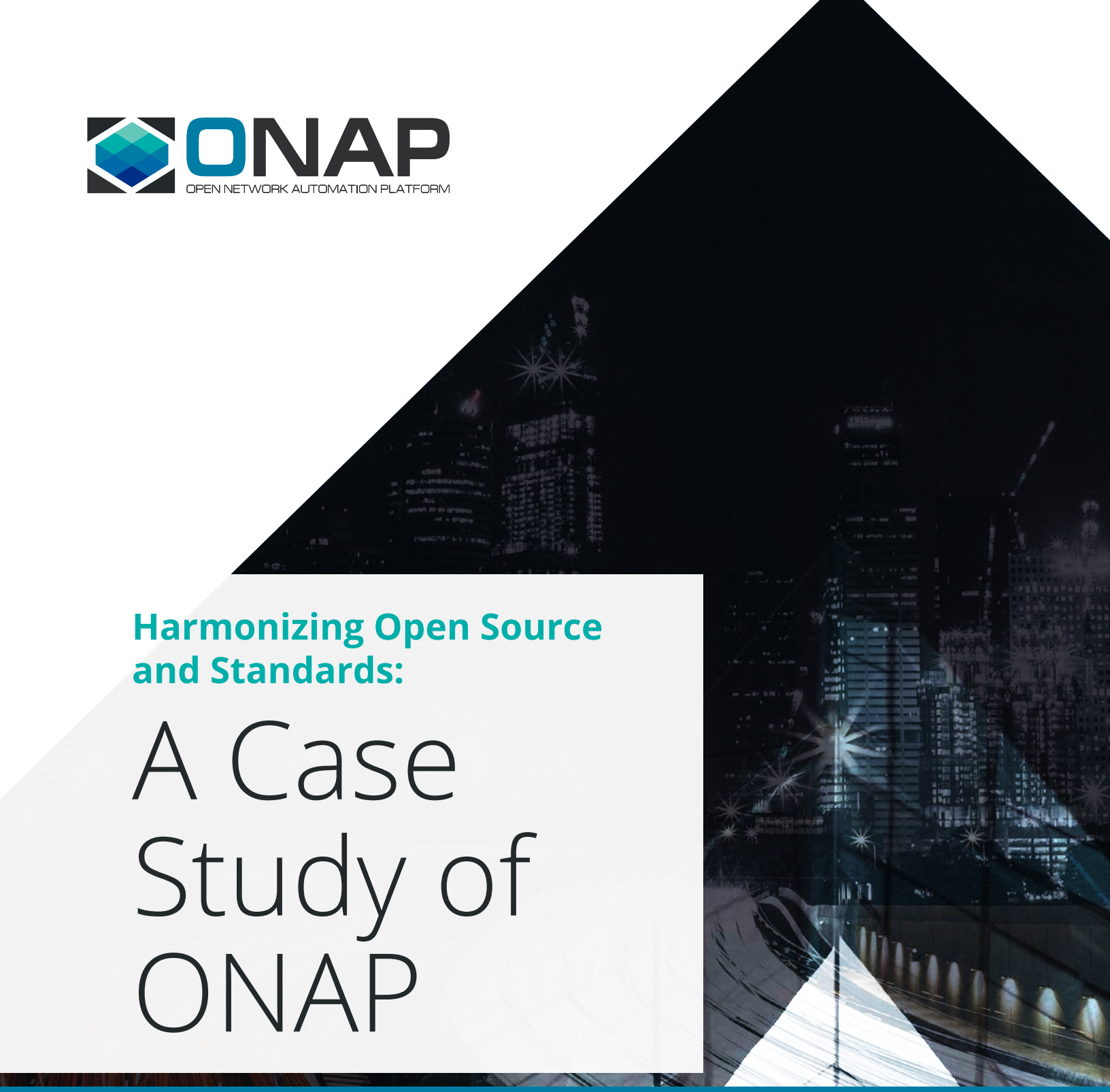

ONAP
OPEN NETWORK AUTOMATION PLATFORM

Harmonizing Open Source and Standards:

A Case Study of ONAP

Draft by Jenny Huang (AT&T),
Lingli Deng (CMCC), and
Hui Deng (Huawei)

As part of the broader evolution of open networking, the Linux Foundation networking projects have been working closely with a range of networking standards groups to align complementary efforts. This work has been described in "Harmonization 2.0: How Open Source and Standards Bodies Are Driving Collaboration Across IT."

