Information Model for Managing Autonomic Functions in Future Networks

Makis Stamatelatos[1], Imen Grida Ben Yahia[2], Pierre Peloso[3], Beatriz Fuentes[4], Kostas Tsagkaris[5], and Alex Kaloxylos[1]

[1] National and Kapodistrian University of Athens, Panepistimiopolis, Ilissia, 15784, Athens
{makiss,agk}@di.uoa.gr
[2] France Telecom, 78 Olivier de Serres, 75015, Paris
imen.gridabenyahia@orange.com
[3] Alcatel-Lucent Bell Labs, Site de Villarceaux, 91620, Nozay (France)
pierre.peloso@alcatel-lucent.com
[4] Telefónica I+D, Distrito C, Ronda de la Comunicación s/n, 28050 Madrid
fuentes@tid.es
[5] University of Piraeus, 80 Karaoli & Dimitriou, 18532, Piraeus
ktsagk@unipi.gr

Abstract. Future Internet (FI), a dynamic and complex environment, imposes management requirements, complexity and volume of data which can hardly be handled by traditional management schemes. Autonomic network and service management can be a powerful vision; a promising solution paving the way towards fully autonomic systems provides a three-level management approach and develops Information Modelling extensions for semantic continuity. This paper aims at proposing an Information Model for abstracting autonomic mechanisms for network management tasks and convincing on the relevance of using/extending standardized information models for system specification.

Keywords: Information Model, Autonomic Communications, Network Management, Future Networks.

1 Introduction

Operators today are facing large scale issues: they serve hundreds of millions of customers and mass of customization; they rely on thousands of different network elements with proprietary implementations; they spend M-euros for the adaptation and integration of Network Elements (NEs), Element Management Systems (EMSs) and Network Management Systems (NMSs); they need to handle thousands of alarms per day in each medium-size Network Operating Centre. In this sense, Operators are seeking for advanced management solutions implementing self-* functions to handle complexity, alleviating integration issues, reducing both CAPEX and OPEX and minimising Time-to-Market of new services.

Autonomic network management is expected to solve these issues, but this adoption is yet far from being generalized. That is why in UniverSelf project [1],

D. Pesch et al. (Eds.): MONAMI 2013, LNICST 125, pp. 259–272, 2013.

which has the goal to foster the conditions for such an adoption, the paradigm shifted from autonomic network management to management of autonomic functions which are themselves managing the network. To achieve such a result, a novel Unified Management Framework (UMF) has been specified. Hence, the UMF revolve around autonomic functions which embody intelligence into network entities (physical or virtualized). The set of specifications imposed to these autonomic functions define what is hereafter named Network Empowering Mechanisms (NEMs). UMF also defines a set of core functions, operations and mechanisms for the proper governance, coordination and knowledge exchange among the NEMs.

The design of an Information Model able to support the management operations of thousands of vendor-specific NEMs, becomes of utmost importance. This Information Model needs to support the seamless integration with existing management systems.

The aim of this paper is to propose subsets of Information Model for autonomic functions (NEMs) based on the TM Forum's Information Framework (SID) [2]. The extensions of the SID model were designed to achieve the specification of UMF interfaces and cover the NEMs structure and lifecycle of NEMs, actions and information manipulated by NEMs, as well as the policies driving the NEM behaviour. The rest of the paper is organized as follows. Section 2 presents the UMF and essential information flows. Section 3 presents the information modelling within the UMF framework whilst Section 4 details the UMF Information Model following SID patterns. Section 5 presents the usage of the UMF Information Model in deriving object-level information items utilised during the NEMs lifecycle.

2 UMF-Framework for Autonomic Mechanisms

UMF has been designed based on FI autonomic networks requirements and encompasses a set of functionalities resolving manifold networking problems. UMF primal objective is to enable trustworthy integration (plug-and-play) and interworking of NEMs within the operator's management ecosystem. The successful deployment of NEMs necessitates their governance/administration, their orchestration/coordination and corresponding information and knowledge sharing. These demands steered to introduction of the concept of UMF core (Fig. 1), which consists of three functional blocks: Governance (GOV), Knowledge (KNOW) and Coordination (COORD). UMF follows a three level approach; UMF manages NEMs which, in turn, manage and optimise network resources and services (Fig. 1).

NEMs specifications details that a NEM is designed as a piece of software implementing an algorithm forming a control loop that can be deployed in a (part of a) network to enhance/simplify its control and management (e.g. take over some operations). An intrinsic capability of a NEM is to be deployable and interoperable in an UMF context (e.g. an UMF compliant network).

The GOV block provides interfaces and functions for the Operators to deploy, pilot, control and track progress of NEMs in a unified way, including fusion of business goals and respective translation to NEM-policies thus realising the policy-continuum [3]; the COORD block provides tools for identifying and avoiding conflicts among autonomic functions (as realized by NEMs) and ensure stability and performance when several NEMs are concurrently working; the KNOW provides tools to make NEMs find, formulate and share relevant information and knowledge towards enabling or improving their operation.

