INSTRUCTIONS

The project involves:

- OSGI_SIB v3.2, Semantic Information Broker for Windows platforms;
- KPI, software APIs for interacting with the SIB;
- initializer.jar, software module for populating the SIB with the triples describing the ontology and some random values for test purposes only;
- sensor.jar, mock replacing an FPGA or whatever computational device for detecting the plates of entering vehicles;
- passRegistation.apk, android mobile app for binding a plate to a purchased access ticket;
- monitor.jar, java interface which displays both registered vehicles with their access rights (on the right) and issued fines (on the left);
- reasoner.jar, the intelligence of the systems: it compares plates with the timestamp of entrance and the access rights in order to decide if issuing a fine is necessary or not.

If you want to *execute* the project:

1) run the **SIB** (*java -jar org[+tab] -console*);
2) execute the **initializer** (*java -jar initializer.jar SIB_ip SIB_port*);
3) launch the **monitor** (*java -jar monitor.jar SIB_ip SIB_port*);
4) start the **reasoner** (*java -jar reasoner.jar SIB_ip SIB_port*);
5) lastly, run
    a. the **sensor mock** (*java -jar sensor.jar SIB_ip SIB_port*) to simulate entering vehicles *and/or*
    b. the **mobile app** (*install passRegistration.apk on an Android smartphone*), firstly IP address and Port of the SIB must be set (Options → Configurations Parameters), then a new ticket can be registered.

➔ note that all the *java -jar commands* are supposed to be run from a terminal opened in the folder containing the relative *.jar* file.

➔ please consider also that the Initializer inserts inside the SIB only few triples regarding *Vehicle* and *Person* according to this criteria:

- Private: PP111PP → PP999PP
- Motorbike: MM000MM → MM444MM
- Taxi: TT555TT → TT999TT
- Resident: RR000RR → RR444RR
- Bus: BB555BB → BB999BB

If you want to *edit* the project:

a) ANDROID STUDIO CASE (passRegistration.apk)
    - Open Android Studio
    - Click on File → Open → choose *Interoperability of Embedded Systems M\SORGENTI\progetti android studio in android\PlateSubscription*
    - Press CTRL+Alt+Maiusc+S and open Project Structure → select *app* in the "Modules" menu → click on *Dependencies* → if *libs/jdom-2.0.5.jar* isn't present, add it by clicking on the green "+" and searching that library inside the "libs" folder of the project

b) ECLIPSE CASE (Initializer, Monitor, Reasoner, Sensor)
    - Open Eclipse and go to the workspace

- Click on *File* → *Open project from file system* → *Directory…* → select the desired project from the relative path *Interoperability of Embedded Systems M\SORGENTI\progetti eclipse in java\ZTL* → Click on *Finish*
- Right click on the project folder → Build Path → Configure Build Path… → Libraries → Add external JARs → select all the JARs located in *Interoperability of Embedded Systems M\JAR\da includere nei progetti sorgenti* (if there are other JARs with the same name already imported, delete them), Initializer also includes Apache Jena Libraries (not included, download link: https://jena.apache.org/download/).
- Click on Run → Run configuration → Arguments → in *Program argument* insert IP and port of the running SIB separated by a blank space → Apply

➔ *ModelPackage* is a standalone Java project including the model classes (JavaBeans) used by all the other 4 eclipse projects. If you want to change something inside the model, please import ModelPackage project inside eclipse (no external JARs needed), edit what you need and export all in .JAR format, overriding *ztlModel.jar* previously imported in the other Java projects. Doing this way, your changes will be automatically sent to the other projects.