This paper provides a closer look at the ONAP (Open Network Automation Platform) project within the Linux Foundation in order to provide concrete details about what standards might be related for ONAP project and what ONAP is doing on harmonizing open source and standards. We focus on three areas of ONAP-related industry standards and best practices: architecture, model-driven approaches, and APIs. By sharing our experiences to date, we hope to stimulate broader industry contributions towards shared objectives.

1. ONAP ARCHITECTURE

1.1 ARCHITECTURE DESIGN PRINCIPLES

ONAP is a platform above the network infrastructure layer that automates the operation and management of the entire network—that is, both virtual and physical network functions. It allows operators to connect their products and services through the infrastructure and scale the network in a fully automated manner.

In other words, ONAP aims to provide a utility network abstraction to the business layer, making services that demand just-in-time networking capabilities more attainable.

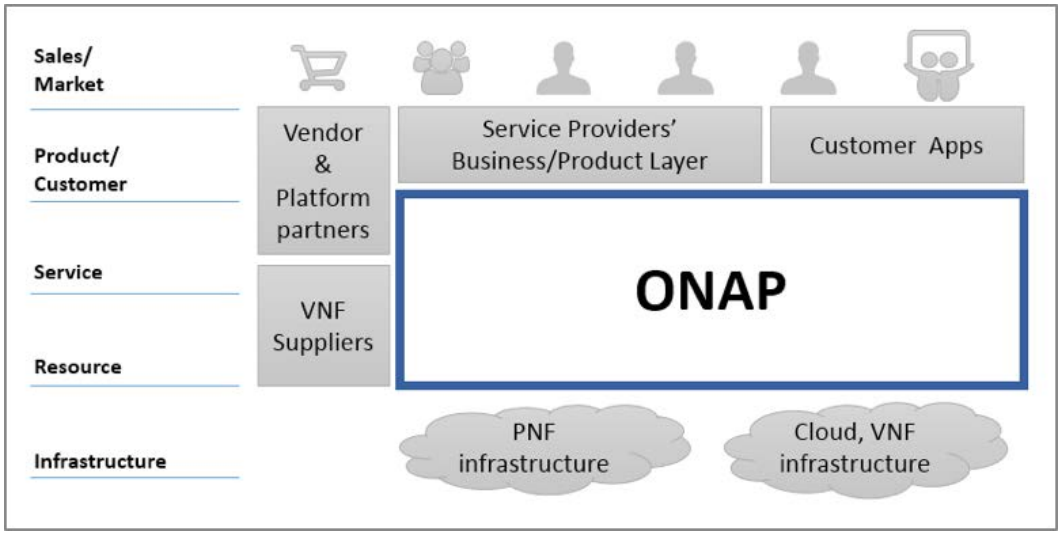

Figure 1 Scope of ONAP and Its Ecosystem

The Separations of Concerns design pattern is key in modeling the scope of ONAP, as shown on the left-hand side of Figure 1. ONAP is focused on modeling the information and related management functions in the **service** and **resource** layers, while its entire ecosystem spans many vertical industries and business scenarios, from end-user products to infrastructure layer.

There are a few ONAP architecture design principles that guide the realization of the platform[1,2]:

1. ONAP creates an open, model- and metadata-driven reference platform for service providers to support full lifecycle management of cloud-centric, software-controlled networks (SDN / NFV). The target goals include:
    - A modular, model-driven, and microservices-based architecture,
    - A layered management architecture including orchestrator, controllers, and multi-cloud (multi-VIM) infrastructure abstractions, and
    - Well-defined APIs for all modules to foster interoperability both within ONAP and across complementary projects and applications

2. ONAP must support a common approach to manage various network functions and related lifecycle management from different vendors. This approach includes:
    - All ONAP platform modules must be product/service/resource-agnostic, with a common information model for all vendors to follow, and
    - Support standards for consistency across vendor products, such as standard templates for instantiations, standard language for configuration, standard telemetry for monitoring and management, and so on.

