Practical Session 5 – Perception, Assembly

Interactive Perception

In this practical session, we will explore the Perception module in practice, showing you how to add perception proxy to monitor components and collect the perceived data. Similar to *Practical 3*, our interaction with the Perception module will occur through the *Interactive Perception* component. All code for this practical is on our repo, please download the latest version of the code from our summer school **GitHub** repository at: https://github.com/barryfp/summer_school.

In this practical session we will use a web server as a target application. All components necessary to assemble the web server software is inside the *repository* folder in *Practical 5.* We are going to use the *InteractivePerception.dn* component to assemble and monitor the web server. In the *pal* folder in this practical, you will find the *InteractivePerception.dn* component, as well as the Perception module itself. As described in today's lecture, the Perception uses the Assembly functions and extends it with its own unique functions. Therefore, *InteractivePerception.dn* works as a wrap around both Assembly and Perception and provides a command line interface for both modules' functions access.

The execution steps are the following:

1. Compile components in both *pal* and *repository* folders:
    a. To compile *pal*: "dnc . -sp ../repository"
    b. To compile *repository*: "dnc ."
2. In *pal* execute: "dana -sp ../repository InteractivePerception.o ../repository/TCPNetwork.o"

Once Interactive Perception is executed, we can see all its functions by typing "help" in the command line. After the web server is completely assembled, you can access it by going to a web browser and requesting the index.html page: "localhost:2012/index.html" or simply "localhost:2012". The index.html page is located in the *htdocs* folder inside *repository* and you may check it to see if the content displayed onto your web browser is actually the same content in that file (:.

The next step is to add a monitoring proxy to our web server. The proxy code is in "*pal/monitoring/proxies*" in *HTTPProxy.dn* component. If you open the proxy file, you will notice that there is a TODO comment. We expect you to implement the metric and event collection as part of your assignment, but for now, it's suffice to know where the proxy component is. To add the component to our web server architecture we need to enter the '*add_proxy*' command followed by the expression that tells our Perception module where to place the proxy. Copy and paste the following line to Interactive Perception to add our monitoring proxy to our web server example:

```
add_proxy |../pal/monitoring/proxies/HTTPProxy.o|*(*:http.handler.GET.HTTPGET[0]:*)|
```

After adding the proxy to our web server example, we can now type "get_perception_data" to see the collected metrics and events. Remember that the perception data (metrics and events) are only

generated after a request has been made to the web server. To verify whether the proxy was properly added, go to the browser and make some requests to our index.html page and type "get_preception_data" in the Interactive Perception command line.

Assignment

For this practical assignment, we expect you to:

- Implement a client program that issues a series of requests to the web server (**suggestion**: use the *HTTPRequest* component);
- Finish the implementation of the monitoring Proxy available in *'pal/monitoring'* to collect **metrics** and **events** (perception data) from the *HTTPGET* interface of our web server example;
- Test collecting perception data using the client program and the *InteractivePerception* module.
- Implement different client programs with different workloads patterns. You may want to create different files in the *htdocs* folder.
- Find two different workloads that have different optimal web server compositions;