ETSI GR ENI 003 V1.1.1 (2018-05)

Experiential Networked Intelligence (ENI); Context-Aware Policy Management Gap Analysis

Disclaimer

The present document has been produced and approved by the Experiential Networked Intelligence (ENI) ETSI Industry Specification Group (ISG) and represents the views of those members who participated in this ISG.
It does not necessarily represent the views of the entire ETSI membership.

Reference
DGR/ENI-003

Keywords
management, network, policy management

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - NAF 742 C
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° 7803/88

Important notice

The present document can be downloaded from:
http://www.etsi.org/standards-search

The present document may be made available in electronic versions and/or in print. The content of any electronic and/or print versions of the present document shall not be modified without the prior written authorization of ETSI. In case of any existing or perceived difference in contents between such versions and/or in print, the only prevailing document is the print of the Portable Document Format (PDF) version kept on a specific network drive within ETSI Secretariat.

Users of the present document should be aware that the document may be subject to revision or change of status. Information on the current status of this and other ETSI documents is available at
https://portal.etsi.org/TB/ETSIDeliverableStatus.aspx

If you find errors in the present document, please send your comment to one of the following services:
https://portal.etsi.org/People/CommiteeSupportStaff.aspx

Contents

Intellectual Property Rights

Essential patents

IPRs essential or potentially essential to normative deliverables may have been declared to ETSI. The information pertaining to these essential IPRs, if any, is publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (https://ipr.etsi.org/).

Pursuant to the ETSI IPR Policy, no investigation, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Trademarks

The present document may include trademarks and/or tradenames which are asserted and/or registered by their owners. ETSI claims no ownership of these except for any which are indicated as being the property of ETSI, and conveys no right to use or reproduce any trademark and/or tradename. Mention of those trademarks in the present document does not constitute an endorsement by ETSI of products, services or organizations associated with those trademarks.

Foreword

This Group Report (GR) has been produced by ETSI Industry Specification Group (ISG) Experiential Networked Intelligence (ENI).

Modal verbs terminology

In the present document "**should**", "**should not**", "**may**", "**need not**", "**will**", "**will not**", "**can**" and "**cannot**" are to be interpreted as described in clause 3.2 of the ETSI Drafting Rules (Verbal forms for the expression of provisions).

"**must**" and "**must not**" are **NOT** allowed in ETSI deliverables except when used in direct citation.

Introduction

A critical foundation of experiential networked intelligence is context- and situation-awareness. The present document will analyse work done in various SDOs on policy management in general, and context-aware policy management specifically, to determine what work can be reused from external SDOs, and what work needs to be developed in ENI. When gaps are found on existing interfaces that have been developed by other SDOs and ISG ENI needs to reuse, then the recommendation on how these gaps should be filled will be discussed in co-operation with the SDO that defined these interfaces within Phase 1 and beyond. The requirements documented in the present document will be considered during the architecture design work.

1 Scope

The present document analyses the work done in various SDOs and open source consortia on policy-based modelling. This information will be used to develop a specification for a context-aware, policy-based management model and architecture for enhancing the operator experience through the use of network intelligence.

2 References

2.1 Normative references

Normative references are not applicable in the present document.

2.2 Informative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long term validity.

The following referenced documents are not necessary for the application of the present document but they assist the user with regard to a particular subject area.

[i.1] IETF draft-ietf-supa-generic-policy-info-model-03: "Generic Policy Information Model for Simplified Use of Policy Abstractions (SUPA)", May 30, 2017.

[i.2] Strassner, J.: "Policy-Based Network Management", Morgan Kaufman, ISBN 978-1558608597, September 2003.

[i.3] Liskov, B.H.: Wing, J.M., "A Behavioral Notion of subtyping", ACM Transactions on Programming languages and Systems 16 (6): 1811 - 1841, 1994.

[i.4] Martin, R.C.: "Agile Software Development, Principles, Patterns, and Practices", Prentice-Hall, 2002, ISBN: 0-13-597444-5.

[i.5] Strassner, J.: editor: "MEF Technical Specification: Policy-Driven Orchestration", Call for Comments v0.7, August 2017.

[i.6] Riehle, D.: "Composite Design Patterns", Proceedings of the 1997 Conference on Object-Oriented Programming Systems, Languages and Applications (OOPSLA '97), ACM Press, 1997, pages 218-228.

