Proceedings of the 2020 Winter Simulation Conference
K.-H. Bae, B. Feng, S. Kim, S. Lazarova-Molnar, Z. Zheng, T. Roeder, and R. Thiesing, eds.

ArTIC-M&S: AN ARCHITECTURE FOR TOSCA-BASED INTER-CLOUD MODELING AND SIMULATION

Paolo Bocciarelli
Andrea D'Ambrogio

Department of Enterprise Engineering
University of Rome Tor Vergata
Via del Politecnico, 1
00133 Rome, Italy
{paolo.bocciarelli,dambro}@uniroma2.it

Umut Durak

German Aerospace Center (DLR)
Institute of Flight Systems
Lilienthalplatz 7
38108 Braunschweig, Germany
umut.durak@dlr.de

ABSTRACT

Modeling & Simulation (M&S) techniques have proven their effectiveness for several intended uses, from complex systems analysis to innovative training activities. The emerging M&S-as-a-Service (MSaaS) paradigm deals with the adoption of service-orientation and cloud computing to ease the development and provision of M&S applications. Due to its relevance for the military domain, the NATO MSG-164 is investigating how the MSaaS potential can be exploited to support NATO objectives. In this context, this work proposes *ArTIC-MS*, a MSaaS architecture that aims at investigating innovative approaches to ease the building of inter-cloud MSaaS applications. ArTIC-MS's main objective is to provide an effective interoperability among M&S services provided by different nations to seamlessly build complex MSaaS applications. Specifically, the work addresses the use of the TOSCA (Topology and Orchestration Specification for Cloud Applications) standard, and also discusses how ArTIC-MS may cope with the orchestration of M&S services available on non-TOSCA infrastructures.

1 INTRODUCTION

In the last decade, Modeling & Simulation (M&S) has gained great relevance in both academia and industry as an effective and valuable approach to support systems engineering activities (Gianni et al. 2014). A M&S approach encompasses the specification of a simulation model that describes a system or process, and the generation of a computer-based model implementation, which is eventually run so as to reproduce the behaviour of the addressed system or process (Sokolowski and Banks 2009). M&S approaches have proven their effectiveness in various application domains, ranging from the evaluation of systems design alternatives, to the execution of innovative training activities that can be too costly or risky to be carried out in a real environment. These approaches provide significant opportunities in the military domain also, in order to address various operational activities including training, decision support, concept development and experimentation (Ford et al. 2018). Emerging M&S-based technologies can be effectively used to develop and deliver innovative training services to provide soldiers and leaders with the required competencies (Caulkins et al. 2017). The M&S community may significantly benefit from the adoption of technologies such as cloud computing and service-oriented architecture (SOA), so to build and provide M&S applications through the composition of services available in the cloud, as addressed by the emerging *M&S-as-a-Service (MSaaS)* paradigm (Cayirci 2013; Siegfried et al. 2018). Due to its relevance for the NATO, the MSaaS paradigm is the subject of research activities undertaken by two Modeling and Simulation Groups: the recently completed MSG-136 and the ongoing MSG-164, which aim at investigating how MSaaS can be effectively adopted to support NATO's objectives and goals (van den Berg and Cramp 2018).

Unfortunately, the delivery of composite services in the cloud can be a non-trivial task. From an *application development perspective*, a set of component services have to be identified and deployed, with each component service providing an interface that meets the composite service requirements. Then, a choreography-based or an orchestration-based approach has to be implemented to handle the execution flow of the several component services. Differently, from an *execution platform perspective*, each component service has to be deployed onto an execution platform which, in turn, requires the integration of a set of hardware and software resources, such as computational nodes, containers, applications, networks connections, databases, and middleware.

In such a context, this work illustrates a contribution that has been partially supported by a research project that aims at investigating innovative architectures to ease the building of inter-cloud MSaaS applications in the defence sector. Specifically, this paper introduces **ArTIC-MS**, an **Ar**chitecture for **T**OSCA-based **I**nter **C**loud **M**odeling and **S**imulation. ArTIC-MS is a conceptual architecture that exploits the TOSCA (Topology and Orchestration Specification for Cloud Applications) standard (OASIS 2019a) to show how advanced M&S applications can be built by integrating simulation components provided by different partners, and deployed onto heterogeneous cloud infrastructures, in order to maximise the reuse of existing components.

Since interoperability is the main concern of ArTIC-MS, this paper also investigates how the proposed architecture may cope with the orchestration of M&S services available on non-TOSCA infrastructures. In this respect, the paper illustrates a case study concerning the integration of ArTIC-MS with OCEAN (Biagini et al. 2017), a MSaaS platform based on the open-source OpenStack cloud infrastructure (Openstack Foundation 2019e).

The rest of this paper is structured as follows. Section 2 reviews existing literature and highlights the novelty of the proposed contribution. Section 3 briefly summarizes the TOSCA standard. Section 4 illustrates the ArTIC-MS rationale and architecture, and Section 6 discusses a real-world application dealing with the use of ArTIC-MS in an existing MSaaS platform. Finally, Section 7 gives concluding remarks.

