SODALITE actors and use cases

Luciano Baresi, Elisabetta Di Nitto
Politecnico di Milano

The SODALITE vision is to support Digital Transformation of European Industry through (1) increasing design and runtime effectiveness of software-defined infrastructures, to ensure high-performance execution over dynamic heterogeneous execution environments; (2) increasing simplicity of modelling applications and infrastructures, to improve manageability, collaboration, and time to market.

Within this vision, SODALITE will provide application developers and infrastructure operators with tools that (a) abstract their application and infrastructure requirements to (b) enable simpler and faster development, deployment, operation, and execution of heterogeneous applications reflecting diverse circumstances over (c) heterogeneous, software-defined, high-performance, cloud infrastructures, with a particular focus on performance, quality, manageability, and reliability (quote from the grant agreement).

In particular, SODALITE is focusing on supporting the entire life cycle of the so-called Infrastructure as Code (IaC). IaC means limiting the need to manually provision resources, configuring them and deploying an application by offering to DevOps teams the possibility to code such tasks into proper scripts that are then executed by proper orchestrators, thus introducing a significant automation in the application life cycle.

If we can define IaC, it means that we can deal with such code as we do with traditional code. Therefore, activities concerning the design of IaC, its verification, its optimization, its dynamic evolution (i.e., the possibility of modifying, on the fly, the IaC associated to a software system so that its deployment and configuration can change) can be envisaged and have the potential to significantly improve the state of practice which is still relatively immature in this area. The purpose of SODALITE is exactly to contribute to this goal.

Following its path toward this objective, the SODALITE consortium has undertaken an iterative and disciplined analysis process that has lead to the identification of the main actors that will interact with the SODALITE system, as well as the use cases such system will address. These two aspects are summarized in Figure 1.

The SODALITE actors can be grouped into two families: the **users**, experts in charge of interacting with and exploiting SODALITE, and the **resources needed by the system** to carry out the different activities. As for users, we have:

- **Application Ops Experts (AOE).** They are in charge of operating the application and, as such, are in charge of all the aspects that refer to the deployment, execution, optimization and monitoring of the application. They are supposed to know the applications to execute and the requirements on both the deployment/execution environment and the quality of services they are interested in.
- **Resource Experts (RE).** They are in charge of dealing with the different resources required to deploy and execute the application. These persons are in charge of application component technologies, of cloud, HPC, and GPU-based computing infrastructures, or of middleware solutions for both storing data and allowing components to communicate.
- **Quality Experts (QE).** They are responsible for the quality of service both provided by the execution infrastructure and required by the executing application. Being part of the SODALITE ecosystem, they are in charge of offering libraries of patterns for addressing specific performance and quality problems in the SODALITE applications.

As for resources, they merely cover all the possible ones needed to operate applications on SODALITE. These resources can be specialized in:

- **Application components** are the executables the applications of interest are partitioned in. These components can be based on diverse technologies and come both as black-boxes and as complete packages, that is, the executables come with source code and with any other external artifact needed to compile, deploy, and execute them.
- **Execution platforms** provide the means to execute the different application components. They can be cloud based elements (e.g., virtual machines or containers), HPC infrastructures, or clusters of GPUs.
- **Middleware frameworks** provide the underlying glue and help both store the different data and artifacts and make the different elements communicate.

