# Intent-Based Networking: a survey on implementation methods

Anne-Ruth Meijer

February 2022

## 1 Introduction

Along with the increase of network usage, networks are expanding in terms of size and their requirements [8]. This growth influences the complexity of managing and configuring the network. Traditionally, networks are configured and managed manually by network operators. However, the increased complexity makes this traditional method more difficult. A Software-Defined Network (SDN) is a versatile software-based network technology that separates the control plane and data plane [8] [17]. This allows network operators to configure, track and control the network through a centralized interface [17]. The benefits of SDN include that it allows for programmable and centralized network management. The architecture of an SDN allows for applications to inform the network what is needed. A method for informing the network is Intent-Based Networking (IBN). IBN offers an improvement to the traditional method of configuring networks by enabling the user to express a desire and translating it into network level configurations. The user is only expected to express what is needed from the network, eliminating the need to express how the network should achieve this need.

The benefit that IBN offers includes providing a better quality of service and experience [13]. It provides high-level, simple and reliable control. IBN can be used to create a system that can quickly accommodate for changes in a network without interrupting the existing services of the network [18]. Industry has expressed interest in the abilities of IBN. Cisco has published a white paper highlighting the benefits of IBN for bridging the gap between business language and IT policies [4]. Another improvement of IBN compared to traditional networks that Cisco highlights is the ability to understand what is expected of the network. Using integrity checks IBN can keep verifying if the network works as intended.

Other industrial use cases for IBN include planning, protecting systems and commissioning [19]. In planning IBN has for example the capability to decide whether the addition of another flow is a possibility. IBN can aid in protecting systems by computing a new secure path when a previous one fails. In terms of commissioning IBN can provide an accurate estimated price for certain network demands. According to a survey conducted in 2020 about the recent advances in IBN the recent developments in Artificial Intelligence (AI) in Natural Language Processing (NLP) will cause improvements in IBN, which will influence the application of IBN into network- and telecommunication [24]. It is interesting to understand what IBN is, knowing that IBN could benefit both academia and industry and with the possibility that it will be included in standardized network technologies. Therefore, the main question for this literature study is:

*What is Intent-Based Networking and how can it be implemented?*

In order to answer the question this research will further explore the concepts of Intent, Software-Defined Networks, Intent-Based Networking, translating Intent and Intent-Based Networking implementations.

| **Subject** | **Characteristics** | **Related works** |
|---|---|---|
| 1. Intent-Based Networking | The definition of Intent; The definition of Intent-Based Networking; What are Software-Defined Networks; What is P4 | [5] [7] [8] [10] [6] [17] [24] |
| 2. Intent-Based Networking relevance | How can Intent-Based Networking improve industry; What are use cases for Intent-Based Networking; Benefits of using Intent-Based Networking | [4] [13] [19] [24] |
| 3. Translating intent | Intent-Based Networking frameworks; Translation techniques with previously defined intents; Translation using Artifcial Intelligence | [3] [16] [13] [14] [15] [20] [21] [23] |
| 4. Intent-Based Networking implementations | Intent-Based Networking with P4; 5G networks, Network virtualization; Self-generating intents; Evolution of the network | [2] [9] [11] [12] [18] [22] [1] |

