Deliverable D4.3

Assessment Results of Trust in Autonomics Release 1

Grant Agreement	257513	
Date of Annex I	25-07-2011	
Dissemination Level	Public	
Nature	Report	
Work package	WP4 – Deployment and Impacts	
Due delivery date	01 June 2012	
Actual delivery date	11 October 2012	
Lead beneficiary	Fraunhofer	Mikhail Smirnov mikhail.smirnov@fokus.fraunhofer.de

SEVENTH FRAMEWORK PROGRAMME

FUTURE NETWORKS

Authors	ALBLF – L. Ciavaglia, S. Ghamri-Doudane Fraunhofer – M. Smirnov, M. Corici UPRC – P. Demestichas, V. Stavroulaki, A. Bantouna Orange-FT - B. Sayrac ALUD – M. Gruber NKUA – V. Kosmatos, E. Patouni

Executive summary

Stability, robustness and security issues arising from future Self-Organising Networks (SONs) must be understood today, and involved in their design, standardisation and certification. We address the issue of Operator trust in SON (e.g. LTE) through the following five requirements and our approaches to meet them: 1. Trust must be measurable (we consider the three facets of operator trust – reliable operation, trustworthy interworking and seamless deployment and suggest a composite metric for SON stability), 2. Trust must be SON-specific (we define a KPI-based envelope of dependable adaptations), 3. Trust must be model-driven (we demonstrate how to construct such models based on predicates), 4. Trust must be propagated end-to-end (we show that trust networks emerge from predicate-enabled behaviours), 5. Trust must be certified (we outline the certification process). Trust predicates that are defined at the design phase as abstract behaviours, and verified at run-time as fully qualified ones, prove to have the power of policies – check them once and re-use many times; rewrite them to cater for new behaviours.

Table of Content

Foreword

In the UMF (Unified Management Framework) design the interests of several groups of stakeholders are to be interwoven, which will be reflected at various design phases in various priorities set to different groups of requirements, both functional and non-functional. However, the importance of the two non-functional requirements – manageability and trustworthiness – is likely to be set at a high level by all involved stakeholders. It is important therefore to attempt to address these two requirements together and to see whether a design for manageability is bringing certain benefits for the design for trustworthiness and vice versa.

Such an attempt is reported in this deliverable. Considered successful this experiment performed in WP4 occupies an important and central place within the project. Thus, WP2 will benefit from a straightforward bottom-up approach, in which the complexity of a problem at hand is being solved by the tools suggested by the complexity itself. The WP3 will benefit from a self-orchestration methodology outlined step-by-step in the reported work with examples from LTE SON and with the study of a generic machine learning method applicability for trust indexing. WP4 will have an opportunity to enrich its UC4, as well as bring similar considerations to other project use cases. last but not least, the WP5 will have an opportunity to include the reported work into its trend-setting activities, because the main stakeholders addressed by the work are Mobile Network Operators (MNOs).

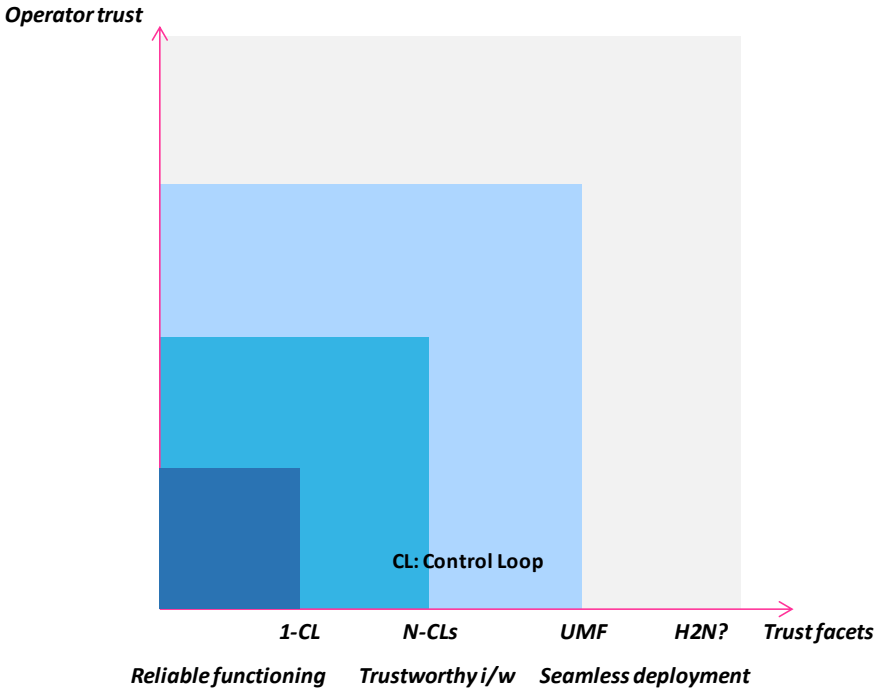


Figure 1. Facets of Operator trust in SON

As Fig.1 outlines our roadmap to build an Operator Trust in SON, and in general in autonomic self-management of an empowered network is to gradually guarantee the following properties of an autonomic network. First, to assure reliable operation of each control loop; second, to guarantee trustworthy interworking between several control loops (this report considers seven CLs); and, third to facilitate seamless deployment of any number of CLs enabled by the UMF. The Human-to-Network (H2N) aspects of trust should probably be the next step on this roadmap.

While the report is a theoretical work it opens novel implementation opportunities, which are already being implemented – the approach reported here is being further detailed in one of the project Network Empowerment Mechanisms (NEMs), namely SOUP – Self-Orchestration via Utility Predicates.