3. Enable service providers to define and onboard resources to support any type of infrastructure and services, and to define analytics and policies that will be used at runtime. The design goals include:
    - Unified models between design-time and runtime modules to facilitate end-to-end, zero-touch operations,
    - Well-defined northbound APIs for all modules, and
    - A central design studio where all required artifacts are designed, tested/certified and distributed.

1.2 RELATED STANDARDS DEVELOPMENT ORGANIZATIONS (SDOS)

Figure 2 is a snapshot of the ONAP Beijing release architecture with modules that are either influencing or relying on industry standards highlighted in orange.

[1] ONAP Project Charter
[2] ONAP Architecture Principles

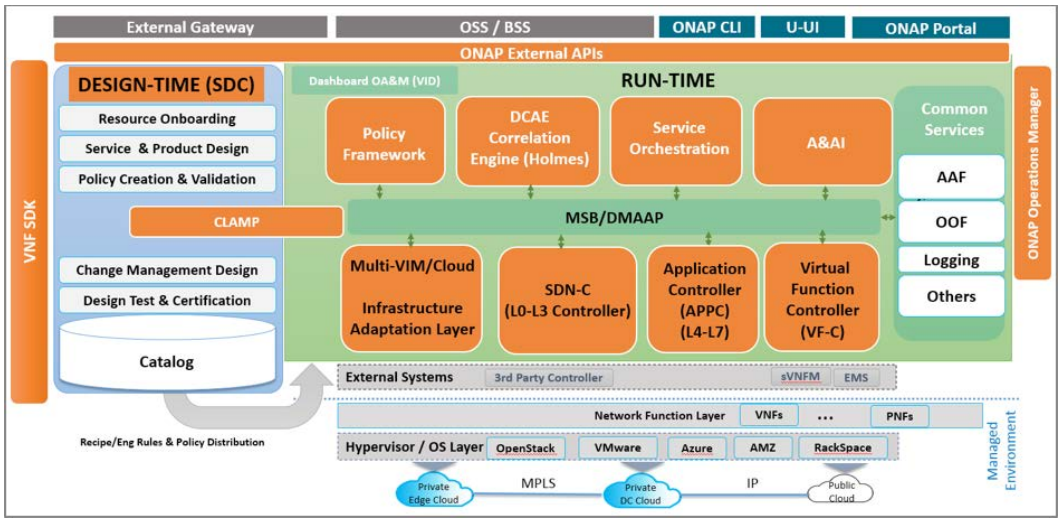

Figure 2 ONAP Modules Related to Standards

Table 1 shows the relationship between various ONAP Platform modules and related SDOs in a more detailed manner by grouping those modules highlighted in Figure 2 into Design Time and Runtime modules, describing their functions and the related SDOs.

ONAP Platform Modules	Key Functions	Related SDOs
Design Time		
SDC, VNF-SDK, VVP	Catalog management	TM Forum, MEF, ETSI NFVO
	Onboard VNF	TM Forum, ETSI NFVO
	Services and Operations Design	TM Forum, MEF, OASIS TOSCA
	Test, certify, and distribute models for Runtime Execution	ETSI NFV plugtests, OPNFV
Closed-loop Control Design	CLAMP Design Artifacts	ETSI ZSM, TM Forum
	Policy Design Artifacts	ETSI ZSM, TM Forum
Run Time		
External Framework APIs	Expose ONAP capabilities to OSS/BSS and partner ecosystems	(see more details in API section)
OOM	ONAP Operations Manager	TM Forum, OASIS TOSCA
Orchestrator	Service coordination, instantiation, and lifecycle management	MEF, ETSI NFVO, TM Forum
Generic NF controller*	Resource Lifecycle Management	ETSI NFV (VNF)
	Resource Configuration	3GPP SA5 EMS
SDN-C controller	Common SDN management abstraction	ONF, IETF
Close Loop Control Runtime	DCAE	3GPP SA5, ETSI ZSM
	CLAMP	ETSI ZSM
	Policy	ETSI ZSM

Table 1 ONAP Architecture and Related SDOs

*Note. Generic NF controller is a functional module which is implemented by VF-C and APP-C.