[i.7] Davy, S., Jennings, B., Strassner, J.: "The Policy Continuum - A Formal Model", Proc. of the 2^{nd} Intl. IEEE Workshop on Modeling Autonomic Communication Environments (MACE), Multicon Lecture Notes, No. 6, Multicon, Berlin, 2007, pages 65-78.

[i.8] Gamma, E., Helm, R., Johnson, R., Vlissides, J.: "Design Patterns - Elements of Reusable Object-Oriented Software", Addison-Wesley, 1994, ISBN 0-201-63361-2.

[i.9] Strassner, J., de Souza, J.N., Raymer, D., Samudrala, S., Davy, S., Barrett, K.: "The Design of a Novel Context-Aware Policy Model to Support Machine-Based Learning and Reasoning", Journal of Cluster Computing, Vol 12, Issue 1, pages 17-43, March, 2009.

[i.10] MEF: "Lifecycle Service Orchestration Architecture: Reference Architecture and Framework", MEF 55, March, 2016.

[i.11] MEF: "Service Orchestration Functionality", Call for Comments v0.7, August 2017.

[i.12] MEF: "Policy Driven Orchestration Kickoff v3".

NOTE: Available at https://wiki.mef.net/display/MTA/PDO+Contributions.

[i.13] Dey, A.: "Providing architectural support for building context-aware applications". Ph.D. Thesis (2000).

[i.14] MEF: "MEF Core Model", Call for Comments v0.95, January 2018.

[i.15] ETSI ISG ENI(17)0002-014r3: "Improved Operator Experience through Experiential Networked Intelligence (ENI)", first edition, May 2017.

[i.16] IETF draft-ietf-supa-generic-policy-data-model-04: "Generic Policy Data Model for Simplified Use of Policy Abstractions (SUPA)", June 18, 2017.

[i.17] Standford Encyclopedia of Philosophy: "Deontic Logic".

NOTE: Available at https://plato.stanford.edu/entries/logic-deontic/.

[i.18] Standford Encyclopedia of Philosophy: "Modal Logic".

NOTE: Available at https://plato.stanford.edu/entries/logic-modal/.

[i.19] IEFT: "Simplified Use of Policy Abstractions (supa)".

NOTE: Available at https://datatracker.ietf.org/wg/supa/about/.

[i.20] IETF draft-ietf-supa-policy-based-management-framework-03: "SUPA Policy-based Management Framework", July 2017.

[i.21] IETF draft-cheng-supa-applicability-01: "Applicability of SUPA", March 2017.

[i.22] TM Forum: "Information Framework (SID); Common Business Entities - Policy", GB922 Policy, Release 14.5.1, March 2015 (part of Release 17.0).

[i.23] TM Forum: "Information Framework (SID); Common Business Entities - Root Business Entities", GB922 Root, Release 17.0.0, June 2017 (part of Release 17.0).

[i.24] Strassner, J.: "Using the MEF Core Model in ONAP", December 2017.

[i.25] MEF PDO CfC: "Policy-Driven Orchestration", v0.8, February 2018.

[i.26] MEF: "Lifecycle Service Orchestration (LSO): Reference Architecture and Framework", MEF 55, March 2016.

[i.27] IETF RFC 3060: "Policy Core Information Model -- Version 1 Specification".

3 Definitions and abbreviations

3.1 Definitions

For the purposes of the present document, the following terms and definitions apply:

abstraction: process of focusing on the important characteristics and behaviour of a concept, and realizing this as a set of one or more elements in an information or data model

NOTE: When applied to modelling, it defines a generic set of characteristics and behaviours for a class that all of its subclasses inherit.

action: set of operations that may be performed on a set of managed entities. It represents a transformation or processing in the system being modelled

NOTE: An Action either maintains the state, or transitions to a new state, of the targeted managed entities. The execution of an Action may be influenced by applicable attributes and metadata [i.25].

capability: set of features that are available from a component

NOTE: These features may, but do not have to, be used. All Capabilities should be announced through a dedicated Interface.

condition: set of attributes, features, and/or values that are to be compared with a set of known attributes, features, and/or values in order to determine what decision to make

data model: representation of concepts of interest to an environment in a form that is dependent on data repository, data definition language, query language, implementation language, and/or protocol (typically, but not necessarily, all five)

declarative policy: type of policy that uses statements to express the goals of the policy, but not how to accomplish those goals

NOTE 1: State is not explicitly manipulated, and the order of statements that make up the policy is irrelevant.

