Dynamically Adaptive Policies for Dynamically Adaptive Telecommunications Networks

Sven van der Meer, John Keeney, Liam Fallon
Ericsson NM Lab, L.M. Ericsson, Athlone, Ireland
[Sven.van.der.Meer | John.Keeney | Liam.Fallon]@ericsson.com

***Abstract*—New technologies are changing the world of communication networks and even more so their management. Cloud computing and predictive analytics have removed the need for specialized compute hardware and created products that continuously search for and find insights in management data. Virtualization of networks and network functions, SDN and NFV, are beginning to be mature enough for production networks resulting in much more flexible and dynamic networks. IoT and M2M traffic and new customer demands are driving new thinking and demands for 5G networks. Almost every aspect in the control and management of networks has seen new dimensions of flexibility and dynamicity, with the notable exception of the *policies* that drive them. This paper discusses the need to add adaptiveness to classic policies, describes a novel approach for adaptive policies, shows how adaptive policies will form part of future network frameworks and architectures, and finally discusses early use cases developed for mobile operators.**

Keywords—Automation, Policy, Adaptive Policy, Control Loop

I. Introduction

Telecommunication networks, driven by new technologies and new customer demands, are now changing faster than ever before. New technologies (cloud, analytics, SDN, NFV) are pushing operators and vendors to be much more flexible and dynamic. One example is AT&T's Domain 2.0 architecture [1].

While new technologies already show promised added value, 5G networks will create new and unanticipated demands. IoT and M2M traffic already compete with human-initiated traffic, changing the traffic mix we control and manage. Future users, devices, and applications will demand seamless and extreme customization never before required. The industry needs to "rethink" network architectures from radio to transport [2].

We see two core tools: *automation* to tame this new complexity and *adaptivity* to cope with the continuously increasing dynamicity and required flexibility. The way forward for operators (e.g. [1]) and vendors (e.g. [3]) is multi-domain, adaptive automation, supported by continuous analytics creating insights from telecom data that have not been seen before.

However, one aspect is left behind and remains rather static: the actual decision-making processes in control and management. Little effort is spent to exploit and drive this new flexibility and dynamicity for business, user, service, or network management.

This paper introduces an adaptive control loop approach for network management, discusses how to integrate new technologies for multi-domain adaptive automation, and shows how to make control and management policies adaptive. We introduce our Adaptive Policy concept, detail how these policies can adapt, and discuss how an adaptive policy engine maintains stability and control. We conclude with use cases managing operator networks and new network stacks.

II. The Context: COMPA Adaptive Control Loop

COMPA [3] is a new operational and management framework derived from the key tasks of *Control*, *Orchestration*, *Management*, *Policy* and *Analytics*. It provides for a modular architecture for a federation of control units exhibiting a common underlying pattern – the COMPA adaptive control loop.

COMPA operates at different levels in a carrier network. At a given layer, it orchestrates and manages the layer's resources to deliver services. Resources may in turn contribute to the services exposed by the receiving layer. This is recursive pattern with multiple, nested levels of control using the following building blocks: *COM* executes control decisions; *P* provides adaptive decision-making; *A* provides insights about resources. The architecture defines a number of federated responsibility domains and cross-cutting layers for continuous operations.

Within each domain, inside each adaptive component we find then the *C*, *O*, *M*, *P* and *A* components that manifest an adaptive, closed control loop as the main automation pattern of COMPA. This is a simple replication of a control loop as multiple nested loop instances. This pattern allows for multi-dimensional configuration and customization to be adapted to the requirements of the domain it resides in.

Figure 1 shows three of the four different flows (without top level interactions) to manage an automation target. Translated into the COMPA framework, the automation target being managed are those services exposed by a lower layer.

The red outer loop begins with Analytics (*A*) to lift, analyze, and process incoming data (e.g. event streams from automation targets and context sources). *Lift* aims to mediate, normalize and correlate the data by transforming streams. *Analyze* aims to use advanced analytics to discover and define trends and patterns in the event stream. *Process* aims to monitor inputs and match understood patterns. Analytics results are patterns and predictions as described in [6][12][13].

