A Framework for the Dynamic Generation of Workflows for Network Management

Andrzej Jasinski
Software Research Institute
Athlone Institute of Technology
Athlone, Ireland
a.jasinski@research.ait.ie

Yuansong Qiao
Software Research Institute
Athlone Institute of Technology
Athlone, Ireland
ysqiao@research.ait.ie

Enda Fallon
Faculty of Engineering & Informatics
Athlone Institute of Technology
Athlone, Ireland
efallon@ait.ie

Ronan Flynn
Faculty of Engineering & Informatics
Athlone Institute of Technology
Athlone, Ireland
rflynn@ait.ie

***Abstract*—A demonstration of the application of dynamically generated workflows to the management of a network environment is presented. A proposed workflow management framework (WMF) uses dynamically generated workflows to control the managed environment. Exception detection and handling in the workflow generation produce recommendations for mitigating incidents that might occur. The key features of the proposed WMF are its ease of implementation, flexibility and scalability. These features facilitate a quick response by the definition of new recommendations in the form of a workflow that can be executed in a controlled environment. The effectiveness of these recommendations can be assessed through feedback from the managed environment.**

***Index Terms*—dynamic workflow generation, proactive management, incident detection, incident prevention, incident mitigation**

I. Introduction and Problem Description

A workflow is designed for humans to control job processes within an organization. It is characterized as a set of actions that need to be performed in a repeating sequence [1]. The implementation of workflows is not a trivial task. For example, human error and design issues in a complex manufacturing process can lead to maintenance problems, waste generation and unwanted expenditure. When a problem occurs, the traditional workflow approach can handle known (predefined) issues only [2]. Traditional workflow approaches are not suited to managed environments where unpredictable events/incidents can happen at random. In such environments, the process is stopped, and human intervention must identify and fix the problem. There has been a growth in environment infrastructures in recent times, driven by advances in technology, especially in networking (SDN, 5G) and IoT [3]. Managing such infrastructures is a challenge, in particular the problem of incident detection and avoidance. Therefore, there is a need for proactive solutions, especially technology that can dynamically handle exceptions, ideally without human intervention, or with minimal human intervention. Research to date in the area of workflows is primarily anchored in business management. The literature shows that workflow in business management is strictly related to, and dependent on, the particular environment in question and the elements associated with it.

This demonstration presents a workflow management framework (WMF) that uses a novel approach to control, maintain and manage a network environment using dynamically generated workflows. The proposed framework can detect an environment incident and, using self-adaptive behaviour, make an autonomous decision to either update the existing workflow or replace it with a new one. The effectiveness of the generated change is assessed based on feedback generated by the environment controlled by the workflow. For the demonstration, the proposed workflow management framework is applied to a software-defined network (SDN) environment.

II. Implementation Architecture

The key elements of the WMF are shown in Fig. 1, which includes the dynamic workflow generation engine (DWGE). In the WMF, there is a client-server relationship between the managed environment and the DWGE. The DWGE server waits for a message from the environment to initialise. When the server-client link is activated, the first workflow is generated and forwarded to the environment. It should be noted that the environment can exist and operate without support from the server (autonomous environment). Alternatively, the environment can be supported partly or entirely by the server; this depends on the specifics of the environment and the initial configuration of the principles of cooperation between the WMF's environment and the DWGE. These principles include the prerequisites necessary to initiate environmental functions, e.g. disabled by default services or modules. To complete a job, referred to as a Task, a workflow must be executed in a controlled environment. The workflow engine server (WES) in Fig. 1 is responsible for dynamic workflow generation. All incoming requests from the environment are forwarded by the Listener in the DWGE to the WES. The engine uses the repository in Fig. 1 to read and save both structured and unstructured data.

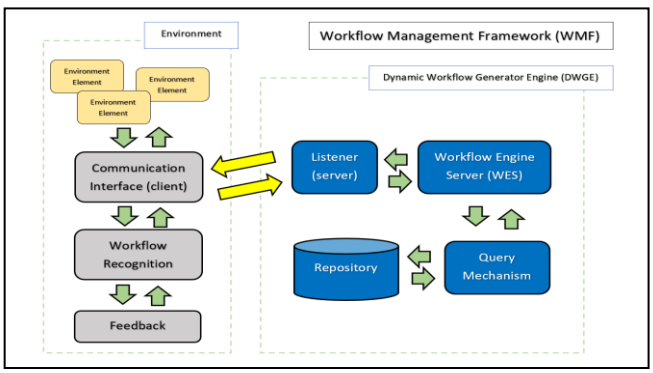

Fig. 1 Workflow management framework (WMF)

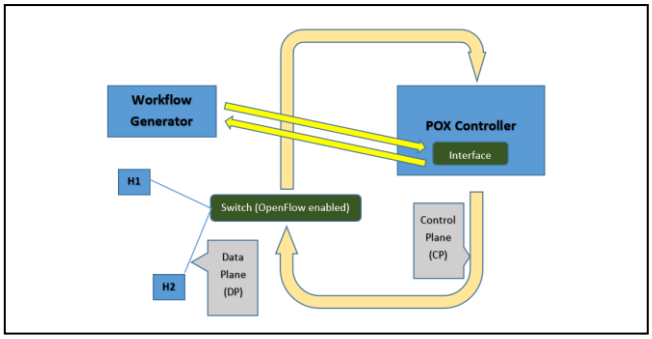

Fig. 2 SDN – workflow generator relationship

III. Test Environment and Demonstration

The WMF functionality was tested in a software-defined network (SDN), which was emulated using Mininet software [4] and managed by a POX controller [5]. This is illustrated in Fig. 2, which shows the workflow generator using a direct connection through the programming interface installed in the POX network controller. The network topology used is shown in Fig. 3, which has hosts (H1-H6) connected through switches (S1-Sn). By default, the POX controller manages the network so that all traffic between Network A and Network B is routed via Network C using switches S1, S2 and S3.