Fig. 1 illustrates also the key information flows among UMF blocks as well as between the UMF and the NEMs. Operator's business goals and services description form the key input for GOV block which provides policies to the other UMF blocks. COORD sends the "Call-for-GOV" notification, in cases a problem needs direct invocation of GOV mechanisms; KNOW provides (i) COORD with information enabling NEM coordination, (e.g. NEM objective, type, etc) as well as network-level knowledge, and (ii) GOV with aggregated knowledge and performance measurements targeting policy evaluation (i.e. whether and in what extent the policies communicated by GOV have achieved the intended improvement on the network performance). Moreover, UMF supports certain interactions with NEMs, such as NEMs registration to the UMF core blocks; GOV communicates the NEM mandate (a "turn-on" command); COORD communicates the NEM Control Policy whilst KNOW communicates Information Exchange Policy (for knowledge production and sharing).

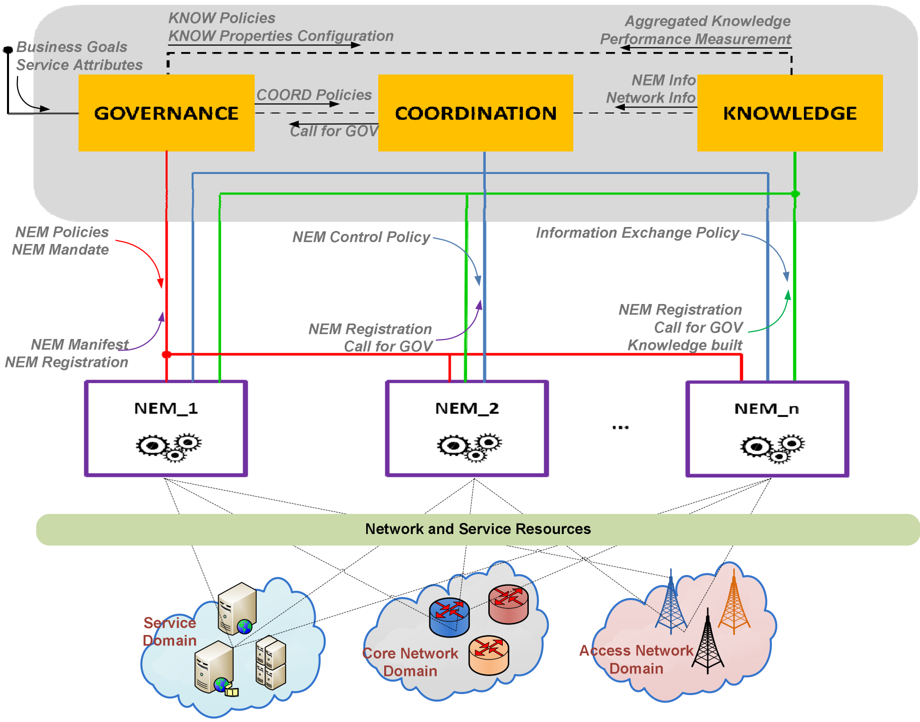

Fig. 1. UMF: 3-level management and information flows.

The need for an Information Model to formalize the interactions between NEMs and the UMF core and address interoperability among different NEM developers is apparent. Information Model enables semantic interoperability among management systems (e.g. EMS, NMS, etc.) which are being managed in a unified way by the UMF and in turn manage network and service resources through NEMs' deployment.

3 Information Modelling

An Information Model provides a system conceptualisation; several definitions have been provided by standardization fora and initiatives. According to IETF, an Information Model is "*an abstraction and representation of the entities in a managed environment including definition of their properties, operations and relationships. It is independent of any specific type of repository, software usage, platform, or access protocol.*" [4]. TMF states that "*an Information Model is a representation of business concepts, their characteristics and relationships, described in an implementation independent manner*" [2]. In 3GPP "*Information Model denotes an abstract, formal representation of entity types, including their properties and relationships, the operations that can be performed on them, and related rules and constrains.*" [5].

The presented (and other) definitions conclude on conceptualisation and formality. Within UMF, Information Model is considered as an enabler for convergence/unification of management systems. It is applicable to legacy and emerging management systems, in particular those featuring autonomic capabilities [6]. Moreover, an Information Model is also used to define management system interfaces, communication interfaces between application and upper management layers as well as repository data models. From a software engineering point of view, Information Model is an enabler for software development, ontology development and conceptual reasoning as well as model transformation (e.g. Model-Driven Architecture (MDA), Model-Driven Engineering (MDE)).

The following sections present the UMF Information Model Approach, the SID as the model basis and the identified extensions.

