EXTENDED ABSTRACT

Application Lifecycle Management for Smart Grid Use Cases in the Intelligent Secondary Substation

Stephan Cejka[1*], Florian Kintzler[1], Lisa Müllner[1], Felix Knorr[1], Marco Mittelsdorf[2] and Jörn Schumann[2]

*Correspondence:
stephan.cejka@siemens.com
[1]Siemens AG Österreich, Siemensstraße 90, 1210 Vienna, Austria
Full list of author information is available at the end of the article

Introduction

It is of utmost importance to provide mechanisms for the installation or update of software on devices in the field. This requirement is further increased since the transition to the Internet of Things (IoT) introduces a high number of affected field devices that are required to be maintained. Our previous projects in the Smart Grid and Smart Building domain led to the requirement for a device and application management mechanism [1–3]. Important components in the affected parts of the electric power grid are the secondary substations located on the borders between the medium and the low voltage grid to connect local commercial and residential users to the power grid. The transition from the traditional grid to the Smart Grid includes equipping those traditionally passively operated isolated secondary substations with improved computational power and communication abilities, easing interaction with them and reducing maintenance efforts and costs (intelligent secondary substation – iSSN). Within the LarGo! project, we are investigating processes for the large scale rollout of software applications for energy and grid management [4]. The vast number of application modules, as well as the high number of substations a distribution system operator (DSO) has to maintain requires to issue typical application lifecycle management tasks from the remote site without staff required on-site.

Requirements and Tasks

The main requirements for the iSSN's application lifecycle management are:

- R1 Scalable device management by a central control center
- R2 Automatic deployment of software components to the devices
- R3 Modular system to easily compose required features on demand
- R4 Module dependency management
- R5 Automatic updates and configuration

The system's architecture is expected to consist of the DSO's central backend and control system, responsible for the management of the field devices' software components. Many target devices are controlled by this backend system; they are connected with the backend by a communication channel on which messages and artifacts are exchanged. Target systems allow for modular applications, thus the number of concurrently executed applications may be high [2]. Listed application lifecycle tasks [1] are issued on this backend system:

- T1 Installation of a software module
- T2 Start of this software module
- T3 Stop of this software module

T4 Uninstallation of this software module
T5 Update of this (possibly running) software module
T6 Configuration of this (possibly running) software module
T7 Information on the current state of the software modules (e.g., health check)

A resulting simplified application lifecycle graph is shown in Figure 1. Note that, T5 basically is a sequence of stopping and uninstalling the old version, and installing and starting the new version (T3-T4-T1-T2, if the module is currently running, or T4-T1, if not) with the old persistent state reused, in contrast to T6 which should not require a restart. Transitions to a *failed* state could occur at any task; they are left out in the figure for simplicity reasons.

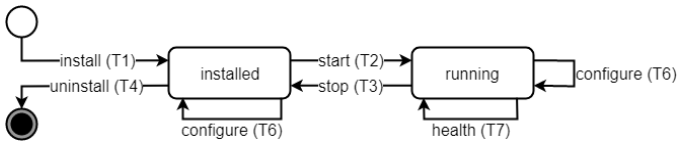

Figure 1 A simplified application lifecycle graph

State-of-the-art Evaluation and Implementation

The implementation of applications can differ depending on the use case (e.g., OSGi modules, Docker containers). As there exist no domain-specific solutions for software management, existing IoT solutions (Apache ACE[1], Apache Felix[2], Apache Karaf[3], balena[4], Eclipse hawkBit[5], Eclipse Virgo[6], Gridlink Application Framework Provisioning [1], SWUpdate[7], and cloud-provider solutions, such as AWS IoT Greengrass[8], and Azure IoT Edge[9]) were evaluated regarding the listed requirements and tasks. While the list is not exhaustive, it provides a good overview over the spectrum of currently available solutions. However, not all requirements and tasks can be fulfilled by the frameworks in evaluation: Most of the frameworks have only limited support of deployable components' types or they do not allow a central management of multiple devices. Furthermore, most of the frameworks show only limited support for the expected lifecycle tasks: an installation task usually already includes the start of the module, and modules cannot be stopped once running.

In result, the support for some of the defined main lifecycle tasks is limited. Thus, special challenges in the domain of Smart Grid application rollouts require a tailored solution. The architecture of a generic implementation for application lifecycle management in Industrial IoT (IIoT) use cases [3] is shown in Figure 2, including an optional *App Store*. In contrast to consumer IoT solutions, in IIoT use cases it needs

[1] https://ace.apache.org/
[2] https://felix.apache.org/
[3] https://karaf.apache.org/
[4] https://www.balena.io/
[5] https://www.eclipse.org/hawkbit/
[6] https://www.eclipse.org/virgo/
[7] https://github.com/sbabic/swupdate/
[8] https://aws.amazon.com/greengrass/
[9] https://azure.microsoft.com/services/iot-edge/

to be necessarily avoided that external systems have any access to field devices, especially since the App Store may not be within the sphere of the operator. Thus, purchased applications are first downloaded by the *Application Lifecycle Management Service (ALMS)* to the *Local Application Repository*, all situated within the operator's backend. The operator manages the several field devices by use of an *User Interface (UI)* on the backend side; there the enumerated application lifecycle tasks are issued. They are communicated to the *Application Lifecycle Management Agent (ALMA)* on the device, which executes specific shell scripts based on the file type and the task (Table 1). Those shell scripts can be implemented for any file type, thus the implementation shows to be very flexible and extendable.