Table 1: Researched related works

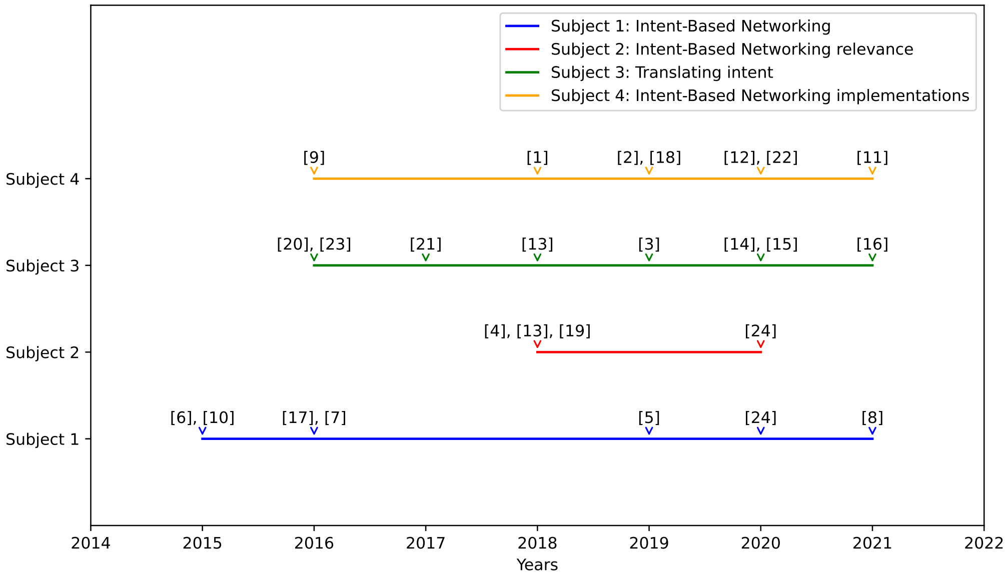


Figure 1: Timeline of the publish years of the related works. For each subject a timeline is shown with the references to the work shown on the year they were published.

To research these concepts we have reviewed multiple papers. To gather the papers we used Google Scholar. With the queries: "Intent-Based Networking" and "Intent-Based Networking implementations", the most cited papers were found. Including a survey about recent advances from 2020 [24]. The references of this paper were used to research varying methods for Intent-Based Networking. Table 1 shows the four main subjects for which we searched papers, their main characteristics, and the related works about the subject. In total 24 related works were read and used for this literature study. Seven of those works were for the first subject, four for the second subject, eight for the third subject and seven for the fourth subject. As visualized in figure 1 most of the related work is published in 2018 or later. This research includes reviews of company white papers, academic papers and software manuals.

The rest of the survey is structured as follows. First, Intent and Software-Defined Networks with Intent-Based Networking is explained in section 2. Second, the method of translating intent is described in section 3. Third, existing implementation efforts are compared in section 4. A final discussion on the literature study is given in section 5.

# 2 Intent-Based Networking

## 2.1 Intent and policy

To understand what Intent-Based Networking is it is important to understand the definition of intent. In 2016 Han et al. [9] stated that there was a lack of a standardized definition of intent. Their explanation is that intent is a set of system level policies. Whereas intent is independent from specific network features, such as network topologies. However, in 2019, the network working group of the Internet Engineering Task Force (IETF) published a working document in the form of an Internet-Draft about the concept intent [5]. The reason for the need of this working document is to clarify the concept of intent, its difference to the concept of policy, and to contribute to a common shared understanding of the term and concept. According to this working document the definition of intent is:

> *" An abstracted, declarative and vendor agnostic set of rules used to provide full lifecycle (Design/Build/Deploy/Validate) to a network and services it provides."* [5]

The definition of policy is given as:

> *"A rule, or set of rules, that governs the choices in behavior of a system."* [5]

Meaning, intent is the abstract expression of what is wanted of the network without expressing how the network is expected to achieve it. It is an expression of the expectation of the network, not an expression of the design or configuration of the network. This makes it applicable on different hardware, so it is not network or manufacturer specific. A policy on the other hand is a much lower level concept. A policy expresses a rule that influence the behavior of the network. With an intent a user can for example express that all traffic from node A should not go through node B, while a policy would be for example: when node A sets up a route the route should not include node B, if it does then create a new route or block the traffic. The policy includes more details of how the traffic should be handled.

## 2.2 Intent-Based Networking in Software-Defined Networks

To understand how Intent-Based Networking works, it is important to understand the network technology it operates on. As mentioned in the introduction it is able to work with Software-Defined Networks. Figure 2 visualizes the architecture of an SDN. There are three separate layers: the application layer, the control layer and the infrastructure layer. The application layer consists of the applications that use the SDN services. The applications communicate their requirements of the network in terms of security, virtualization, load balancing and other functions using the Northbound Interface (NBI). The control layer consists of a set of controllers, including the functionalities of the network, the management and an abstract topology. Through the Southbound interface the control layer interacts with the physical devices in the network. Through the west and eastbound interfaces the control layers communicate to each other to keep a consistent view over the network. The infrastructure layer consists of the devices in the network [10] [17].

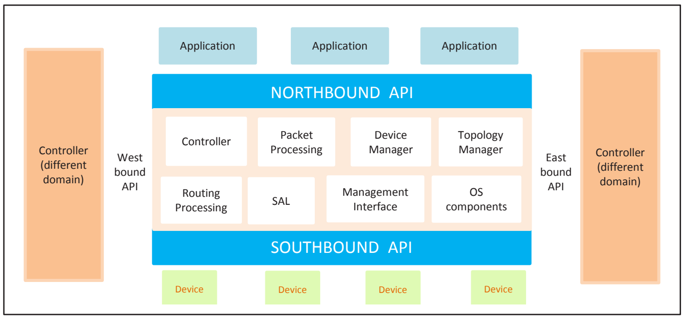


Figure 2: Software-Defined Network architecture [10]

