-like boards (ATmega328: 32KB code, 2KB RAM, 3 timers, ADC, PWM, …)

DomoBus "high-level" Objectives

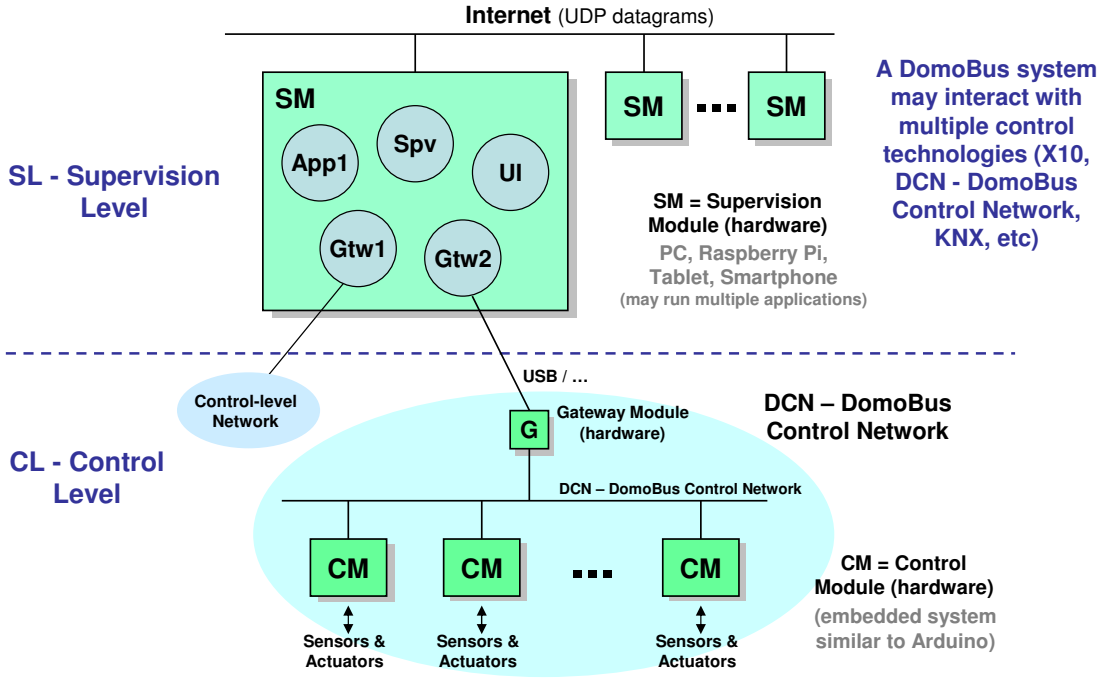
- Offer a generic platform that supports interoperation with different technologies (not just the "low-level" DomoBus)
- Offer generic tools for system specification and system operation (independent of the type of the "low-level" technology being used)
  - Allow development of generic applications that can be applied to any house and any system
  - Allow a common user to define or change how the system performs

"High-level" Architecture Interoperation Support



- SM – Supervision Module (hardware: Raspberry-Pi, PC, …)
- DS – DomoBus Supervisor application
  - Manages a set of home automation devices
- One SM may hold multiple DS applications

DomoBus Architecture



Key aspects of the DomoBus approach

- Generic model for a “home-automation device”
- System specification language (XML based)
  - Defines systems composition
  - Defines house structure
- Generic model for specifying how a system behaves, based on
  - Rules (if <condition> then <actions> else <actions>)
  - Timing Actions (wait *t* time, then execute <actions>)
  - Schedules (at time *T*, with repeat period *P*, execute <actions>)
  - Automation Blocks (automation component with generic inputs and generic outputs that implement a given behavior; a DAB – DomoBus Automation Block – can be instantiated multiple times in any given system; see www.domobus.net for more info)