1 Introduction

Our goal is not to identify exploits of future technology but to design technology add-ons that shall prevent the exploits. Our agenda is to design add-ons that shall be mutually beneficial to both - management of future technology (we term it governance, perhaps unconventionally but with the clear goal to differentiate it from conventional management) and to increase in the level of operator trust in that future technology. We are also interested to contribute to the certification process of future equipment that shall cover the entire manufacturing chain from design technology to the deployment process.

Network designers are talking most of all these days about cognitive, software defined and self-organising solutions for future networks. We try to concentrate on self-organisation, even more - only on one element of it – self-orchestration of multiple control loops. We also believe that alongside with self-orchestration the deployment of practical machine learning techniques should not be underestimated.

Which possible exploits could be eliminated during the design of add-ons? We are not trying to solve general security problem (where the major paradigm is the comparison of access vector to the attack vector); in our case there is one more dimension, namely operational instability (it does not matter whether it was caused by an exploit or by a poor design), which will damage the operator trust in future network. That's why we concentrate on self-orchestration as stability foundation.

Our major use case is LTE SON [7, 10] that we select for the reasons of mature specification, large deployment interest, availability of industrial prototypes, and last but not least huge expected impact on Mobile Network Operator (MNO) networks. LTE SON is our target also because the risks associated with the deployment of SON features are well understood by the MNOs. These risks are: uncontrolled adaptations, conflicting operation, and possibility of new types of attacks.

First, the MNOs have a strong fear of losing control over their source of revenue, the risk can be eliminated by the rigorous proof of the fact that adaptations of all nine groups of LTE technical parameters made automatically by the SON mechanisms will never leave the network behaviour envelope that defines the sustainable revenue – these dynamic adaptations within this envelope define for us the important aspect of operational stability. The above rigorous proof is likely unfeasible for all possible network situations, therefore our approach is to demonstrate (at the design phase) operational stability for a representative number of situations and to equip SON mechanisms with a human interface. The human to network interface will allow a human operator not only to parameterise and to follow SON automatic adaptations (at the deployment phase) but gradually to learn the internal logic of these adaptations and be able to improve human governance skills (at the service phase). The ability to learn is also useful to improve the run-time behaviour stability of the mechanisms over time.

Second, almost all considered LTE SON mechanisms are potentially conflicting either in respect to monitored Key Performance Indicators (KPIs) or in respect to controlled parameters, or both. The risks arising from the unsolved conflicts are multiple (LTE cell coverage and capacity inefficiency, throughput degradation due to unnecessarily strong interference, poor mobility management, energy wasting, etc.) and will lead to poor quality of experience. In general, the risk is to increase losses instead of promised optimisations.

Third, since SON mechanisms empower the network they at the same time create possibilities for new types of attacks. We see at least two new attack types that can be termed sensing attack and suggestive attack. The sensing attack is possible because LTE SON mechanisms learn about actual network situations largely from radio sensing reports issued by mobile terminals; the terminals generally trusted by an MNO can be hijacked, mis-configured, spoofed, etc. by attackers intentionally to disrupt the SON. The suggestive attack known from psychology can be used to disorient cognitive engines proposed for almost all SON mechanisms.

We address the first two types of risks through the proposed SON operational stability and conjecture that it will be also helpful in eliminating risks from new attack types. The rest of the paper is organised per five topics defined in the executive summary, with the next section making an overview and placing our work against the related work.

2 State of the Art and Beyond

Relevant papers are referenced in the main part of the report as appropriate. We use this section in order to highlight what we bring beyond the state of the art as the result of the cross-disciplinary nature of the reported work. To the best of our knowledge this is the first report that applies ecosystem modelling to LTE SON use cases in combination with machine learning techniques and group communication. At the same time the driving model of the self-orchestration is the set of unified KPIs, which set also bears certain novelty, the main body of publications use that KPIs that are relevant for a service in question; in our case handling of the service quality is also estimated by the conventional set of KPIs but self-orchestration is driven by the utility that in turn is computed based on unified KPIs.

Network operators traditionally always trusted their networks because of the extensive testing all network equipment had to undergo in order to meet internationally standardised performance and reliability requirements. The situation is different today, when networks are being empowered by self-management features. This empowerment theoretically allows network elements to solve previously unsolved problems by relieving traditional management systems from routine tasks and thus promising to significantly reduce the cost of network management. In such networks exhaustive testing is not possible; to keep trusting their networks, operators must be able to verify multiple facets of network operation, hence we define the three main facets of operator trust: reliable functioning, trustworthy interworking, and seamless deployment shown in Fig.2 as a trust hierarchy.

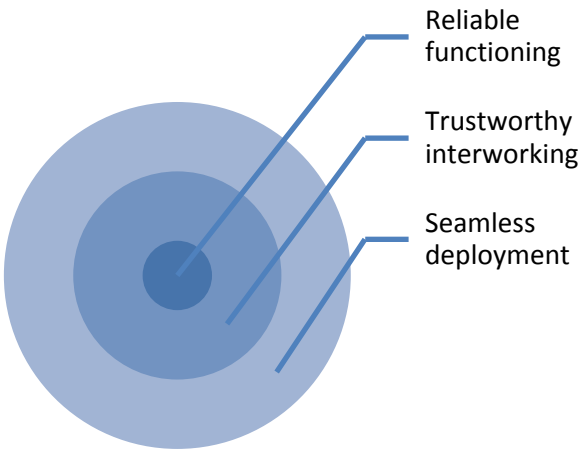


Figure 2. Hierarchy of Trust Facets

The **reliable operation** applies to every Control Loop (CL) (or by extension to the NEMs in UniverSelf terminology) that can be deployed in the network, and includes the following phases:

Characterise – Test – Verify – Certify.