978-3-901882-77-7 (c) 2015 IFIP

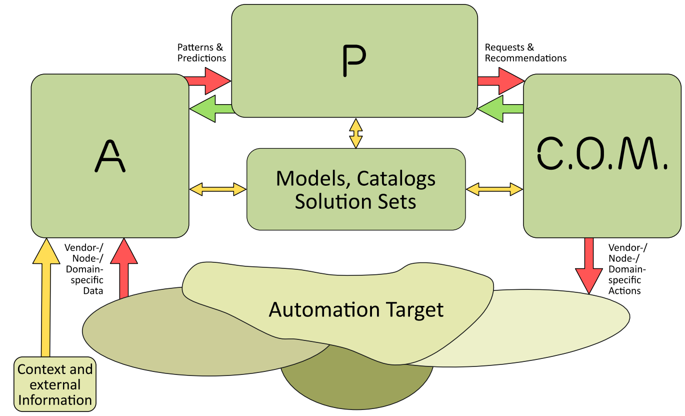

Fig.1 COMPA Adaptive Control Loop with three flows

Patterns and predictions are passed to the Policy (*P*) phase. This component can either recommend actions [4] or automatically decide on actions (request). Results are recommendations and/or action requests for the *COM* phase.

COM performs a set of validations of recommendations and requests. A semantic evaluation will test them against a semantic model to infer any potential deviations from given business or domain goals. This is followed by a test of the functionally validity. A final non-functional validation will check concerns such as security, access control rights, and resource availability. Validated, action(s) can be translated into the vendor-, node-, domain-specific representations and finally be committed.

The inner green feedback flow supports self-stabilization, i.e. to reach equilibrium. Since the loop represents a dynamic system, we need a static set of functions that facilitate the dynamic aspects of the loop while enabling a consistent and stable operation. This supporting flow is represented by the yellow arrows to and from the "Models, Catalogs, Solution Sets" component providing models of the artifacts and semantics, catalogs of patterns, context, algorithms and solution sets.

Details on how the COMPA and the adaptive control loop realize multi-domain, adaptive automation are discussed in [5]. An example for a management system using predictive analytics and recommendations is presented in [6].

III. Problem: Static Management Decision Making

The *A* part of the control loop benefits from recent advances in data-/stream-analytics. For instance, [7] addresses the task of network management from an event stream processing point of view. Wang et al. [8] show how to identify network faults based on spatio-temporal patterns using topological-aware reasoning. Meng et al. [9] prose automated profiling of network events by modelling event sequences. Zirkel and Wirtz [10] developed a process to discover predictive patterns. More discussions can be found in [12] and [13].

In the network management domain the *COM* part of the control loop exploit the benefits of SDN and NFV. The flexibility of SDN controllers and controllers-of-controllers provides infrastructure for orchestration and flexible control. While some challenges remain [11], the full deployment of SDN and later NFV will facilitate dynamic *c*ontrol, *o*rchestration and *m*anagement for and with the control loop.

However, the *P* part of the control loop currently suffers from a distinct lack of adaptability in decision making processes. The deficiencies have been discussed for some time. For instance, [14] describes some of these deficiencies: (i) aggregates of elements may exhibit behavior not predictable from knowledge of individual behaviors, (ii) causal determinacy is still limited by simple statistical analysis and rudimentary correlation approaches, (iii) no ability of the system to "go beyond" static knowledge and procedures.

Furthermore, [15] describes current policy-based management tools as being based on an event-condition-action loop, but being limited by (i) no or limited feedback within the loop (leading to incorrect/suboptimal decisions), (ii) only taking context into account at the decision point, and (iii) no tie back to the business or system goal in a dynamic manner.

IV. Solution: Adaptive Policies

The concept of adaptive policies has six key characteristics. (i) In a control loop, Analytics frees policy from complex event specifications and COM frees policy from complex action definitions. Furthermore, a feedback loop facilitates policy optimization. (ii) Boyd's OODA control loop [16] provides a clear separation of policy states with main focus on establishing a situation to make a decision about. (iii) Ontologies allow for a semantic description of domain models and policies as a basis for (runtime) inference for conflict handling and system stability. (iv) Context-awareness provides the basis for a flexible linking A, P and COM and for runtime flexibility. (v) Design patterns [17] provide dynamic decision-making behavior: the State pattern for a flexible state machine and the Strategy pattern for interchangeable state logic carrying behavior. (vi) Domain-driven Development (DDD) connects a policy engine with an evolving domain model [18].

