Distributed Programming II

A.Y. 2017/18

Special Projects Proposals

This is the list of proposed special projects for DP II. If interested in applying, send an e-mail message to d1943@studenti.polito.it by October 29, 2017 with subject “Special Project Application”. The message must include the following information:

- The id of the project(s) you are interested in (in case you specify more than one project, please sort them in order of preference)
- Your CV including the transcript of your exams and the marks you got

The special projects will be assigned by November 2, 2017. Students who have been assigned a special project will have to confirm their decision of undertaking it and get the full documentation for their work from the teacher. When a special project is completed, the material produced for the project has to be submitted to the teacher and discussed with the teacher. Special projects must be completed by the deadline indicated in each project. Completing a special project exempts from the final test in the Lab and from the mandatory assignments (the exam terminates with the discussion of the special project).

1) TOSCA language support in Verigraph (max 3 students)

Verigraph (https://github.com/netgroup-polito/verigraph) is a verification service (based on JSON) that can receive a description of an NF-FG (richer than the one used for the DP2 assignments) and of network function configurations and that can analyze the given NF-FG and related configurations in order to check if some reachability policies (e.g. the possibility for packets of a certain flow generated by a certain node of the NF-FG to reach another NF-FG node) are satisfied or not. The service is written in Java and it is available with both RESTful and gRPC APIs.

TOSCA (Topology and Orchestration Specification for Cloud Applications) is a standard XML-based description language for defining network service descriptors, i.e. NF-FGs, which includes information about network topology of the desired service, network functions, their relationships and configurations, and policies (http://docs.oasis-open.org/tosca/TOSCA/v1.0/TOSCA-v1.0.pdf).

This project consists of adding the support of TOSCA into Verigraph, to describe NF-FGs, with related network function configurations and policies. This means that Verigraph has to be extended in order to also accept TOSCA descriptions (in addition to the ones already accepted) in the requests to run verification of reachability properties. The assignment can be divided into the following parts:

1a) Analysis and improvement of the current catalogue of network functions offered by TOSCA, with any missing functions available in Verigraph. Analysis and improvement of the current catalog of policies supported by TOSCA to add the support for reachability-based policies (needed for verification).

1b) Extension of the current interfaces of Verigraph (i.e. RESTful and gRPC) to support NF-FG descriptions in TOSCA format too, and corresponding Java implementation.

1c) Implementation in Java of a CLI to make the verification process runnable with both RESTful and gRPC interfaces and with any of the two supported NF-FG description languages (i.e. JSON and TOSCA).

A documentation of the new TOSCA's network function descriptions and of the CLI, a client for testing the extended service, and an ant script that automates compilation and running of the various programs and tests must also be provided as part of the solution.

The work, as described here, is sized for 3 students (each student will be responsible of a part of the work), and it can be associated with a thesis on the same topic (to be discussed with the teacher). The work can be assigned to less than 3 students, in which case it will be reduced accordingly.

Deadline: **The end of the February exam session**

2) RESTful interface for Verifoo and MeDICINE (max 3 students)

Verifoo (Verification and optimization orchestrator) is an extension of Verigraph (see project 1 for a brief description), which is capable of performing joint optimization and verification. It requires, as input, an NF-FG, the topology of an infrastructure network (IN), and a set of network policies to be verified. Verifoo produces, as an output, verified optimal placement of the nodes of the NF-FG on top of physical hosts of the IN, in text format.

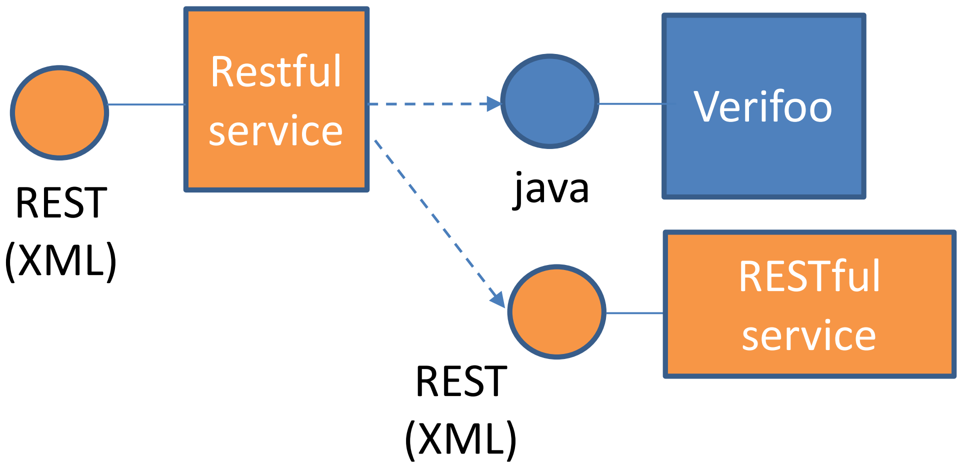
MeDICINE is an emulation platform, created to support network service developers to locally prototype and test complete network service chains in realistic end-to-end multi-PoP (point of presence) scenarios. It allows the execution of real network functions, packaged as Docker containers (another form of virtualization), in emulated network topologies running locally on the network service developer's machine.