2 RELATED WORK

As pointed out in (Tolk 2013), the adoption of simulation-based techniques in the military domain has a long history. In this context, simulation has been widely used to support different operational needs, ranging from strategic games, computer-assisted exercises (CAX), and training (Loper et al. 2012; Tolk and Mittal 2014; Lorenz et al. 1997; Straßburger et al. 1998). The primary role of M&S in the military domain has been underlined in (Siegfried et al. 2014), where M&S is recognized as a key enabler for the delivery of capabilities to NATO in training, analysis and decision making. This contribution also argues how the adoption of a MSaaS paradigm, e.g., the introduction of service orientation for building M&S applications, may be a relevant opportunity to better utilize M&S capabilities in support of NATO critical needs.

In this respect, the main objective of this paper is the definition of a MSaaS-based architecture that aims at providing an effective tool to support NATO and allied nations in the discovery, deployment, integration and orchestration of M&S services for building complex simulation applications.

As regards the development of MSaaS platforms, several contributions have been proposed that address such an issue from different perspectives (Tolk 2013; Tolk et al. 2013; Hannay et al. 2020; Ford et al. 2018). In (Tolk 2013; Tolk et al. 2013), interoperability and composability have been identified as two of the most relevant challenges for M&S systems. While interoperability is defined as the ability to exchange data among simulation components, composability emphasizes the need of a conceptual alignment of data among the components part of a simulation system. In this regard, the main objective of the proposed architecture is twofold, namely the provisioning of an approach for the specification of the simulation system conceptual model, so to identify the available existing M&S components, and the use of the TOSCA standard for describing both the M&S components and the composed simulation, in order to address interoperability issues.

In (Hannay et al. 2020), it is argued that MSaaS-based applications should be easily composed by integrating loosely coupled shared components (in other words, simulation services), in a cloud-based environment. The pillars of the MSaaS ecosystem have been discussed in (Ford et al. 2018). According to the proposed architecture, a MSaaS system should be composed of M&S Services, which are the building blocks of simulation applications, Registries and Repositories, containing M&S Services descriptions and implementations, respectively, Processes, which define how services are discovered, composed, deployed and executed, an Infrastructure, which describes the Simulation Environment and, finally, a Portal, the entry point to start the MSaaS process. Such contributions have inspired the essential building blocks at the basis of the ArTIC-MS's architectural design, which also introduces a step forward by addressing interoperability in case of inter-cloud service composition.

Finally, existing literature also provides relevant examples of already available MSaaS platforms, such as CloudSME (Taylor et al. 2018), Simulation Platform (Yamazaki et al. 2011), and OCEAN (Biagini et al. 2017). Due to their architecture and rationale CloudSME and Simulation Platform are largely different form ArTIC-MS. CloudSME (Taylor et al. 2018) is a multi-cloud platform for developing and executing commercial cloud-based simulations and its primary target audience includes commercial software vendors and consultant companies in the IT domain, as well as Small and Medium-sized Enterprises (SMEs).

Simulation Platform (Yamazaki et al. 2011) consists of a cloud of virtual machines (VMs) upon which a GNU/Linux OS runs. Various software including scientific software, compilers, libraries, and simulators are pre-installed on each VM. The platform allows users to request the assignment of a set of VMs to build and run a scientific simulation, according to the specific requirements.

Differently, OCEAN (Biagini et al. 2017) is a MSaaS platform based on Openstack and specifically designed and developed for supporting training and simulation-based exercises in the defence sector. Similarly to the ArTIC-MS platform, OCEAN's architecture includes a cloud infrastructure, a service repository and a portal which allows users to discover, select, compose and deploy M&S components. Moreover, it also addresses the same target audience. The main difference is that OCEAN is not specifically designed to support inter-cloud interoperability. Its orchestration engine (i.e., the Openstack HEAT (Openstack Foundation 2019a)) requires the use of an implementation-specific technology, namely HOT (HEAT Orchestration Template) (Openstack Foundation 2019d), for the description of a simulation application. Differently, ArTIC-MS makes use of the TOSCA standard. Due to the relevance of OCEAN for the NATO and allied nations, this paper also discusses how ArTIC and OCEAN can effectively cooperate.

Finally, the architecture proposed in (Bocciarelli et al. 2018) exploits model-driven techniques to automate the application-level orchestration of M&S services according to an abstract composite service specified by use of different notations (e.g., UML, BPMN, etc.). As further detailed in Section 4, ArTIC-MS deals with the infrastructure-level orchestration, e.g., the execution of the actions required for deploying and executing the several M&S services onto the required execution platform.

3 TOSCA OVERVIEW

The description of a cloud application's topology and its deployment and configuration is a complex task that cloud vendors address by adopting different approaches and technologies, such as Amazon AWS CloudFormation (Amazon 2019) or HOT (Openstack Foundation 2019d).

In this context, in order to pursue the harmonization of existing approaches, the Organization for the Advancement of Structured Information Standards (OASIS) has released the Topology and Orchestration Specification for Cloud Applications (TOSCA), a standard language for describing cloud-based service orchestrations (OASIS 2019a; OASIS 2019b).