3.1 UMF Information Model Selection

The set up of an Information Model can be of – at least – two types; (i) defining from scratch to fill gaps within a domain; or (ii) selecting and extending an existing Information Model. When it comes to the selection of a model for UMF, the second approach has been applied. Particularly, various Information Models have been established and are in use in Telco's domain, covering service, resource, customer and device management: TMF Information Framework (SID) [2], the DMTF Common Information Model (CIM) [7] and the Directory Enabled Network next generation (DEN-ng) [8][9]; therefore, the second approach has been applied. Table 1 compares the above mentioned Information Models. A key criterion for selecting a reference Information Model for UMF is standardisation.

Typically, Operators are involved within the TM Forum Frameworx in general and specifically for the definition of the SID in use within the Operator Information System (Operations Support System, OSS and Business Support Systems, BSS). Moreover, SID covers various management domains (e.g. customer, resources, services, and partnerships) whilst it also defines a set of common business entities, specifically policy domain modelling including policy architecture, policy specification, and policy management. These are key elements to UMF objectives in particular managing autonomic functions through the GOV block, and the reasons to take SID as a basis for the UMF Information Model.

Table 1. Information Model Comparison

Comparison Features	CIM	SID	DEN-ng
Software patterns	No	Some	Many
Compatibility to OMG [14]	No[1]	Yes	Yes
Standardized	**No**	**Yes**	**No**
Link with business	Not clear	Yes[2]	Yes[3]
Context model	No	No	Yes[4]
Finite state machine/dynamic diagrams	No	No	Yes[5]

3.2 UMF Information Model Basis and Extensions

The approach for establishing the UMF Information Model and extending the SID starts with the identification of concepts defining and characterising the exchanged information items within UMF and between the UMF core blocks and the NEMs. A mapping was made to SID equivalent concepts followed by either semantic alignment to SID concepts or elaborating on extensions following the "SID usage Guide" and the "SID patterns" [2]. Iterations ensured consistency within the identified concepts.

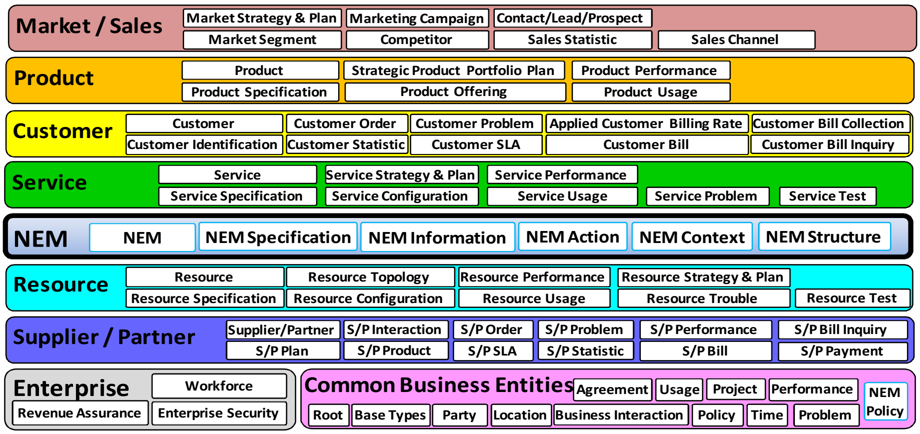

Fig. 2. Proposed NEM layer in SID ABEs

The resulted SID extensions are mainly related to the NEM concept. For example, for service and resource performance, service class or service profile we reuse existing concepts from the SID. With the deployment of UMF and NEM, there is a need to model the NEM data and information that Operators need to be aware of for

---

[1] It has its own Meta-Object Facility (MOF).

[2] Part of global Business Process Framework (ETOM), Application Framework (TAM) and Information Framework (IF).

[3] Policy continuum for translating business goals to low-level configuration actions.

[4] To apply policies with respect to the context of resource, service and customer.

[5] To describe the state/behaviour of a managed entity.

communication with these NEMs, switch them on/off, customize their actions and configurations, etc. The NEM is a new managed entity that Operators need to handle; in this sense, we propose extending the SID layers with new NEM layer (Fig. 2) containing specific ABEs (Aggregation Business Entities, a group of entities belonging to a common domain).

Apart from basing the UMF Information Model on the SID, we also adopted the DEN-ng context diagram [10] and then tried to reuse it as we considered it mandatory for managing autonomic entities. In fact, from the literature one can see the potential and adequacy of DEN-ng to manage autonomic mechanisms; albeit the fact that DEN-ng has not been standardised or even open sourced so far, is still a crucial issue.

4 UMF Information Model

The proposed Information Model components aim at conceptualising NEMs as autonomic functions and potentially incorporating them within SID framework. Following initial modelling attempt [13] UML diagrams have been elaborated following SID modelling approach for proposing the main concepts: the NEM Structure, NEM Policy, NEM Action, and NEM Information. Concepts in NEM Structure abstract NEM as provided functionality, software package, and manager of specific network resources. NEM Policy provides specification related to policies for governing the NEM's behaviour. NEM Action diagram specifies the possible NEM actions (linked to management actions). Finally, NEM Information specifies the information items managed by NEMs and their relation to specification of management information.

