Distributed Programming II
A.Y. 2018/19

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 November 5, 2018 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 8, 2018. 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). All special projects offer the possibility to continue the project work as a subsequent thesis on the same topic.

1) New RESTful interface for Verigraph (max 1 student)

Verigraph (https://github.com/netgroup-polito/verigraph) is a verification service (based on JSON) that can receive a description of a network topology and of security network function (NSF) configurations and that can analyze the given input 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 topology graph to reach another node) are satisfied or not. The service is written in Java and it is available with both RESTful and gRPC APIs.

This project consists of improving and optimizing the capabilities of the Verigraph verification service by extending and re-designing its RESTful API. The assignment can be divided into the following parts:

a) Design extended/improved data format representations including some new input parameters.

b) Design the new REST API of Verigraph, using the previously defined data formats.

c) Implement the new REST API in Java, by adapting the already existing implementation.

A documentation of the data 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.

Deadline: The end of the February exam session
2) Policy-driven NSF selection (max 1 student)

Verifoo (verification and optimization orchestrator) is an extension of Verigraph (see project 1 for a brief description), which is capable of performing joint refinement and verification.

Refinement is the process that determines the resources needed to satisfy security requirements, that translates high-level requirements into low-level configurations that may be enforced by the system, and that verifies that the set of lower level policies or configurations actually meet the requirements.

This project consists of implementing a specific task of the refinement process: the optimal selection of the best network security functions (NSFs) that correctly enforce the security policy (i.e. security requirements). The assignment can be divided into the following parts:

a) Design data representations (described by means of an XML schema) respectively of a policy and of a catalog of functions.

b) Design a RESTful web service that can receive both a policy and a catalog using the previously defined data formats (XML or JSON) and that can perform the selection.

c) Implement a library of Java classes for selecting (just selection, without ordering) from a given catalog the security functions necessary to enforce a given a policy, by using Z3 OPT (https://rise4fun.com/Z3/tutorial/optimization), a tool that can solve optimization problems under given hard constraints. Finally, the library has to be used to implement the RESTful web service designed at point b). The heuristics for selection (i.e. the hard constraints and cost functions for optimization) will be made available when the project work starts.

A documentation of the data 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.

Deadline: The end of the February exam session

3) Data models for NFV architecture (NSF description, policies, forwarding and service graph, allocation places) (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 Network Functions (e.g., firewalls, VPN-gateways, traffic monitors) that packets have to traverse in the route from source to destination.

Although there are several languages for specifying information in the NFV/SDN frameworks (topology, resources, NFS types ...), these languages are not well integrated with each other.

This project consists of the following parts (only part a if the project is assigned to a single student, both parts if the project is assigned to two students):

a) Design a data format (described by means of an XML schema) for the representation of all the most relevant information in the NFV and SDN contexts. A documentation of the data formats and some examples must also be provided as part of the solution.

b) Design and implement a RESTful web service that permits to store and retrieve the NFV/SDN
information and call an Orchestration service like Verifoo or Verigraph (see project 1 and 2 for a brief description). A documentation 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.

The work, as described here, is sized for 2 students (each student will be responsible of a part of the work. The work can be assigned to less than 2 students, in which case it will be reduced accordingly.

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

4) Design and implementation of a simple vehicle tracking service (max 2 students)

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 has to use a data model similar to the one used for the DP2 assignments, but more complex (for example, there will be the possibility to have nested places). 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.

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