2. ONAP MODELING

2.1 A MODEL-DRIVEN APPROACH

Model-driven is a widely adopted principle of IT system design, and often a business requirement in large enterprises or complex ecosystem operations.

In this approach, the business logic of the software application is specified through the model at a higher level of abstraction, which is decoupled from the implementation code in a specific programming language. Running code can be generated or behaviours can be changed through model transformation techniques, such as code generation or interpreting/executing the models. Therefore, a model-driven approach enables enterprises to sustain technology changes and gain the agility to support multiple business and service scenarios.

For example, within the current ONAP release ('Amsterdam'), only a few modules are using the model-generated code, such as A&AI. The majority of the ONAP core modules are "template-driven," i.e., using the common execution engine as a service-independent platform to parse and execute templates for services and resource lifecycle management. Those models/templates are described in domain-specific languages (DSLs), such as TOSCA, YANG, etc.

To support service and resource management that is model/template-driven, ONAP features the separation of Design Time and Run Time environments: the Service Design & Creation module (SDC) in Design Time is responsible for the design, encapsulation, certification, and distribution of the related models/templates; the Run Time modules are responsible for parsing and executing the distributed templates.

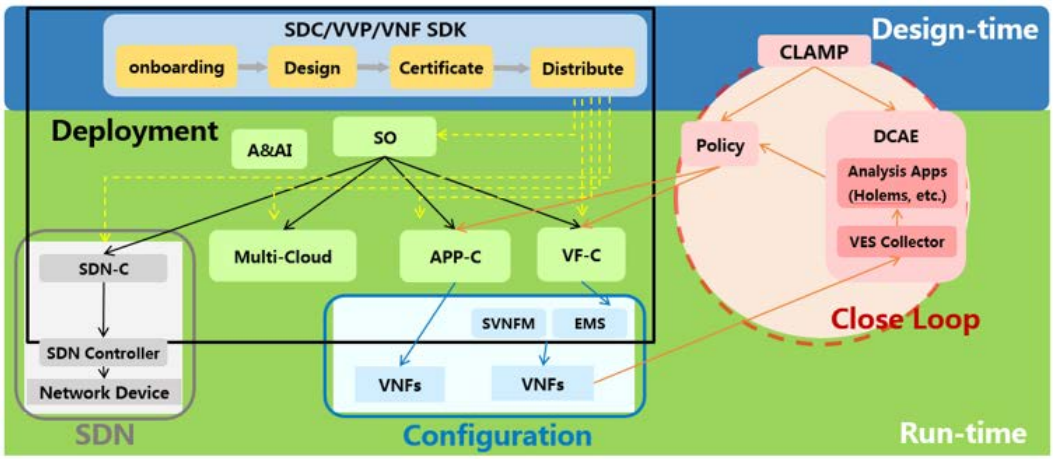

Figure 3 ONAP Modelling Scope/Distribution

As shown in Figure 3 above, there are four modeling domains in ONAP: deployment, closed-loop, SDN, and configuration. Further, each domain model can be subdivided into an information model and a data model: The information model describes the concept and the relationship among those concepts at an abstract level, while the data model adheres to the semantics described in the information model with strict syntax specifications within its domain. The data model facilitates system coding without ambiguity.

For example, with the template-driven approach, there is no need for any code modification to ONAP when deploying a new service if its deployment requirements can be described with the ONAP information model using ONAP data modeling templates.

2.2 RELATED SDOS

ONAP has set up a modeling subcommittee to work on unified modeling across modules in the community. Following are the external SDOs which may be related to in that work.