Those phases are applied to the performance and to the conformance of a device/function/system to be deployed and must be compared with that of a “reference” device/function/system, which can be a canonical implementation, but also a behavioural model, structured set of requirements and/or specifications, or a set of benchmarks. The last three phases do exist at both design phase and at the run-time.

The **trustworthy interworking** applies to multiple co-existing control loops; this is achieved by their coordination (orchestration), and ideally by means of self-orchestration. The unification of coordination is a big challenge because it may include a huge variety of options, such as dynamic (at run-time coordination, static (pre-defined) one, centralised or distributed, etc. The coordination must be constrained by the stability requirement. Such challenge is being addressed in the design and specification of the Unified Management Framework (UMF), in particular with the development of the Coordination block and its mechanisms, as part of the deliverable D22.

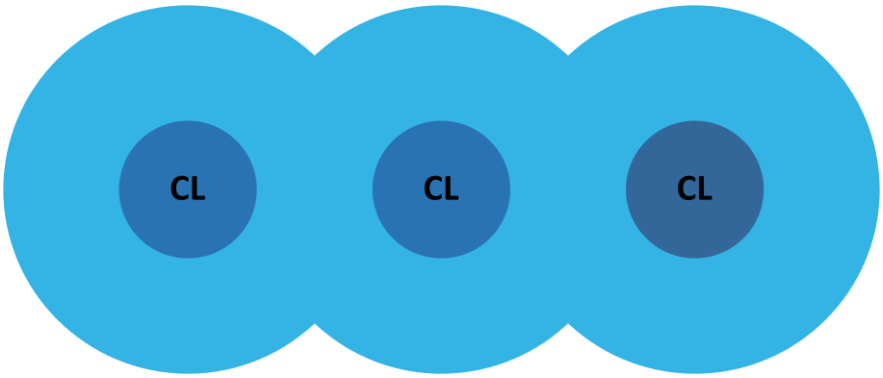


Figure 3. Coordination of multiple CLs

As Fig.3 demonstrates the coordinated control loops continue to stay autonomic in the sense that their operations have now two set of states that are necessarily organised in a hierarchical manner. Inner operation is being defined by CL's behaviour that is optimising certain network parameter(s), while outer operation is that of coordination. The outer operation necessarily constrains the inner operation in such a way that coordination goals are met. Such organisation of the two behaviours poses a novel challenge to the design phase – namely the co-design of functional and coordination mechanisms.

The **seamless deployment** applies to network modifications that a network operator might wish to make in a plug and play mode. From now on an operator will be able to deploy seamlessly and dynamically any needed number of control loops organised (by the coordination) in a needed number of control groups. As the way to achieve this we define the requirement for each deployed CL to be UMF-compliant.

UMF compliance that enables seamless deployment is yet another behaviour of a control loop that implements all needed steps and phases of the UMF life-cycle for a deployed CL. Being the most outer behaviour of a CL the seamless deployment guides (constrains) the operation of coordination, and jointly with the coordination guides (constrains) the most inner operation of a control loop.

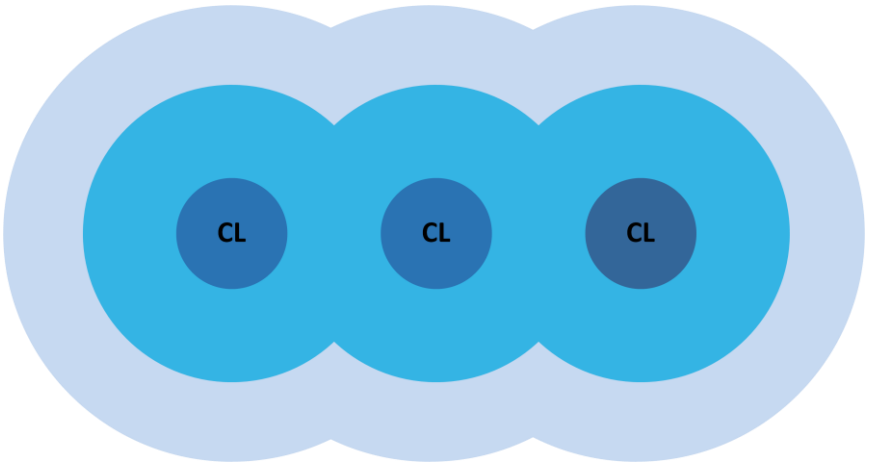


Figure 4. Life-cycle management of multiple CLs

The key concepts in modelling and the analysis of all three trust facets detailed in the main body of this report are the concepts of trust metrics and the utility.

By these definitions of the trust facets we are able to distinguish between **vertical trust** (User to Network Interface) that is implemented via UMF governance interface, and **horizontal trust** (Network to Network Interface) that is implemented via coordination interface, knowledge interface and enforcement interface.

The above outlined trust facets were obtained as deliberate abstraction and aggregation of many more facets that were defined at the beginning of the reported work and are summarised in the Annex A.

We conclude this section with a very specific example of operator trust in automatic control borrowed from the aviation.