The final aim of this project is to provide RESTful interfaces for Verifoo and MeDICINE, implemented in Java. The work is divided into 3 parts that can be assigned to different students, as specified in the description below, and it can be associated with a thesis on the same topic (to be discussed with the teacher).

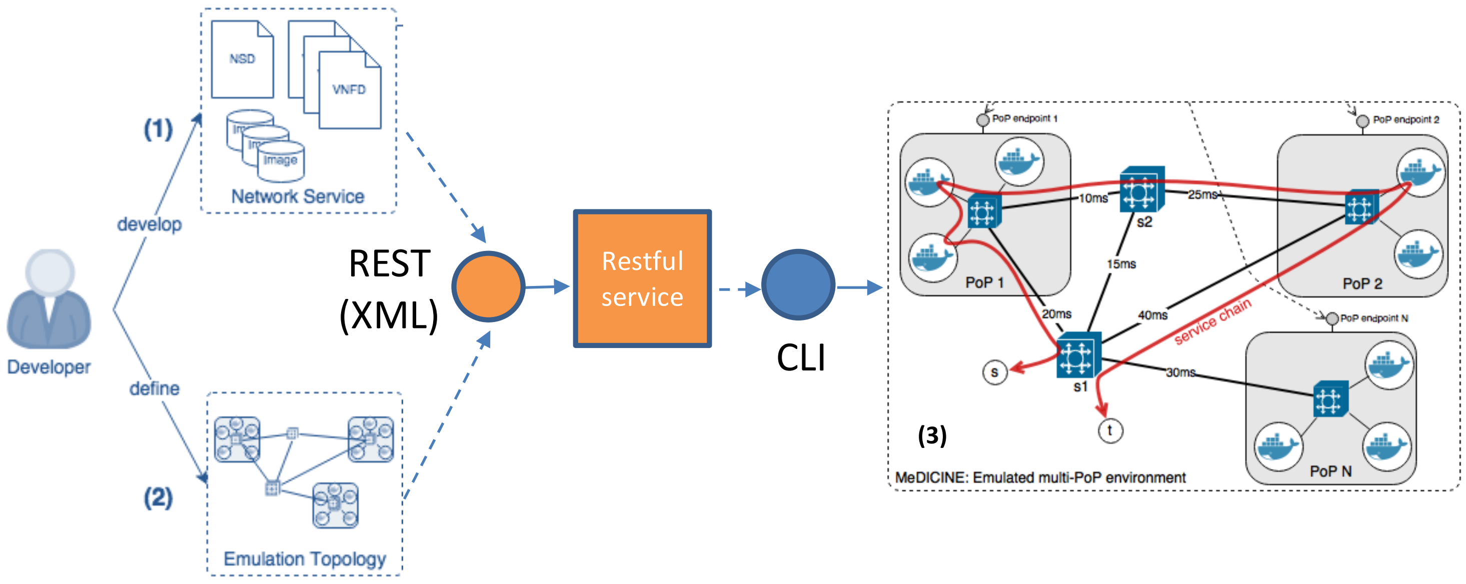
The work will start with a common task, which will be performed jointly by all the students who will undertake this project. This task consists of extending the XML format that was designed for Verigraph so that it can store all the information necessary for Verifoo and MeDICINE. Specifically, it is necessary to add information about the topology of the IN and the allocation of NF-FG nodes onto the IN.

Then, the work will continue with 3 different tasks, each assigned to a different student:

1. Design and implement a RESTful web service that can receive an XML document written according to the format specified above and the text output of Verifoo, parse the text output of Verifoo, and incorporate the parsed information into the XML document.
2. Design and implement a RESTful web service that exposes the functionalities of Verifoo and uses the service developed at point 1. The web service must be able to receive an XML description of NF-FG, IN topology, and policies, in the previously defined XML format, and return a similar XML document that also includes the allocation information returned by Verifoo and turned into XML by the service developed in point 1.



3. Design and implement a RESTful web service that exposes a subset of the functionalities of MeDICINE. It can be divided into the following parts:
    a. The web service must be able to receive XML documents written in the format designed in this project, which consists of NS (Network Service - richer than the one used for the DP2 assignments) **(1)** with the possible mapping on the infrastructure network (IN) and the IN topology **(2)** to be emulated by MeDICINE.
    b. The input files must be translated into appropriate commands of the MeDICINE platform. In order to execute these commands, the web service should host MeDICINE's command line client (CLI). The supported commands and short examples for the CLI can be found here. Finally, **(3)** the service is deployed and runs inside the platform.