Scope	Model Types	Components	SDOs
Deployment	Service/Resource Topology, LCM Workflow, Policy, etc.	SO, VF-C, Multi-Cloud	TM Forum, ETSI NFV, OASIS TOSCA,
Closed-Loop	Data Collection, Analysis Rules, Automatic OPs Policy, etc.	CLAMP, DCAE, Policy	OASIS TOSCA, MEF, IETF
SDN	SDN Device Configuration and Management	SDN-C	ONF, MEF, IETF
Configuration	VNF Application Configuration	APP-C, VF-C	3GPP SA5

Table 2 ONAP Modelling Related SDOs and Open Source Projects

The ONAP modeling subcommittee might implement or refer to several industry standards in its upcoming Beijing Release for deployment modeling as shown in TABLE 3:

Model Types	Information Model Standards	Data Modeling Standards
Topology Model	TM Forum SID, ETSI NFV MANO, ONF CORE, OASIS TOSCA	OASIS TOSCA, OpenStack HOT
Workflow Model	ETSI NFV MANO, OASIS TOSCA	BPEL, OASIS TOSCA
Homing Policy Model	ETSI NFV MANO, IETF SUPA, MEF	OASIS TOSCA, IETF SUPA, MEF

Table 3 ONAP Deployment Model and Related SDOs

3. ONAP APIS

3.1 ONAP API DESIGN PRINCIPLES

To enable service providers and users of ONAP to quickly integrate ONAP with their existing systems, such as the OSS/BSS, ONAP embraces an architecture with well-defined APIs that fosters interoperability both within ONAP and across complementary projects and applications.

The ONAP API design principles include:

- Support for self-service & user-focused business objectives,
- Ease of integration via standardized APIs, and
- Model-driven approach (API code generation instead of static coding per scenario) and agnostic to VNF, resource, product, and service type

There are two categories of APIs in the ONAP platform, which adhere to the above design principles:

1. ONAP External APIs: These allow ONAP to be viewed as a "black box" by providing an abstracted view of the ONAP platform's capabilities. They can also be used for connecting to systems where ONAP uses the capabilities of other systems.
2. ONAP Internal APIs: These are APIs exposed by individual ONAP modules with the primary goals of exchanging information with other modules and jointly fulfill the functions provided by ONAP.

The ONAP External API Framework project (ExtAPI, also shown in Figure 2) provides the entry point for external API interfaces for the northbound OSS/BSS interface. It shields the ONAP details from the consumer interfaces as well as providing the consistency required for internal modules, such as authentication and authorization.

3.2 RELATED SDOS

Currently, most of the ONAP APIs are ONAP-specific. As we continue with our standards harmonization and alignment efforts, we expect ONAP internal APIs will become standards-compliant, and in turn, ONAP may influence the industry standards development as well.

The following table outlines the standards organizations which may be related to ONAP APIs development.

Purpose	Standards	Remarks
Northbound: **OSS/BSS APIs**	TM Forum APIs MEF Legato Reference Point ETSI NFV – SOL 005	TM Forum Open APIs and their Poly-morphism design pattern provide intent-based, product, service resource agnostic APIs. MEF LSO framework and ETI NFV SOL 005 provide more detailed and service-specific interaction patterns and data payload
East-West: **Partner, Developer APIs**	TM Forum APIs MEF Interlude Reference Point	Same as above
Southbound: **Resource Management APIs**	ONF TAPI, TM Forum API (HIP), 3GPP and others. MEF Adagio and Presto Reference Points ETSI NFV – SOL 003	Technology & domain-specific resource management interface will be embraced. E.g., 3GPP for wireless, TMF, ONF for PNF, VNF, fixed and wireline, MEF for Ethernet, etc.
ONAP Internal APIs	MEF LSO Presto Reference point ETSI NFV – SOL 003 and 005	

Table 4 ONAP APIs and SDO Collaborations

4. OPEN SOURCE AND STANDARDS COMMUNITIES WORKING TOGETHER

ONAP takes both top-down and bottom-up approaches to harness the differences and complementary features between the open source and standards communities. There are several significant efforts that the standards organizations, ONAP and other open source communities have taken to achieve the progress thus far as discussed in early part of this paper.

