ASSL as an Intent Expression Language for Autonomic Intent-Driven Networking

Solmaz Jaberi  
Computer Science and Software Engineering  
Concordia University  
Montreal, Canada  
orcid.org/0009-0009-3896-0596

J. William Atwood  
Computer Science and Software Engineering  
Concordia University  
Montreal, Canada  
orcid.org/0000-0002-5973-5832

Joey Paquet  
Computer Science and Software Engineering  
Concordia University  
Montreal, Canada  
orcid.org/0000-0002-4922-6989

Abstract—Autonomic Networking is one of the main models proposed for the deployment of Networking Intents. Ideally, Intent should be specified by the user in a natural language (e.g., English), but it must then be transformed into a computer-executable representation that can then be mapped onto interactions with existing networking components. There currently is no consensus as to what language can be used to express the operational version of Networking Intents. The Autonomic System Specification Language (ASSL) was designed for the specification and verification of autonomic systems in general. We propose to use ASSL as the intermediate language to express the executable version of Networking Intents. Starting from a set of Intent examples (expressed in English), transformations of the examples into ASSL have been obtained. Using these examples, we show that ASSL is capable of expressing a very wide range of Networking Intents.

Index Terms—intent, intent expression, intent-driven network, autonomic network, network management

I. INTRODUCTION

From the network user’s perspective, a Networking Intent represents a set of operational goals (that a network should meet) and outcomes (that a network is supposed to deliver) expressed in a declarative manner, i.e., without specifying how to achieve or implement them [1]. The IETF, as well as several authors [2], [3], have proposed that organizing a network driven by Intent (an Intent-Based Network (IBN)) as an Autonomic Network will be an effective approach. Even though the originally-expressed Intent is a high-level construct expressed in some sort of problem-specific (high-level) language (ideally, English), it eventually needs to be translated/expressed/mapped onto a computer-executable version that represents a bridge between the original Intent and the operational considerations of the network upon which it is to be deployed. One of the main operational models currently proposed for the enactment of networking Intents is Autonomic Networking, which can be characterized as self-managing system model (self-configuring, self-protecting, self-healing, self-optimizing) [2]. If Intent is to be useful as a network management tool, it is necessary to be able to map a wide variety of Intents into concepts related to Autonomic Networks, which in turn must be mapped onto specific commands that are understandable to the network components involved in the operational version of the Intent. Following this premise, our research addresses the following questions:

1) For Intents expressed in English, is it feasible to express them in a computer-executable specification language intended for Autonomic Systems?

2) For what range of different kinds of Intents is this possible?

Since the definition of Intent given above is too general to permit evaluation of the expressiveness of an Intent representation, or the range of its applicability, we propose a set of Intent Objectives and adopt a set of Intent Categories. We then present a set of Intent Examples, and transform each of them into a target Intent representation. The set of examples has been chosen to cover all of the Intent Objectives, and a large subset of the Intent Categories. In this paper, we demonstrate that the Autonomic System Specification Language (ASSL) is suitable as an intermediate representation for the expression of a wide variety of Intents, which can then be executed and be connected upon an existing set of networking components to effectively deploy the originally expressed Intent.

II. PREVIOUS WORK

Mehmood et al. [3] provide a structured literature review of Intent-Based Networking, and an architectural framework that represents a possible way to integrate an Intent Controller into present and proposed cellular networking systems. They propose three layers 1) the Intent Layer, 2) the Network Management and Orchestration Layer, 3) the Infrastructure and Resources Layer. This layered model proposes that Intents be created by interactions with network users through a northbound interface, and translated into an operational version that maps with existing networking services defined at the Intent level that use the underlying networking resources, which can then be deployed and monitored by an Intent Controller. However, the authors do not mention any specific operational model or language used to express or execute the operational versions of the Intents. There has been considerable work
The expressiveness of such a language, we need to properly define the axes of variation of all the possible Intents that the language may be expected to express. Therefore, we propose below a set of Intent Objectives, based on ideas expressed in the IRTF’s RFC 7575 [2] and RFC 9315 [1], and other sources. This set of objectives makes it much easier to assess the expressiveness of a particular target representation. The objectives are as follows:

- **Abstract Formulation** [2]: Expression of Intents so as to abstract the operational details of its deployment.
- **Declarative Outcome Formulation** [1]: Declaration of goal instead of a procedure to achieve this goal.
- **Portability**: An Intent’s formulation should not need to be changed when it is deployed in a different context.
- **Local Behavior** [2]: Enable elements to express local goals independently from the system-level goals.
- **Composability** [1]: Modularity that defines minimally-coupled and reusable interactions with the exterior.
- **Efficiency** [16]: Succinct expression of Intents, whose translated meaning corresponds to an operational Intent.
- **Scalability** [17]: Scaling up of controlled nodes or Intents does not result in unacceptable resource consumption.
- **Monitoring** [16]: Observes the Intent’s behavior to verify that the network executes the defined behavior.
- **Security** [2]: Relies on secure interactions in the deployment of Intents to minimize possibilities of intrusion.
- **Reporting** [2]: Reports state aggregated from across the entire network, at the abstract level of the intent.

An additional dimension for expressiveness comes from the Intent Classification provided in RFC 9316 [4]:

- **Intent Type** (Carrier, Data Center, or Enterprise Networks)
- **Intent User** (Customer, or Network or Service Operators)
- **Intent Context** (Carrier, Data Center, or Enterprise Networks)
- **Intent Scope** (Carrier, Data Center, or Enterprise Networks)
- **Network Scope** (Campus, Radio Access)
- **Abstraction Level** (Technical, Non-technical)
- **Life Cycle** (Persistent, Transient)

**III. INTENT CLASSIFICATION AXES**

The problem that we address in this paper concerns the expressiveness of a specification language to be used for Intent representations that can be interpreted/executed to effectively deploy Intents. In this paper, we suppose that such a specification language is used as an intermediate representation between the high-level (e.g., English) Intent as expressed by the user, and the underlying networking elements and resources upon which the Intent is eventually deployed. This paper does not claim to describe the process by which the originally-expressed Intent is translated into this specification language, which has been recognized as one of the main problems of Intent-based networking. Since we wish to evaluate the expressiveness of such a language, we need to properly define the axes of variation of all the possible Intents that the language may be expected to express. Therefore, we propose below a set of Intent Objectives, based on ideas expressed in the IRTF’s RFC 7575 [2] and RFC 9315 [1], and other sources. This set of objectives makes it much easier to assess the expressiveness of a particular target representation. The objectives are as follows:

- **Abstract Formulation** [2]: Expression of Intents so as to abstract the operational details of its deployment.
- **Declarative Outcome Formulation** [1]: Declaration of goal instead of a procedure to achieve this goal.
- **Portability**: An Intent’s formulation should not need to be changed when it is deployed in a different context.
- **Local Behavior** [2]: Enable elements to express local goals independently from the system-level goals.
- **Composability** [1]: Modularity that defines minimally-coupled and reusable interactions with the exterior.
- **Efficiency** [16]: Succinct expression of Intents, whose translated meaning corresponds to an operational Intent.
- **Scalability** [17]: Scaling up of controlled nodes or Intents does not result in unacceptable resource consumption.
- **Monitoring** [16]: Observes the Intent’s behavior to verify that the network executes the defined behavior.
- **Security** [2]: Relies on secure interactions in the deployment of Intents to minimize possibilities of intrusion.
- **Reporting** [2]: Reports state aggregated from across the entire network, at the abstract level of the intent.

An additional dimension for expressiveness comes from the Intent Classification provided in RFC 9316 [4]:

- **Intent Type** (Carrier, Data Center, or Enterprise Networks)
- **Intent User** (Customer, or Network or Service Operators)
- **Intent Context** (Carrier, Data Center, or Enterprise Networks)
- **Intent Scope** (Carrier, Data Center, or Enterprise Networks)
- **Network Scope** (Campus, Radio Access)
- **Abstraction Level** (Technical, Non-technical)
- **Life Cycle** (Persistent, Transient)