4.1 Extensions to SID

In the SID root diagram (Fig. 3) the *RootEntity* class defines the attributes common to define/select SID entities related to service, resources and policies. The *commonName* attribute enables SID users to refer to a specific object using terminology as defined by application-specific needs. The *description* attribute (optional) enables SID users to customize the description of a SID object. The *objectID* attribute provides a unique identity to each entity. The abstract class *Entity* extends the *RootEntity* class and represents those entities playing a business function .

An abstract class *NEM* extends the class *Entity* (Fig. 3). NEM captures functionality related to management of network resources and services which is part of the Operator's mission. This is captured by the "*manages*" association showing the link to the set of *Managed Entity* (i.e. a resource or a service) managed by a given NEM. The *NEMpolicy* defines the policies applicable to a given NEM and extends the SID policy class, whilst the *NEMStates* capture the state of a NEM (section 5.2).

Following the SID specification pattern, *NEMSpecification* and *NEMPolicy Specification* classes have been defined for the *NEM* and *NEMPolicy* classes respectively. The specification classes describe the invariant part/information of the entity, which enables the construction of an entity.

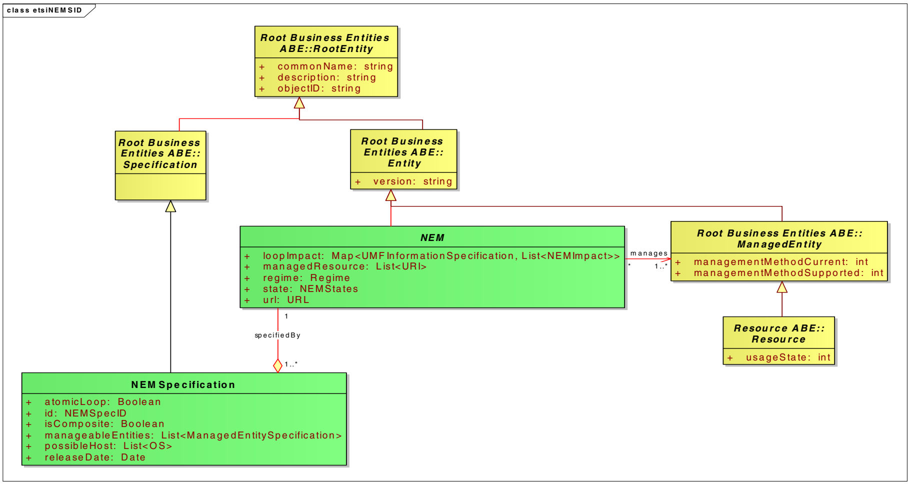

Fig. 3. NEM linked to TMF-SID root diagram

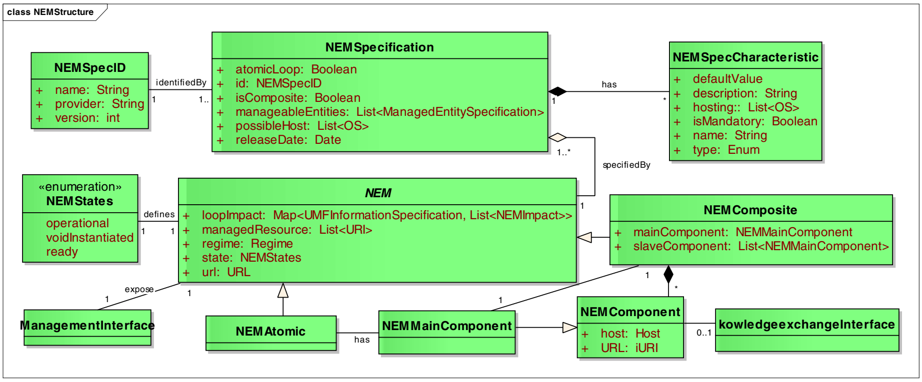

Fig. 4. NEM Structure

4.2 NEM Structure

Fig. 4 represents the structure of *NEM* which is specified by attributes grouped in a *NEMSpecification* which is *identifiedBy* the *NEMSpecID* allowing a unique identification of the “NEM class” in the catalogue. The *NEMSpecID* regroups three (3) attributes, namely, *name*, *provider* and *version*. A NEM exposes a management interface enabling UMF control over NEM.

A NEM can be either *NEMAtomic* or *NEMComposite*. An atomic NEM has centralized software running on a single machine, while a composite NEM has distributed software running on multiple machines. This is slightly different from the SID pattern as the *NEMComposite* is not composed of multiple NEMs but of multiple *NEMComponents*, and a *NEMAtomic* is composed of a single *NEMComponent* which may expose a *KnowledgeExchangeInterface*. The *NEMMainComponent* handles the

NEM control tasks manages the relation with UMF core for ensuring that the NEM instance is behaving accordingly to UMF instructions.