DomoBus Device Model

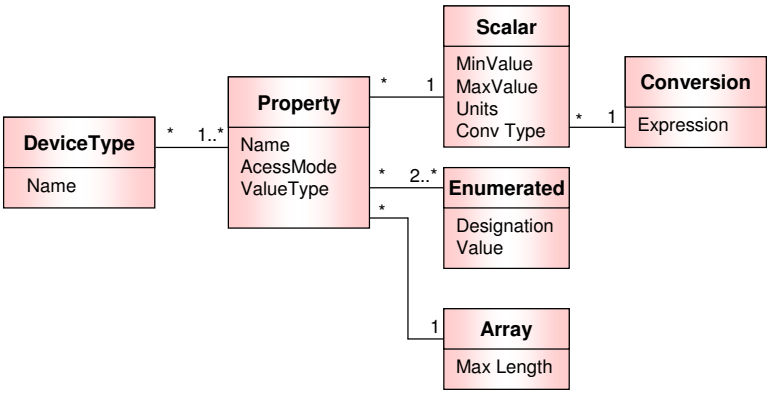
- DomoBus Device:
  - Abstract entity characterized by a set of "properties"
  - Standard operations over properties:
    - GET – Read a property value
    - SET – Modify a property value (may imply an action over the environment)
    - NOTIFY – Each device can be configured to, autonomously, notify its DomoBus Supervisor (DS) when a property's value has changed, informing the new value.

DomoBus Specification Language

- System specification language (XML)
  - Specifies types of devices
  - Specifies home structure (floors, divisions)
  - Specifies the system's composition
  - In the future, also:
    - system behavior
    - users profiles
    - . . .
- Promote **generic applications** that work with all houses/buildings and all systems
  - Just load a XML file and obey to its content

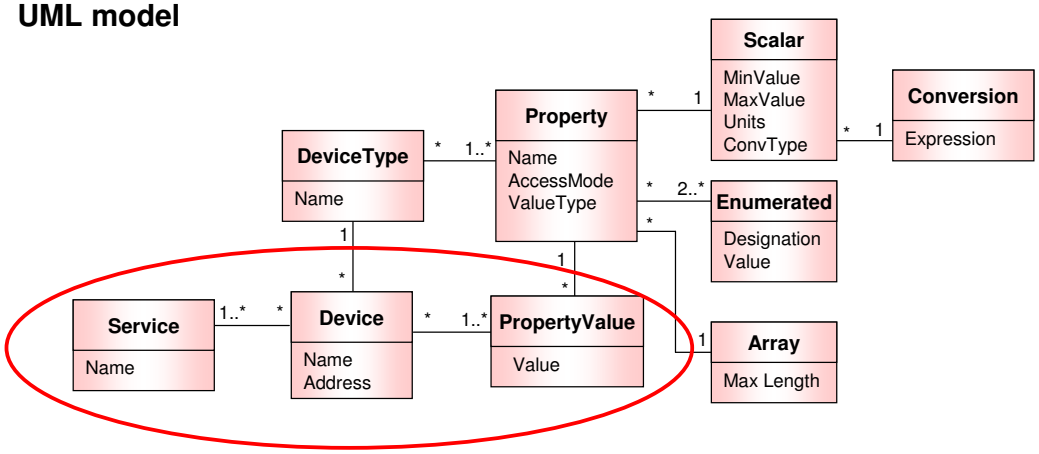
Properties Types and Devices Types

UML model



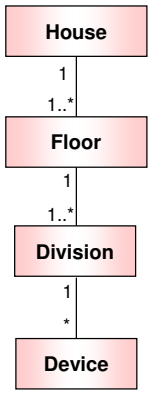
Instantiation of Devices

UML model



House Structure

UML model



DomoBus Specification Language

- General Structure (main “sections”)

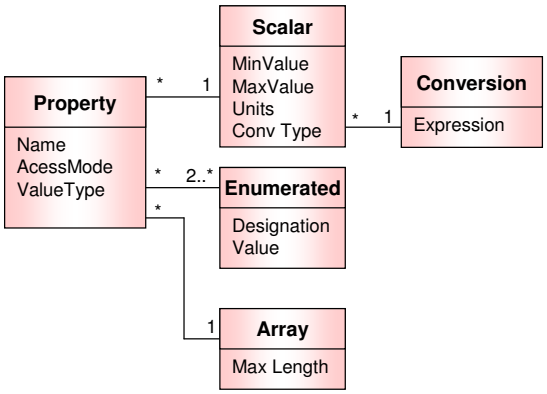
<DomoBusSystem ID="#" Name="x" Type="#.#" Version="#.#" Date="x">
. . . Value types (allow definition of Properties). . .
. . . Device types . . .
. . . Users and Access levels . . .
. . . House structure . . .
. . . Services . . .
. . . Devices . . .