2.1 Learning from Aviation

Self-organization including self-configuration, self-optimization, and self-healing will be an integral feature of future telecom network equipment. In this context, network equipment manufacturers will have to ensure a reasonable level of *trust* in this feature by the operator. In particular, it needs to be shown that

- The performance of the self-* process excels the performance of manual intervention in terms of
  - Convergence time until optimal solution is achieved
  - Key performance indicators during convergence
  - Key performance indicators after convergence
  - Decision when to re-launch a self-* process (after the process had already converged at an earlier point in time)
- The cost of purchasing and maintaining the self-* process is lower than the cost of providing service personnel for the same purpose
- Multiple self-* processes interact with each other in a way that oscillations and instabilities are avoided

The most important thing is to demonstrate the operator that an autonomic management does not come along with incalculable risks such as network outages and the like. Intuitively, human beings always have a tendency to distrust features that are not under their full control. However, in other domains, such as in aviation, automated systems have already taken a substantial part of the control over a system with much higher security requirements (i.e. the safety of aircraft passengers) than those in the telecom area. Even beyond, an article of the IEEE Spectrum Magazine [39] highlight the remaining challenges and obstacles but also the close opportunity that unmanned or pilotless airplane become a daily reality. A good example is the autopilot of a commercial aircraft that relieves the actual pilot from routine actions. But not only that, the autopilot is also able to control complex manoeuvres and supports the safe landing of the aircraft. With this in mind it should not be too hard to assure a network operator that a trustworthy deployment of self-* features is possible.

In the following we will transfer some key principles from aviation to the self-* domain. The basis for this is a flight crew training manual for the Airbus 330/340[1]. We will cite relevant sentences / paragraphs from this manual in italics and explain how this principle can be applied to autonomic management of a telecom network (instead of an aircraft). The idea behind this exercise is to apply security principles that have already proven to work.

The autopilot is designed to fly the aircraft within the normal flight envelope
The management of the network is autonomic unless there are severe problems

The autopilot automatically disengages if the aircraft flies significantly outside the normal flight envelope limits
The automatic management of the network is automatically disengaged if the key performance indicators significantly differ from normal (good) values

With one engine inoperative, the AP can be used throughout the entire flight envelope without any restriction, including autoland
There are self-healing features. For instance, if there are outages of network equipment, the network management system is able to recover without manual intervention

[1] A330 & A340 Flight Crew Training Manual, FCTM Presentation

The relationship between sidestick input and the aircraft response is called the flight control law. Depending upon the status of the fly-by-wire system, three sets of control laws are provided, i.e. Normal Law, Alternate Law and Direct Law
Depending on the status / situation of the network, there are different levels of how much control the autonomic management has with respect to the human being in charge

Under most circumstances, the aircraft is operated in Normal Law
Usually the autonomic management has full control

Normal Law provides five different protections [e.g. high angle of attack protection or high speed protection].The protections are complementary and together work to maintain the aircraft in the safe flight envelope
There is the concept of protection during “normal law” which does not let the pilot / human operator make any dangerous moves. For example, during take-off of an aircraft, the pilot usually cannot let the tail touch the ground – without protection that would be possible with all its dangerous consequences. Likewise, during “normal law” the network operator should not be allowed to leave certain parameter limits that are known to have negative or nonlinear effects

In some cases of double failure, e.g. double hydraulic failure, the integrity and redundancy of the computers and other required systems are not sufficient to achieve normal law with its protections. In this case, Alternate Law is triggered
In case of multiple problems that cannot be accommodated automatically, the human operator gains more (yet not full) control

If the aircraft is operated outside the normal flight envelope, the pilot must take appropriate corrective action to avoid losing control and/or to avoid high speed excursions, since the normal law protection features may not be available
In case of “alternate law” there are fewer protections. This allows human operators to do moves that would be dangerous in normal situations, but necessary in exceptional situations

In most cases of triple failure […] direct law is triggered. Autopilot and auto-trim are not available
In some pathological cases, the human operator is in full control and no autonomic functionality is active. This is clearly the worst case and should be considered an extremely rare event

Cautions and warnings
An interesting aspect of the flight manual is the differentiation between “cautions”, which are solely prudent forethought to minimize risks, and “warnings”. Please note that this differentiation would allow the human network operator to prioritize information that he/she gets from the autonomic system. What looks like a minor detail at first sight is in fact a major security feature (as information overflow can cause delays or failures on the human reaction side) that deserves attention

Computer replication / backup
Obviously, a network management system must not be a single point of failure

Conclusion:

Concerning trust in autonomics, we can learn and apply a number of principles from an autonomic system that is yet more security-critical than a telecom network, namely the autopilot of an aircraft:

- The partition of control between autonomic management system and the human network operator should be changeable to different discrete control levels. A first example of these control levels in the context of the UMF and NEM is the change of state of a particular NEM from under trial to operational (and vice-versa) depending on its trust index estimations (see Section 3, and section 3.4 in particular).
- Depending on the control level, the human network operator should be protected from dangerous changes of parameters. In the context of UMF, these “boundaries” can be defined and applied via the use of policies and policy based management principles.
- The information the human network operator receives from the network management system should be on clearly defined hierarchical levels so that information can be easily filtered depending on the

current needs. In the context of the UMF and NEM, this information may take the form of a Call for Governance

One of our next steps in the Trust in Autonomics activity will be to investigate further the translation of the notions presented above into IT and Telecommunication specific context(s) and precise the identification of their networking equivalent.

2.2 State of the Art in Certification

A true Autonomic Computing (AC) system is one that is able to automate the management decision making process and reflect on the quality of the decisions made. It must do regardless of the environmental context and within the goals set by the human operator. The ultimate aim of autonomic computing systems is to allow complex Information Technology (IT) infrastructure evolve into more complex systems to handle more difficult tasks, without significantly increasing the cost of management.