For the demonstration, a non-invasive proactive application of dynamic workflow generation is implemented, which is designed to support infrastructure and management mechanisms already implemented in the environment. Non-invasive behavior means that the DWGE will communicate with the environment using an interface installed in the managed environment without altering the environmental structure (i.e., no hardware or software changes). Also, the managed environment can still operate, while retaining its original functionality, independent of the WMF. Proactive management can add new functionality based on the request of the environment and using its resources. Some of the workflows do not require dynamic updates because they are responsible for the consistency of prerequisite, e.g., checking the POX modules or collecting data. In this demonstration, two types of dynamic workflow are implemented: management-oriented and monitor-oriented.

The aim of the management-oriented workflow is to build a path (or tunnel) between H1 and H2, must be dynamically updated when a link problem is detected because the previously known network topology will not be valid anymore. To build a dedicated tunnel between hosts H1 and H2 (available for these two hosts only), the workflow generated by the WES creates a tunnel (static route) through the unused and possibly shortest path S1-S4-S5-S2 (switch S3 has been excluded because it is part of the default route managed by the POX controller). This workflow is executed once every minute because the installed entry in the switch table expires after 60 seconds and then deleted from the switch. This task is repeated until an incident (link failure) is detected.

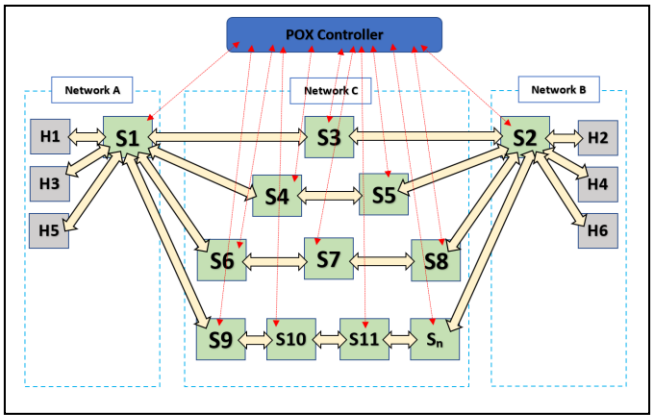

Fig. 3 SDN topology

Executed in parallel with the management-oriented workflow is a monitor-oriented workflow, which checks a live link using the link layer discovery protocol (LLDP). When LLDP detects a link failure, this information is stored in the repository (Fig. 1). Incident detection does not affect monitor-oriented workflows but executing the management-oriented workflow will not be possible. The management-oriented workflow uses the repository data to validate itself. When a broken link is reported, the workflow will change its own state to indicate that it is damaged, then the environment (interface) will send a request to the server to update this workflow, considering the topology changes caused by the incident.

In this experiment, a dynamically generated workflow was used to detect a broken link and recommend an alternative route for the affected traffic. For test purposes, a Python script manually terminated a working link between the last two switches in the established tunnel. For the traffic generation, ICMP ping packets were sent between H1 and H2 through the established tunnel formed using 4 switches (S1-S4-S5-S2). In parallel, the same command was executed to generate traffic between H3 and H4 to show that any other traffic managed by POX uses the shorter path (S1-S3-S2) by default and is carried out without interruption. As expected, when the link was terminated, the traffic in the tunnel was lost. However, default traffic continued without interruption. The system reacted to the incident by the generation of a new workflow that redirected traffic to the new tunnel S1-S6-S7-S8-S2.

Acknowledgements

This work was supported by the Irish Research Council, under grant number EPSPG/2017/336, and by Ericsson, the industrial partner.

References

[1] C. U. Press, “Cambridge Dictionary,” Cambridge, 01 Jan 2020. [Online]. Available: https://dictionary.cambridge.org/dictionary/english/workflow. [Accessed 01 06 2020].

[2] I. W. Salemme, “What types of workflow exist?,” 17 Apr 2020. [Online]. Available: https://www.pipefy.com/blog/types-of-workflow-exist/. [Accessed 20 Jun 2020].

[3] K. Nash, “5G accelerates network management challenges,” Ericsson, 27 Mar 2018. [Online]. Available: https://www.ericsson.com/en/blog/2018/3/5g-accelerates-network-management-challenges. [Accessed 16 06 2020].

[4] Bob Lantz, Brandon Heller, Nikhil Handigol, Vimal Jeyakumar, “Mininet - Virtual Network Emulator,” 2020. [Online]. Available: http://mininet.org. [Accessed 03 04 2020].

[5] J. McCauley, “The POX network software platform,” POX GitHub repository, 23 Nov 2017. [Online]. Available: https://github.com/noxrepo/pox. [Accessed 04 04 2020].