The combination of these characteristics allows for policies with three capabilities. (I) *Context-aware decision making* uses simple triggers from Analytics with different context describing the insights and external context to describe a situation. Not understood context might signal a shift in the automation target and the policy can escalate that. (II) *Adaptable decision making* meaning that a policy *can changes decision-making behavior based on external activity* (e.g. context or changed goals). This capability can be simple (e.g. external set of policy parameters or external change of policy logic) or context-aware (e.g. using the trigger context or external context). (III) *Adaptive decision making* meaning that a policy *can change decision making behavior based on internal activity*. Examples for those internal activities are a change on automation target (by the policy) resulting in new/altered context or the set of policy parameter/state logic due to policy internal context. In the extreme case one could implement a policy that can even change its own logic. The internal activity can be task selection based on policy internal context or changing the automation target itself.

All capabilities result in fewer policies which are more flexible. The adaptive capability also allows (within limits) a policy to adapt to changes in the automation target. Furthermore, the resulting policy system introduces a flexibility to control and management policies similar to that found in programming languages, general purpose rule engines, or workflows.

We have designed an adaptive policy model, built an Adaptive Policy Engine (APEN, realizing full context-aware, adaptable and adaptive decision making behavior), additional runtime components (deployer, context handler, knowledge base authoring system) and developed various operator and network use cases. Our policy system is event-sourced and can be connected to virtually any trigger system and actuator.

A. Adaptive Policy Model

An Adaptive Policy defines four states with clear separation of concerns: *M*atch, *E*stablish, *D*ecide and *A*ct. The resulting policy is called MEDA Policy (based on the OODA loop [16]). Figure 2 shows the policy model uses the state pattern [17] for policy states and the strategy pattern [17] for policy state logic (tasks).

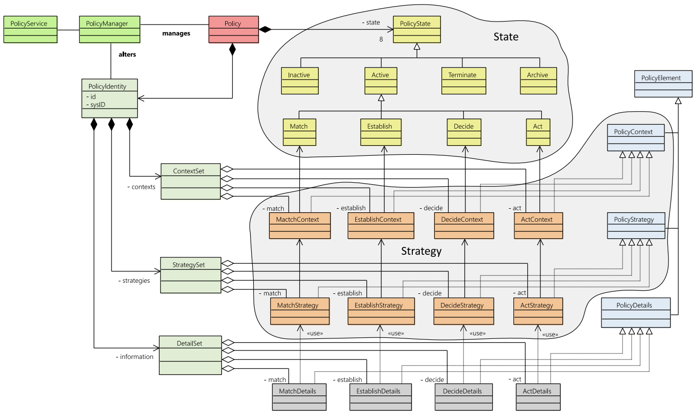

Fig.2 APEX Adaptive Policy Class Model

The *Match State* receives an incoming trigger event (with context). It is responsible for translating all information into a form understood by later tasks. It can only fail if the trigger does not match the trigger the policy is designed for. The *Establish State* establishes a situation using the available context information. It prepares all required and available information about the situation for the decide state. It can never fail, but can result in a virtually empty situation description. The *Decide State* uses the established situation to make a decision about how to address the established situation. It can fail, i.e. not result in any decision. Upon completion it emits a decide event encoding a decision or set of decisions to be processed by the subsequent act state. Finally, the *Act State* selects the best strategy to realize or fulfil the decision. The resulting grounded action request(s) is sent to an appropriate actioning system, e.g. a workflow engine, an email or SMS system or a configuration management system.

B. Adaptive Policy Execution

The policy state logic (or task) executed in each phase is selected when triggered from a number of possible policy task logic definitions based on the business and domain goals of the policy, policy context, optional policy parameters, and the context of the trigger. This allows dynamic changes to the decision making behavior of a policy at run time. APEN is built to offer policy as a service for application or as a policy server (e.g. a PDP).

APEN uses metadata for handling conflicts. The input, outputs, and context that each policy uses or modifies is modelled, allowing conflicts to be identified at authoring, deployment, or runtime and eventually mitigated. Further, runtime interference between policies can be monitored and identified as they execute because all interactions between policy instances and context are modelled and tracked.

This approach scales easily. Adaptive policies can be distributed across APEN instances running on multiple physical or virtual machines using technologies such as Distributed Hash Tables [19] for sharing context information. Policies, and policy tasks, are loosely coupled. All communication between policy tasks is via events. State shared between policies or policy instances is modelled and controlled allowing distributed policy instances to be deployed and coordinated to handle high load.