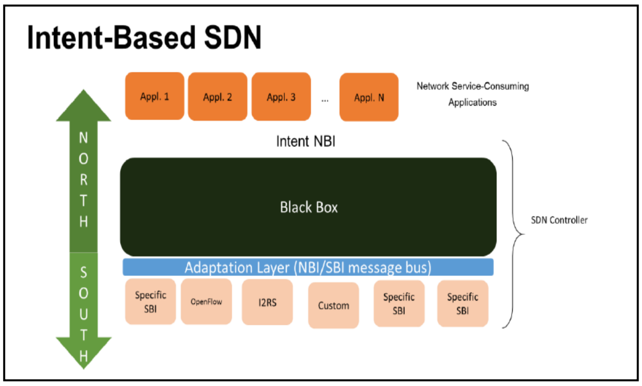


Figure 3: Software-Defined Network architecture using an Intent-Based Northbound interface. [17]

In IBN an Intent-Based Northbound Interface is used. Figure 3 shows a visualization of an SDN architecture from the view of the application when IBN is used. The architecture of SDN does not change, but the applications need less information of the network to change it. The main difference is that the control layer acts like a black box. Figure 3 visualizes the view of the applications where the information of the network and how it is configured is unknown. It is no longer needed for the application to understand the components of the network. The Northbound interface translates the intent and the control layer configures the network to comply with the intent [7] [10]. This method reduces the visibility and transparency of the network configuration and capabilities. For network operators tasked with analyzing the network the IBN approach introduces an extra challenge [22]. Since the network changes are not configured by the network operators, they loose information about the network. Section 4.5 elaborates on a method to mitigate this challenge.

# 3 Translating intent

Intent-Based Networking uses intent to express the high level expectations and requirements of the network. Different approaches are available to translate the intent. The extent of these translation capabilities depends on the method used for expressing the intent. An intent expressed in plain English or an intent expressed in a programming language differ in difficulty to translate. Existing intent translating techniques include ONOS intent translation [9], the NEMO language [23], the NIC project [14], iNDRIRA [13], and the Nile language [18] [11]. Recent advances in NLP combined with neural networks allow for more powerful conversion of natural language queries into network representations [24]. Thus other approaches to translate intent could use AI in the translation technique [11] [3]. The different techniques are further compared in the following sections.

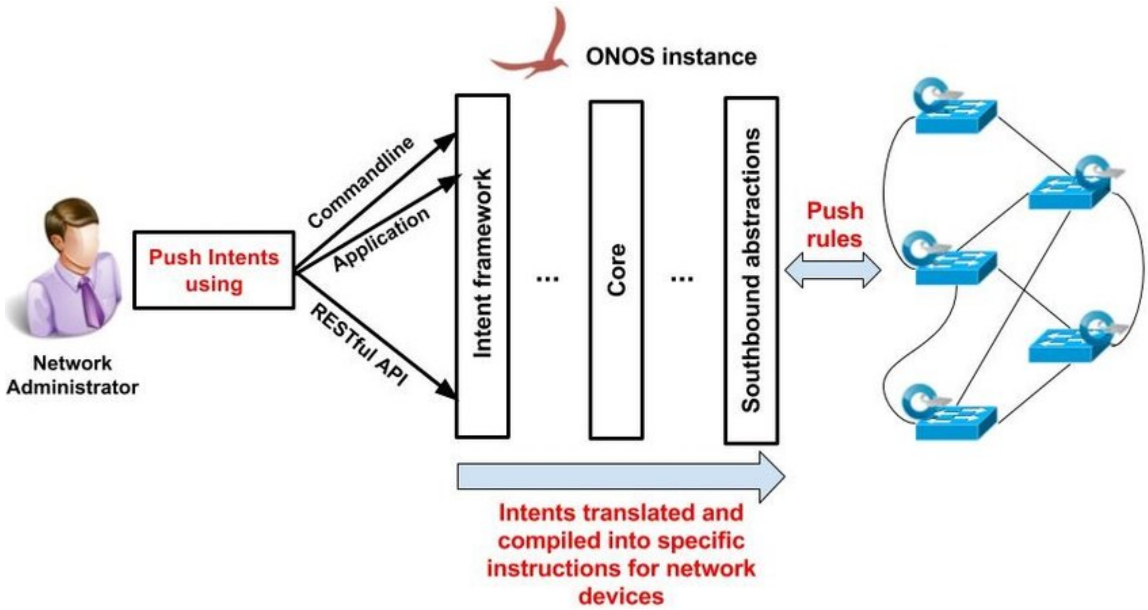


Figure 4: The ONOS framework architecture [1]

