OPEN NETWORKING
FOUNDATION

Intent NBI – Definition and Principles

October 2016

ONF TR-523

Abstract

This document describes the Intent NBI paradigm, its utility and properties, and its essential implementation structure.

OpenFlow™

ONF Document Type: Technical Recommendation
ONF Document Name: Intent NBI – Definition and Principles

Contents

1 Introduction 4
2 Executive Summary 4
3 Scope 6
4 Definitions 6
5 Intent NBI: Defining Principles 7
6 Intent NBI: Properties and Structure 9
6.1 Examples 10
6.2 Consumer-Provider Associations & Service Contexts 11
6.3 Mappings 11
6.4 Notifications 13
6.5 Intent Engine 14
6.6 Intent – Operational Example 17
7 Intent NBI: Benefits 19
Appendix A: Back Matter 20
A.1 References 20
A.2 Contributors 20

Figures

Figure 1 Architectural representation of Intent NBI and mapping. This representation is nominal and not prescriptive (e.g., implementations may place mappings within provider system architectures). 5
Figure 2 Getting from A to B. 7
Figure 3 Consumer-provider interactions using Intent NBI. 8
Figure 4 Architectural representation of Intent NBI, mapping and potential sources of mapping contents and/or their dynamic updates. The illustration of mappings as standing outside of provider systems is not prescriptive. 13
Figure 5 Schematic representation of "engine" implementation of Intent NBI (such implementation is not prescriptive). The illustration of mappings as standing outside of the Intent engine and provider systems is not prescriptive. 15
Figure 6 An example of Intent NBI system operation for a simple service function chaining use case. The specific Intent NBI and mapping API syntax/expressions are hypothetical examples. 18

1 Introduction

This document describes the principles defining Intent NBIs, as well as the operational, architectural & structural aspects of Intent NBIs. Intent NBIs are declarative in nature and serve to separate consumer and provider system implementations, and to render human and/or machine consumer requests forwarded to provider systems as simple as possible. Intent NBI requests do not specify or prescribe any service implementation detailed methods, such as specific resources to be used in service fulfillment. As a result they are provider system-independent and provide a common basis for consumer service requests to diverse provider systems. An important aspect of the Intent NBI paradigm is the concept and use of mappings. Mappings provide for information intermediation between consumers and providers, permitting provider-independent Intent NBI requests to be properly interpreted by individual providers in their own system-specific terms. Intent NBI implementations make use of continuous-loop comparison among: existing and new Intent requests, mappings, and controlled-resource sets and states; to correctly enact and maintain service Intents, even as such Intents, mappings and controlled resource sets and states may evolve. Intent NBI systems may usefully be represented and even implemented as discrete Intent NBI “engines” lying above provider systems (i.e., SDN controllers).

Document structure:

Section 2 is an executive summary.

Section 3 delineates the scope of this document.

Section 4 defines significant terms used in this document.

Section 5 describes the defining principles of Intent NBI: “what is Intent NBI”.

Section 6 describes Intent NBI properties and structure.

Section 7 describes the benefits of Intent NBI.

2 Executive Summary

Intent NBI is a declarative paradigm/methodology for interaction between service consumers and service providers. The use of the term “consumer” does not necessarily/specifically imply a customer; nor does it, generally, imply a human: it is used in the broad sense of a consumer of services that are offered by one or more providers. With respect to providers, network controllers - as described in the ONF SDN Architecture v1.1 [1] - are of special interest and particular focus in this document. Within the meaning of the architecture and definitions discussed in [1], and to the extent that controllers and Intent implementations are viewed collectively, Intent NBIs are - from the server perspective - A-CPIs. Viewed from the perspective of consumer applications, and applying the recursion principle discussed in [1], Intent NBIs are D-CPIs.

NOTE: this document generally uses the terms “consumer” and “provider” in, effectively, direct replacement of/equivalence to the terms “client” and “server”, as the latter terms are used in [1].

Intent NBIs may be implemented in both distributed (e.g., n-tier or peer-to-peer) and client-server architectures. The client-server-oriented SDN architecture described in [1] is of special interest and particular focus in this document.

In part, Intent NBI aligns with general NBI roles, purposes and practices, e.g., as described in [1]. A major departure of Intent NBI from other NBI methodologies lies with the use of mappings, which serve as a mechanism to translate Intent NBI requests into forms that lower-level entities can understand. Mappings represent an information intermediation mechanism that effectively permits consumer and provider systems to communicate in terms "natural" to each. The architectural relationships among consumer and provider and systems, and mappings, are illustrated in Figure 1.

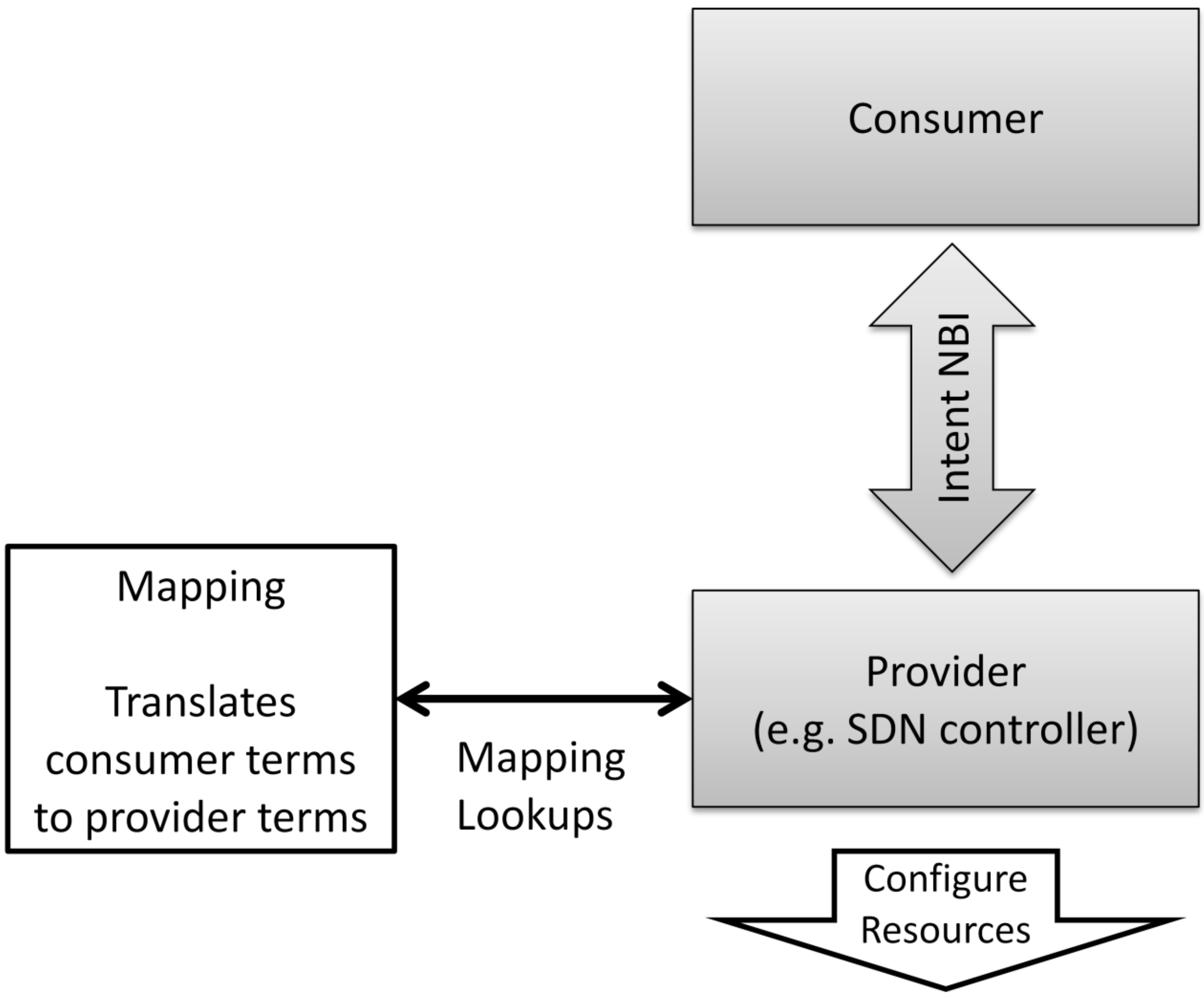
Consumer
Intent NBI
Mapping
Translates consumer terms to provider terms
Mapping Lookups
Provider (e.g. SDN controller)
Configure Resources