NOTE 2: In the present document, Declarative Policy will refer to policies that execute as theories of a formal logic [i.25].

event: anything of importance to the management system (e.g. a change in the system being managed and/or its environment) occurring on a time-axis

formal logic: use of inference applied to the form, or content, of a set of statements

NOTE: The logic system is defined by a grammar that can represent the content of its sentences, so that mathematical rules may be applied to prove whether the set of statements is true or false [i.25].

formal methods: set of mathematical theories, such as logic, automata, graph or set theory, and provide associated notations for describing and analysing systems

functional block: modular unit that defines the properties, behaviour, and relationships of a part of a system. Some functional blocks may also define relationships to other functional blocks outside of its enclosing system

imperative policy: type of policy that uses statements to explicitly change the state of a set of targeted objects

NOTE 1: The order of statements that make up the policy is explicitly defined.

NOTE 2: In the present document, Imperative Policy will refer to policies that are made up of Events, Conditions, and Actions [i.25].

information model: information model is a representation of concepts of interest to an environment in a form that is independent of data repository, data definition language, query language, implementation language, and protocol

intent policy: type of policy that uses statements in a natural language to express the goals of the policy, but not how to accomplish those goals

NOTE 1: Each statement in an Intent Policy may require the translation of one or more of its terms to a form that another managed functional entity can understand [i.25].

NOTE 2: In the present document, Intent Policy will refer to policies that do not execute as theories of a formal logic. They typically are expressed in a restricted natural language and require a mapping to a form understandable by other managed functional entities.

Lifecycle Service Orchestration (LSO): open and interoperable automation of management operations over the entire lifecycle of Layer 2 and Layer 3 Connectivity Services

NOTE: This includes fulfilment, control, performance, assurance, usage, security, analytics and policy capabilities, over all the network domains that require coordinated management and control, in order to deliver the offered Service [i.26].

LSO reference architecture: high-level functional architecture that characterizes the management and control domains and entities that make up a system, and the interfaces among them, to enable cooperative orchestration of offered Services

managedEntity: manageable object that may be related to a Product, Service, and/or Resource

NOTE: A ManagedEntity has the following common semantics:

1) each has the potential to be managed;

2) each can be associated with at least one ManagementDomain;

3) each is related to Products, Resources, and/or Services of the system being managed [i.14].

management: set of procedures that are responsible for describing, organizing, controlling access to, and managing the lifecycle needs of information and entities of an organization

management abstraction: abstraction used for management purposes

managementDomain: domain whose contents are governed using a common set of management mechanisms

NOTE: A ManagementDomain is a type of ManagedEntity that has 3 key characteristics:

1) it has a set of administrators defined to perform management operations on the ManagedEntities that it contains;

2) it defines a set of applications that are responsible for different governance operations, such as monitoring, configuration, and so forth;

3) it defines a common set of management mechanisms, such as policy rules, that are used to govern the behaviour of ManagedEntities contained in the ManagementDomain [i.14].

metadata: set of objects that contains prescriptive and/or descriptive information about the object(s) to which it is attached [i.14]

NOTE: While metadata can be attached to any information model element, the present document only considers metadata attached to classes and relationships.

meta-policy: policy that governs that operation, administration, and/or management of another set of policies

orchestration: set of processes that collectively automate the management and control of digital information systems

NOTE: Orchestration processes coordinate the actions of disparate systems and functions, ensuring that they all act towards a common set of goals.

party: abstract class that represents either an individual person or a group of people

NOTE: A Party may take on zero or more PartyRoles. A group of people can also be structured as an organization made up of organizational units [i.14].

PartyRole: abstract class, and specializes Role. It represents a set of unique behaviours played by a Party in a given context [i.14]

pattern: reusable, object-oriented framework to a commonly occurring problem

policy: set of rules that are used to manage and control the changing and/or maintaining of the state of one or more managed objects

Reference Point (RP): logical point of interaction between specific ManagedEntities

resource: Resource provides capabilities that may be consumed by different internal and external users [i.14].