### 3.1 ONOS intent translation

Open Network Operating System (ONOS) provides a framework for IBN. The framework translates intents and installs these, changing the network environment [20]. The framework could for example change *flow rules* that are installed on a switch. Flow rules are rules specified in the forwarding table of a switch, the rules state what needs to be done with a packet that matches a certain constraint. The ONOS framework allows applications to specify their need from the network by translating intents into network configurations[9] [20]. The framework provides a set of built-in intents with their compilers, which can be extended. Examples of built-in intents are *point-to-point*, *single-to-multi* and *multi-to-single* [1]. The translation process works exclusively on the previous declared intents. The intent needs to be provided to the framework along with a compiler to convert the intent into a network configuration. Figure 4 visualized the architecture for the ONOS framework. The architecture shows that the ONOS framework is responsible for translating the intent and configuring the network. The intents can include requests about the network resources, constraints, criteria and instructions.

### 3.2 NEMO

The NEtwork Modeling language (NEMO) project is part of the OpenDayLight framework [24]. Instead of providing a framework for general Internet-Based Networking, OpenDayLight focuses on the implementation of the Northbound Interface. NEMO provides an interface to ask for resources and to specify what service is desired [15]. The framework for NEMO uses *Virtual Network spaces* (VN spaces) for the different users of the network. Each VN space is used to express intents and is isolated from other users. NEMO is a domain specific language allowing users to describe demands for the network [23]. Users can get information from the network using queries or notifications. To control the network, users have to express a certain operation in the following form:

*" when <condition>, do <action>, with <constraint>"* [23]

The translation of intent is limited to the NEMO syntax. A drawback of NEMO is the lack of ability to change the network configurations in response to changing network conditions [21]. To mitigate this drawback a method is proposed to extend the NEMO language [21]. The method allows users the express *reactive configurations* through intent. An example of a reactive configuration is that bandwidth is limited when excess usage of bandwidth is detected. Thus users can express that bandwidth needs to be limited when its usage reaches a given threshold.

## 3.3 NIC

Network Intent Composition (NIC) is another project by OpenDayLight to allow network operators to describe intent to the controllers through the NBI [14]. Both NEMO and NIC are sub projects for the OpenDayLight framework. NIC uses functionalities of the OpenDayLight networks framework and Southbound plugins to configure the network devices [16]. Similar to the methods described in sections 3.2 and 3.1 the intents need to be declared before they can be translated. NIC includes commands to add, remove, show, list and compile intents to a configuration tree [16]. An example of adding an intent to the configuration:

*intent:add –actions [ALLOW] –from <subject> –to <subject>* [16]

## 3.4 Nile

The Network Intent Language (Nile) is a language that acts as a layer between intents expressed in a natural language and network configurations [11]. Nile supports a set of intents, which can be extended. The intents have the form of definition functions that are close to the English language, while being understandable for a controller. An example of a function in Nile is shown in figure 5. Again only defined actions can be applied where the intent is defined in a function. The intent function can be used to apply code templates that can be loaded into the network [18].

```
define intent drop_hh_any_any:
  from   endpoint('any')
  to     endpoint('any')
  for    traffic('any')
  apply  drop_heavy_hitters
  with   threshold('more or equal', 20)
```

Figure 5: An example of an intent that applies an action to drop heavy hitters [18].

## 3.5 iNDIRA

Intelligent Network Deployment Intent Renderer Application (iNDIRA) is a system deployed between an application or user and the NBI of a SDN [13]. The purpose of the system is to translate queries from a natural language into network commands. To achieve this iNDIRA uses natural language and *ontology engineering*. Ontology engineering is used to understand what the given intent is by converting the query to graphs. These are parsed to network commands. In order to generate the network commands, files about the topology and user profile need to be rendered. When a query is given to iNDIRA, the query is parsed. An example of a sentence that could be parsed is:

*"For project 1, connect sites ANL to LBL, with condition nobwlimit, nolimit and start time now"* [13]

First, iNDRIRA identifies the keywords of the sentence and each of the keywords are turned into a Resource Description Framework (RDF) graph. The process includes replacing the sites with their urn address, converting time statements as "now" to system time, and recognizing words that mean the same. This results in the graph shown in figure 6.