Figure 1 Architectural representation of Intent NBI and mapping. This representation is nominal and not prescriptive (e.g., implementations may place mappings within provider system architectures).

Note that in Figure 1, mappings are shown as standing outside the provider system; this should not be taken as prescriptive: i.e., implementations may place mappings within provider system architectures.

Intent NBIs serve to fully separate consumer and provider system implementations, and to render human and/or machine consumer-originated requests to provider systems as simple as possible.

Intent NBI systems make use of continuous-loop comparison among: existing and new Intent requests, mappings, and controlled-resource sets and states; to correctly enact and maintain service Intents, even as such Intents, mappings and controlled resource sets and states may evolve. Intent NBI systems may usefully be represented, and possibly implemented, as discrete Intent NBI "engines" lying above provider systems - i.e., controllers, though this is not prescriptive and is not as depicted in Figure 1.

The Intent NBI paradigm may be applied to any pair of elements in a distributed architecture, irrespective of the location and role of the pair of elements. Note that this also applies to a recursively-defined hierarchical control structure. However, Intent NBI may be particularly applicable to at least two different cases:

1. Relatively direct interactions between humans and provider systems;
2. Interactions between higher-level applications and lower-level system components.

The use of Intent NBI carries valuable benefits in such cases. These include provider system implementation independence and simplicity of Intent NBI request formulation.

3 Scope

Intent is a declarative NBI paradigm. The objective of this document is to describe that paradigm, its utility and properties, and its nominal implementation structure. It situates Intent NBI with respect to larger SDN architectural and interface concepts.

Specific Intent NBI syntax, mappings etc. are developed in Project Boulder in alignment with the principles and paradigms described in this document and as animated by use case-driven requirements. Their specific definition is outside the scope of this document.

Example use cases appear in this document for concept illustration purposes. However, this document does not seek to specifically or comprehensively consider Intent NBI use cases.

4 Definitions

Association: In essential alignment with the definition in [1]: A condition binding two contracting system entities, with information exchanged, permissions set and, generally, conditions established so as to permit operational interactions between the entities.

Client: Per [1]: An entity that receives services from a server.

Consumer: An entity that receives services from a provider.

Mapping: Mappings represent an information intermediation mechanism that permits consumer and provider systems to communicate in terms "natural" to each. Terms appearing in service requests from consumers may be translated, using mapping lookups, into terms directly relevant to providers. Mappings thus resolve aliases, addresses, etc. into terms relevant to lower-level systems (i.e., specifically, SDN controllers). Mappings may be generated or effectively obtained in different ways.

Non-prescription: Consumer service requests are non-prescriptive when they do not specify or constrain the selection and allocation, virtualization and abstraction, or assembly and concatenation of resources needed to satisfy the request. Intent NBI, as a declarative interface paradigm, uses non-prescriptive consumer service requests.

Notifications: Notifications appear within the context of a consumer-provider association, and are used to indicate loss-of-service or incipient loss-of-service to consumers. In this document, notifications are represented as appearing on the Intent NBI.

Provider: An entity that provides services to a consumer.

Provider independence: A property of Intent NBI service requests that derives, effectively, from non-prescription. The same Intent NBI request may be presented – without variation – to any provider with which the consumer is permitted to inter-operate (i.e., for which an operational association exists).

Server: Per [1]: An entity that provides services to a client.

Service context: In essential alignment with the definition in [1]: Information about, for example a reference uniquely identifying, a particular service delivered by a provider to a consumer.

5 Intent NBI: Defining Principles

Intent NBI simplifies service request interactions between consumer and provider systems. It does this by permitting consumer systems to request services using terms that are intrinsically known and relevant to them. Mappings are used to bridge any resulting "comprehension gaps" between consumer and provider systems – see section 6.2. The choice of detailed service fulfillment mechanics is left entirely to the provider system. Among other desirable consequences, this allows consumer systems to be developed, modified and operated independently of provider systems, and vice versa. This helps to separate, architecturally and operationally, the management and presentation of service demands from the mechanics of service delivery. To a large degree, this objective of Intent NBI mirrors practices that have been developed and refined in other industries. It is therefore useful to begin a consideration of Intent NBI's defining principles by looking at analogies from other industries.

Figure 2 Getting from A to B.

One useful analogy is from the transport-for-hire market: taxis, Uber, etc.; see Figure 2. The consumer of a transport-for-hire service is generally concerned with one thing: getting from point A to point B. He or she generally does not want to have to know or specify how to get from A to B. A declarative, Intent NBI-style request might thus be: "get me from A to B". Note that this request involves only information relevant and intrinsically known to the consumer. Such information is either naturally comprehensible to the transport provider, or becomes so through the equivalent of a mapping lookup – e.g., by consulting maps, business or residential address databases, etc. It does not contain any references to the transport provider's infrastructure, operational methodologies, or constraints. Similarly, why the customer should want to get from A to B is neither known by nor relevant to the transport provider. See Figure 3.