4.3 NEM Policy

Quite extended argumentation exists on the importance of Policies in autonomic management of networks [11]. UMF framework provides a Human-to-Network interface enabling Operator to fuse own business goals which will be translated to business, service and NEM policies [12] following the Policy Continuum [3]. NEM-level policies are targeting NEM management.

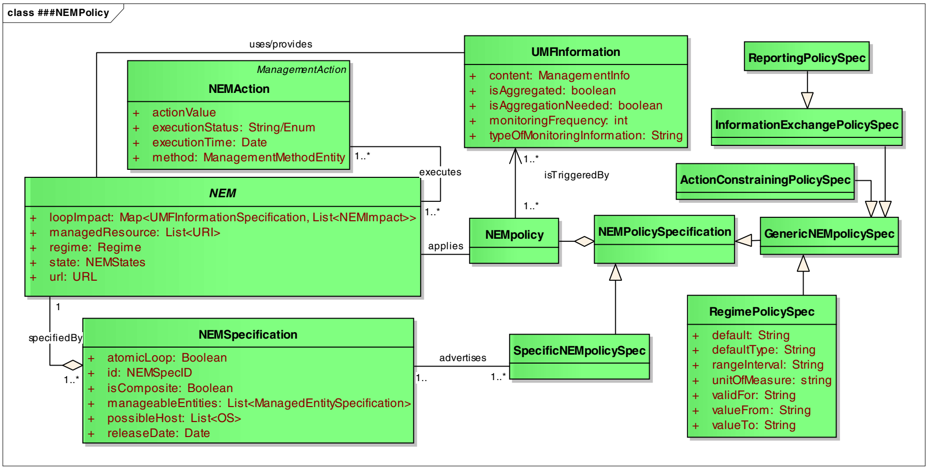

Fig. 5. NEM Policy

Fig. 5 depicts the inheritance of different policy types within the NEMs scope following the inheritance of SID *PolicySpecification* . Different policy types have been specified applicable to NEMs inheriting from *NEMPolicy* class. *GenericNEMpolicySpec* specifies the *GenericNEMPolicy* abstract class representing policy types applicable to any NEM; the format is defined by the UMF specification.

RegimePolicySpec specifies the *RegimePolicy* communicated to NEMs by COORD for setting the frequency and the modalities at which the NEM autonomic loop is invoked (e.g. “run once every 10 min”, “run continuously”, “run now only once”, “run when such X condition is true”, etc.).

ActionConstrainingPolicies (specified by the *ActionConstrainingPolicySpec*) are issued by COORD to constraint a NEM instance possible actions, aiming for example, at avoiding conflicts among NEM instances and can either disable specific NEM actions, or suspend the enforcement of the planned action to a validation by COORD or constrain the range in which a parameter can be set. *InformationExchangePolicies* (specified by *InformationExchangePolicySpec)* are issued by KNOW for organizing the information exchange, as for example, when a NEM share information needed by another NEM; KNOW is to organize a subscription between these NEMs. *ReportingPolicies* are specific *InformationExchangePolicies* issued by GOV to set the rules of information reporting from the NEM to GOV. *SpecificNEMPolicies* (specified by *SpecificNEMPolicySpec*) are tailored to NEM behaviour and objectives. In traffic engineering, for example, such policy sets whether NEM targets energy saving or congestion avoidance.

4.4 NEM Action

NEM actions include management, configuration and optimization actions to be applied by a NEM instance to network resources or services, driven by respective policies, as presented in section 4.3. Fig. 6 depicts the inheritance of Actions: *NEMActions* are executed by *NEMs* onto *ManagedEntities* (i.e. resources or services).

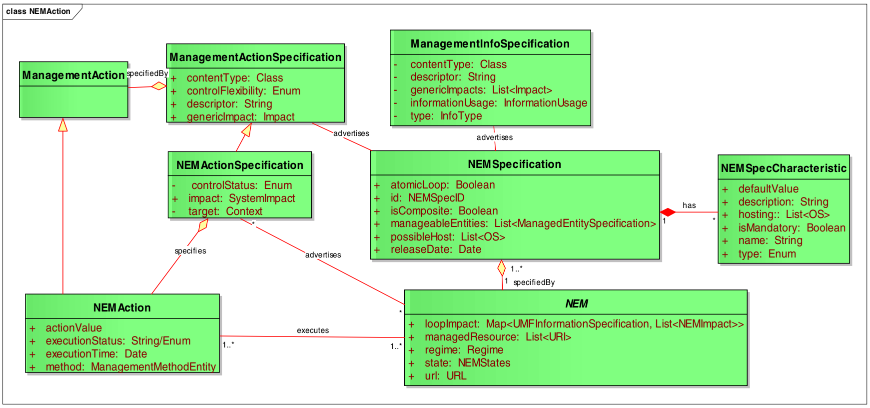

Fig. 6. NEM Action

Three levels of NEM actions specifications are depicted. *ManagementAction Specification* corresponds to the nature of the action, e.g. “switch on/off a router’s port” and is used to build catalogues of actions e.g. the list of the nature of all the actions performed by a given NEM, which corresponds to the *Possible_Actions* field of the NEM Manifest (section 5.1,Fig. 8). A NEM-agnostic catalogue should be also used to complete an ontology describing the relations among the network elements. This ontology could describe, for example, that “switching on/off a port” is changing “link capacity” if “port” is “composing” the “link”.