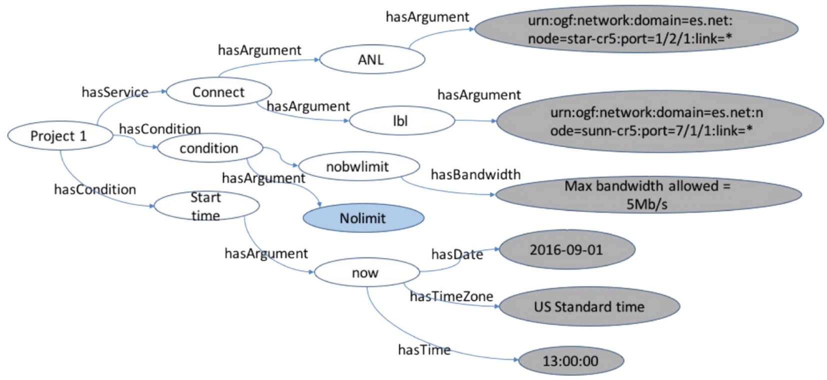


Figure 6: The process of parsing a sentence in iNDRIRA [13]

## 3.6 Natural language processing

Previous mentioned methods of translating intent to network configurations translate a defined notation of intent into a network configuration. Most methods don't include support for translating natural language. Translating natural language introduces additional challenges to the translating operation. There are many different possibilities to express the same demand and the language typically contains a large vocabulary. To be able to translate English, Natural Language Processing methods are used.

One implementation of translating intent is by using NLP to translate to Nile [11]. This approach uses AI on top of the existing translating method. Jacobs et al. created Lumi to translate English to Nile using machine learning and feedback [11]. Lumi uses Named Entity Recognition (NER) on queries provided in a chatbot interface to label the intent. NER classifies entities in the query, for example classifying the source and destination. However, to work properly it is important that there is sufficient training data available to learn to select the appropriate class for words with enough accuracy. Lumi avoids the lack of training data by using user feedback to learn if the query has been classified correctly. Over time this improves the accuracy of Lumi. An example of the feedback system is shown in figure 7. The feedback shows that the user can improve on the suggestion given by Lumi.

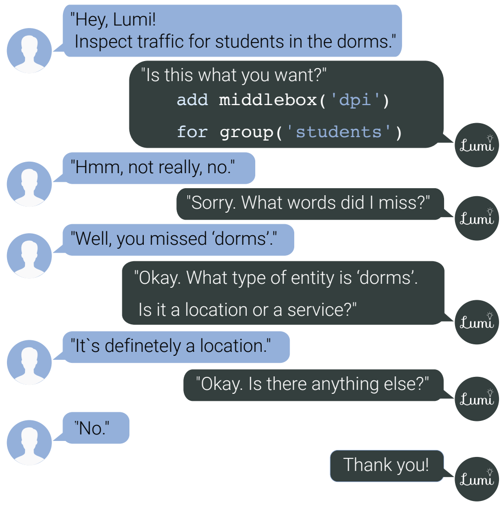


Figure 7: An example of the feedback system used by Lumi [11].

Another implementation of translating intent uses AI with Bidirectional Encoder Representations from Transformers (BERT) [3]. AI approaches for intent classification have proven to achieve good performance [3]. However, with the lack of labeled data it cannot be generalized. In order to make a general intent classification model the pre-training technique of BERT is applied. This technique solves the weak understanding of the complete language [3]. BERT is pre-trained using unlabeled text instead of human labeled text. The model provides an output representation of the input sentence. This representation can be used to give a predicted classification. The classification is based on the word and the predictions of the surrounding words. BERT has been trained and evaluated on audio recordings of reserving flights. The evaluation shows promising results and the research mentions the possibilities of future research with other types of data [3].

# 4 Implementation efforts

With an understanding of the concept of Intent-Based Networking and the available translation techniques explained we need to gain an understanding on the implementation efforts of IBN and the translation techniques they use. ONOS and OpenDayLight provide frameworks to implement IBN, using some of the translation techniques explained in 3. However, other implementation efforts have been published without the use of these frameworks. The rest of this section highlights different use cases of IBN with or without the provided frameworks.