The request "get me from A to B" may be offered to any transport provider, and is therefore independent of transport providers: the request contains no references to any transport provider's particular infrastructure or operational methodologies. The transport provider is free to use its available resources, according to its own policies, to provide the service requested, most efficiently in conjunction with parallel service fulfillment requests, and in response to unforeseen or changing operational and/or infrastructure resource circumstances. In SDN terms: by working with a larger pool of resources flexibly usable across many consumers and services, a controller may often fulfill a greater volume of consumer service requests, with better overall resource efficiency and greater resilience to changing operational conditions; than by assigning smaller, fixed pools of resources to individual consumers.

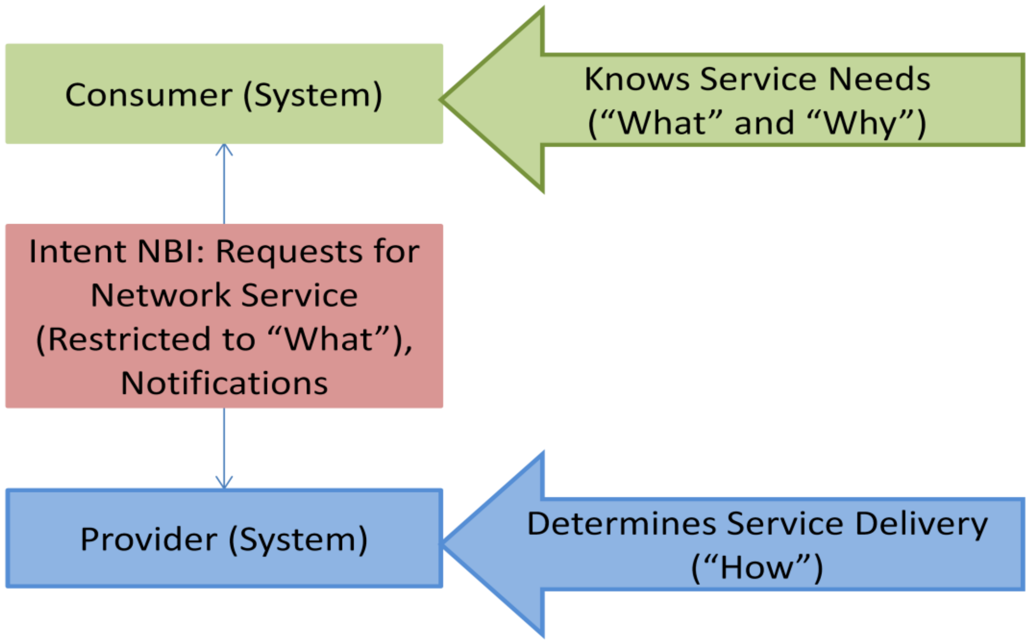
Consumer (System)
Knows Service Needs ("What" and "Why")
Intent NBI: Requests for Network Service (Restricted to "What"), Notifications
Provider (System)
Determines Service Delivery ("How")

Figure 3 Consumer-provider interactions using Intent NBI.

"Get me from A to B" represents a typical Intent NBI request: it defines, in simple – even intuitive – declarative terms, the service that is requested. Intent NBI requests may include "modifiers" that constrain or add detail to the request. For example, the transport-for-hire customer's request may additionally include "before time C" or "for a price not exceeding D".

Intent NBI request components and values derive from consumer operational service needs, but potentially also from consumer-specific policies. For example, an airline service consumer, looking to travel on business, may be precluded by his company's policies from purchasing a business class ticket. However, the airline needs only to know whether business class travel is requested, or may be accepted, in a particular service request case: the consumer's governing policy does not need to be known by the provider. Similarly, providers may have policies that inform or constrain service implementation operations: e.g., aircraft may not fly under certain weather or other conditions; or, aircrews may not be placed on duty more than X hours within any Y-hour period. Such policy frameworks inform the operations of the provider, but they do not need to be known by the consumer.

An Intent NBI-based consumer-provider interaction consists, minimally, of a request from the consumer to the provider. In circumstances in which it may be possible that the provider cannot fulfill a requested service – due to resource unavailability, provider policy constraints or for other reasons – the interaction should also include either an agreement by the provider to deliver the service requested, or a declination to do so, or some variation of these. In principle, further consumer-provider interaction phases could exist: for example, a provider could respond to a service request with one or multiple service options, requiring the consumer to make a selection before an agreement is concluded or declined. In such cases, a complete Intent NBI-based system inter-interaction is (something resembling) a negotiation; however, the decisions required between phases of such negotiations are made by the consumer and provider systems astride the Intent NBI. As such, they lie wholly or partly outside the Intent NBI implementation itself and are not considered further.

Notifications from provider to consumer - related to unavoidable service-impacting delivery system events or circumstances - also form an important component of an Intent NBI system. Intent NBI requests indicate in declarative terms a service state to be produced and, in general and importantly, *maintained* by the provider unless and until a subsequent Intent NBI request modifies or annuls the original service request. The provider must notify the consumer when it can no longer maintain the service condition requested, or expects not to be able to do so. This is discussed again in sub-section 6.4.

6 Intent NBI: Properties and Structure

Intent NBI is significantly characterized by two related properties: non-prescription and provider independence. It is also declarative.

Non-prescription: Per the discussion in section 5, an objective of Intent NBI is to extricate and separate consumer systems from involvement in the detailed operational control of provider-managed resources. Intent NBI requests therefore specify "what" (services are requested) but do not detail "how" (those services are to be delivered). Decisions concerning provider use of resources are left entirely to the operational discretion and determination of the provider. Implementation choices regarding technology, vendor, media, node, port, link, path, server, virtual machine, etc. are left to the provider (i.e., in SDN, the controller) with the support of mappings, and therefore such parameters never appear in Intent NBI requests. As discussed more fully in section 7, non-prescription brings a number of benefits to both consumers and providers. Strict non-prescription represents one end of a continuum of possible compositions of information and service request exchanges between consumer and provider, but strict adherence to non-prescription is significantly what defines Intent NBI.