The ability to self-configure, self-optimize, self-protect and self-heal, have been identified as the four cardinal characteristics of AC systems, leaving the human operator to simply design and communicate the overall business goal to the autonomic computer. Since then, the autonomic research community has worked towards building machines that fit squarely with the original vision set out for autonomic computing systems. Indeed, significant progress has been made in this regard, with several architectures defined for autonomic machines, the most prominent being IBM's MAPE (Monitor, Analyze, Plan, Execute) architecture [13]. In turn, this architecture has inspired a number of (semi-) autonomic applications including; the Autonomic Toolkit, ABLE, Kinesthetics eXtreme, the Open Services Gateway initiative (OSGi) platform and applied to several other projects. However, there is lack of efforts in the area of autonomic computing certification i.e., there are only a few frameworks which guide the process by which two or more autonomic machines are rated in relative terms, assuming these machines target the same application domain.

2.2.1 The importance of certification

With a huge effort devoted to the design and development of ASs, emphasis is lacking on the certification of these systems.

Authors in [18] suggest that ASs must reach trustworthy status and be "certifiable" to achieve the full vision of AC. Appropriate measures for validating AS decision-making processes should be defined. They identify this as the core challenge facing the success of AC. Another major problem facing the AC research field is the lack of standards. There are proliferation of approaches and the misuse of AC terms –different terms mean different things to different researchers. This shortcoming can only be addressed by standards.

Certification of ASs is a specific work area that needs attention and this can be achieved through defining proper AS validation mechanisms. AC systems are designed and deployed across many application domains to address the challenge of human management complexities. We may come to a point where these systems take over full control of operations in those domains (e.g., businesses, telecoms, military, health etc.) and any failure can be extremely costly –in terms of down time, danger to life, loss of control etc. This underpins the criticality of AS validation and assessment. Robust self-management in AC systems resulting in dynamic changes and reconfigurations requires that ASs should be able to continuously perform self-validation of their own behaviour and configuration, against their high-level behavioural goals and be able to reflect on the quality of their own adaptation behaviour. Such systems are considered trustworthy and then certifiable. It is then necessary to have a testing approach that combines design/run-time elements and is also an integral part of the self-management architecture. By trustworthiness, we mean a state where we can be confident that an AS will remain correct in the face of any possible contexts and environmental inputs and sequences of these; this is achieved through robust validation and assessment.

2.2.2 Certification Process

The ultimate goal of AC should be the certification of AC systems. Yet to achieve certification requires a process and the meeting of some conditions. For unknown reasons and in a bid to get things working faster, the AC research community has concentrated efforts on designs and architecture with little or no emphasis on system validation. Only very few researchers have identified trustworthiness as a major AC challenge and yet fewer

[28], [29], [30] have actually suggested or proposed techniques. The problem in clear terms is the ignoring of AS trustworthiness and the general lack of validation efforts that specifically target the dynamic aspects of these systems.

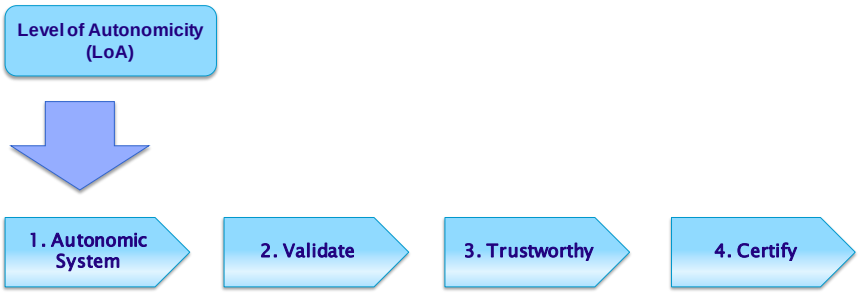


Figure 5. Certification process

Figure 5 represents a section in the journey towards full AC. At point (1) we assume that a system is developed and is considered autonomous at some level. This level is determined by a LoA measurement methodology which needs some form of standardization. The definition of LoA at this point is prerequisite to the next step. At point (2) is the system self-validation distributed across design-time and run-time. When it is ascertained that a system is validated then it is trustworthy and trustworthiness is a vital foundational step on the road towards certification. It then follows that for a system to be certified, it must be trusted and only validated systems can be trusted. We draw a conclusion here that for an AS to be certifiable there must be a standard for measuring the level and extent of its autonomicity as it makes no meaning to certify a system whose extent of autonomic capability cannot be measured.

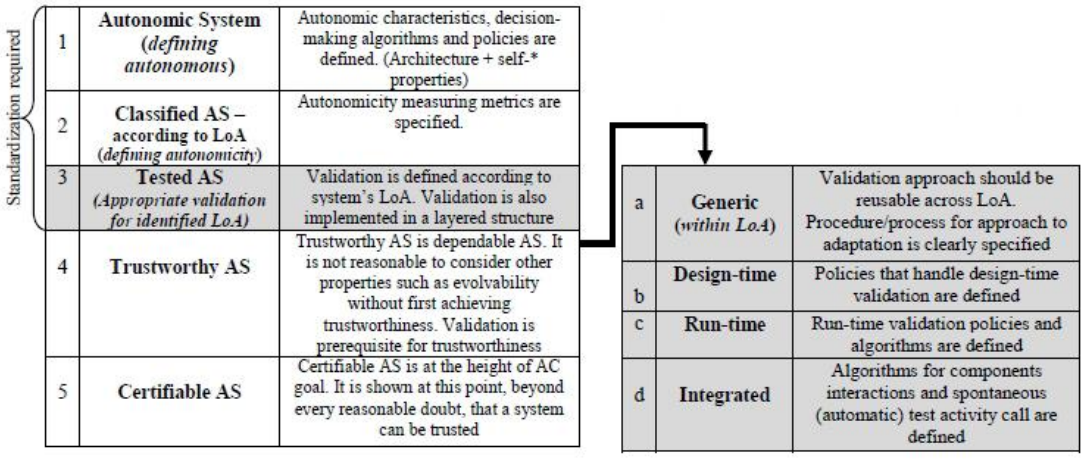


Figure 6. Roadmap toward Certifiable AS

The authors in [18] proposed a roadmap towards AS trustworthiness and therefore the way to achieve certifiable AC systems. The proposed layered solution for autonomic certification is illustrated in Figure 6. Firstly, characteristics or features should be identified, that a proper validation approach should possess. Then a number of inter-related steps should be taken towards certifiable ASs. In [18] the authors identify that the validation step is the most important toward certification and they argue that a proper validation and assessment methodology is the only reliable process towards certifiable AC systems. A proper validation/assessment approach should have the following characteristics:

- Generic: Reusability reduces complexity and cost (in terms of time and effort) in developing validation processes for AS. A good validation approach should be flexible to be adapted to different adaptation processes and the procedure or process for this adaptation clearly detailed.

- Design/Run-time: The dynamic changes and reconfigurations in AS could result in drawbacks such as the possibilities of policy conflicts and incorrect goal specifications. Again it is clear that some AS frameworks facilitate decision-making both at design-time and run-time. It is then necessary to consider testing both at design-time and run-time.
- Integrated: Testing should be an integral part of the whole self-management architecture. Testing being integrated to the management structure achieves real time validation which is necessary to mitigate adaptation conflicts and promote consistency.
- Automatic: Validation activity should be human independent (i.e. should be triggered by a change in application context, environmental volatility or a locally-detected failure requiring reconfiguration) following a defined validation process. But proving that a validation mechanism actually meets its set requirements is another issue of concern.

2.2.3 Models for Autonomic Assessment and Software Quality

As a novel research domain, autonomic computing is the main development direction of large-scale network and computer. The autonomic evaluation has important meanings. IBM has created an autonomic assessment software tool to measure the level of autonomic function against each of the six operational areas within the I/T environment. Since autonomic computing aims essentially at improving the QoS of systems, [19] and [20] try to define an autonomic computing evaluation model based on the ISO/IEC 9126 standard [21]. They describe the qualitative binding between autonomic characteristics and the ISO/IEC 9126 factors (Figure 7).

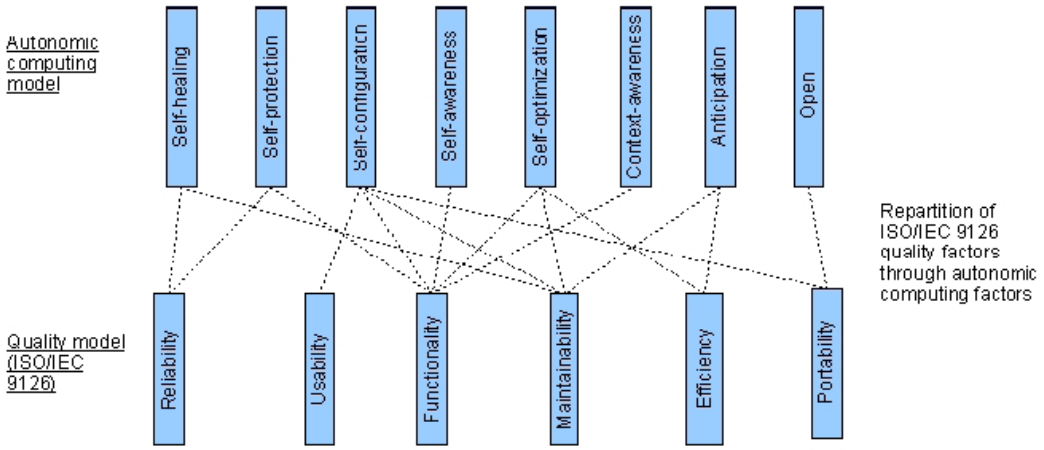


Figure 7. Organization of autonomic computing characteristics based on ISO/IEC 9126 standard quality factors

At the same time, [22] provides a qualitative hierarchy between the eight autonomic characteristics (the four main ones and the four secondary ones) (Figure 8). Finally some works focus on the definition of metrics in order to evaluate autonomic capabilities. For example, [23], [24] and [25] propose a non-exhaustive set of criteria and metrics that are not related to any evaluation standard such as ISO/IEC 9126. In detail, in [23] a set of metrics are defined by which the autonomic systems can be evaluated and compared, namely: Quality of Service (QOS), cost, granularity/flexibility, robustness, degree of autonomy, adaptivity, reaction time, sensitivity and stabilization.

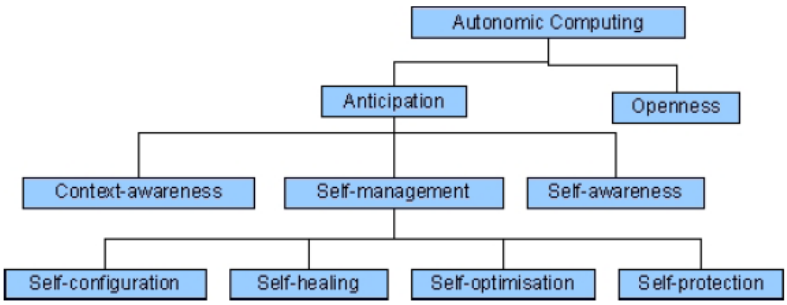


Figure 8.: Hierarchy between autonomic computing characteristics

Recently (2010), [16] defined the constituents, i.e. factors, criteria and metrics, of an hybrid (refined) ISO/IEC 9126 model for the autonomic computing area, similarly to the refinement of ISO/IEC 9126 model proposed in [26] while high-level indicators are defined for qualifying autonomic features. In addition an exhaustive set of metrics participating in the empirical evaluation of the ISO/IEC 9126 model has been identified (by using and enriching the list defined by [23], [24] and [25]). A hybrid ISO/IEC 9126 model for autonomic computing is defined (Figure 9.) in line with the proposals of [20] and [27] concerning the coupling between autonomic characteristics and software quality factors.