## 4.1 Intent-Based Networking with P4

One of the uses of Intent-Based Networking is with P4 [18]. This research presents a new approach to programming the data plane based on IBN. To program the data plane a language P4 is used. The language allows to program network devices by indicating how the devices should process packets [6]. The data-plane language P4 allows network operators to program network hardware with custom functions. However, a drawback of using P4 is the static nature of the language [18]. Whenever the requirements of the network changes, the code needs to be rewritten accordingly. This makes it complicated to accommodate changes in the network and apply them in a limited time frame. IBN provides a solution. The researchers developed an IBN framework to translate expressed intent by users into P4 code. The framework uses their so-called *Extensible Intent Definition Language* to allow users to express their intent in. The language is an extension of the Nile language explained in section 3.4. To translate to P4 code a repository is created with P4 code templates for relevant network functionalities. The templates of the different network functions are combined based on the expressed intent. The merged code based on the combined templates is a valid P4 program ready to be loaded onto the hardware.

## 4.2 Intent-Based Networking in 5G networks

An area of networking that could benefit from IBN is 5G networks [12]. 5G networks are aimed to provide good quality of service over a wide range without interruptions. Managing the services on such a network proves to be a challenging task for network operators. When the deployment of 5G networks increase, the challenge of manually controlling the network will increase. Intent-Based Network can be used to automate the management and control of the network. A method for automating 5G network configurations is introduced by Khan et al. [12]. The method uses IBN for configuring the network and machine learning to control the stability of the network resources. To achieve IBN the system consists of three main components. The first component is an Intent-Policy engine. The engine translates high-level intents into required resources and policies. These policies are compiled into network configurations. To express intent an interface is provided. The second component is in charge of the management. The second component consist of a platform for implementing mobile networks, ONOS is integrated in this platform. The third component is the Intent Assurance and Update Engine. This component monitors the network and integrates improvements if they are available. The improvements are only implemented when they comply with the intents. This method for configuring and managing 5G networks not only relieves the task for network operators, but in addition eliminates the need for technical experts for configuring the network.

## 4.3 Intent-Based Network Virtualization platform

Network Virtualization (NV) is a technique to create Virtual Networks (VN) on top of a physical network infrastructure [9]. Using this technology multiple users are able to use the network independently from each other, while sharing the resources of the network. NV can create different slices of the physical resources of the network, allocating a slice to each VN. However, a drawback of this technique is the complexity of creating and managing the VNs [9]. Again IBN is used to propose a solution. The research presents a platform to manage VNs for SDN. The platform uses the ONOS intent framework, with pre-defined intents as described in section 3.1. When an intent is configured the platform verifies whether the new intent does not conflict with other intents of the network. The platform traces the entire existence of intents besides checking for conflicts. This is used for VN recovery when the hardware fails or to determine whether the intents are satisfied by the VN.

## 4.4 Self-Generated Intent-Based System

Intent is by definition expressed by the user or application. However, another approach is to allow the system to generate intent itself [2]. The systems implements expressed intents by users and applications. During this process the system determines if improvements can be made by the network. When there is an improvement available, the system implements it by expressing the improvement as an intent. The research uses the ONOS framework to implement the Intent-Based Networking. The system collects data from multiple resources, for example social and political networking. Using machine learning and big data algorithms intent from the system itself is generated. A feature of the network is selected to react to and to improve on, for example throughput. The research presents a demonstration using their change detection algorithm. The algorithm uses the monitored data to detect changes, for example detecting a saturated link between two nodes. The system can then express an intent for the SDN controller to forward the traffic through another node.

## 4.5 Increasing transparency and visibility of an Intent-Based Network

A drawback of IBN is the reduction of visibility and transparency of the network as mentioned in section 2.2. To mitigate this a framework extension for the SDN control plane is created, called ProvIntent [22]. ProvIntent records the *provenance of the network*, the evolution of the network over time. In the research ProvIntent is implemented using the ONOS framework. The extension uses intent provenance graphs, which give insight to how intents have affected the configuration of the network. The nodes in the graph represent the agents, events and data structures of a system. The edges represent the relationships between the nodes, for example an event uses a data structure. In the provenance graph nodes include an IntentState, a state that includes data about which application submitted the intent, when it was submitted and the requirements of the intent. Edges include transitions between intents, network changes that affected the intent and the effect on the network of changing an intent. The evolution of an intent is captured by the path of IntentState nodes in the graph. Figure 8 shows the evolution of an expressed intent that is installed at t=3 and withdrawn at t=100. The graph showing the past intent and network configurations allows network operators to review the past effects of intents on the network.