Provider independence: A property of Intent NBI service requests that derives, largely, from non-prescription. The same Intent NBI request may be presented – without variation – to any provider with which the consumer is permitted to inter-operate (i.e., for which a consumer-provider operational association exists). However, mappings generally will be provider-specific (i.e., they may, and generally will, be consumer-provider association-specific). Terms appearing in service requests from consumers may be translated, using mapping lookups, into terms directly relevant to providers. Mappings may thus resolve aliases, addresses, etc. into terms relevant to lower-level systems (i.e., specifically, SDN controllers). Mappings may be generated or effectively obtained in different ways and are discussed further in sub-section 6.3.

Declaration: In the Intent NBI paradigm, the consumer system "declares" what it wants the provider system to deliver. It does not have to "ask" the provider system for any information (such as topology) in order to do so. This may be one source of the utility of, or requirement for, mappings, in that the consumer system may declare service Intents using specific terms not fully or naturally comprehended by the provider system, such that mappings must provide the reconciliation.

6.1 Examples

Let us illustrate the above by considering some networking-related examples.

In a first example, an Intent NBI request expression might resemble: "connect Bob to the (public) Internet". If this request is made to a corporate network controller (i.e., as the provider), a circumstantial context is implied: connect Bob to the public Internet, through the corporate network.

> NOTE: Context interpretation issues may present potentially complex challenges in some circumstances. Consideration of such cases and related challenges is for further study.

That this request is non-prescriptive is clear: it does not specify an Internet access point on the corporate network, it ignores the question of Bob's point of connection to the corporate network, and it does not specify how to route Bob's public Internet traffic between these points. A mapping is needed, for example, to translate "Bob" into terms relevant to the provider (e.g., a host address, or set of host addresses, corresponding to Bob's connecting device(s)). Obviously, for multiple reasons, this may be a dynamic mapping, in the sense that the specific mapping table entries may need to be modified over time.

In a second example, an Intent request expression might resemble: "create a connection service of type A and class B, between C and D" (C and D might represent, for example, consumer service end points). Mapping lookups might need to resolve the meaning of A (e.g., A = virtual private line service for Ethernet frame carriage) and B (e.g., B = bandwidth 10Gb/s, latency < 10ms, etc.)

> NOTE: in some cases, service type and class definitions may be both static and pre-defined, so that mapping lookups may not be required to resolve them. See sub-section 6.3.

The object identifiers C and D may or may not require mapping lookup, depending on whether the specific terms are held in common by both consumer and provider (i.e., if they are not held in common, then mapping lookup is required). For example, if C and D represent consumer service end points identifiers, then mapping look-up may be required to provide applicable provider service end points to the controller. Note that, since service performance-related constraints form part of the Intent NBI request (i.e., the service class specification) it may be possible that resource constraints could prevent the provider from delivering the requested service. In such a case, negotiation of alternative service parameters (e.g., a reduced bandwidth or higher latency or cost profile, or fulfillment at a later time) might enable a modified, but still consumer-acceptable, service to be delivered. Such negotiation processes would be based on further behaviors and decision-making capabilities built into consumer and provider systems, and as such are beyond the scope of Intent NBI itself.

In a third example, an Intent NBI request might resemble: “apply functions X then Y to traffic between A and B”. Implicit in this request is a requirement to find or create implementation instances of functions X and Y (e.g., using NFV methods), and to connect A to (an object corresponding to) function X, then to (an object corresponding to) function Y, and thence finally to B. All of A, B, X and Y might need to be subject to mapping lookup. Various specific approaches could be taken. For example, X and Y mapping lookups could yield groups of objects corresponding to available instantiations of function types X and Y, from which the provider could choose. Alternatively, the mapping lookup could trigger an NFV-based instantiation of a new VNF, for which an object is subsequently entered into the mapping table. Additionally, ordered sets of function types could be represented by aliases – with interpretation of the aliases delivered by mapping lookups, as could other objects; e.g., extending prior examples: “Apply security package A and filter package B to Bob’s Internet traffic”.

6.2 Consumer-Provider Associations & Service Contexts

Intent NBI interactions occur between specific system element pairs for which consumer-provider operational associations, including association-specific mappings, have been created (or, at any rate, which effectively exist). Such associations correspond in many respects to the client-server associations described in [1]. For example: “various contextual, relationship and business-related factors may lie behind the establishment” of an operational association, and the creation and/or operational enablement of an association may be, in many respects, an administrative (i.e., management) operation.

The identification of multiple (e.g., sequentially presented) Intent NBI expressions as pertaining to a particular service instance (or, in the language of [1], a particular service context) would permit a provider to properly manage multiple concurrently-existing but otherwise independent services requested by the same client and/or by multiple clients.

6.3 Mappings

As noted earlier, non-prescription, provider independence and declaration all generally impose a requirement for an adjunct to Intent NBI service interfaces, which we call mappings. Mappings are key-value stores, e.g., accessible by API, and are usefully considered, conceptually, as tables which permit providers to interpret the terms that appear in Intent NBI requests, by translating from the (“simple and intuitive”) consumer frame-of-reference terms that appear in Intent NBI requests to the detailed, specific provider frame-of-reference terms that permit the provider to properly interpret the requests.

Mappings represent an information intermediation mechanism that permits consumer and provider systems to communicate in terms “natural” to each. As a rule, in applications wherein the Intent NBI paradigm is a natural “fit”, consumer frame-of-reference terms may tend to be relatively human-oriented and intuitive, slowly-evolving and persistent, etc.; whereas corresponding provider frame-of-reference terms may tend to be relatively more “machine-oriented”, potentially faster-evolving and more ephemeral, etc.

As discussed in sub-section 6.2, mappings are, in the general case, consumer-provider association-specific. Both Intent NBI specific terms (i.e., keys - the “indices” for potential mapping lookups) and mapping entries (i.e., values - the “results” of mapping lookups) may be fixed; alternatively, one or both may evolve dynamically. We explore some of the possibilities here.

In some cases, specific terms appearing in Intent NBI requests may be defined by widely-known actual or *de facto* standards; e.g., aliases used to represent service and/or function types or classes. In such cases, effective interpretation may be directly engineered in provider systems – mapping lookup thus is not necessarily required (though it may be used).

In other cases, specific terms appearing in Intent NBI requests may be defined by explicit agreement between particular consumers and providers. In such cases, effective interpretation may be directly engineered in provider systems – mapping lookup thus is not necessarily required (though it may be used); however, this applies only to the particular consumer(s) in question.

In yet further cases, mapping entries – keys and/or values - may be dynamically-evolving, with such evolution driven by consumer (keys), provider (values) and/or external systems or authorities (keys and/or values). An example of the latter case might be an address resolution authority or service that either assigns or learns aliases applicable to particular consumers/users, and that generates or learns their applicable provider-relevant addresses. Another example might be an NVF environment that provides, to mapping tables, addresses for available instantiations of various VNF-based network functions.