TOSCA defines a YAML-based specification for describing an IT service in terms of both its computing infrastructure and the required procedures for deploying, instantiating, executing and managing the service. The standard also specifies a packaged file format, namely CSAR (Cloud Service Archive), which allows one to store in the same package the YAML service description and the set of software artifacts (e.g., OS virtual images, libraries, scripts, DMBS installation files, middleware, application software in executable

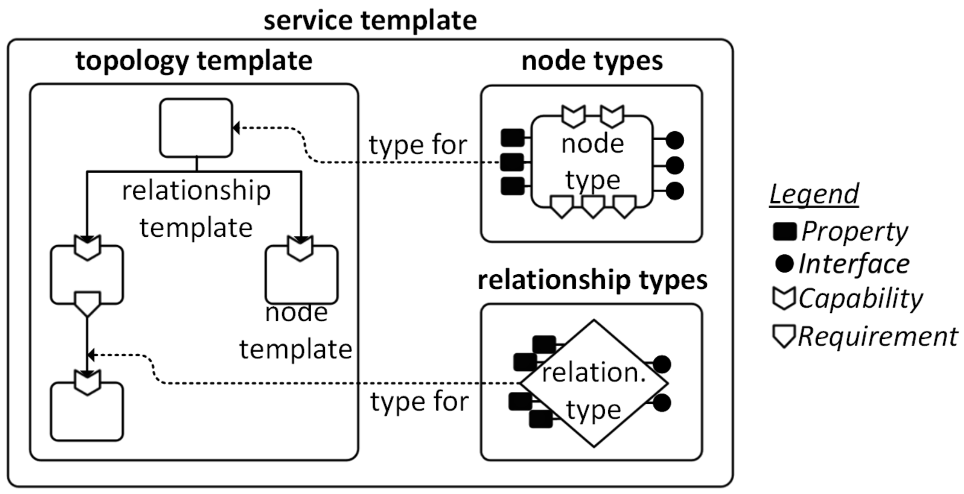

Figure 1: Logical structure of a TOSCA Service Template.

form, etc.) that actually implement the service. The CSAR package is given as input to a *TOSCA Engine*, which is responsible for processing the YAML description so as to oversee the deployment of the several artifacts composing the IT service and to orchestrate the execution of the related managing procedures.

In TOSCA a *service* is specified in terms of a *Service Template*, which is a YAML description that completely specifies the service structure and its related characteristics. A Service Template is composed by different elements, as shown in Figure 1.

A *Topology Template* is the essential element of a service template as it describes the structure of the service in terms of its building blocks. The service structure is specified by use of a direct graph in which nodes represent the building blocks of a service (e.g., servers, network interfaces, virtual images, databases, etc.), and edges represent the relationships between nodes (e.g., deployment relationship, connection relationship, etc.). According to a hierarchical structure, TOSCA nodes and relationships are, in turn, further specified by use of *Node Templates* and *Relationship Templates*, respectively. Moreover, a TOSCA *Service Template* also allows the specification of *Lifecyle Operations*, i.e., the actions that a TOSCA engine performs to deploy, execute, handle and undeploy nodes and relationships.

The use of TOSCA allows the specification of vendor-agnostic service orchestrations. In this paper contribution, TOSCA is used as the reference notation for specifying reusable and interoperable service descriptions.

4 ArTIC-MS CONCEPTUAL ARCHITECTURE

ArTIC-MS deals with the concept of *orchestration*, which is a term that needs to clearly defined, being applied to different kind of activities. A MSaaS application is built by integrating different M&S services. The *application orchestration* refers to methods and tools used to coordinate the execution flow and the information exchange among those services, so as to meet the functional requirements of the MSaaS application. Differently, as each M&S service has to be deployed on top of a given execution platform (which consists of computing nodes, system and application software, databases, network connections, etc.), the term *infrastructural orchestration* includes those activities needed to set up the execution platform, and deploy, configure and eventually start the required M&S services.

This work focuses on the *infrastructural orchestration*, and exploits the TOSCA standard to improve portability and interoperability of M&S services. This is achieved by specifying a YAML-based and vendor-independent orchestration description, which can be automatically processed by any TOSCA-compliant engine.

The ArTIC-MS conceptual architecture is shown in Figure 2. The main objective is to ensure the highest degree of interoperability among services provided by different partners, which potentially use different underlying technologies and cloud infrastructures. This scenario is referred to as the *inter-cloud service orchestration*.

In order to properly address the inter-cloud service orchestration scenario, ArTIC-MS is founded on the following assumptions:

- *Service Descriptions*: to ensure the interoperable orchestration of services implemented by use of different technologies, each partner shall provide an agnostic and technology-independent service description. ArTIC-MS assumes service descriptions defined in terms of:
  - *Metadata*: used to describe services and to allow users to identify the most suitable services that are to be integrated in the MSaaS application. Metadata are stored in a *Service Registry*;
  - *Infrastructural Orchestration Description*: the information required to deploy and execute the service, as stored in a *Service Repository*. Even though ArTIC-MS assumes that such a description is provided as a TOSCA CSAR package, Section 6 discusses the case of integrating non-TOSCA compliant services.
- *Service Discovery*: in an inter-cloud orchestration scenario, users must be provided with a discovery feature in order to retrieve services available on different cloud platforms, each under the responsibility of a different partner. In this respect it is assumed that the Service Registry provides an API for implementing inter-cloud service discovery.
- *Service Availability*: each partner is responsible of making available a given set of MSaaS services which, in turn, other partners can retrieve and integrate to build more complex MSaaS applications. In this respect, each service can be provided by a partner in two different configurations, according to the preferred business model:
  - *running services*: a partner might provide a running service which is deployed and configured under the responsibility of the providing partner. In such a case, the service metadata includes a service *endpoint* (e.g, an URI) and an interface description (e.g., by use of WSDL);
  - *deployable services*: a partner might provide an instance of the service specified in terms of artifacts and an infrastructural orchestration description (e.g, a TOSCA CSAR package). In such a case, the service metadata include a reference to the Service Repository that stores the CSAR package.