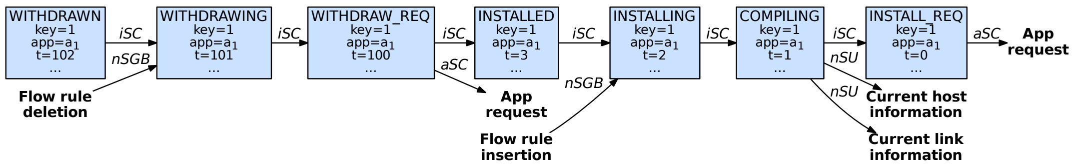


Figure 8: A provenance graph showing the evolution of an intent. The blue objects are the IntentStates at different time intervals [22]

# 5 Discussion

To conclude this survey on Intent-Based Networking and the implementation methods we summarize the discussed topics, discuss the answers on the research question and discuss some key observations that we made after surveying the literature.

In this study we have researched the networking concepts of intent, policy, Software-Defined Networks and Intent-Based Networking. We have looked into the benefits of Intent-Based Networking and how it can aid configuring networks. With a clear view on the underlying concepts we have researched the intent translation techniques and implementation methods for Intent-Based Networking. The translation techniques include methods that translate intents expressed in plain English and methods that translate only pre-defined intents. The reviewed implementations show the benefit of IBN in different fields of networking, for example 5G networks and Network Virtualization.

The question asked at the beginning of this survey was: *What is Intent-Based Networking and how can it be implemented?* Studying the available literature has provided definitions of Intent-Based Networking. Multiple translation techniques are available to implement Intent-Based Networking. Most provided intent translation methods cannot translate plain English into network configurations. However, machine learning and NLP provide interesting opportunities. The use of NLP and AI could benefit the field as literature suggested [24]. For now there are no standardized methods, only methods to solve specific issues in configuring the network. Interesting to note is that the implementations do make use of provided frameworks for IBN, although only examples using the ONOS framework have been discussed. There have been no recent developments with the OpenDayLight framework, and the projects for the framework seem to be archived.

Some key observations can be made from surveying the literature about Intent-Based Networking. Most literature highlights the improvements that IBN could bring to a system. However, IBN could potentially introduce some drawbacks. An improvement of IBN is the potential of the network to verify whether it is working properly, this ability could be used to detect possible security issues in the network. IBN could also open the network up for more security risks, since it uses user input. Users can influence the way the network works, thus they could potentially influence the network in a malicious manner. Another key observation is the trade off between using NLP to translate intent and using pre-defined intents. NLP allows for more variety in expressing intents, but gives the users more freedom. It needs a system that can verify whether the requirements can be achieved by the functionalities of the network, as well as a system to verify whether the network is working optimally. The translation mechanism has to be able to translate multiple variations of expressing the same intent. Using a pre-defined method of implementing IBN allows for a simplified approach, but it gives the users less freedom. Whenever new network functionalities are available the intent still need to be defined, requiring network experts. In complicated networks this simplified approach would introduce a lot of implementation overhead.

Concluding, Intent-Based Networking could benefit network management and configurations and there exist multiple methods to implement the concept. However, it would be interesting to see some research testing the concept on larger scale and reporting on potential drawbacks. Whenever IBN is implemented, it is important to choose a method that suits the expectations of the system and how it will be used.

# References


[1] Rami Akrem Addad, Diego Leonel Cadette Dutra, Miloud Bagaa, Tarik Taleb, Hannu Flinck, and Mehdi Namane. Benchmarking the onos intent interfaces to ease 5g service management. In *2018 IEEE Global Communications Conference (GLOBECOM)*, pages 1–6. IEEE, 2018.

[2] Mehdi Bezahaf, Marco Perez Hernandez, Lawrence Bardwell, Eleanor Davies, Matthew Broadbent, Daniel King, and David Hutchison. Self-generated intent-based system. In *2019 10th International Conference on Networks of the Future (NoF)*, pages 138–140. IEEE, 2019.

[3] Qian Chen, Zhu Zhuo, and Wen Wang. Bert for joint intent classification and slot filling. *arXiv preprint arXiv:1902.10909*, 2019.

[4] Cisco. Intent-based networking: Building the bridge between business and it. `https://securenetworkers.com/wp-content/uploads/2018/08/Cisco-nb-09-intent-networking-wp-cte-en.pdf`, 2018. Accessed: 19-01-2022.

[5] Alexander Clemm, Laurent Ciavaglia, Lisandro Zambenedetti Granville, and Jeff Tantsura. Intent-Based Networking - Concepts and Overview. Internet-Draft draft-clemm-nmrg-dist-intent-03, Internet Engineering Task Force, November 2019. Work in Progress.

[6] Open Networking Foundation. P4 open source programming language. `https://p4.org/`, 2015. Accessed: 20-01-2022.