> NOTE: in all of the cases described above, it is only the specific mapping entries (values), and/or the specific terms in Intent NBI requests that index the lookup of such entries (keys), that may be susceptible to dynamic evolution. The fundamental composition of mappings – i.e., the “meaning” of any given row/column and entry/cell in an effective mapping table – is presumed fixed. In principle, that fundamental composition could itself be determined by automated interactions between a consumer and provider; for example, during the establishment of a consumer-provider association. This represents a level of functional ambition that is likely beyond either need or practical possibility of implementation in the near future, and as such, is reserved for future study.

The nominal architectural relationships among consumer, provider and external systems, and mapping tables, are illustrated in Figure 4. Consumer and provider systems interact directly via the Intent NBI. Mappings are shown as standing outside both consumer and provider systems (N.B.: as discussed in section 2, Executive Summary, this is not prescriptive: e.g., implementations may incorporate mappings architecturally within provider systems) but are associated with particular consumer-provider pairings, or at least, particular provider systems. Consumer systems may, case-depending, write index lookup values – keys - corresponding to specific terms appearing in Intent NBI requests, to the mapping.

Provider systems read, and in some cases may write, lookup entries (values) in the mapping. External systems may - case-depending - write both index lookup values (keys), and lookup entries (values), to the mapping. The Intent NBI transmits provider-independent, declarative service-defining information from the consumer to the provider. The mapping-provider API delivers provider-specific information to the provider system in response to lookup index information (keys), taken from Intent NBI request content.

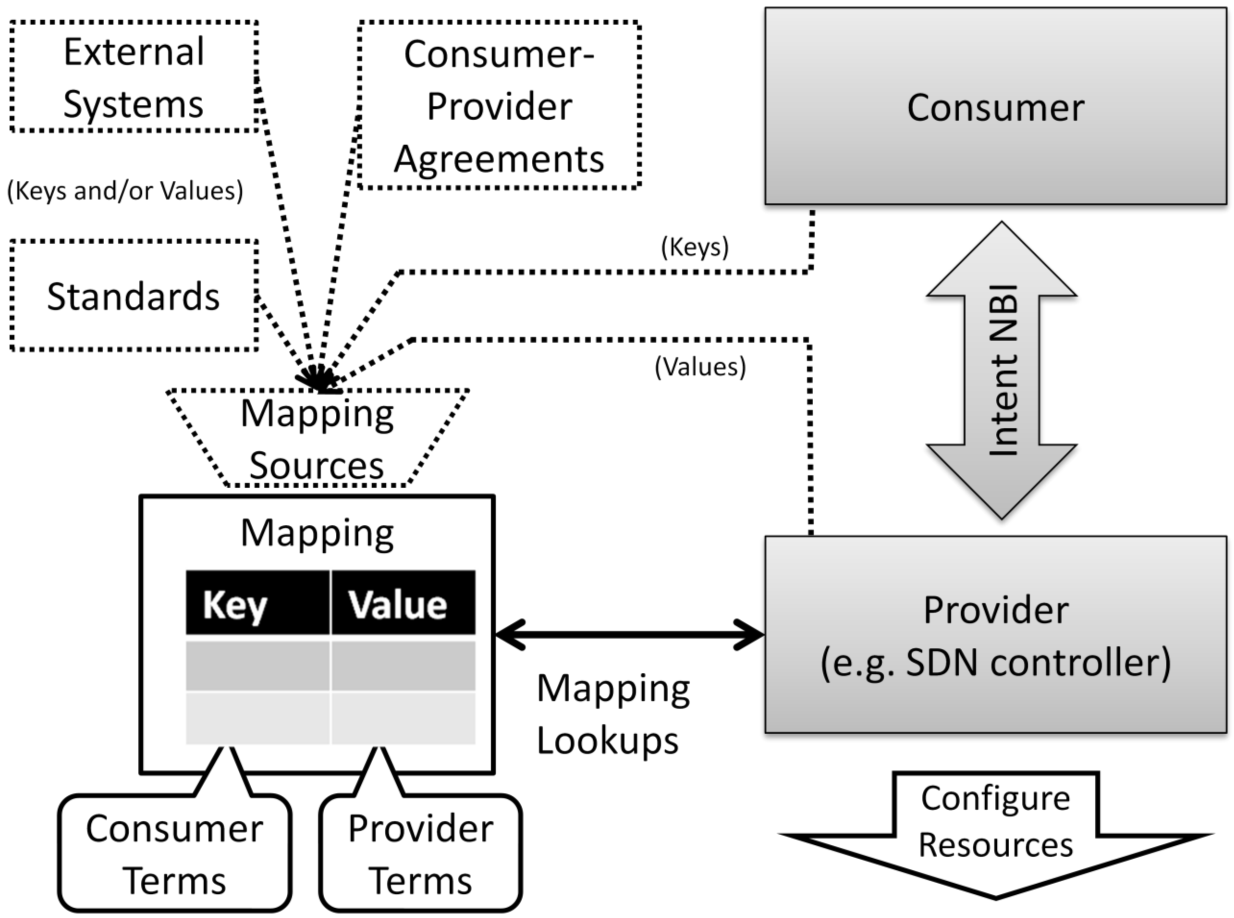

Figure 4 Architectural representation of Intent NBI, mapping and potential sources of mapping contents and/or their dynamic updates. The illustration of mappings as standing outside of provider systems is not prescriptive.

6.4 Notifications

Notifications are discussed in [1]. In [1], notifications refer to autonomous messages that provide information about events: e.g., alarms, performance monitoring (PM) threshold crossings, object creations/deletions, attribute value changes, state changes, etc.

In the Intent NBI paradigm, notifications are a mechanism by which providers signal actual or incipient service failures to consumers. The “what” not “how” nature of service requests in the Intent NBI paradigm means that providers (and/or Intent engines, depending on implementation) are entirely responsible for the active management of resources to fulfill and maintain requested service states. Therefore, consumers do not normally receive, and generally/presumably are not equipped to interpret, detailed resource or other state information that might permit them, on their own, to determine and classify or localize, or – more so – to anticipate, a service failure. In the Intent NBI paradigm, notifications are transmitted on the Intent NBI and should signal to consumers what aspect(s) of the requested service definition cannot or will not be delivered or maintained. Consumer systems have the responsibility to react to the indicated failure(s), e.g., by suspending traffic to the service, by requesting service modification or suspension, etc.