It should be underlined that it is not required that all partners adopt the whole ArTIC-MS platform as shown in Figure 2. In an inter-cloud orchestration scenario it is unrealistic to impose the adoption of a completely specified application platform. The main idea behind ArTIC-MS is that each partner shall just provide service descriptions and discovery features.

5 ArTIC-MS USE CASES

This section describes the role and responsibilities of ArTIC-MS users when building and deploying a MSaaS application resulting from the orchestration of a set of available M&S services.

The scenarios discussed in the following sections involves the following users:

- *Simulation Expert*: responsible for the elicitation and specification of MSaaS application requirements.
- *Integrator*: responsible for the MSaaS Application development. Specifically, the Integrator deals with the identification and composition of the required M&S services, in order to satisfy the MSaaS application requirements provided by the Simulation Expert. The Integrator skills also include knowledge of the TOSCA standard, which is used to specify YAML-based templates.
- *End User*: the final user of the simulation experiment.

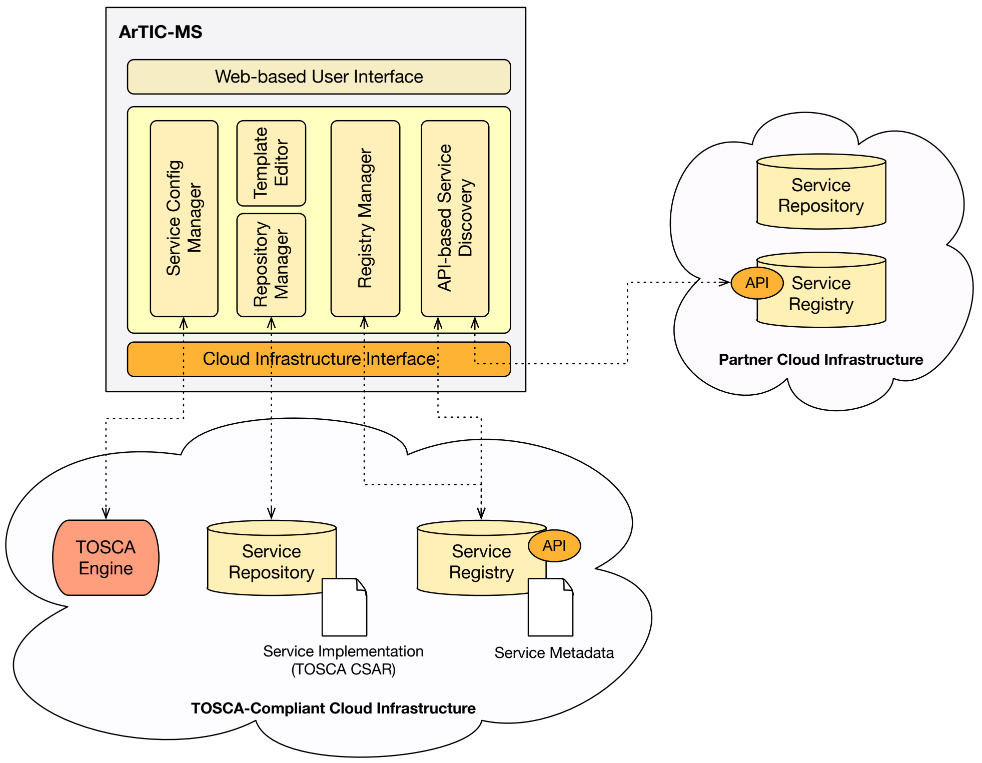

Figure 2: Conceptual Architecture of ArTIC-MS.

- *ArTIC-MS Administrato*r: the user who possesses the required skills for managing the ArTIC-MS platform. Its main task is to provide other users with the required environment for building, executing and monitoring simulation experiments.

The scenario proposed in this section addresses the development of MSaaS applications composed by M&S simulation services available in the cloud infrastructure of different partners. As discussed in 4, the underlying technology adopted by the cloud infrastructure of each partner is not required to satisfy any assumptions. Moreover, it is also assumed that each partner's infrastructure includes a metadata-based Service Registry which, in turn, provides an API-based discovery interface. Finally, the provided services do not need to satisfy any technology requirements and are made available either as running services (via URI endpoints) or as deployable services packaged in a CSAR file. The development of MSaaS applications is based on the follwoing workflow:

1. the Integrator interacts with the Simulation Expert to retrieve the objectives and requirements of the given simulation;
2. the Integrator logs into ArTIC-MS and executes a query on the Service Registry to identify a MSaaS application, i.e., an existing composition of available M&S services that meets the given objectives and requirements. Thanks to the API-based interface provided by the various service registries, the Integrator is allowed to discover services provided by any partner;
3. if an existing MSaaS applications is found, then:
   (a) the related CSAR package is retrieved from the Service Repository;
   (b) the Integrator deploys and configures the CSAR package and configures the required parameters of the various M&S services;
   (c) the simulation is ready to be used by the End User.
4. else (i.e., if a suitable existing MSaaS application is not found):