NOTE: In addition, a Resource may consume other Resources. A Resource has a distinct state. Resources are typically limited in quantity and/or availability. Resources may be logical or virtual in nature [i.14].

service: service represents functionality that can be used by different internal and external users (e.g. a management system and a Customer, respectively) for different purposes [i.14]

NOTE: Services may be consumed by other Services, but not by Resources. A Service has a distinct state.

Service Orchestration Functionality (SOF): set of functional blocks and associated processes to translate requests from business and customer applications to a form that the infrastructure of the Service Provider can understand, and similarly, to translate responses from the infrastructure of the Service Provider to business and customer applications

NOTE: It also manages and controls the functional components that make up the infrastructure of the Service Provider.

3.2 Abbreviations

For the purposes of the present document, the following abbreviations apply:

AI	Artificial Intelligence
ANIMA	Autonomic Networking Integrated Model and Approach
API	Application Programming Interface
CLI	Command Line Interface
DSL	Domain-Specific Language
ECA	Event-Condition-Action
ENI	Experiential Networked Intelligence
EPRIM	Eca Policy Rule Information Model
FOL	First Order Logic
FTP	File Transfer Protocol
GBP	Group Based Policy
GPIM	Generic Policy Information Model
GS	Group Specification
ICM	Infrastructure Control and Management
IP	Internet Protocol
LDAP	Lightweight Directory Access Protocol
LF	Linux Foundation®
LSO	Lifecycle Service Orchestration
LSO RA	LSO Reference Architecture
MANO	MANagement and Orchestration
MCM	MEF Core Model
MEC	Multi-access Edge Computing
MEF	MEF Forum
NFVRG	Network Functions Virtualisation Research Group
ODL	OpenDayLight project
ONAP	Open Network Automation Platform
ONF	Open Networking Foundation
ONOS	Open Network Operating System
PBM	Policy-Based Management
PDO	Policy-Driven Orchestration
RA	Reference Architecture
RDBMS	Relational DataBase Management System
RP	interface Reference Point
SAP	SoftwAre Product for enterprise resource planning
SDN	Software Defined Networks
SID	Shared Information and Data model
SLS	Service Level Specification
SNMP	Simple Network Management Protocol
SOF	Service Orchestration Functionality
SUPA	Simplified Use of Policy Abstractions
SVGA	Super Video Graphics Array
TM	TeleManagement
TMF	TeleManagement Forum
UML	Unified Modeling Language
WG	Working Group

4 Introduction and Approach

4.1 Base Assumptions

ENI has recommended the use of *context-aware* policy management in its white paper. This is because one of the most important goals of ENI is to respond to dynamic changes. This is best handled by defining the concept of *context*, and then modelling context in the system, so that the Policy Management system can detect and respond to changes in context. In this respect, the goal of ENI is to detect changes in context, and as a result, to change the working set of policies being used. This causes the behaviour of the system being managed to be adjusted to follow changes in context (according to appropriate business goals and other factors, of course) in a closed loop manner.

One of the most popular definitions of context is: "Context is any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and application themselves" [i.13]. This definition has a number of shortcomings when applied to modern system and network management, as detailed in [i.9]. Therefore, the definition of context used in ENI is:

> *"The Context of an Entity is a collection of measured and inferred knowledge that describe the state and environment in which an Entity exists or has existed".*

This definition emphasizes two types of knowledge - facts (which can be measured) and *inferred* data, which results from machine learning and reasoning processes applied to past and current context. It also includes context history, so that current decisions based on context may benefit from past decisions, as well as observation of how the environment has changed [i.9].

4.2 Introduction to Policy Management

4.2.1 Policy Definition

The purpose of policies is to ensure that consistent decisions are made governing the behaviour of a system. More specifically, [i.5] and [i.12] provide the following definition of Policy:

> *"Policy is a set of rules that is used to manage and control the changing and/or maintaining of the state of one or more managed objects."*

Organizations are policy-driven entities. Policy is a natural way to express rules and restrictions on behaviour, and then automate the enforcement of those rules and restrictions. However, the number of policies can be very large (e.g. 100 000+), and the relationships between policies can be complex. In addition, policy can change *contextually.* For example, different actions can be taken based on type of connection, time of day, and network state.

4.2.2 Uses of Policy Management

There is a distinct difference between Policies that operate in a hierarchy of systems (e.g. north-south) vs. Policies that are exchanged between systems (e.g. east-west). The former is used to control behaviour, while the latter is used to represent and possibly negotiate behaviour. This is shown in Figure 4.2-1.