> NOTE: In the Intent NBI paradigm, “service failure” means, strictly, the failure of the provider to maintain the service in accordance with the full set of terms that defined by the consumer’s service request. Depending on circumstances, various flavors of “hard”, “soft” and “partial” service failure or anticipated failure might be defined, and subject to notifications.

6.5 Intent Engine

Intent NBI systems make use of continuous-loop comparison among: active and new Intent requests, mappings, and controlled-resource sets and states; to correctly enact and maintain service Intents, even as such Intents, mappings and controlled resource sets and states may evolve. Intent NBI systems may usefully be represented and even implemented as discrete Intent NBI “engines” lying above provider systems (i.e., controllers) although this is not prescriptive. This is depicted in Figure 5.

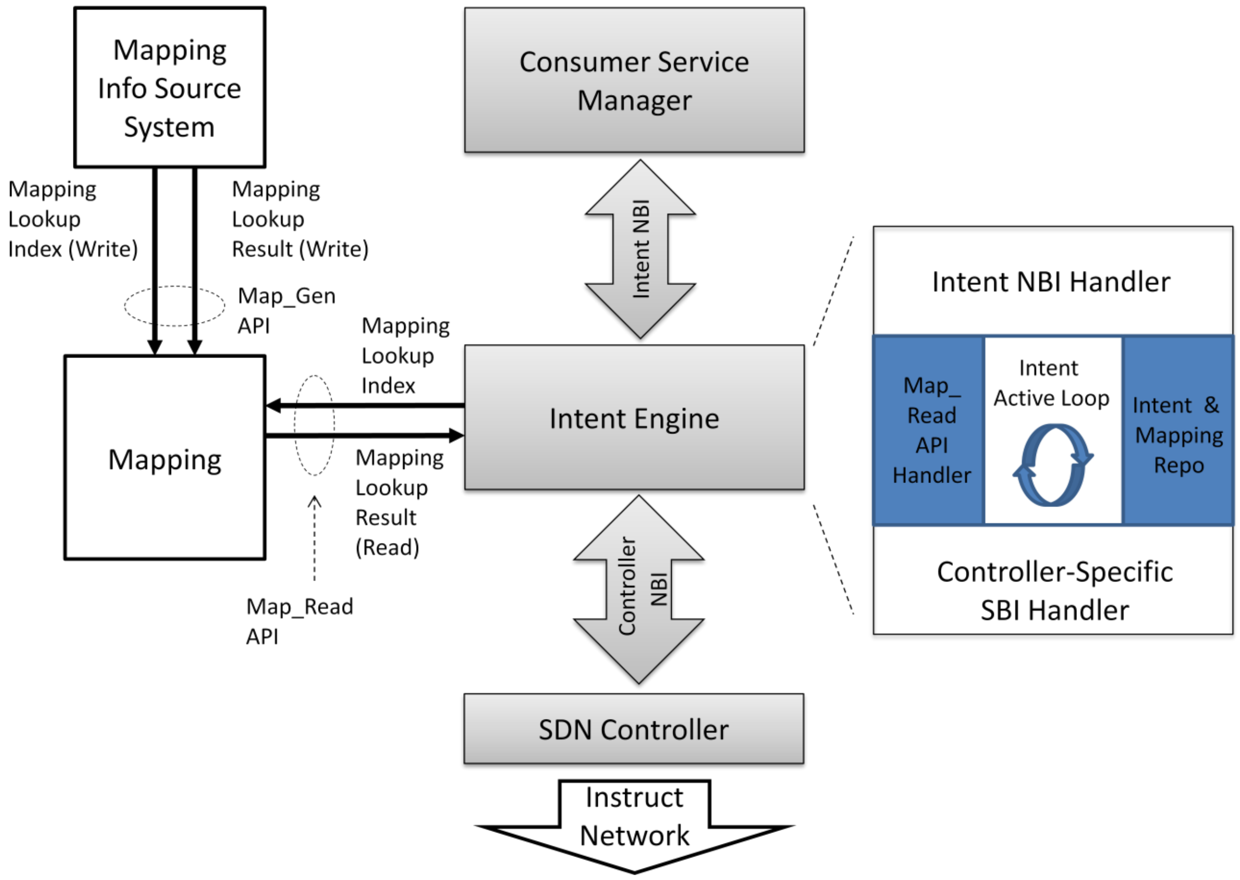

Figure 5 Schematic representation of "engine" implementation of Intent NBI (such implementation is not prescriptive). The illustration of mappings as standing outside of the Intent engine and provider systems is not prescriptive.

The Intent engine is, in effect, a client of the SDN controller and a server of the consumer service manager (i.e., the system labeled "consumer" in some previous diagrams). The Intent engine might be taken to consist of five major components (NOTE: this description is illustrative only: it is not intended to be construed as definitive or prescriptive):

1) An information repository ("repo") containing the set of active service Intents & mapping lookup values;
2) A Map_Read API handler. This is responsible for reflecting to the repo an up-to-date capture of mapping lookup values;
3) An Intent NBI handler. This is responsible for receiving service Intents from the consumer system and reflecting them to the repo (in native form), and for reflecting notifications to the consumer system;
4) An "Intent active loop". This element is responsible for continuously evaluating active service Intents & mappings from the repo and network information from the SBI handler; and taking actions required to instantiate new, or appropriately modify existing, service configurations as a function of detected Intent changes (repo), and/or of mapping changes (repo), and/or of

network changes (reflected by the SBI handler); by computing appropriate inputs to be reflected to the SBI handler:

        i. Inputs to the SBI handler may adhere to some model-based form, e.g., specifying one or more pairs of end points for piece-wise interconnection, along with parametric constraints on such interconnections. The function of the Intent active loop would then include any required and appropriate parsing and translation of Intent request terms into such model form. Instructions relayed to the SBI handler would be based on this model and use the specific terms that result from mapping lookups;
        ii. Where network conditions do not permit instructions to be issued that deliver all aspects of specified service Intents, to synthesize appropriate notifications to be reflected to the Intent NBI handler;

5) A controller-specific SBI handler. This is the only controller-specific component of the Intent engine and has the following functions:
    a. Receives inputs from the Intent active loop, and provides an appropriately modified form of those inputs, as provisioning instructions, to the SDN controller;
    b. Receives information (e.g., topology information) from the SDN controller, and forwards it to the Intent active loop in appropriately modified form.

As an example, let us consider an SDN controller that uses topology and tunnel provisioning model-based northbound interfaces. Because Intent does not specify or constrain implementation details, the Intent active loop only needs to specify domain-wise service end point pairs and connection parameters to the SBI handler. The controller can “intercept” this by presenting domain-wise single node abstracted topologies to the SBI handler. In this case, the adaptation functions in the SBI handler may be relatively simple.