(a) the Integrator executes a query on the Service Registry to discover a set of candidate M&S services. In the most general case, the Integrator identifies N services, where N_L services are available in the local infrastructure, N_E services are provided by remote partners via an URI endpoint and an interface specification (e.g., WSDL, etc.), and finally N_R services are available from remote service repositories as CSAR packages, being $N = N_L + N_E + N_R$;
(b) the Integrator retrieves from the local Service Repository the N_L locally available services;
(c) the Integrator retrieves from remote service repositories the identified set of N_R remote CSAR packages;
(d) the Integrator uses the ArTIC-MS visual interface to create a template which describes the given MSaaS application, by composing the retrieved CSAR packages;
(e) the Integrator configures the required parameters of the various M&S services;
(f) the Integrator configures their composition, so to make it possible invoking the operations provided by the endpoints of remote M&S services;
(g) the Integrator gives the template as input to the TOSCA engine, so to carry out the automated deployment of the simulation;

5. The MSaaS application is ready to be used by the End User.
6. the Integrator can optionally save the so obtained MSaaS application as a new composite simulation service by i) storing in the Service Repository the CSAR file that wraps the YAML service template together with the required artifacts, and ii) updating the Service Registry with the required set of service metadata.

For the sake of clarity, it should be noted that the TOSCA engine addresses the infrastructural orchestration of only those services available as CSAR packages. The integration of the remote running services via URI endpoints is addressed by the *application orchestration*, which is not discussed in this work. In other words, each simulation component, during its execution and according to the related business logic, invokes the given remote services by sending messages to the correct endpoint, according to the remote service's interface specification. In this case, the Integrator's task is limited to the configuration of the orchestration template, in order to make each local service able to reach the URI of the given remote services.

6 ArTIC-MS IN PRACTICE: INTEROPERABILITY WITH A NON-TOSCA PLATFORM

The most important objective of the ArTIC-MS's architecture is to ensure interoperability among heterogeneous services provided by different partners, without assuming either a specific cloud implementation or one or more technological constraints for the implementation of each M&S service. In this respect ArTIC-MS is founded on i) the availability of registries that provide an API-based discovery interface and ii) the availability of services in terms of TOSCA CSAR packages.

An additional relevant objective of this paper contribution is to investigate how to make ArTIC-MS interoperable with any other non-TOSCA infrastructure. Thus, as mentioned in Section 1, this work specifically addresses the OCEAN MSaaS platform.

OCEAN includes a visual interface to specify templates by composing the several elements (software artifacts, computing nodes, network connections, etc.) available from a dedicated toolbox. OCEAN includes also a feature for searching existing services that can be used as building blocks of more complex services. As OCEAN is founded on Openstack, it makes use of an orchestration engine, namely HEAT, to parse topology templates specified in the YAML-based description format named HOT.

The objective of this section is to outline how M&S services and MSaaS applications available from ArTIC-MS can be easily reused in OCEAN.

In order to cope with HOT-based service descriptions, *HEAT-Translator* (Openstack Foundation 2019b), the Openstack component for translating TOSCA templates to semantically equivalent HOT templates, has been evaluated.

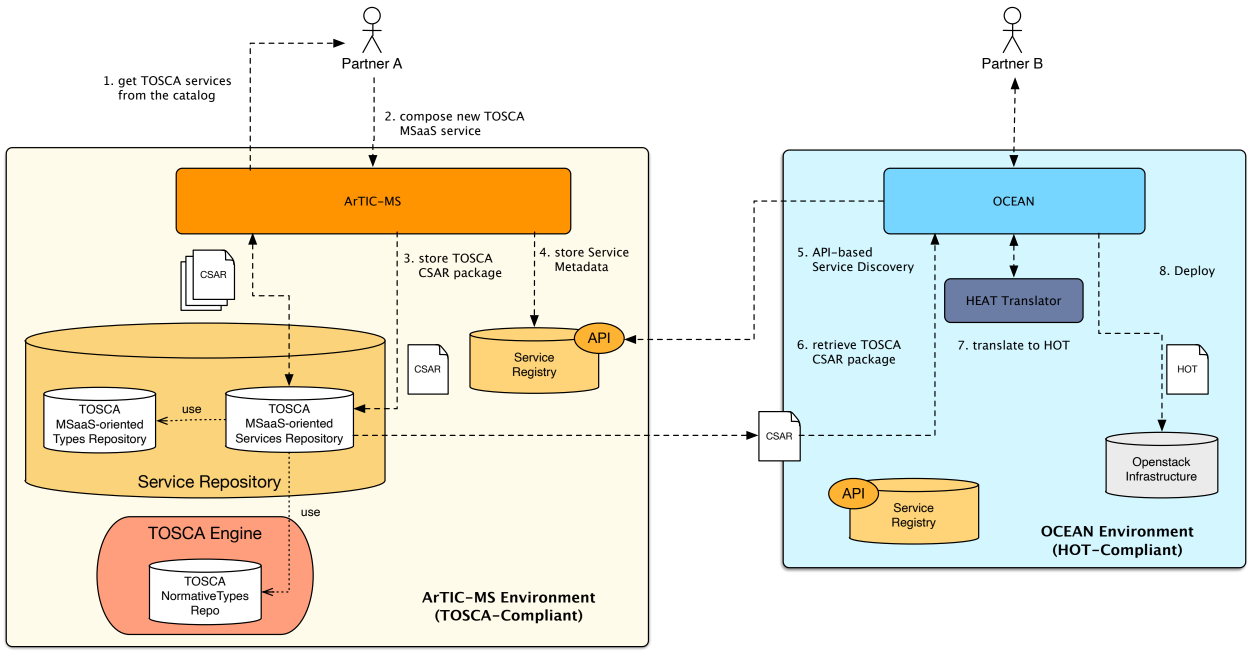
Partner A
1. get TOSCA services from the catalog
2. compose new TOSCA MSaaS service
ArTIC-MS
3. store TOSCA CSAR package
4. store Service Metadata
CSAR
API
Service Registry
TOSCA MSaaS-oriented Types Repository
use
TOSCA MSaaS-oriented Services Repository
Service Repository
use
TOSCA Engine
TOSCA NormativeTypes Repo
ArTIC-MS Environment (TOSCA-Compliant)
Partner B
OCEAN
5. API-based Service Discovery
HEAT Translator
8. Deploy
6. retrieve TOSCA CSAR package
7. translate to HOT
HOT
Openstack Infrastructure
OCEAN Environment (HOT-Compliant)

