Report

on

Assignment 4: Web Services and SOA
Connected Systems and Devices
(DA614A)

Submitted

by

Andreas Gustafsson
Adrien Lebret
Fatimah Majid
Prakriti Dhang
Saif Alhuttaitawi

Supervisor

Zahra Ghaffari

Department of Computer Science,
Faculty of Technology and Society
Malmö University
Malmö, Sweden

I. Intoduction

In this assignment we have to build a **SOA style system** which consists of a **motion detection logger**. We have to build a web service that will provide motion logs. A client will able to request motion logs from the camera and display the motion log in a chart.

The ACAP application Axis Video Motion Detection (VMD 3), creates motion detection events whenever a motion is detected. We build a web service that will detect motion on request from the client. Whenever a "motion" event occurs, it prints to the syslog with specific texts. This texts in the syslog file is parsed by AxisCamWebservice component and keeps it ready to be used by front-end application.

The assignment has implemented the motion log display using Google chart. The RESTful web services were implemented to monitor the camera motion.

II. System Description

The system is composed of four components; The camera with the motion detection application (1), a python tcp request handler (2), a python HTTP server (3), and a client with HTML and javascript (4). In the following section a more in depth explanation for each will be provided.

The motion detection application used on the camera is provided by axis. In other words, it has not been developed for this specific assignment. By using the event system provided by axis an event is set up to be fired to a targeted connection whenever motion is detected. As we have set it up it will send a '1' to the target ip. However, for our application it does not matter what will be sent, only that something is sent.

The above mentioned TCP request handler is made in python and can be seen as a logger of sorts. It can be seen as a lightweight server which continually listens after any data on a specific IP and port. When something is sent to the handler it will write down a timestamp in a log file. It is worth noting that it appends a new line to an existing log in the format of '%Y-%m-%d %H:%M:%S'.

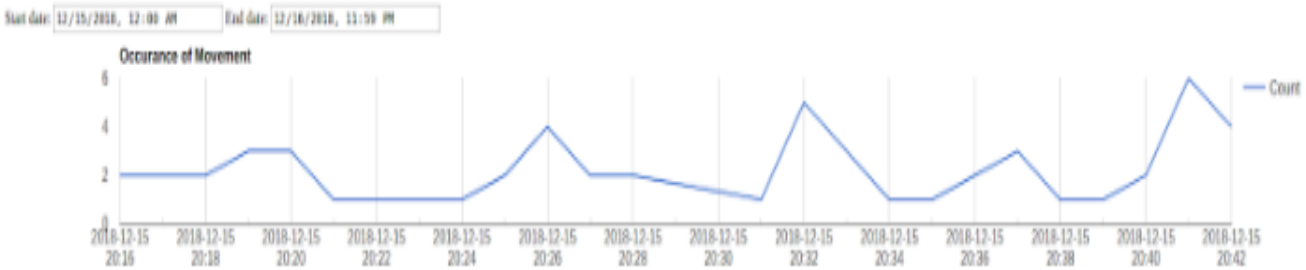
As for the web service, it is constructed as a simple http server with python. One could see this service as a REST service. This service takes two arguments, one being the start date time and the other being the end date time of an interval. As an interval has been specified and sent to the service it will return all the timestamps, from the log file, within the interval.

Finally, the client has two date time fields, for start date time and end date time respectively. As change occurs in either of these fields a new request is sent to the previously explained webservice. After having sent a request to the web service the client will await a response. Once a response has reached the client it will commence some basic manipulation of the data received to ultimately have a two-dimensional array where the first index is a date time and the second index is the number of movements recorded during that time. The reason for such transformation of data is due to how google charts work when inserting values into a data table. Lastly, the data is put into a line chart to visualize the number of movements occured at specific times within the specified interval of time.

It is worth mentioning that while the logger logs movements occurred as precise as seconds the client only displays data as precise as within minutes. The reason for this is due to how uninteresting the visualization became when it separated entries on a seconds basis, as no more than one movement can occur at the same second by the motion detection in the camera.

The choice of making most of the application in python was made due to how trivial it is to create these forms of applications with the language. In other words, that specific language was a great tool for the given assignment. Similar reasoning could be said about the client, as this meant not having to create any own logic about graphical representation or any of that like. It was also hinted within the assignment given that it would be of great aid to use google charts.

Below is a picture of the client to further increase understanding.



Two date times are set in the top-left corner. The data is then visualized with a line chart

III. Particular Interests and Challenges

It was particularly interesting to get to design a system as if it was part of a larger system. Although the assignment itself is rather small it is easy to imagine how one could extend this into something larger without any particular problems arising from the architecture itself. To get to pick what kind of tools to use for the completion of the assignment was also a nice addition to keep it interesting, this opened up for the possibility to try out languages one might not be experienced within. In our case we decided to create a majority of the system with python due to both wanting to try that out as well as knowing that it is a language known for being easily used within these tasks.

As for challenges, with the combination of several different languages and tools that we did not have much experience within (such as javascript and html) some problems arise. Especially with how to actually transform the data in the client into the format google charts wanted input in was troublesome. However, due to mostly being a question about how to work with javascript rather than a difficult problem to solve in itself it was more a question of time to come up with a solution.

IV. Feedback

- Demonstration of generating a web services during web services lecture help us to understand the concepts clearly.
- Events were generated when motion was detected and most of the time accessible camera was not showing any motion detection logs.
- University's camera were only accessible in our LAN. This slowed down the overall development activities.

V. Conclusion

We have created a system as described in the assignment which is capable of logging motion occurrences at specific times as well as visualize it with a graph of sort in a client that specifies the time interval of interest. The system architecture has split it into four different parts which take of one specific task each.

A lot of freedom was given in how to actually implement the solution, hence the tools we decided to be used were python, javascript and html. The motion detection application was an off-the-shelf application by axis.