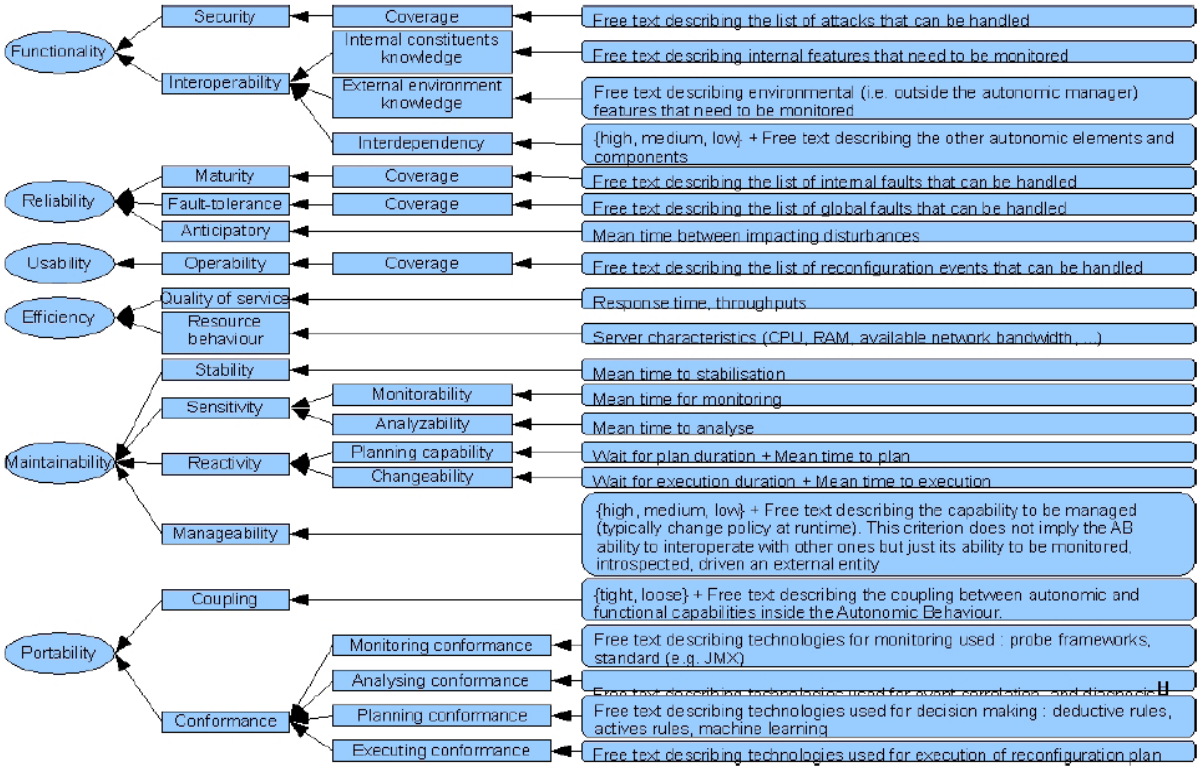


Figure 9. Quality model for autonomic computing assessment

This model is also used in the context of the Deliverable D46 - Synthesis of Deployment results which reports on the deployment assessment of the UniverSelf technologies and solutions. The different factors, criteria and metrics are or will be used accordingly for the evaluation of the deployment readiness of the project solutions as a step towards the validation of the solution and their conformance evaluation, as part of the certification process highlighted previously.

3 Results and Analysis

In this section we report the motivation, formalisation and proposed approach for the unification of trust and self-management using seven LTE SON control loops as our working example.

3.1 Shaping Trust in Autonomics

"Business wants trust and predictability above all else. Predictability makes it easy to make a simple story to sell to someone, it brings trust in what you are selling, and trust is the basis of all human interaction" [12]. Exactly because of this our choice of desired technological add-ons is in favour of a technology that helps to mediate between humans and networks. This technology is policy. We quickly outline what we borrow from the field and what are our contributions to policy.

Our approach to policy and policy conflict avoidance closely follows that of M. Sloman. According to [8], policy is a rule defining a choice in the behaviour of a system; policy domain is a set of objects and subjects with similar policies; policies are of the two types – obligation (**O**) that are set on subjects, and authorisation (**A**) that are set on objects, both with positive and negative modalities. Sloman's recommendations for policy conflict avoidance are: 1. Sub-domain policy overrides domain policy; 2. **A-** overrides **A+**; 3. Recent policy overrides older; 4. Short term policy overrides long term policy, etc. It is easy to see that all these recommendations remove conflicts between policies by separating conflicting ones into non-overlapping policy domains, where non-overlapping has *spatial* (or organisational) and *temporal* dimensions.