Figure 3: Integration of ArTIC-MS and OCEAN.

HEAT-Translator is a command-line tool that allows the generation of HOT templates by converting non-HOT service description templates. Currently, HEAT-Translator only supports the conversion of TOSCA templates to HOT ones, but according to the referred documentation, it is designed to be easily extended for being used with any other input format. In addition, ongoing work is claimed to be planned for implementing the transformation in the opposite direction (i.e., from HOT to TOSCA), as well as transformations from additional orchestration languages. The translator architecture consists of two modules, the *TOSCA Parser* and the *HOT Generator*. The TOSCA parser analyzes the input template in order to identify `Types` and `Nodes` element in the source model, so to build a graph representing the TOSCA topology. Then, the *HOT Generator* is responsible for mapping elements in the TOSCA graph to HOT concepts (*MAP* sub module) and generating a corresponding HOT template that preserves the TOSCA semantics (*Generate* sub module).

The translator is developed in Python and, even though part of the Openstack project, is being developed as an independent and self-contained open source project (Openstack Foundation 2019c).

The next subsection illustrates the proposed architectural solution to cope with the interoperability between ArTIC-MS and OCEAN by use of the HEAT-Translator.

6.1 Integrated Architecture

Figure 3 illustrates the conceptual model that describes how to achieve the required interoperability between ArTIC-MS and OCEAN.

In the proposed scenario, *Partner A* is responsible for creating M&S services (and also MSaaS applications exported as CSAR complex services, as discussed in Section 5) by using the ArTIC-MS platform (left part of the figure). The MSaaS application is built by integrating a set of M&S services available in the catalog. The right part of the figure shows the OCEAN infrastructure, in which Partner B aims at creating a MSaaS application orchestrating some local components with services provided by Partner A.

The scenario is specified as follows:

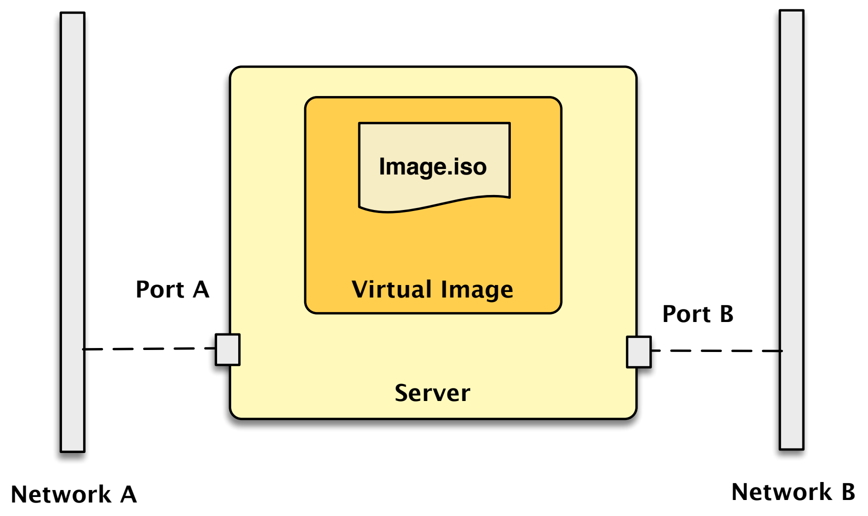

Figure 4: Service Topology Conceptual Model.

Precondition: *the user Partner A created a set of M&S services and MSaaS applications, which are available as CSAR packages in ArTIC-MS.*

1. the user *Partner B* logs into the OCEAN platform and executes a query to the Service Registries in order to identify the available M&S services. As the query makes use of the API interfaces provided by each partner's Registry, *Partner B* is able to identify the TOSCA-based service provided by *Partner A*;
2. *Partner B* identifies the set of needed services;
3. *Partner B* retrieves the required set of locally available services from the local repository, and also retrieves the CSAR descriptions of M&S services provided by Partner A through the ArTIC-MS Service Repository;
4. *Partner B* translates the TOSCA template contained into the CSAR package to a HOT template;
5. the generated HOT template can be used to build and deploy the required MSaaS application in OCEAN, by integrating local services which other services built, packaged and made available by different infrastructures.

The proposed integrated infrastructure has been tested using a simple TOSCA based service template which has been translated into a HOT template and deployed onto OCEAN.