The Intent engine is not only a useful conceptual vehicle; it may in some circumstances be an advantageous implementation method. By implementing Intent separately from controllers, we allow diverse controllers to be used with specific adaptations limited to the SBI handler component of the Intent engine. This approach therefore imposes no requirements or changes on controller implementations. This is valuable in that controllers are, advantageously, designed to deal with a range of client systems. For example, an SDN controller using topology and tunnel model-based NBIs is capable of forwarding a range of topology abstractions to clients, and receiving tunnel configuration inputs accordingly. While Intent NBI is content to work with “full” – i.e., single node – topology abstractions, other client systems might not be. Each possible client, including but not limited to an Intent engine, may thus request an appropriate topology abstraction from the same (implementation of) controller. Intent NBI thus does not require a unique “flavor” of controller to be delivered for its own purposes, reducing implementation diversity and complexity in the controller space.

Note that the description of Intent engine functional decomposition given above permits an Intent active loop “interior” model to differ from the explicit or implicit model reflected in the controller NBI, and also permits the Intent NBI to differ – even in basic construction – from the form and construction reflected by an interior model. The utility of the latter in Intent NBI is demonstrated in the next section.

6.6 Intent – Operational Example

By way of illustration, we return to a use case considered in section 6.1, related to service function chaining. See Figure 6.

In this example, the Intent NBI client is an enterprise security and/or service policy manager (assumed to be a "thin pane of glass" system with a human standing behind it). A network manager (a system, potentially – again – a "thin pane of glass") is responsible for creating user groups, internet access points, etc., and for indicating current, applicable IP addresses for each of these to the mapping service. A VNF Manager (system) is responsible for instantiating different classes of VNFs – a firewall function is referred to here - and for indicating current, applicable IP addresses for those VNFs to the mapping service.

Let us say that the enterprise service manager wishes to allow all sales users to connect to the internet, while requiring those connections to pass through a firewall. This Intent could be reflected to the Intent engine using a simple declaration, e.g.: Service_A=Path(Sales,Firewall,Internet). This hypothetical Intent NBI expression is taken to mean: create – as (by way of reference) Service_A – chained connections from sales user end points, to firewall VNF end points, to internet access end points. The label "Service_A" constitutes a service context identifier, per section 6.2. With mapping lookups, the Intent engine obtains IP addresses currently corresponding to sales users, available firewall VNFs and internet access points. With this information, the Intent engine is able to generate a series of pair-wise end point connection instructions to the SDN controller (e.g., to connect each user to an available firewall, and that firewall to an available internet access point). The end points are given to the controller as IP addresses which are correctly comprehended by it. The controller is free to construct the ensemble of specified, piece-wise point-to-point connections however it is able and sees fit to do. Note that here, "Firewall" is a proper name created to identify a network function of a specific type; if multiple firewall types exist, then multiple proper names (e.g., Firewall-A, Firewall-B, etc.) could be created to represent the respective end point groups in the mapping. Appropriate systems would have to understand the respective proper name-specific function relationships.

Note that the service Intent described may be entered by the enterprise service manager, at any time, irrespective of whether (and how many or which) sales users, firewall VNFs, and/or internet access points, are currently specified or available. The Intent engine's active loop is responsible for maintaining a current list of IP addresses corresponding to the defined groups, as each may be independently updated by the network and VNF managers, and for updating connection instructions to the controller on occurrence of such updates. Similarly, the Intent engine's active loop may be continually monitoring the connection topology/resources made available by the controller, modifying – if need be – the specific connection instructions it provides in order to maintain complete user-firewall-internet connection chains.

A notification corresponding to active, correct and complete fulfillment of the service Intent might consist of a return, to the service policy manager, of the originally issued Intent, i.e., Service_A=Path(Sales,Firewall,Internet). Let us now suppose that, at a given point, no sales users are actually defined in the mapping. In this case, the Intent engine might return the notification: Service_A=Path(NULL,Firewall,Internet). This would allow the service policy manager to understand that the Intent engine currently does not see, and is not actively connecting, any sales users. The service policy manager may feel no need to react to this. Let us say, on the other hand, that at some point in time, an insufficient number of firewall VNFs is available given the number of sales users being connected to the internet. In this case, the notification might consist of: Service_A=Path(Sales,PARTIAL,Internet). This "shorthand" indicates that firewalls are not (or are no longer) configured in all service chains. This notification may well be of greater concern to the service policy manager, who may, in consequence, cause traffic to be suspended, or issue a service deletion or suspension command to the Intent NBI engine (not shown).

