Integration with DOMIQ/Base

One of the most desirable features of a smart home/building is the possibility to integrate multiple systems and devices into one complete solution. This task is greatly facilitated by the integration capabilities of the **DOMIQ/Base** module.

The **DOMIQ** system offers active and passive integration. Active integration is implemented based on extension modules (**Serial**, **Expander**, **Light**) or realized by software of a **Base** module. The main principle of this type of integration is the fact that the data exchange is initialized and controlled by the **DOMIQ** system. An example of this type of integration is the integration of Satel Intrusion Detection System, where a **Base** module in conjunction with a **DOMIQ/Serial-2SI** module periodically polls status of variables from alarm panel. In general, the **DOMIQ** system acts as a master in this type of integration.

In passive integration, the third party software initiates communication with a **DOMIQ/Base** and the **Base** module processes received data based on programmed rules, but it doesn't initiate data exchange by itself. **Base** module acts as a slave. This type of integration gives almost unlimited possibilities to integrate all kinds of network devices that are connected to the same LAN, where a **Base** module is connected.

1. Protocol and capabilities

1.1. Establishing connection

Passive integration is based on the TCP/IP protocol. **Base** module listens on the port **4224**. Communication is not encrypted thus this port shouldn't be forwarded to be accessible in the Internet. In order to connect, a master device must establish connection with the IP address of a **Base** module on the port 4224. For testing using PC you can use netcat (`nc <IP> 4224)` or telnet (`telnet <IP> 4224`).

1.2. Capabilities

By establishing connection with a **Base** module, a master device can realise the following capabilities:

1. Data aggregation from a **Base** module.
2. Control state of all variables and devices available in a **Base** module.
3. Passing data to a **Base** module.

The latter part of this tutorial describes each capability in more details.

1.2.1. Data aggregation from a Base module

After establishing connection with a **Base** module, a master device will receive notifications about changes of state of identifiers available in the **Base** module. Notifications are sent automatically by a **Base** when any change of state occurs. Based on notifications, a master device can visualize the status of all identifiers (e.g. state of the LCN system, Modbus registers, alarm system, etc.) in the real time. An example of using data aggregation can be an application that collects data about temperatures or energy consumption in a building for later analysis.

Base module can return state of all identifiers in one packet. In order to do that, a master device has to send the question mark character followed by the carriage return and newline characters: `?\r\n`. In case of testing from a PC, please type the `?` (in terminal/command line window) and then press the **Enter** button to confirm the command. After receiving this query, **Base** module will return state of all available identifiers in the following format: `name=value` (each entry in a separate line).

You can also query state of a single identifier. In order to do that, a master device has to send query string in the following format: `identifier=?\r\n`, for example: `LCN.output.0.10.1=?\r\n`

In case of testing using command line window on PC the `\r\n` sequence need to be ommited and instead the command has to be confirmed with the **Enter** key.

1.2.2. Control

Using passive integration, a master device can control state of all identifiers available in a **Base** module, therefore it can control all devices and variables that **Base** module is able to control. In this case, **Base** module needs to receive a command in the following format: `identifier=value\r\n`, for example `LCN.output.0.10.1=100\r\n` - this command will set output no. 1 in a LCN module with the ID 10 to 100%. List of all available identifiers can be found in the **Configuration Manual** of the **DOMIQ/Base** module.

This feature allows to use **Base** module as a gateway between any master building automation system and subsystems controlled by a **Base** module.

1.2.3. Passing data to Base

Passive integration also allows to pass data from a master device to a **Base** module. This feature allows to integrate any device that is capable to establish connection on the 4224 port and send packets as presented in earlier chapters. Examples of this kind of integration was presented in tutorials about integration of Mobitix cameras and Mobotix Video Door Station. In those tutorials we presented how video door station can pass events for example about pressing door bell button or about motion detection and so on.

By passing data to **Base** module you can integrate:

- **Microcomputers** - for example: Arduino, Raspberry PI, BeagleBone etc. Capabilities of microcomputers are huge, from these closely related to the topic of smart homes we should mention possibility of using cheap and commonly available digital and analog sensors, for example temperature sensors, humidity sensors, carbon dioxide sensors, air quality sensors etc.
- **Video door stations**
- **PLCs**
- **Other home automation software**, for example **OpenHAB**.

Any data received by a **Base** module is interpreted as a command, therefore it is prefixed with the `C.` So if a master device sends a packet in the following format: `identifier=value\r\n`, then a **Base** module interprets it as `C.identifier=value\r\n`.

Received data can trigger an event that will excute any sequence of programmed commands.

Example

As an example we will present sending of a packet by **Mobotix** Video Door Station that informs about pressing bell button. The result of receiving this packet is sending a push-notification to the **Remote** app. In this example we will assume that **Mobotix** sends the following string when the bell button is pressed: `MOBOTIX.event.t24=CameraBell-Button\r\n`

In order to handle this packet in **Base** module, we will define the following event :

- **Channel:** *C.MOBOTIX.event.t24*
- **Data:** *CameraBellButton*
- In the **Actions** section, click on the **Add Command...** button, and in the new window that appears enter:
  - **Name:** *C.REMOTE.notify*
  - **Value:** *Door bell*.

Similarily you can create other events and process any data that **Base** receives over the 4224 port.