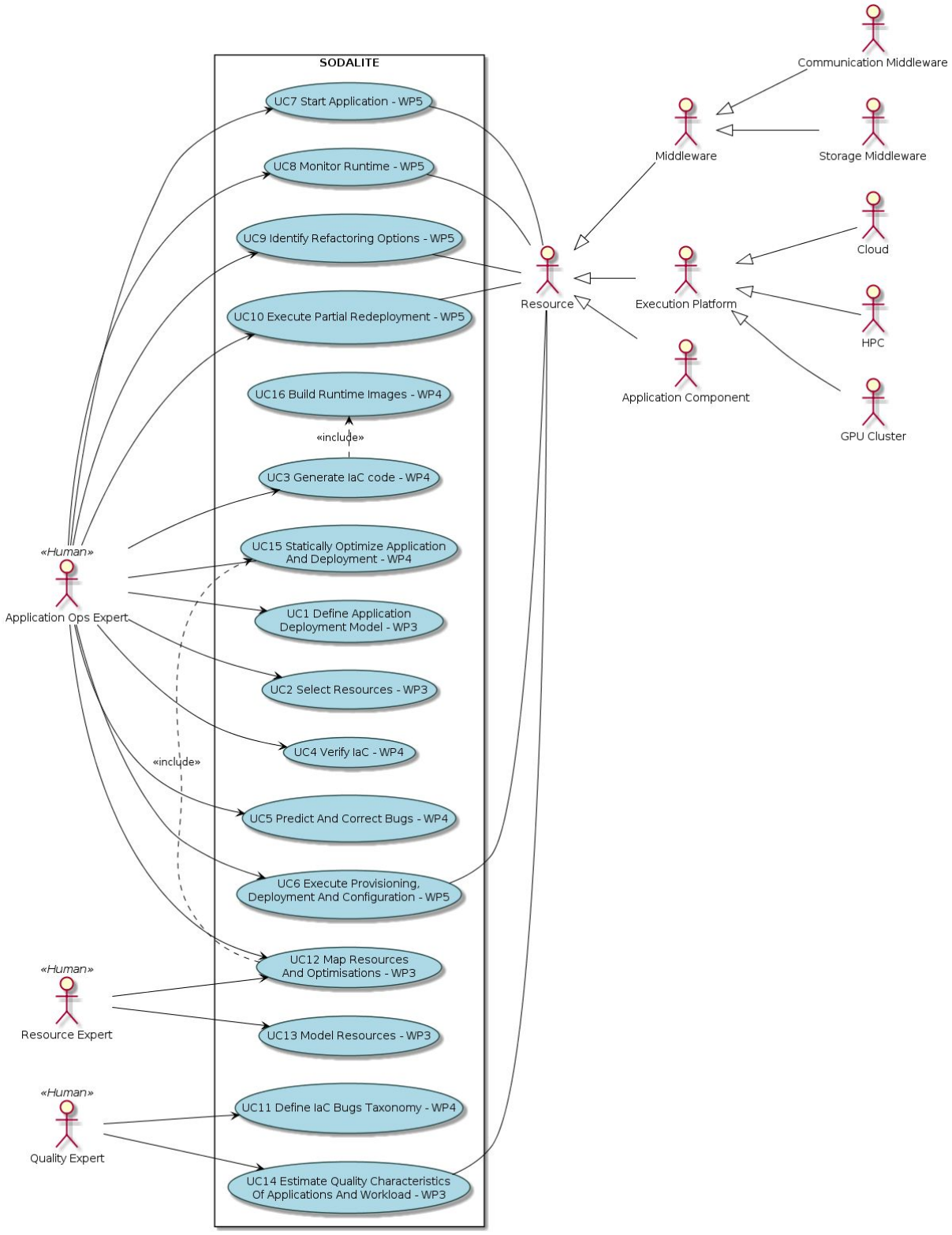

Figure 1: Use cases and actors.

The use cases in FIgure 1 reflect the main activities human actors can trigger or participate in as part of the life cycle management of IaC. More specifically:

- To make the SODALITE framework usable by AOEs, it must be populated with information concerning the resources that can be exploited at runtime. This requires modeling resources (UC13) and making them available, as part of the SODALITE Domain Specific Language, to AOEs. This activity is performed

by Resource Experts which are also in charge of mapping the modeled resources into specific optimization patterns (UC12).

- The Quality Expert defines a bug taxonomy for IaC (UC11) that helps AOEs in predicting bugs (UC5). Moreover, he/she experiments with application components and prototypes to estimate their quality characteristics (UC14).
- AOEs start their activity by defining an application deployment model (UC1). This model includes the main components of an application and any constraint or requirement on their deployment, configuration or execution. At this point they can either rely on the resources the SODALITE system would assign by default, or they could select specific resources (UC2). After this step, they are ready to trigger the automatic generation of IaC code (UC3) and its verification (UC4) as well as bug prediction and correction (UC5) and static optimization (UC15) aiming at improving application performance. Of course these activities may lead to some reiteration in the mentioned use cases until the point in which, as part of the IaC code generation, AOEs generate the needed runtime images (UC16). Then AOEs can trigger the execution of provisioning, configuration and deployment (UC6), start the application (UC7) and start monitoring the execution (UC8) with the purpose of checking that everything is working well and, in case of problems, of identifying possible refactoring and deployment improvement options (UC9). As a result of this identification, they can go back to the modeling and IaC generation/verification/optimization phases and, at this point, trigger a partial redeployment of the system (UC10).

The definition of all use cases has been an intense and joint effort within the consortium. This ensures these use cases are fully endorsed by all consortium members. The first use cases will be available by February 2020 and we will be very happy to share them with all interested stakeholders.