**IV. USING ASSL TO EXPRESS NETWORKING INTENTS**

ASSL is a declarative specification language used to represent the structure, behavior, and communication within a group of collaborating elements wishing to achieve a common task, by expressing system-level and element-level goals [16]. Generally, ASSL views autonomous systems (AS) as being made up of autonomic elements (AE) that communicate via interaction protocols (ASIP and AEIPs). Operationally, upon compilation, an ASSL specification is translated into an event-driven reactive system that enables the expression of the goals, behavior, and structure of a collectivity of elements that participate in global behavior that is achieved by orchestrating each of their local specified behaviors. ASSL has been used to represent the autonomic behavior for NASA multi-agent-based exploratory space probes collaborative missions, group-based space probes telecommunication behavior, the specification of real-time reactive systems, self-scheduling robotics, and autonomic pattern-recognition systems.
Analogous to an autonomic system specified with ASSL, an autonomic network is generally developed from behavioral models and a control loop to manage the deployment of the Intent onto the network in an autonomic manner. The hierarchical/multi-tier structure of ASSL allows us to express a Networking Intent as an autonomic system in the following manner: First, the AS tier provides the means to create a global perspective for the Networking Intent and its underlying autonomic network topology by representing the network rules implemented in the Intent in terms of self-management policies and metrics necessitated to define the specification of the public characteristics of the Intent. Second, for every element involved in the specification of the Intent, a local set of rules can be defined, which specify the local perspective/responsibility of the part of the Intent that is managed by this particular AE. Finally, a communication protocol is declared across the AS and the AEs. Operationally, each of the AS and AE tiers can then be executed independently and communicate via their respective interaction protocols to achieve their collective/local goals and behaviors.

One of the defining characteristics of networking stack implementation is that each additional layer should be implemented using abstractions that do not require reimplementation of the underlying layers. ASSL achieves this as each AE can be defined to connect to one or more Managed Elements (ME), which are in this case a pre-existing networking component/resource that is assumed to expose an interface that can be used to interrogate their state and/or send them commands to effectuate actions as the definition of the Intent may require. Through such a structure, the ASSL specification can interact with pre-existing networking components to effectuate an Intent without any required change in their implementation. Such deployed intent thus represents an “overlay network management layer” which works independently from the orchestrated elements.

The fundamental basis for executing a Networking Intent using ASSL involves an underlying reactive system control loop that evaluates the abstract self-management policies, including their conditions and specific measurements to evaluate the network’s managed elements’ state/performance, and, whenever the desired state/performance threshold is not met, the mapping to appropriate actions triggered on other managed elements that are expected to make the system’s state/performance to go back within the desirable threshold.

V. ASSL INTENT EXPRESSION EXAMPLE

Many Intent examples are provided in RFC 9316 [4]. We have chosen 10 of these, plus one from the paper by Jacobs et al. [14]. These cover all of the Intent Objectives, and most of the elements of the Intent Classification from RFC 9316 [4]. They are hereby referred to as I1 to I11. Based on our analysis of our group of specified Intents, ASSL can specify almost every Intent category. In [4], a classification divides various networking concepts into Intent types used by diverse users. We selected scenarios from all different groups to cover almost every category and adequately analyze the ASSL specifications’ strengths and weaknesses. We could express all the context at an abstract level by simulating the concepts as ASSL fluents, actions, and events with their specified conditions. Therefore, theoretically, there is no lack of expressiveness power from ASSL. Also, in two stages, we analyzed the consistency of the expressions through the consistency checker feature of the ASSL toolkit. Then we assessed if the autonomic system respects the autonomic behavior by running the generated code and tracking its detailed execution traces. The resulting coverage is summarized in Table I. We proceed to evaluate the appropriateness of ASSL to express and deploy Intents based on the set of 11 examples that we developed with regards to each of the Intent Objectives that we have identified in Section III. The results of the breadth of applicability of ASSL to express the different Intent Objectives and Intents Categories are summarized in Table I, Table II, and Table III.

In order to demonstrate the breadth of expressiveness of ASSL to express networking Intents, we first proceeded to manually write a corresponding ASSL specification for each of the Intent Examples identified in Section V. Second, we analyzed the design elements of each ASSL Intent Example such as to demonstrate that ASSL can be used to achieve some of the Intent Objectives as stated in Section III.