NEMActionSpecification designates the action, e.g. “Switch on/off the port 12 of router 1.1.1.1” and is used to build catalogues such as the indexation in COORD of NEMs actions for conflict identification. *NEMActionSpecification* extends the *ManagementActionSpecification* with the *context* attribute (in the above example the designation of the port 12 of the router 1.1.1.1) taken from DEN-ng [10].

NEMAction represents the action actually performed by the NEM and contains the *actionValue*, which in the above example can be either “On” or “Off”. The

NEMActionSpecification describes (with its *controlStatus* attribute) the allowed control of the action, while the *ManagementActionSpecification* describes (with its *controlFlexibility* attribute) the allowed control of this kind of action (this property only depends on the flexibility offered by the NEM designer at implementation time).

4.5 NEM Information

Fig. 7 depicts the inheritance of NEM Information. *UMFInformation* objects are exchanged among UMF blocks as well as between UMF blocks and NEMs. Fig. 7depicts three levels regarding information.

ManagementInfoSpecification correspond to the nature of the information, e.g. “link load” and is used to build information catalogues. Such class can capture the information types acquired by a given NEM class which can be “optional” or “mandatory” (as reflected by the *Acquired_Inputs, Optional_External_Input* and *Mandatory_External_Input* of the NEM Manifest, Fig. 8 in section 5.1) as well as the NEM outputs (captured by the *Available_Outputs* within the Manifest). The mentioned ontology could describe that “link load” is related to “link capacity” which is the “sum” of “ports capacity” “composing” the “link” whilst it would be further used to assist COORD identifying conflicts among NEMs.

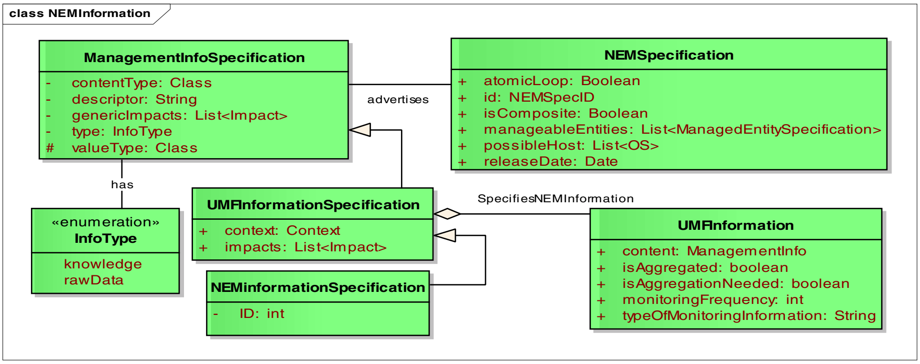

Fig. 7. NEM Information

UMFInformationSpecification designates the information, e.g. “The link load between router 1.1.1.1 and router 2.2.2.2” for building catalogues, e.g. the indexation in KNOW of all NEMs’ available outputs for the identifying knowledge source when organizing knowledge exchange with other UMF entities, and the indexation in COORD of NEMs’ inputs for identifying NEM conflicts. *UMFInformationSpecification* extends the *ManagementInfoSpecification* with the *context* attribute (in the above example the designation of the link: router 1.1.1.1 to 2.2.2.2), taken from DEN-ng extensions.

UMFInformation represents the information actually exchanged through a Knowledge Exchange. To this aim, KNOW takes in charge its organization, which will be materialized by an *InformationExchangePolicy* (Fig. 5). *UMFInformation* inherits from *ManagementInformation* (from SID) specified by an *UMFInformation Specification* and enriched with a *context* (in order to know that the "load" which is "70%" is actually referring to the "link" between "router 1.1.1.1" and "router 2.2.2.2".). The actual value is of any sub-class of *ManagementInformation* (in SID). The *ManagementInformationSpecification* and the *contentType* which sub-class *ManagementInfo* will be used to describe the value of the *UMFInformation*.

5 Information Model in Action

In the following paragraphs a case for the utilisation of the UMF Information Model is presented. Initially, the derivation of object level information is provided; in turn, the usage of those objects in the NEM lifecycle is described.