A documentation of the XML format and of the web services, a client for testing each service, and an ant script that automates compilation and running of the various programs and tests must also be provided as part of the solution.

3) ASP-VER (Advanced Security Policy- VERification tool) in SFC (max 2 students)

Two emerging architectural paradigms, i.e. Software Defined Networking and Network Function Virtualization, enable the deployment and management of Service Function Chains (SFCs). A SFC is an ordered sequence of service functions (e.g., firewalls, VPN-gateways, traffic monitors) that packets have to traverse in the route from source to destination. This project consists in checking the correct configuration, ordering and interactions of service functions in a chain to satisfy overall security policies, following the verification approach proposed in [1]. In particular, the work consists of the following parts:

1a) Design a XML data format representation (XML schema) of the SFC model described in [1].

1b) Implement a library of Java classes for managing SFCs and storing them into the NEO4J graph-oriented database (http://neo4j.com/). The library must expose methods to read and validate XML representations of SFCs, serialize SFCs to XML, upload SFCs into a NEO4J DB.

1c) Implement a library of Java classes for performing policy verification in a SFC, referring to the algorithm and approach described in [1].

1d) Design and implement a RESTful web service that uses the two libraries developed in the previous parts. The web service must be able to receive an SFC using the previously defined XML format and to perform the operations exposed by the two libraries.

A documentation of the XML format and of the web service, a client for testing the service, and an ant script that automates compilation and running of the various programs and tests must also be provided as part of the solution.

[1] https://arxiv.org/abs/1710.03486

Deadline: **The end of the February exam session**

4) Verification and Resolution of Access Control Policies (max 1 student)

An Industrial Networked System (INS) needs to control accesses to system resources, so as to prevent unwanted interactions of unauthorized users with the system itself. A Role Based Access Control (RBAC) paradigm is typically used to describe both positive and negative Access Control (AC) policies and check them in typical INS implementations. This assignment consists in designing and implementing an access control software tool, which is able to check whether policies are correctly implemented in the system (i.e. verification), starting from an already existing prototype of the tool. In case policies are violated, the tool must be able to perform a (re)assignment of credentials (automatically computed) to the users (i.e. resolution). Documentation is available in [2].

In particular, this assignment can be divided into the following parts:

2a) Design XML data format representations (described by means of an XML schema) respectively of the INS model and of the AC policies

2b) Analyze the current prototype of the AC tool and improve its sub-process organization by making it a service-oriented application

2c) Design and implement a RESTful web service that can receive both INS and AC policies using the previously defined XML formats and that can perform the verification and resolution processes of the AC system.

2d) Make the web service persistent in order to retrieve the data about INSs and AC policies and the partial results of both verification and resolution processes.

A documentation of the XML formats and of the web service, a client for testing the service, and an ant script that automates compilation and running of the various programs and tests must also be provided as part of the solution.

[2] https://arxiv.org/abs/1710.03491

Deadline: **The end of the February exam session**

5) Design and implementation of a simple vehicle tracking service (max 1 student)

The aim of this project is to design and implement a RESTful web service that can track the presence of vehicles in an area with restricted access and, based on the tracking information, can provide permission to new vehicles to enter the area and suggestions about the path to their destinations. The service uses a model of the area that is a graph, with nodes representing roads and parking areas, and arcs representing the possibility to go from one node to another one. When a new vehicle wants to enter the area, it has to ask the service for permission, specifying both the entry point (i.e. its current position) and the desired destination in the area. The service will check if it is possible to allow the entrance of the new vehicle, based on the area model (which can include a number of constraints, such as capacity of parking areas and of roads) and based on the current and expected future positions of the other vehicles in the area. If the service allows entrance, it responds by also indicating a suggested path that the new vehicle is expected to follow from the entry point to the destination, and a unique identifier assigned by the service to the new vehicle. While the vehicle is moving to its destination it will periodically send information about its current position to the service, which will update its tracking information. If the service realizes that a vehicle is not following the suggested path, it will compute an updated path that starts from the current vehicle position and will send it back to the vehicle in response to the position tracking message. The area model must be specified in an XML document, for which a schema has to be designed. The service must upload the area model from the XML file at startup and it must allow administrators to collect various kinds of information about the vehicles currently in the area and their expected routes.

A documentation of the XML formats and of the web service, a client for testing the service, and an ant script that automates compilation and running of the various programs and tests must also be provided as part of the solution.

This project can be associated with a thesis on the same topic (to be discussed with the teacher) which extends the system with other functionalities and performs validations by means of simulations.

Deadline: **The end of the February exam session**