Due to space restrictions, we only chose Intent I1: “Always maintain a high quality of service and high bandwidth for gold-level subscribers” to represent in this paper. For an ex-

TABLE I
<table>
<thead>
<tr>
<th>Intent Objective</th>
<th>RFC9316 Example(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mission Formulation</td>
<td>I1, I2, I3, I4, I5</td>
</tr>
<tr>
<td>Monitoring Capabilities</td>
<td>I6, I7, I8, I9</td>
</tr>
<tr>
<td>Security</td>
<td>I10, I11</td>
</tr>
<tr>
<td>Anomaly Detection</td>
<td>I11</td>
</tr>
</tbody>
</table>

TABLE II
<table>
<thead>
<tr>
<th>Intent Objective</th>
<th>RFC9316 Example(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operational Tasks</td>
<td>I1, I2, I3, I4, I5</td>
</tr>
<tr>
<td>Monitoring Capabilities</td>
<td>I6, I7, I8</td>
</tr>
<tr>
<td>Security</td>
<td>I10, I11</td>
</tr>
<tr>
<td>Anomaly Detection</td>
<td>I11</td>
</tr>
</tbody>
</table>

TABLE III
<table>
<thead>
<tr>
<th>Intent Objective</th>
<th>RFC9316 Example(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operational Tasks</td>
<td>I1, I2, I3, I4, I5</td>
</tr>
<tr>
<td>Monitoring Capabilities</td>
<td>I6, I7, I8</td>
</tr>
<tr>
<td>Security</td>
<td>I10, I11</td>
</tr>
<tr>
<td>Anomaly Detection</td>
<td>I11</td>
</tr>
</tbody>
</table>
tensive presentation and analysis of all the examples, see [18]. This Intent is related to quality of service issues (QoS), which is one of the most important criteria for assessing the network’s performance from a network end-user perspective. The underlying definition of “gold level” likely is translated into very specific operational details, which we are not explaining here due to space limitations. The ASSL specification that we wrote is based on network traffic packet classification, which results operationally in a traffic monitoring control loop specification that reads each kind of traffic’s throughput from a traffic monitoring managed element, evaluates to check the quality of service performance, and triggers actions to rectify the situation if the monitored values are below the acceptable thresholds. ASSL encapsulates the technical details of the procedures to achieve the intent by expressing the underlying logic of control mechanisms of QoS. For the specification, we divided the network’s goals into two main sub-goals, including maintaining a high quality of service and high bandwidth.

This starts with configuring the autonomic network according to the inAutonomicNetworkConfiguration fluent specified under the AS block (named AutonomicNetwork).

Then the required phases to achieve QoS as a self-configuration policy are specified under the AE Controller. In addition, we specify AE GoldLevelSubscriber to serve as the customer agent with its regulations to ensure that the customer agent behaves according to its agreed definition.

Regarding the general structure of this specification, AS and AES represent constructs bound to the hierarchical topology of the intent, which then correspond to general and private rules, respectively. The former includes the self-configuring policy to configure the whole autonomic network using an IMPL action called ConfigureAutonomicNetwork. The latter includes network stages to simulate QoS specification under the AE Controller, and the self-healing policy to maintain the quality of service based on the desirable metrics under the AE GoldLevelSubscriber. This hierarchical view provides an abstract security for the accessibility of policies for different autonomic elements in the system, demonstrating a limited form of the Security objective. Also, due to this hierarchical design, AES can share the distributed policies defined under the AS, while also demonstrating their localized autonomic behavior to meet the Distributed and Local Behavior Management Intent objective.

The protocol channels provide interaction between the autonomic elements at two public and private levels. AEIP is responsible for a private interaction through GoldLink which is only accessible for the AES defined as friends.

The self-healing policy helps the autonomic network to reconfigure itself if the quality of service for the Gold-level is not met. While two self-management policies, such as self-configuring and self-healing, can work individually but under one autonomic element, this gives an example of the potential capability of ASSL to support the Composability objective.