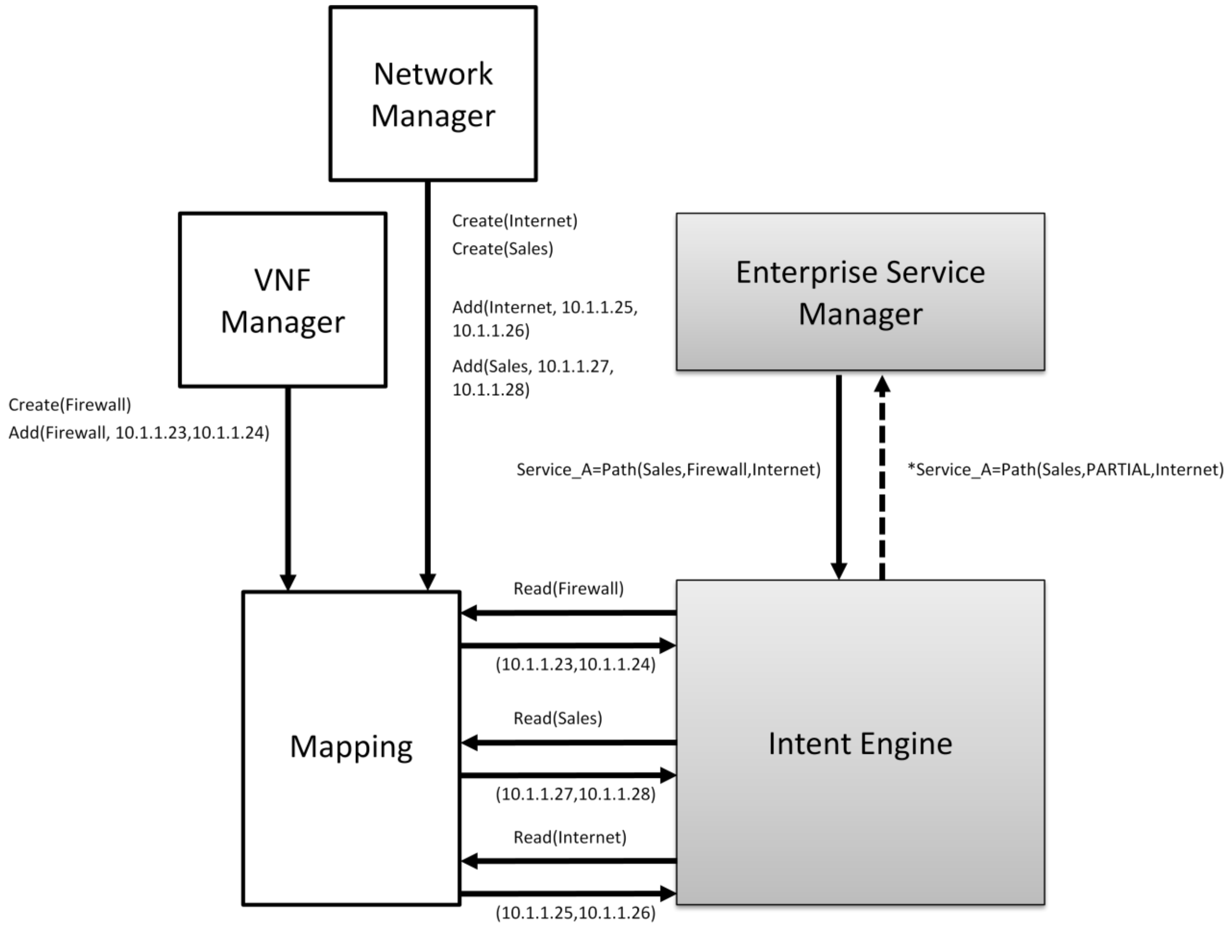
Network Manager
VNF Manager
Enterprise Service Manager
Mapping
Intent Engine
Create(Internet)
Create(Sales)
Add(Internet, 10.1.1.25, 10.1.1.26)
Add(Sales, 10.1.1.27, 10.1.1.28)
Create(Firewall)
Add(Firewall, 10.1.1.23,10.1.1.24)
Service_A=Path(Sales,Firewall,Internet)
*Service_A=Path(Sales,PARTIAL,Internet)
Read(Firewall)
(10.1.1.23,10.1.1.24)
Read(Sales)
(10.1.1.27,10.1.1.28)
Read(Internet)
(10.1.1.25,10.1.1.26)

Figure 6 An example of Intent NBI system operation for a simple service function chaining use case. The specific Intent NBI and mapping API syntax/expressions are hypothetical examples.

The specific Intent NBI and mapping API syntax used above, is meant only as hypothetical example. But it illustrates that there is no reason why Intent NBIs need to use object model or similar specific constructions. In general, more intuitive/natural (for humans) formulations may be envisaged. In a concrete implementation, any such syntax would need clearly defined meanings to ensure proper use by consumers and proper interpretation by the Intent engine. The Intent engine would parse and convert expressions as required into object model or other forms as appropriate for internal operations and interaction with controllers. Intent NBI expression syntax may or may not be standard, or fully standard. Specifically, a core set of expressions might be agreed (by standard or de facto standard definitions) but case-specific extensions may be easily and freely defined. The impacts of such extensions are limited, effectively, to the Intent Active Loop portion of the Intent engine implementation we described in the preceding section.

7 Intent NBI: Benefits

Let us now (re-)examine some general properties and benefits of Intent NBI.

An Intent NBI request is **non-prescriptive** with respect to detailed provider implementation of a request. Choices regarding technology, vendor, media, path, etc. are left to the provider (i.e., in SDN, the controller) and therefore such parameters never appear in Intent NBI requests. This has affirmative value, including in affording providers the greatest degrees of freedom in fulfilling service requests, for example in seeking greatest resource efficiencies, or reacting to changing resource and operational circumstances; in hiding implementation complexities from consumers; and in allowing for independent evolution of consumer and provider systems.

An Intent NBI request is **independent** of provider implementations and their operational policies. An Intent NBI request is, at a point in time, determined only by consumer operational and policy considerations and is expressed strictly in consumer frame-of-reference terms; the same Intent request would be made by a consumer to any provider regardless of details of provider implementation and controlled resource particularities. Mappings (exclusively) provide the bridge between consumer and provider frame-of-reference terms.

The Intent NBI paradigm is **universal** (i.e., is universally applicable), in the sense that it is always possible (if not necessarily always strictly desirable, given the detailed role and function of the consumer) for a consumer to express its service requirements in Intent NBI paradigm-compatible terms.

Intent NBI may **mitigate resource allocation conflicts** that otherwise might arise among concurrent consumer service requests to a given provider. This useful property derives from non-prescription: since Intent NBI requests do not specify what resources providers must allocate to specific services, providers gain relative freedom to assign resources to services in ways that reduce – and perhaps strictly minimize – the possibility of potentially irremediable resource contention among services.

Intent NBI requests may be **composable**, in the sense that Intent NBI requests may represent the effective sum of multiple specific inputs. An example is seen in the potential Intent NBI request given section 6: “apply security package A and filter package B to Bob’s internet traffic”. The operating consumer might specify that “Bob is allowed to access the internet”; a consumer-related security authority might additionally specify that “if Bob is allowed to access the internet, then apply security package A and filter package B to his internet traffic”. Composition may map to useful concepts of micro-services.

Intent-based systems may be more **secure** than prescription-based systems**.** There are a wide variety of security vulnerabilities that have been identified for systems that allow direct access to forwarding tables (e.g., via OpenFlow) or that, more generally, are based on resource prescription. In an Intent NBI-based system, a smaller security attack surface exists, because the Intent NBI exposes no ability for consumers directly to control any aspect of provider resource management – the details of which could otherwise affect other services, belonging to other consumers.

By architectural definition, per [1], an Intent NBI is an interface northbound from a provider (e.g., a controller). However, an Intent NBI may in practice be implemented as an "engine" in **overlay** to a (non-Intent NBI) provider implementation. This approach may present a number of practical benefits as discussed in sections 6.5 and 6.6.

Appendix A: Back Matter

A.1 References

[1] ONF TR-521, SDN architecture, Issue 1.1, 2016 (onf2015.420)

A.2 Contributors

Key contributors to the discussions that have supported generation of this document and shaped its contents include:

Christopher Janz, Huawei (Principal Author)

Nigel Davis, Ciena

David Hood, Ericsson

Mathieu Lemay, Inocybe

David Lenrow, Huawei

Li Fengkai, Huawei

Fabian Schneider, NEC

John Strassner, Huawei

Andrew Veitch, NEC-Netcracker