Selene upgrade bundle

To make selene upgrade process flexible the following bundle format is proposed.

Install/upgrade bundle has the following structure:

```
/
distribution
   jar
          selene-engine-x1.y1.jar
          selene-web-x2.y2.jar
          some-jar-x.y.jar
          ...

   lib
          librxtxSerial.dll
          librxtxSerial.so
          someDll.dll
          someSo.so
          ...

   resources
       config
              SomeDao.properties
              AnotherDao.properties
              OneMoreDao.properties
              ...

       scripts
              telemetry.js
              restart.sh
              ...

       ...
               ...

scripts
   install.bat
   install.sh
   ...

jar
       ...

config
            backup.cfg
            install.cfg
            ...
```

The bundle archive consist of 2 directories:

- `scripts` contains installation scripts and additional files (configs, rules, etc) for different operating systems;
- `distribution` contains files to be installed or upgraded.

Installation/upgrade can be performed manually by unpacking archive and execution installation script corresponding to current OS; or automatically by selene on receiving corresponding request. To support automatic execution names of installation scripts must be hardcoded as `install.*`.

Scripts

The main purposes of installation scripts are:

- to define files to be removed, replaced or created in destination directory, and perform corresponding actions;
- to make OS configuration changes by updating OS config files and/or running external commands (e. g. configure application autostart and watchdog tools restarting application after unexpected crashes);
- to re/start installed/upgraded application if needed.

The main installation/upgrade logic can be located directly in scripts. If complex logic is required (e.g. fix inconsistencies or upgrade DB), it is allowed to write it on java and place corresponding `.jar` files into `jar` directory. Anyway `install.*` scripts must be main entry points, then they can start java to execute the mentioned files.

The main principle of installation scripts — do not hardcode names of files and directories located in `distribution`. Installation scripts should contain main logic only. All configs and rules should be moved to external files and should be used by installation scripts logic (e.g. templates of filenames to be removed or replaced in case if old and new names have difference not in version only).

Note: during files removal installation scripts must take into account that some files might be locked by OS. Such files should be renamed by adding `.bak` extension for further removal (e.g. after application restart at the end of upgrade phase).

Fallback

Installation scripts should support fallback scenarios to prevent incomplete upgrade and further application malfunctioning. For example, they can backup installation directory and OS configuration files and restore them if upgrade fails for some reason.

Logging

Installation scripts should provide detailed logging to determine root cause of upgrade failure, if any.

Distribution

Content of the `distribution` directory is not strictly limited. It can contain any files required for normal application functioning: jars, shared libraries and other resources like configs and properties, scripts and other text or binary data. However, to not overload installation scripts logic, it is recommended to keep structure of this directory minimally different from expected installation result.