V. SON Coordination Use Case

Self-Organizing Network (SON) functions are autonomic functions operating on or across radio network elements [20]. 3GPP and NGNM [21] have standardized a number of SON functions. When different SON features are operating in a distributed environment their goals and features can easily conflict. For example, an Energy Saving (ES) function strives for minimal energy usage according to live traffic requirements while a Cell Outage Compensation (COC) function increases power in neighbor cells around outage gaps. Conflicts occur when a particular spatio-temporal pattern exists: ES reduces power usage and COC increases power output on the same cell.

The top of figure 3 shows two networks managed by OSSs with different policies. Net#1 (left) is coordinated by an adaptive policy. Net#2 (right) is managed statically. The bottom shows the event stream for both networks (including SON and OSSS events) split into four. *Phase I* shows the initial setup of the two networks. This process populates nodes, cells and SON functions.

Phases II and *III* show how cell outage is mitigated by COC with ES being temporarily switched off by our coordination policy (left network). The events show that once cell outage is discovered, the policy disables ES for the neighboring cells selected to provide cover allowing COC to increase coverage by decreasing antenna down-tilt (blue bars) and increase cell radius (red bars) by increasing antenna power of those cells.

Phase IV shows a fault scenario where ES prevents COC for a given cell since there is no coordination policy in place. The event stream clearly shows how COC tries to change down angle and radius while ES immediately reverses the action.

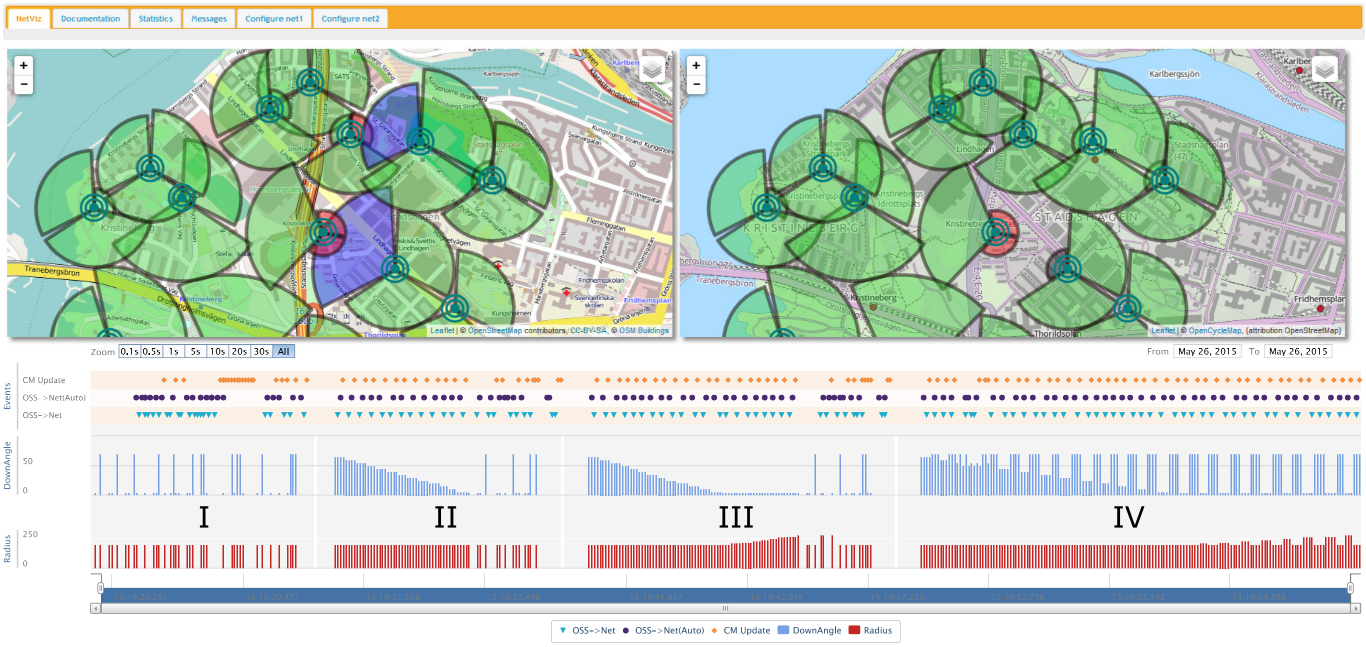

Fig.3 SON Use Case, top: Net#1 (left) with adaptive policy coordination, Net#2 (right) without; bottom: event stream and correlated

The adaptive policy then can establish the situation that a SON conflict is emerging and decide how to mitigate it. Here we decided to temporarily disable ES, and so prioritize COC over ES. Depending on business goals, real-time traffic information, time, and operational constraints, the same policy can also decide to prioritize ES over COC if the negative impact on coverage is acceptable. The policy will chose from the available candidate behaviors depending on the operating context of the network and the business and operational goals.