Abstract Formulation in this Intent is met with using abstractions of networking components states, including no details about how the metrics are actually computed, or how the actions are implemented. This is shown in Figure 1 where the value of the bandwidthPolicer metric is extracted from the monitoringTool managed element.

The Declarative Outcome Expression Intent objective is met in several parts of the autonomic behavior, as QoS intensively depends on various metric declarations. In this scenario, we used two metrics, one called CoS as the class of service for QoS and one as a bandwidth policer to maintain the high quality and high bandwidth. The bandwidth threshold is defined between 5 and 10. Suppose the metric’s value is less than this threshold, like what is declared as an example in Figure 1 as 3. In that case, it results in bandwidth metric violation guiding the autonomic network to perform increaseBandWidth action to set the metric value to a number within the threshold range to maintain the high bandwidth. Also, observing these metrics by ASSL control loops shows the capability of ASSL to meet the Monitoring Intent objective.

VI. EVALUATION

Our list of ten Intent Objectives has been given in Section III. Five of these are satisfied because of the nature of ASSL: Abstract Formulation, Declarative Output Formulation, Distributed and Local Behavior Management, and Monitoring Capability. The remaining five are discussed below:

a) Portability: An ASSL specification is inherently portable, in that what is specified is independent of the details of a particular implementation environment. ASSL (and any other Intent Specification Language) will always need to be built on top of management modules that more precisely control the vendor-specific details of the underlying hardware.  

![Fig. 1. ASSL code excerpt for intent example II](image-url)
b) Efficiency: Since ASSL provides a high-level specification of Intent and the autonomic network concepts together in one formal language, without requiring the use of other modeling languages such as YANG, it is efficient from a programming expressiveness perspective.

c) Scalability: Scalability refers to either (1) deploying an Intent on networks of varying scales without having to change the formulation of the Intent, or (2) deploying a large number of Intents on a network. As we focused on Intent expression rather than Intent deployment, the actual deployment on a networking testbed was out of scope for this work, and we were thus not able to test for either of these aspects of scalability.

d) Security: Security must be pervasive in systems design. ASSL has top-level security through the concept of FRIENDS. Lower-level security needs to be provided by the underlying system modules; the specifications of the ANIMA Working Group [5]–[10], for example, have security built-in from the lowest level.

e) Autonomic Reporting: ASSL has the built-in capability to link to node-specific (and vendor-specific) modules of the underlying system. The specifications will have to be written to ensure that the necessary data are summarized and reported as needed, but the raw data-gathering is there, because it is essential to the feedback model that ASSL has as a core concept.

VII. Conclusion and Future Work

In this paper, we have demonstrated that it is possible to transform Networking Intents, expressed in English, into ASSL, a specification language for Autonomic Systems. To show the wide range of Intents for which this is possible, we have presented a classification set of Intent Objectives, and adopted a set of Intent Classifications. We have developed a set of Intent examples drawn from the research literature, which have been shown to cover all of the Intent Objectives, and most of the Intent Categories from the literature. We have identified inadequacies that ASSL has when representing Intent. They are expressed below, along with our future work solutions:

ASSL does not have the ability to express a topology that dynamically changes at runtime. In our future work, we intend to improve the ASSL specification language and its underlying execution engine to include dynamic discovery/creation and destruction of elements.

ASSL does not provide a way to express advanced security concerns, and does not currently implement secure communication channels. In order to alleviate this, we plan to integrate ASSL with the specifications of Autonomic Networks produced by the IETF ANIMA Working Group.

At this point, the generated code for Managed Elements is represented by very primitive dummy classes, which need to be manually programmed in order to effectively connect to existing components using sockets. In the future, we plan to rely on (YANG+NETCONF) and/or (CoAP+RESTCONF) to map to existing operational networking components that can then be interacted with by the ASSL code to interrogate their state, and/or trigger some actions.

Currently, the generated code is only a monolithic simulation of a real distributed system where each of the AS and AEs and ME components are running on different computers. ASSL needs to compile to a real distributed system for the solution to be applicable. Fortunately, the semantics used in the code generation is thread-based, so it is just a question of retargeting the generated code to a distributed model, and to implement the communication channels over a medium that provides secure interactions.

References