Note:
“#” – Represents a number
“x” – Represents a string

and other additional relevant data…
. . . Scenarios . . .
. . . Favorites (most used devices, etc) . . .
. . . System behaviour . . .
. . . System state . . .
. . .
</DomoBusSystem>

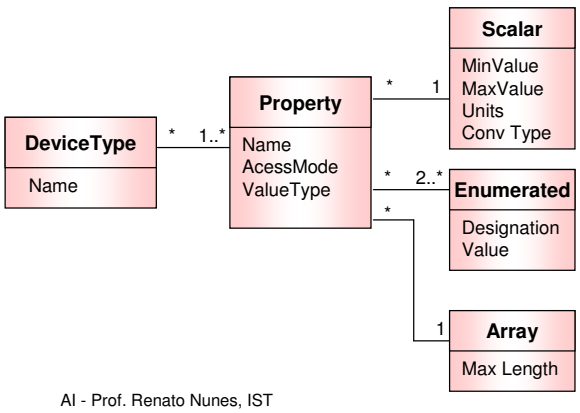
Property types

```
<ScalarValueTypeList>                    <!-- NumBits: 8 or 16 -->
 <ScalarValueType ID="#" Name="x" NumBits="#" Units="x" MinValue="#"
    MaxValue="#" Step="#">
  <ValueConversion Type="x" Ref="#" />
   </ScalarValueType>
</ScalarValueTypeList>

<EnumValueTypeList>
 <EnumValueType ID="#" Name="x">
    <Enumerated Name="x" Value="#" />
    <Enumerated Name="x" Value="#" />
 </EnumValueType>
</EnumValueTypeList>

<ArrayValueTypeList>
 <ArrayValueType ID="#" Name="x" MaxLen="x">
    <ValueConversion Type="x" Ref="#" />
 </ArrayValueType>
</ArrayValueTypeList>
```



Device types

```
<DeviceTypeList>
 <DeviceType ID="#" Name="x" RefDeviceClass="#" Description="x">
   <PropertyList>
    <Property ID="#" Name="x" AccessMode="x" ValueType="x" RefValueType="#" />
                              <!-- Value types: "SCALAR", "ENUM" or "ARRAY" -->
   </PropertyList>
 </DeviceType>
</DeviceTypeList>
```

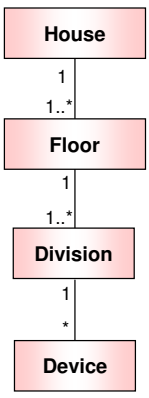


Users and Access Levels

```
<AccessLevelList>
    <AccessLevel Level="#" Name="x" />
</AccessLevelList>


<UserList>
    <User ID="#" Name="x" Password="x" AccessLevel="#" />
</UserList>
```


House structure

```
<House ID="#" Name="x" Address="x" Phone="x">
  <FloorList>
    <Floor ID="#" Name="x" HeightOrder="#" />
  </FloorList>
  <DivisionList>
        <Division ID="#" Name="x" RefFloor="#" AccessLevel="#" />
  </DivisionList>
</House>
```



Services

```
<ServiceList>
    <Service ID="#" Name="x" />
</ServiceList>
```


Devices

```
<DeviceList>
  <Device ID="#" RefDeviceType="#" Name="x" Address="#" RefDivision="#"
          AccessLevel="#,#" UserBlocked="#,#">
    <DeviceServiceList>
      <DeviceService RefService="#" />
    </DeviceServiceList>
  </Device>
</DeviceList>
```

Scenarios

```
<ScenarioList>
  <Scenario ID="#" Name="x">
     <ActionList>
       <Action ID="#" RefDevice="#" RefProperty="#" Value="x" />
     </ActionList>
  </Scenario>
</ScenarioList>
```


Favorites

```
<Favorites>
  <FavoriteList RefUser="#">
     <FavoriteDevice ID="#" RefDevice="#" />
     <FavoriteDivision ID="#" RefDivision="#" />
  </FavoriteList>
</Favorites>
```

Specification Language Reference

- See document that describes the DomoBus Specification Language (available in fenix and in www.domobus.net)

Questions?