VI. Recursive Network Architecture Use Cases

The Recursive InterNetworking Architecture (RINA) is detailed in [22] with tutorials and other resources available at [23]. RINA's core statement is that networking is Inter Process Communication (IPC) [24]. Key added value is a network architecture that can be recursively applied to any number of layers, each of which has the same instrumentation differing only in objectives and configuration. The recursive nature of RINA, along with build-in mobility [25] and security, makes it an ideal environment to explore the similar recursive COMPA architecture with its control loop and our adaptive policies.

In the first phase of the FP7 project Pristine [26] we have developed a Distributed Management System (DSM) as detailed in [27]. The DMS allows defining and implementing adaptive policies realized in a simple way since all information about the network is provided by the Resource Information Base (RIB). Those simplified yet adaptive policies are then used in an orchestrated way to manage RINA layers.

With the common infrastructure in place, the second phase of Pristine will build adaptive policies for three main use cases (i) distributed cloud, (ii) datacenter networking and (iii) virtual networks. These use cases are detailed in [28]. The management aspects covered are congestion control, security, and multi-layer management. We are planning to then use the results of this work in large-scale testbeds with network conditions close to real communication networks.

VII. Conclusions

In this paper we have discussed how telecommunication control and management policies need to evolve to cope with the flexibility and dynamicity other network components are already mastering. We have described our approach for Adaptive Policies to exploit runtime context and runtime selection of task logic (and the availability of multiple such logics at runtime) to allow a policy to adapt to new situations.

The developed policy engine allows for easy deployment as well as for controlled policy execution. The concrete models for context facilitate a complete understanding of individual runtime policies and facilitate new mechanisms for conflict detection. We also discussed a SON coordination use case and use cases for RINA (ongoing research work). Use cases not presented in this paper, such as autonomic anomaly detection as presented by Ericsson at the Mobile World Congress in Barcelona in 2015 and control of transport networks, are expected to go on trial by the end of 2015. Large scale RINA scenarios will be deployed by the end of 2016 and in 2017.

Future work will focus on the integration of meta-data and context for advanced conflict detection as well as on sophisticated authoring tools for adaptive policies.

Acknowledgement

This research is partly funded by the European Commission via the PRISTINE project (Grant 619305), part of the Future Networks objective of the 7th Framework Program (FP7).

REFERENCES

[1] AT&T Vision Alignment Challenge Technology Survey - AT&T Domain 2.0 Vision White Paper. Online Document, Available: https://www.att.com/Common/about_us/pdf/AT&T%20Domain%202.0%20Vision%20White%20Paper.pdf

[2] IEEE, 1st IEEE 5G Summit, Princeton, NJ, USA, May 26, 2015, Available: http://5gsummit.org/ (presentations) https://ieeetv.ieee.org/ (recordings)

[3] Rune,G., Westerberg, E., Cagenius, T., Mas, I., Varga, B., Basilier, H., Angelin, L., "Architecture evolution for automation and network programmability", Ericsson Review, Nov 28, 2014, Available: http://www.ericsson.com/news/141128-er-architecture-evolution_244099435_c

[4] Keeney, J., van der Meer, S., Hogan, G., "A Recommender-System for Telecommunications Network Management Actions", IFIP/IEEE International Symposium on Integrated Network Management (IM 2013), Ghent, Belgium, May 27-31, 2013

[5] van der Meer, S., "5G & Autonomic Networking - Challenges in closing the Loop", IEEE First International 5G Summit, Tuesday, May 26 at Princeton University, NJ, USA

[6] Zaman, F., Hogan, G. ; van der Meer, S. ; Keeney, J. ; Robitzsch, S. ; Muntean, G.-M., "A recommender system architecture for predictive telecom network management", IEEE Com Mag, Vol. 53, Issue 1, 2015

[7] Cugola, G., Margara, A., "Processing flows of information: From data stream to complex event processing", ACM Computing Survey, Vol.44, No.3, Article 15, Jun 2012

[8] Wang, T., Srivatsa, M., Agrawal, D., Liu, L., "Spatio-temporal patterns in network events". 6th International Conference on emerging Networking EXperiments and Technologies (CoNEXT '10). ACM, New York, 2010