The conceptual model of the service template is shown in Figure 4. The service's main artifact is an `.iso` virtual image implementing the application that has to be deployed on top of a virtual machine, which in turn is executed on a server connected to two different networks.

The TOSCA-based YAML service template is provided in Listing 6.1, while the related HOT-based YAML template is provided in Listing 6.1.

7 CONCLUSIONS

This paper has introduced ArTIC-MS, a MSaaS platform for supporting the development of inter-cloud M&S applications. ArTIC-MS exploits TOSCA, a standard for specifying vendor-independent orchestrations of services and applications in the Cloud.

The ArTIC-MS's conceptual architecture has been designed to ensure and maximise the interoperable reuse of M&S services provided by partners using different underlying Cloud infrastructure. The proposed

```
tosca_definitions_version: tosca_simple_yaml_1_0
description: Server connected to 2 networks
topology_template:
...
node_templates:
        virtual_machine:
                type: SoftwareComponent
                artifacts:
                vm_image:
                        file: images/vm_image.iso
                        type: tosca.artifacts.Deployment.Image.VM.iso
                requirements:
                        host: server
        interfaces:
                Standard:
                        create: vm_image
        server:
                type: tsca.nodes.Compute
                capabilities:
                host:
                        properties:
                        disk_size: 10 GB
                        num_cpus: 1
                        mem_size: 4096 MB
        network1:
                type: tosca.nodes.network.Network
                properties:
                cidr: {get_input: cidr1}
                network_name: net1
        ...
```

Figure 5: Input TOSCA Service Template (fragment).

architecture has been developed to be used in the defence sector but is flexible enough to be used in any other M&S-based application domain.

In order to assess how M&S services provided by ArTIC-MS and other non-TOSCA MSaaS platforms can be effectively integrated in a heterogeneous MSaaS application, this work has experimented with the Openstack HEAT Translator, which in the current implementation only provides the automated translation of TOSCA templates into HOT ones.

The analysis of such a concrete scenario has confirmed the feasibility of the proposed approach and has also revealed some issues that deserve to be further discussed and addressed. Indeed, the relevant standardization effort carried out by the well known OASIS (Organization for the Advancement of Structured Information Standards) consortium, which led to the publication of the TOSCA specification, didn't result in a full acceptance and support by important players in the cloud marketplace (e.g., Amazon, Google, Microsoft). This is probably due to the lack of a clear roadmap and to some changes that have been recently introduced in the standard, both in terms of the template specification language, which shifted from XML to YAML, and in terms of the TOSCA conceptual model, which has been slightly revised. As a consequence, any effort dealing with the specification of interoperable MSaaS platforms should appropriately address compliance with alternative orchestration standards and technologies.

In this respect, ongoing work includes the analysis and implementation of a full fledged adaptor component, based on the current implementation of the HEAT Translator, in order to provide the translation of HOT templates to TOSCA ones and vice versa. Finally, in order to ensure compliance with other cloud-based infrastructures, the adaptor is being designed to be easily extended, so to make ArTIC-MS compliant with alternative template specification languages and standards.

```
heat_template_version: 2013-05-23

description: >
  Server connected to 2 networks
...
resources:
  virtual_machine_create_deploy:
    type: OS::Heat::SoftwareDeployment
    properties:
      config:
        get_resource: virtual_machine_create_config
      server:
        get_resource: server
  network1:
    type: OS::Neutron::Net
    properties:
      name: net1
  server:
    type: OS::Nova::Server
    properties:
      flavor: m1.medium
      user_data_format: SOFTWARE_CONFIG
      networks:
      - port: { get_resource: port1 }
      - port: { get_resource: port2 }
  ...
```

Figure 6: Output HOT Service Template (fragment).

REFERENCES

Amazon 2019. *AWS CloudFormation*. https://aws.amazon.com/cloudformation.

Biagini, M., F. Corona, M. Picollo, M. L. Grotta, J. Jones, A. Scaccianoce, A. Mursia, C. Faillace, and D. Prochazka. 2017. "Modeling and Simulation as a Service from End User Perspective". In *Proceedings of the Interservice/Industry Training, Simulation and Education Conference (I/ITSEC)*.

Bocciarelli, P., A. D'Ambrogio, A. Giglio, and E. Paglia. 2018. "Model transformation services for MSaaS platforms". In *Proceedings of the Model-driven Approaches for Simulation Engineering Symposium*. Society for Computer Simulation International.

Caulkins, B., B. Goldiez, P. Wiegand, G. Martin, P. Dumanoir, and T. Torres. 2017. "Emerging Network and Architecture Technology Enhancements to Support Future Training Environments". In *Proceedings of the Interservice/Industry Training, Simulation, and Education Conference (I/ITSEC)*.

Cayirci, E. 2013. "Modeling and Simulation as a Cloud Service: a Survey". In *Proceedings of the 2013 Winter Simulation Conference*, edited by R. Pasupathy, S.-H. Kim, A. Tolk, R. Hill, and M. Kuhl, 389–400. Washington, D.C.: Institute of Electrical and Electronics Engineers, Inc.

Ford, K., B. Mason, L. Simpson, and L. Stewart. 2018. "AIMS Approach to Providing Modelling & Simulation as a Service". In *ITEC 2018*.

Gianni, D., A. D'Ambrogio, and A. Tolk. (Eds.) 2014. *Modeling and Simulation-Based Systems Engineering Handbook*. CRC Press.