- **Harmonizing IPR modes of standards organizations and open source projects**: Sometimes a standards effort will also create a reference implementation or snippets of code demonstrating an implementation. This implementation might quite valuable in furthering an open source project, but the standards licensing model might be incompatible with inclusion in an open source project. Unfortunately, this issue must be made on a case by case based on respecting each others' IPR modes, as each standards body has its own specific IP rules and governance, especially for code-based contributions such as APIs.
- **Establishing a two-way synchronization/feedback loop:** Although most of the standards organizations have adopted agile development methodologies and practices that mirror those used in open source projects, the synchronization between any two communities (regardless of whether it's an open source project, standards group) still largely relies on community members to ensure changes made in the open source implementation are included in the next release of the standards development and vice versa. Hence, decoupling the ONAP schedule from SDO schedule while establishing a two-way feedback loop is key to accelerating innovation and expanding the solution ecosystem.

  A major success factor relies on having high-quality tracking methods and governance. In addition, open source collaboration should be a part of the strategic program within the standards organization to ensure the feedback loop is there.

  Besides feeding open source enhancement back to the standards bodies, a few standards organizations are also using proof-of-concept projects to introduce new features into open source. Proof-of-concept projects that use ONAP-defined use cases enable SDOs to focus, prioritize their work accordingly, and to identify complementary areas where SDOs might lead.
- **Continuously learning and harmonizing** is important to ONAP. ONAP has an SDO Coordination function under the Technical Steering Committee (TSC), which consists of volunteers active in both the affiliated standards organization and ONAP projects. The sub-committee carefully provides permitted updates of the latest and applicable standards development to the ONAP community and coordinates the joint development efforts as discussed in this paper.
- **People make the difference:** Despite all the processes and governance that are put in place to foster collaboration between open source and standards communities, incorporation of standards into open source implementations requires careful architecture planning, community consensus, and code implementation. It is the individuals who invest the time and careful focus that is required to bridge both communities who will make this effort successful. It is truly a labor of love for those who believe in standards and who invest the time to make it happen in the open source projects.

5. SUMMARY

For the first time, the networking industry is researching and developing together in the open among service providers and vendors. It is evident that as ONAP matures, with more platform capabilities introduced in each release, standards become increasingly important to ensure an extensible and interoperable ecosystem that the ONAP platform can support.

ONAP strives to use best-of-breed standards and technologies in achieving this goal; it provides a proving ground for the benefits of both communities, and the results have been promising.

6. ACRONYMS

	ONAP Specific Terms		General Industry Terms
CLAMP	Closed Loop Automation Management Platform	DSL	Domain Specific Language
DCAE	Data Collection, Analytics, Events	IPR	Intellectual Property Rights
ExtAPI	External API Framework Module	LCM	Lifecycle Management
OOM	ONAP Operations Manager	NFV	Network Function Virtualization
SDC	Service Design and Creation Module	OSS/BSS	Operations Support Systems/Business Support Systems
SO	Service Orchestator Module	PNF	Physical Network Function
TSC	Technical Steering Committee	RAND	Reasonable and non-discriminatory, a form of IPR terms
VF-C	Virtual Function Controller	SDK	Software Development Kit
VVP	VNF Validation Project	SDN	Software Defined Network
		SDOs	Standards Development Organizations
		VIM	Virtual Infrastructure Manager
		VNF	Virtual Infrastructure Manager

Standards and Open Sources	
3GPP	The 3rd Generation Partnership Project http://www.3gpp.org
CNCF	Cloud Native Computing Foundation https://www.cncf.io/
ETSI	European Telecommunications Standards Institute http://www.etsi.org/
IETF	Internet Engineering Task Force https://www.ietf.org/
LSO	Lifecycle Services Orchestration, a specification developed by MEF
MEF	Metro Ethernet Forum http://www.mef.net/
NFVO	Network Function Virtualization Orchestrator, a key component from ETSI MANO specification
OASIS	Organization for the Advancement of Structured Information Standards https://www.oasis-open.org/
ONF	Open Networking Foundation https://www.opennetworking.org/
Open Daylight	https://www.opendaylight.org/
OpenStack	https://www.openstack.org/
OPNFV	https://www.opnfv.org/
SA5	Telecom Management working group under 3GPP
TM Forum	https://www.tmforum.org/
TOSCA	Topology and Orchestration Specification for Cloud Applications, a specification of OASIS
YANG	A data modeling language spec. developed by IETF
ZSM	Zero touch network and Service Management, an Industry Specification Group under ETSI