Initial steps for Driver development

This lesson will guide you through the first steps of development of Plugins for integration of external systems.

Plugins for integration of external systems are components connecting the C4 System and an external system. External system can be any device or software which enables integration. Concept of these Plugins allows integration of communication protocols via TCP/IP, UDP/IP, RS232 or any remote calls like Rest API, or SQL call to any database.

Within the C4 System, there are two types of Plugins used for integration of external systems.
The first one is Driver. It communicates with an external system continuously from the moment when the communication begins, until it is stopped. Driver is mostly used for communication with security systems that should be monitored and managed non-stop.
The second type is IS Connector. It is designed especially for processing and exchange of large volumes of data between the C4 System and external system. It is usually used for scheduled synchronization of personal data with customer's information systems, such as Active Directory, payroll system, or HR management system.

Now we will focus on development of a Driver.

Task of the Driver is to monitor and control individual parts of an external system in the C4 System environment.

In our lessons, external system is a virtual application which emulates simple devices with one input and one output. We will use this example to demonstrate, how to integrate such device to the C4 System. Type of a communication protocol depends on a particular device, in our case it's a binary protocol.
Protocol description is a part of the source codes for the Demo Driver.

Each device consists of several address units (subsystems), which can be clearly identified within a communication protocol. In this case, the first subsystem is an entry point for communication with the device. The second one is a central unit which ensures device logic. We also have an input and an output. In our case, it's a

non-supervised input (two states).
Since each of these subsystems must be addressable, in this case the address of the input is 1, and address of the output is 1.

Now, we will talk about environment in which the Drivers are developed. In standard operation mode, Drivers are hosted and managed by an application called Plugin Runner. Each instance of Plugin Runner is hosting Drivers of one type. There can be an unlimited number of Drivers running within one Plugin Runner.
Plugin Runners are managed by Plugin Manager, which runs as a separate service within the C4 System architecture.

In Plugin Runner, it is possible to launch only Drivers with licenses.
Since licenses are issued only for the finished Drivers, developing a Driver in Plugin Runner is a big challenge. Therefore, Gamanet developed an application called Developer Console, which emulates environment of both Plugin Runner and Plugin Manager. It allows to run only one instance of a Driver operating one integrated device at a time. The advantage is, that in Developer Console, this instance of Driver is not checked for licenses.
Developer Console provides developers environment, where they can develop and debug the Driver conveniently.

Driver development consists of four steps:
registration, initialization, development and publishing.

In the first step, a developer must register development of a new Driver.

This registration should be done on Gamanet Portal Services web, section Plugin Store.

Afterwards, a developer needs to define structure of his Driver in mySDK section.

To create a new Driver definition, it is necessary to define all important parameters of your Driver. You should name a vendor, which is a company owning rights for the Driver. Also, you need to define parameters of environment in which your Driver will be able to work smoothly.

On my.c4portal.com/sdk/devicetypes we can find an overview of all subsystems which Gamanet provides to the developers to set up representation of the

integrated device. If you want to integrate a subsystem that is not listed here, you should contact Gamanet on sdk@gamanet.com and ask for its registration.

Each device has its graphical representation in the form of icons of different sizes and formats for various parts of the application.

In the next step, a developer defines all subsystems of a particular device. Each subsystem which we want to monitor and control individually, must be defined here.
An important step is to identify license policy for the Driver. License can be defined for one or more subsystems. If there are more licensed subsystems, C4 System considers all of them as equal. A customer always buys one type of license for one Driver, regardless of the number of subsystem types to which the license will be applied. It is recommended to choose one licensed subsystem, the best option is a control unit. It allows optimal scalability of license fees, and saves customer's time and work regarding licenses when the installation changes.

When the list of subsystems is created, the next step is to define relationship between them.

Usually, the starting point is a defined communication unit, in our case it's represented by an Ethernet input. Under the communication unit there is a central unit, and then input and output.

In this step, we determine how the topology of the device can be defined on the customer's installation. Based on this definition, items are displayed in the C4 System after right clicking on the device, under the "Add" menu.
In case we need to define a possibility to add a subsystem under several units, at first we should define the real representation of the subsystem where it will be registered. Afterwards, it can be placed as a shortcut to other groups, simply by holding a Ctrl button and drag and dropping it to the particular group.