Hannay, J. E., T. van den Berg, S. Gallant, and K. Gupton. 2020. "Modeling and Simulation as a Service infrastructure capabilities for discovery, composition and execution of simulation services". *The Journal of Defense Modeling and Simulation*:1548512919896855.

Loper, M. L., C. Turnitsa, and A. Tolk. 2012. "History of combat modeling and distributed simulation". *Engineering principles of combat modeling and distributed simulation*:331–355.

Lorenz, P., T. J. Schriber, H. Dorwarth, and K.-C. Ritter. 1997. "Towards a Web Based Simulation Environment". In *Proceedings of the 1997 Winter Simulation Conference*, edited by S. Andradóttir, K. J. Healy, D. H. Withers, and B. L. Nelson, 1338–1344. Atlanta, GA: Institute of Electrical and Electronics Engineers, Inc.

OASIS 2019a. *OASIS Topology and Orchestration Specification for Cloud Applications (TOSCA)*. https://www.oasis-open.org/committees/tosca.

OASIS 2019b. *TOSCA Simple Profile in YAML v.1.2*. http://docs.oasis-open.org/tosca/TOSCA-Simple-Profile-YAML/v1.2/TOSCA-Simple-Profile-YAML-v1.2.pdf.

Openstack Foundation 2019a. *HEAT - Openstack Orchestration Engine*. https://docs.openstack.org/heat/train/.

Openstack Foundation 2019b. *HEAT-Translator*. https://wiki.openstack.org/wiki/Heat-Translator/.

Openstack Foundation 2019c. *HEAT-Translator GitHub Repository*. https://github.com/openstack/heat-translator.

Openstack Foundation 2019d. *HOT - Heat Orchestration Template*. https://docs.openstack.org/heat/train/template_guide/hot_spec.html.

Openstack Foundation 2019e. *The OpenStack Platform*. https://www.openstack.org/.

Siegfried, R., J. Lloyd, and T. V. D. Berg. 2018. "A New Reality: Modelling & Simulation as a Service". *Journal of Cyber Security and Information Systems* 6(3):18 – 29.

Siegfried, R., T. van den Berg, A. Cramp, and W. Huiskamp. 2014. *M&S as a Service: Expectations and Challenges*. Orlando, FL: SISO.

Sokolowski, J., and C. Banks. 2009. *Principles of Modeling and Simulation: A Multidisciplinary Approach*. Wiley.

Straßburger, S., T. Schulze, U. Klein, and J. O. Henriksen. 1998. "Internet-based Simulation using off-the-shelf Simulation Tools and HLA". In *Proceedings of the 1998 Winter Simulation Conference*, edited by D. J. Medeiros, E. F. Watson, J. S. Carson, and M. S. Manivannan, 1669–1676. Washington, DC: Institute of Electrical and Electronics Engineers, Inc.

Taylor, S. J., T. Kiss, A. Anagnostou, G. Terstyanszky, P. Kacsuk, J. Costes, and N. Fantini. 2018. "The CloudSME simulation platform and its applications: A generic multi-cloud platform for developing and executing commercial cloud-based simulations". *Future Generation Computer Systems* 88:524 – 539.

Tolk, A. 2013. "Interoperability, composability, and their implications for distributed simulation: Towards mathematical foundations of simulation interoperability". In *2013 IEEE/ACM 17th International Symposium on Distributed Simulation and Real Time Applications*, 3–9. IEEE.

Tolk, A., S. Y. Diallo, J. J. Padilla, and H. Herencia-Zapana. 2013. "Reference modelling in support of M&S—foundations and applications". *Journal of Simulation* 7(2):69–82.

Tolk, A., and S. Mittal. 2014. "A Necessary Paradigm Change to Enable Composable Cloud-based M&S Services". In *Proceedings of the 2014 Winter Simulation Conference*, edited by A. Tolk, L. Yilmaz, S. Y. Diallo, I. O. Ryzhov, S. Buckley, and J. A. Miller, 356–366. Savannah, GA: Institute of Electrical and Electronics Engineers, Inc.

van den Berg, T., and A. Cramp. 2018. "Container orchestration environments for M&S".

Yamazaki, T., H. Ikeno, Y. Okumura, S. Satoh, Y. Kamiyama, Y. Hirata, K. Inagaki, A. Ishihara, T. Kannon, and S. Usui. 2011. "Simulation Platform: A cloud-based online simulation environment". *Neural Networks* 24(7):693 – 698.

AUTHOR BIOGRAPHIES

PAOLO BOCCIARELLI is a postdoc researcher at the Department of Enterprise Engineering of the University of Roma Tor Vergata (Italy). His research interests include software and systems engineering, business process management, model-driven development and distributed simulation. His email address is paolo.bocciarelli@uniroma2.it.

ANDREA D'AMBROGIO is associate professor of systems and software engineering at the Department of Enterprise Engineering of the University of Roma Tor Vergata (Italy). His research interests are in the areas of system performance and dependability engineering, model-driven systems and software engineering, business process management, and distributed simulation. His email address is dambro@uniroma2.it.

UMUT DURAK is a Research Scientist in the Institute of Flight Systems at the German Aerospace Center (DLR). He is also an Adjunct Faculty (Privatdozent) in the Department of Informatics at Clausthal University of Technology. His research interests concentrate around modelling and simulation based development of airborne systems. His email address is umut.durak@dlr.de .