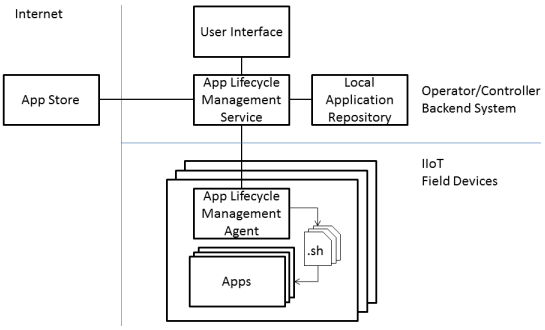

Figure 2 Generic Application Lifecycle Management Implementation

		Docker containers	Docker container compositions
Task	**Shell scripts**	`docker-◯.sh`	`compose-◯.sh`
install (T1)	`▽-install.sh`	`docker image load` `docker container create`	`docker-compose up --no-start`
start (T2)	`▽-start.sh`	`docker container start`	`docker-compose start`
stopp (T3)	`▽-stop.sh`	`docker container stop`	`docker-compose stop`
uninstall (T4)	`▽-uninstall.sh`	`docker container rm` `docker image rm`	`docker-compose down`

Table 1 ALMA calls shell scripts named `▽-◯.sh` (`▽`: file type, `◯`: task; e.g., `docker-install.sh`). Simplifications of the steps called by the according shell scripts are shown for Docker containers, and for compositions of multiple Docker containers.

For the main app-lifecycle tasks (T1–T4) – ⋆: install, start, stop, uninstall – the following five steps are executed:

1 The operator initiates the ⋆ task of an app using the UI on the backend site.
2 ALMS informs ALMA to ⋆ the app. Further steps may be executed by the ALMA; for example, for T1, ALMA downloads the app, and extracts it.
3 ALMA runs the use-case specific configured ⋆ shell script (cf. Table 1).
4 ALMA replies to the ALMS with a message indicating success or failure. In cases of failures, the *failed* state is entered.
5 The result of the ⋆ process is shown in the UI to the operator; for example, after T1, the app it is added to the list of installed apps.

Conclusion and Outlook

We enumerated several requirements and tasks an application lifecycle management in the Smart Grid domain needs to fulfill. Unfortunately, an evaluation of suitable-seeming customer-grade IoT tools showed to be not applicable for this purpose due

to not implementing all of the desired functions. We thus introduced a solution for the iSSN's application lifecycle management and tested it successfully. LarGo! furthermore includes the domain of Smart Buildings utilizing a buidling energy management system (BEMS). Work is ongoing for application lifecycle management on the OSGi-based development of the BEMS fulfilling the listed requirements and tasks. Management systems to roll out software to more devices cover different levels and areas of dependency management. However, some dependencies go beyond of what state-of-the-art software rollout systems support; for example, none of the analyzed systems is able to include knowledge about the physical environment and the functions of the software into software rollout planning and execution. We are thus currently working on a knowledge-based software management layer, utilizing knowledge about the grid's setup and its components for the planning process, including knowledge about the underlying software deployment processes themselves.

Funding
The presented work is conducted in the LarGo! project, funded by the joint programming initiative ERA-Net Smart Grids Plus with support from the European Union's Horizon 2020 research and innovation programme. On national level the work was funded and supported by the Austrian Climate and Energy Fund (KLIEN, ref. 857570), and by German BMWi (FKZ 0350012A).

Availability of data and materials
No additional data and materials are available.

Author's contributions
This paper is a joint work of all listed authors.

Competing interests
The authors declare that they have no competing interests.

Author details
[1]Siemens AG Österreich, Siemensstraße 90, 1210 Vienna, Austria. [2]Fraunhofer-Institut für Solare Energiesysteme ISE, Heidenhofstraße 2, 79110 Freiburg, Germany.

References

1. Faschang, M., Cejka, S., Stefan, M., Frischenschlager, A., Einfalt, A., Diwold, K., Pröstl Andrén, F., Strasser, T., Kupzog, F.: Provisioning, deployment, and operation of smart grid applications on substation level. Computer Science - Research and Development **32**(1), 117–130 (2017)
2. Cejka, S., Diwold, K., Frischenschlager, A., Lehninger, P.: Integrating Smart Building Energy Data into Smart Grid Applications in the Intelligent Secondary Substations. Journal of Electronic Science and Technology **16**(4), 291–303 (2018)
3. Gawron-Deutsch, T., Diwold, K., Cejka, S., Matschnig, M., Einfalt, A.: Industrial IoT für Smart Grid-Anwendungen im Feld. e & i Elektrotechnik und Informationstechnik **135**(3), 256–263 (2018)
4. Kintzler, F., Gawron-Deutsch, T., Cejka, S., Schulte, J., Uslar, M., Veith, E., Piatkowska, E., Smith, P., Kupzog, F., Sandberg, H., Chong, M., Umsonst, D., Mittelsdorf, M.: Large scale rollout of smart grid services. In: 2018 Global Internet of Things Summit (GIoTS'18) (2018)