[9] Meng, X., Jiang, G., Zhang, H., Chen, H., Yoshihira, K., "Automatic Profiling of Network Event Sequences: Algorithm and Applications", 27th IEEE Conference on Computer Communications (INFOCOM 2008), pages 266–270, 2008

[10] Zirkel, W., Wirtz, G., "A process for identifying predictive correlation patterns in service management systems", 7th International Conference on Service Systems and Service Management (ICSSSM), 2010

[11] Keeney, J., van der Meer, S., Fallon, L., "Towards Real-time Management of Virtualized Telecommunication Networks", 10th International Conference on Network and Service Management (CNSM), Brazil, Nov 17-21, 2014

[12] Zaman, F., Robitzsch, S., Qu, Z., Keeney, J., van der Meer, S., Muntean, G.-M., "i-MagNet : A Real-time Intelligent Framework for Finding Specific Needles From Needle Stacks", IFIP/IEEE International Symposium on Integrated Network Management (IM 2015), Ottawa, Canada, May 11-15, 2015

[13] Robitzsch, S., Zaman, F., Qu, Z., Keeney, J., van der Meer, S., Muntean, G.-M., "E-Stream: Towards Pattern Centric Network Incident Discovery and Corrective Action Recommendation in Telecommunication Networks", IFIP/IEEE International Symposium on Integrated Network Management (IM 2015), Ottawa, Canada, May 11-15, 2015

[14] Strassner, J., "Autonomic Systems and Networks: Theory and Practice", Tutorial, 11th IEEE/IFIP Network Operations and Management Symposium (NOMS 2008), Salvador – Bahia, Brazil, Apr 7-11, 2008

[15] Fleck II, J., "Model Driven Semantic Management", Tutorial, 13th IEEE/IFIP Network Operations and Management Symposium (NOMS 2012), Maui, Hawaii, USA, Apr 16-20, 2012

[16] Boyd, J.R. "The Essence of Winning and Losing", Unpublished lecture notes, Jun 1996

[17] Gamma, E., Helm, R., Johnson, R., Vlissides, J., "Design Patterns: Elements of Reusable Object-Oriented Software". Addison-Wesley Pub Co; 1st edition, Jan 15, 1995

[18] Evans, E., "Domain-Driven Design: Tackling Complexity in the Heart of Software". Addison Wesley, 2003

[19] Coulouris, G., Dollimore, J., Kindberg, T. & Blair, G., "Distributed Systems: Concepts and Design", Addison-Wesley, ISBN 978-0-13-214301-1, 2012

[20] Hamalainen, S., Sanneck, H., Sartori, C., "LTE Self-Organising Networks (SON): Network Management Automation for Operational Efficiency", Wiley, Dec 2011, ISBN: 978-1-119-97067-5

[21] NGMN Alliance, "NGMN Recommendation on SON and O&M Requirements", Dec 8, 2008, Available: http://www.ngmn.org/uploads/media/NGMN_Recommendation_on_SON_and_O_M_Requirements.pdf

[22] Day, J., "Patterns in Network Architecture: A Return to Fundamentals", Prentice Hall, 2007. ISBN 978-0-13-225242-3

[23] Pouzin Society, Documents and Publications, Available: http://pouzinsociety.org/documentspublications.html

[24] Day, J., Matta, I., Mattar, K., "'Networking is IPC': A Guiding Principle to a Better Internet" Workshop on Re-Architecting the Internet, (ReArch'08) at ACM CoNEXT 2008. Madrid, Spain, Dec 2008

[25] Trouva, E., Grasa, E., Day, J., Matta, I., Chitkushev, L.T., Bunch, S., Ponce de Leon, M., Phelan, P., Hesselbach-Serra, X., "Transport over Heterogeneous Networks Using the RINA Architecture". 9th International Conference on Wired/Wireless Internet Communications (WWIC), Vilanova, Spain, Jun 2011

[26] FP7 project Pristine website, available online at http://ict-pristine.eu

[27] Pristine Deliverable 5.3: Proof of concept of DIF Management System. Pristine, Apr 30, 2015, Available: http://ict-pristine.eu/wp-content/uploads/2013/12/pristine_d53-proof-of-concept-dms.pdf

[28] Pristine Deliverable 2.1: Use cases description and requirements analysis report. Pristine, May 31, 2014, Available: http://ict-pristine.eu/wp-content/uploads/2013/12/pristine_d21-usecases-and-requirements_draft.pdf