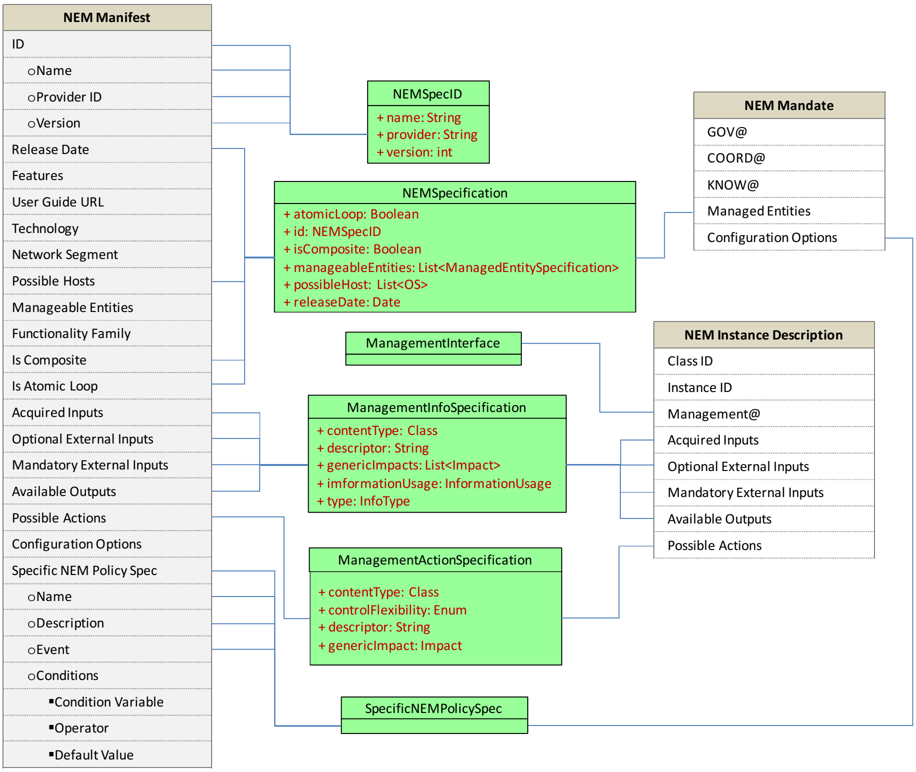

Fig. 8. Derivation of NEM Manifest, NEM Mandate and NEM Instance Description

5.1 Derivation of Object-Level Entities

A NEM class is described by a machine-readable **Manifest**, providing object-level information (e.g. managed network elements, NEM class identification, etc.) for the

Operator to deploy the NEM in its infrastructure. A **NEM Mandate** is issued by GOV to a NEM instance as a set of instructions identifying the NEM instance settings and the network elements, resources, and services assigned to this NEM instance. NEM Mandate provides the needed information for exchanges with all UMF blocks. A **NEM Instance Description** is issued by the NEM during its registration to the UMF system and details information monitored and actions taken by the NEM instance. Fig. 8 presents the derivation of the NEM Manifest, the NEM Mandate and the NEM Instance Description from the UMF Information Model.

5.2 Information Model in NEM Lifecycle

This section presents the NEM lifecycle and the interactions between a NEM and the UMF using the derived objects (i.e. Manifest, Mandate, and Instance Description). The NEM lifecycle can be traced as illustrated in Fig. 9.

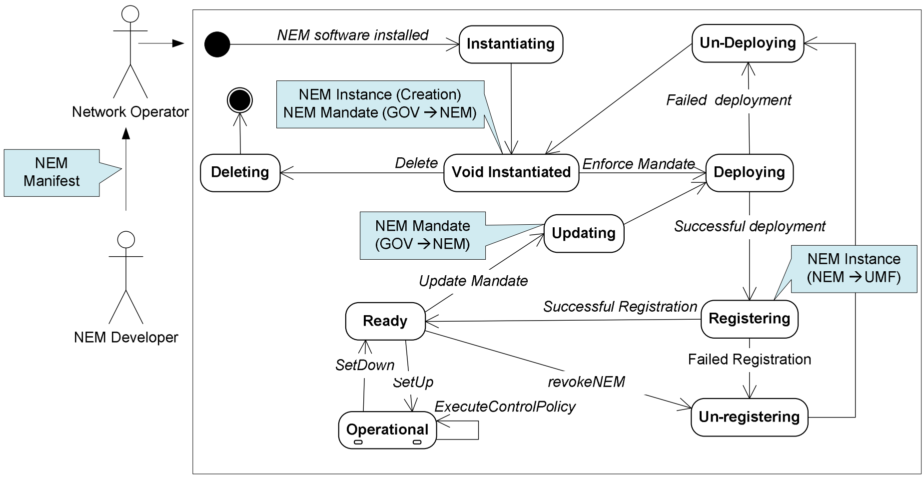

Fig. 9. NEM Lifecycle

The Operator deploys the NEM at the network according to the NEM Manifest, setting the NEM to the Instantiating state. Once the NEM instance has been created, it reaches the Void Instantiated state, and it is ready for receiving a NEM Mandate from GOV. The Mandate determines the network resources that will be managed by this instance and its configuration options, and its delivery completes the deployment of the NEM instance (which passes to the Deploying state). The NEM then proceeds to the Registering state by providing the NEM Instance Description to the UMF core blocks; NEM reaches then the Ready state and, providing no conflicts have been identified by COORD, will move to the Operational state following a Setup command from GOV . In this state the NEM instance is operational and works under the control of COORD block. The Updating trans-state is reached every time an updated NEM Mandate is received from GOV, which forces the NEM instance to get back to