[7] Open Networking Foundation. Intent nbi – definition and principles. `https://opennetworking.org/wp-content/uploads/2014/10/TR-523_Intent_Definition_Principles.pdf`, 2016. Accessed: 19-01-2022.

[8] Saad H Haji, Subhi RM Zeebaree, Rezgar Hasan Saeed, Siddeeq Y Ameen, Hanan M Shukur, Naaman Omar, Mohammed AM Sadeeq, Zainab Salih Ageed, Ibrahim Mahmood Ibrahim, and Hajar Maseeh Yasin. Comparison of software defined networking with traditional networking. *Asian Journal of Research in Computer Science*, pages 1–18, 2021.

[9] Yoonseon Han, Jian Li, Doan Hoang, Jae-Hyoung Yoo, and James Won-Ki Hong. An intent-based network virtualization platform for sdn. In *2016 12th International Conference on Network and Service Management (CNSM)*, pages 353–358. IEEE, 2016.

[10] Doan B Hoang and Minh Pham. On software-defined networking and the design of sdn controllers. In *2015 6th International Conference on the Network of the Future (NOF)*, pages 1–3. IEEE, 2015.

[11] Arthur S Jacobs, Ricardo J Pfitscher, Rafael H Ribeiro, Ronaldo A Ferreira, Lisandro Z Granville, Walter Willinger, and Sanjay G Rao. Hey, lumi! using natural language for {Intent-Based} network management. In *2021 USENIX Annual Technical Conference (USENIX ATC 21)*, pages 625–639, 2021.

[12] Talha Ahmed Khan, Asif Mehmood, Javier Jose Diaz Ravera, Afaq Muhammad, Khizar Abbas, and Wang-Cheol Song. Intent-based orchestration of network slices and resource assurance using machine learning. In *NOMS 2020-2020 IEEE/IFIP Network Operations and Management Symposium*, pages 1–2. IEEE, 2020.

[13] Mariam Kiran, Eric Pouyoul, Anu Mercian, Brian Tierney, Chin Guok, and Inder Monga. Enabling intent to configure scientific networks for high performance demands. *Future Generation Computer Systems*, 79:205–214, 2018.

[14] Vasudevan Nagendra, Arani Bhattacharya, Vinod Yegneswaran, Amir Rahmati, and Samir Das. An intent-based automation framework for securing dynamic consumer iot infrastructures. In *Proceedings of The Web Conference 2020*, pages 1625–1636, 2020.

[15] OpenDayLight. Nemo:user manual. `https://wiki-archive.opendaylight.org/view/NEMO:User_Manual`. Accessed: 25-01-2022.

[16] OpenDayLight. Network intent composition. `https://wiki.opendaylight.org/display/ODL/Network+Intent+Composition#app-switcher`, 2022. Accessed: 25-01-2022.

[17] Minh Pham and Doan B Hoang. Sdn applications-the intent-based northbound interface realisation for extended applications. In *2016 IEEE NetSoft Conference and Workshops (NetSoft)*, pages 372–377. IEEE, 2016.

[18] Mohammad Riftadi and Fernando Kuipers. P4i/o: Intent-based networking with p4. In *2019 IEEE Conference on Network Softwarization (NetSoft)*, pages 438–443. IEEE, 2019.

[19] Barun Kumar Saha, Deepaknath Tandur, Luca Haab, and Lukasz Podleski. Intent-based networks: An industrial perspective. In *Proceedings of the 1st International Workshop on Future Industrial Communication Networks*, pages 35–40, 2018.

[20] Open Networking Operating System. Intent framework. `https://wiki.onosproject.org/display/ONOS/Intent+Framework`, 2016. Accessed: 20-01-2022.

[21] Yoshiharu Tsuzaki and Yasuo Okabe. Reactive configuration updating for intent-based networking. In *2017 International Conference on Information Networking (ICOIN)*, pages 97–102. IEEE, 2017.

[22] Benjamin E Ujcich, Adam Bates, and William H Sanders. Provenance for intent-based networking. In *2020 6th IEEE Conference on Network Softwarization (NetSoft)*, pages 195–199. IEEE, 2020.

[23] Yinben Xia, Sheng Jiang, Tianran Zhou, Susan Hares, and Yali Zhang. NEMO (NEtwork MOdeling) Language. Internet-Draft draft-xia-sdnrg-nemo-language-04, Internet Engineering Task Force, April 2016. Work in Progress.

[24] Engin Zeydan and Yekta Turk. Recent advances in intent-based networking: A survey. In *2020 IEEE 91st Vehicular Technology Conference (VTC2020-Spring)*, pages 1–5. IEEE, 2020.