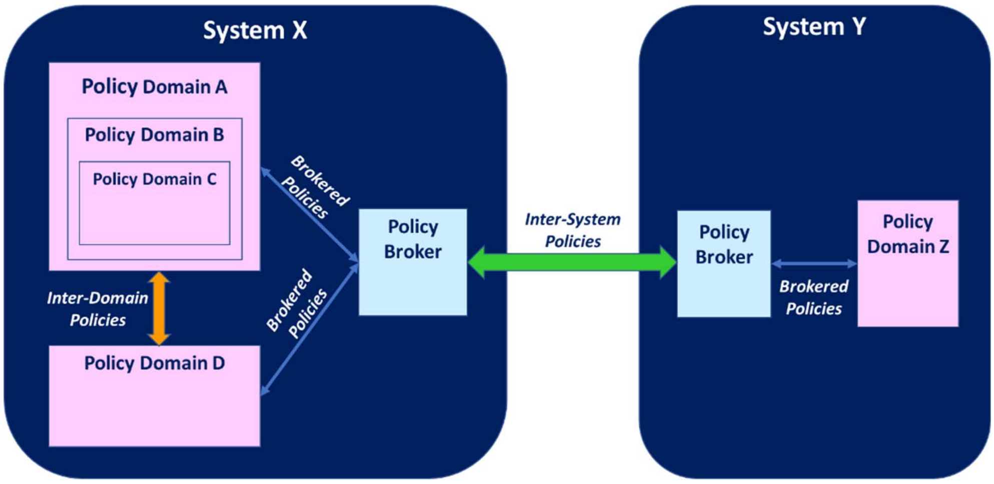

Figure 4.2-1: Policies Used in Management and Control vs. Negotiation

Systems X and Y are different systems (e.g. a Service Provider and a Content Provider) that wish to interact. Inter-System Policies can be used to negotiate behaviour, and use a functional block called a Policy Broker (see clause 4.11.4) to communicate with other Systems that use policy. For example, a Service Provider could specify that a particular service should include HD video; however, the Service Provider cannot tell the Content Provider how to build that service. Furthermore, the use of Policies enables the Content Provider to respond to the request and offer different behaviours based on the current context (e.g. upsell to 4K video, downgrade to SVGA, and so forth). This enables the behaviour of the service to be agreed upon by the Providers, and hence, provide a projected experience for the end user.

Figure 4.2-1 shows three types of Policies, defined as follows:

- Intra-Domain: a Policy exchanged between two domains that are both contained in the same higher domain (e.g. Policies exchanged between domains A and B, or domains B and C).
- Inter-Domain: a Policy exchanged between two domains that are NOT contained in the same higher domain (e.g. between domains A and D, or between B and D or C and D).
- Inter-System: a Policy exchanged between two systems.

The first case (intra-domain) defines Policies according to a strict hierarchy. Policies from the outermost domain should be obeyed by all of its contained domains. Hence, in order for such a Policy to be validated, there should not be any conflict between a newly added (or edited) Policy and all Policies that are in that domain, or in any containing domains.

The second case (inter-domain) defines Policies that are in sibling domains in the same system. The system may use one or more functional blocks to perform conflict detection and remediation.

The third case (inter-system) defines Policies that are in sibling systems. This means that each system should ensure that its policies are compatible with Policies in the systems that it is communicating with.

In general, model-driven engineering uses the abstractions of a *component,* a *module,* and a *system.* A component is the most granular level of reuse in a software-intensive system. Architecturally, it encapsulates a set of related functions, and offers services to the rest of the system via interfaces. This abstraction decouples the functionality offered by a component from its implementation.

A module is a set of related components that are assembled to form a higher-level function.

A system is a set of modules that performs a complete, working application.

Since Policies can be defined at any of these levels, and because Policy depends on context, a Policy Broker is required to communicate the semantics, constraints, and metadata associated with exchanging Policies between systems.

System X is a Carrier. System Y is a Content Provider. In each case, the brokered Policies are a proper subset of the policies from the outermost Domain (i.e. the Policies from a (hierarchical) domain represent a set of capabilities, and the brokered Policies are those policies that the containing System chooses to expose).