Deploying. On reception of a *revokeNEM* message from GOV, the NEM instance reaches the Void Instantiated through the Un-registering and Un-Deploying states whilst on reception of a Delete message from GOV the NEM instance will disappear from the UMF system.

6 Conclusions

Autonomic mechanisms are cornerstones of the next generation of Telco's management approaches. However, to reach large deployment and efficient management of those mechanisms, Operators need specific key enablers. UMF Information Model for autonomic functions abstracts and represents what Operators need to know in order to deploy, configure and activate efficiently autonomic mechanisms.

In UniverSelf project we consider autonomic mechanisms as managed entities and define the main data and information that enable Operators govern, coordinate and develop knowledge about them. Deploying autonomic mechanisms without Information Model will lead to "vendor-specific" and proprietary implementations which will increase the integration issues and prevent adoption. Operators are investing in information model set up in TMForum or in 3GPP, as well as the harmonization between both efforts. It is mandatory that the data exchange, interfaces of Autonomic mechanisms follow these standards bodies. In this paper we selected the Information Framework (SID) in order to be compliant to Operators choices and to reduce integration costs and we proposed a set of classes and concepts towards defining the needed concepts for the management of autonomic functions.

The proposed models have been developed following SID patterns; this means that related semantics can be incorporated in the models in an automated way. This way, as reported in [15] a set of benefits can be gained, regarding implementation challenges as tools can work with the patterns through the transformation, ultimately pulling implementation code from a library written by experts and inserting it into the final application and/or system.

Acknowledgment. The research leading to these results has been performed within the UniverSelf project [1] and received funding from the European Community's Seventh Framework Programme FP7/2007-2013) under grant agreement n° 257513.

References

1. The UniverSelf Project, http://www.univerself-project.eu
2. The TMForum Information Framework (SID), GB922_SID_Rel_13-0_Addenda_Files, http://www.tmforum.org/InformationFramework/1684/home.html
3. Davy, S., Jennings, B., Strassner, J.: The policy continuum—Policy authoring and conflict analysis. Computer Communications 31, 2981–2995 (2008)
4. The Internet Engineering Task Force, IETF, RFC 3198, Terminology for Policy-Based Management

5. 3GPP, TS 32.181, User Data Convergence (UDC), Framework for Model Handling and Management
6. Wong, A.K.Y., Ray, P., Parameswaran, N., Strassner, J.: Ontology mapping for the interoperability problem in network management. Journal on Selected Areas in Communications 23(10), 2058–2068 (2005)
7. The Common Information Model (CIM) Specification, `http://www.dmtf.org/standards/cim/cim_spec_v22`
8. Strassner, J.: DEN-ng: achieving business-driven network management In: IEEE Network Operations and Management Symposium (NOMS), pp. 753–766 (2002)
9. Strassner, J., Hong, J.W.-K., Kyo, K.: A framework for modelling and reasoning about network management resources and services to support information reuse. In: IEEE International Conference on Information Reuse & Integration (IRI 2009), pp. 85–90 (2009)
10. Strassner, J., et al.: The Design of a New Context-Aware Policy Model for Autonomic Networking. In: International Conference on Autonomic Computing, ICAC (2008)
11. Strassner, J.: Policy-based Network Management: Solutions for the Next Generation. Morgan-Kaufman Publishers (2003) ISBN 1-55-859-1
12. Galani, A., Tsagkaris, K., Demestichas, P., Nguengang, G., Grida Ben Yahia, I., Stamatelatos, M., Kosmatos, E., Kaloxylos, A., Ciavaglia, L.: Core functional and network empower mechanisms of an operator-driven framework for unifying autonomic network and service management. In: 17th IEEE International Workshop on Computer-Aided Modeling Analysis and Design of Communication Links and Networks, CAMAD (2012)
13. Ben Yahia, I.G., et al.: Which Information Model for Autonomic Mechanisms Framework. Presentation at 3rd ETSI Workshop on Future Network Technologies (2013)
14. Object Management Group, `http://www.omg.org`
15. Strassner, J., et al.: The Design of a New Policy Model to Support Ontology-Driven Reasoning for Autonomic Networking. In: Latin American Network Operations and Management Symposium, LANOMS (2007)