In the next step, properties for each subsystem should be specified. Gamanet predefined a set of parameters for each type of device. They can be further modified, extended or reduced by a developer. However, some parameters, such as address, are mandatory.

When an event occurs on a device, it is loaded through Driver to the C4 System. In Driver it must always be paired with the particular subsystem which it concerns.

Each device has its own list of event types, which are generated in the form of a journal. Task of the Driver is to transform events from the device to a unified list of events in the C4 System, and add parameters concerning the device, for example address of the subsystem, the person concerned, or time when the event occurred.

The Driver saves this generated event in the database.

A developer can view a predefined list of event types registered in the C4 System on my.c4portal.com/sdk/Messages. They are structured according to the type and severity, and this hierarchy can be used for easy filtering and batch processing within automatization.
If you want to add a new event to the list, you need to contact Gamanet on sdk@gamanet.com.

When developing a Driver, a developer must define for each subsystem all types of events that can occur there. If the events are correctly assigned to the particular subsystem by the Driver, they are displayed to the user within the chosen unit (subsystem) in a device tree. Here, the device tree serves as a filter for displaying only relevant events.

In the next step, a developer defines a list of commands, which will be used to control the particular subsystem. A complete list of available commands can be found on my.c4portal.com/sdk/Messages.
If a developer needs to add a new command to the list, he should contact Gamanet and ask for its registration.

Each device graphically represents possible real statuses of the device to the user. Every status is represented by certain color, sound, or blinking, and every status has its unified status icon. Status can also be represented by change of the basic icon (for example, door open and door closed).

Statuses are predefined by Gamanet, and the developer's task is to ensure that the Driver transforms real statuses of a device to their unified representation in the C4 System. That's why a developer must specify within his Driver, which statuses it will support.

After defining all the necessary information, a developer can export a Visual Studio project and download it to his computer.

Each Driver must contain certain definitions in its source code, in order to be able to run in the C4 System environment.

MySDK portal pre-generates these parameters according to the definitions specified by a developer. To make the development easier, mySDK portal allows to export an extended version of source codes containing predefined parts and recommendations for codes, which need to be completed by the developer.
In case of a repeated export, for example after changing a Driver definition on mySDK portal, it's convenient to export only those parts of the project, which are permanently managed by web generator. These parts always contain "generated" in their name.
Export "Driver Project Initial" generates a project with recommended steps for developer. Export "Driver Project Update" generates only files marked as "generated".

The second step is to initialize the project in environment of a developer's computer.
Within this step, it is necessary to update the required libraries from the NuGet server.
After compiling Driver source codes generated by mySDK portal, the Driver should be registered to the C4 System, where it can be configured during the next steps of development.
After upload to the C4 Server, the installation package is unpacked and the server parts of the Driver are installed. It also provides installation of the client parts to the C4 Client.
It can be done after launching the project in environment of Developer Console, which is within a project defined as a debugging application for development of a Driver.
After registering the package in Developer Console, you can check whether your Driver was successfully registered in the C4 System.
A list of all registered Drivers can be found in the C4 Client, under "Settings" and "Packages".
In the next step, a developer defines topology of external system's demo set to the C4 System.
Demo set should contain all subsystems, so that the developer can test maximum number of situations, which may occur during the real system deployment.

As we have already mentioned, each Driver runs in a predefined Plugin Manager. It is important to set up, that the Driver which we will develop runs in Developer

Console environment. Also, we need to specify the local IP address and the port of the Demo Device. Then we set up the configuration by adding a central unit, one input and one output, exactly according to the construction of the particular device.
In the next step, we download this configuration in Developer Console, and we can start debugging the Driver.

The next step is Driver development itself.
Developer's main task is implementation of the communication protocol, which ensures communication between the Driver and external system.
A developer implements all functionalities required by the customer. He defines ranges of commands controlling the external system, defines and implements all types of statuses which can occur on the external system, as well as their graphic representation in the C4 System environment. Also, he implements registration of all events received from an external system to the central C4 System database.

This topic will be covered in a separate lesson.

When the development is finished, a developer needs to publish the Driver for use by end customers.
This is done through Developer Console. Driver should be published to the Portal Services web, section Plugin Store. Then it can be tested by Gamanet QA department for the purpose of verification, as well as by external system manufacturer for the purpose of certification.
After passing the tests, the Driver is available for implementation to projects.