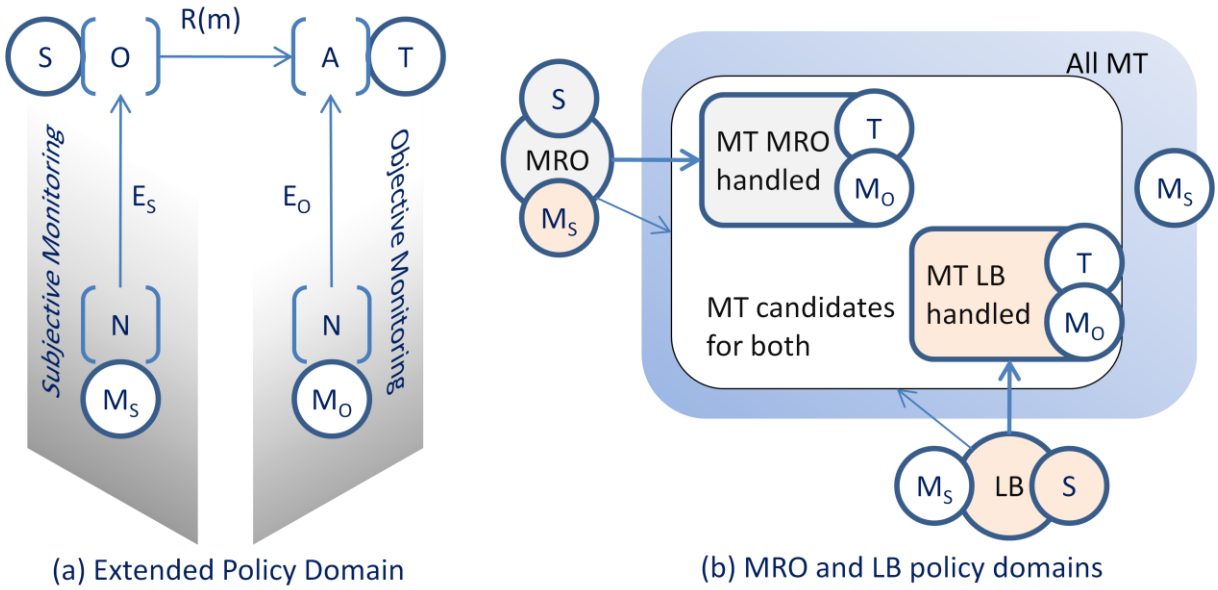


Figure 10. Extended policy domain (a) and its mapping to SON LTE (b)
Legend: S – Subject; T – target object; N – notification obligation; A- action; M – monitor; E – event: R(m) – request for a method; MT – mobile terminal; LB – Load balancing; MRO – Mobility robustness optimisation

For the formalisation of our problem and the approach we use the extended policy domain from the previous work [11]. Conventional policy domain described above is extended with subjective and objective monitoring. As Figure 10.a shows Subject (**S**) configured with obligation policy (**O**) can request from target object (**T**) invocation of its method (**R (m)**). These invocations are conditioned to the authorization (**A**) by **A** policy set on **T**. Both **O** and **A** without loss of generality can be of a conventional *Event:Condition →Action* form. To cater for dynamic self-configuration behaviours the [11] extends the policy domain with the notion of yet another role –

Monitor (**M**), configured by a specific type of obligation policy - obligation to notify (**N**) policy enforcers on relevant changes in the operational context. The relevance is being defined by events and conditions contained in both **A** and **O**. It is postulated that in general there are two types of monitoring – objective ($\mathbf{M_O}$ notifies the **T** on the set of events $\mathbf{E_O}$), and subjective ($\mathbf{M_S}$ notifies the **O** on the set of events $\mathbf{E_S}$), which in general can happen in the non-overlapping realms.

We now instantiate the extended policy domain model for the two SON LTE use cases, namely Mobility Robustness Optimization (MRO) and Load Balancing (LB). Since "the main objective of mobility robustness optimization should be reducing the number of HO-related radio link failures" [7] by means of avoiding too early, or too late handovers (HO), or HO to a wrong cell, the MRO optimises the Time To Trigger (TTT) parameter. Concurrently operating LB can be in obvious conflict with MRO since LB "algorithm decides to distribute the UEs camping on or having a connection to a cell, in order to balance the traffic load. This may be achieved by delaying or advancing the handing over of the UEs between cells" [7].

The first step to avoid said conflicts is to create disjoint policy domains. In LTE environment subjects are mechanisms implementing SON use cases, they are configured with **O** policies to request Mobile Terminals (MT) to perform certain actions (in the both considered cases these actions are HO ones), thus MT's are the objects from the policy viewpoint and they also play the role of objective monitors that sense the radio environment. The role of subjective monitors is played by all MT's in a cell, as well as by MRO for LB and by LB for MRO. Clearly, the set of MT – candidates for both MRO and LB is the same; this is used to separate policy domains as shown in Fig.5.b: any MT that is present within a HO range will be handled either by MRO or by LB but not by both.

The $\mathbf{M_S}$ placed at LB communicates its utility policy to MRO, while the $\mathbf{M_S}$ placed at MRO communicates the MRO utility policy to LB, this exchange emerges during the SON operation and will be detailed later. Here we concentrate on observation and action models that must be part of LB and MRO to correctly share MT's from the common pool of candidates. For this we shall use detailed use cases that hereafter shall be termed *situations* in order to avoid confusion with SON LTE use cases. Information on all possible situations available at the decision process form the so called Situation Awareness Pyramid with the ideal observer, i.e. obvious decision making being its top; the bottom of the pyramid is characterised by impossibility to make informed decisions. We assume that both LB and MRO operate somewhere in the middle; we demonstrate later that LB must be closer to the top of the pyramid than the MRO.

One more extension we make to the policy domain is a novel type of policy that is neither obligation nor authorisation, it is rather an intention declared prior to an action, and as such it is conceptually very close to a promise in promise theory [5,12]. We term the new policy type a predicate; we consider rather grammatical than logical meaning of a predicate. If an atomic behaviour of certain control loop can be considered as a sentence, then the decision process of that control loop will be its grammatical subject. Consider for example the following sentence expressed as our predicate:

"MRO in cell A increments the TTT by 10%" → Predicate (Subject; Parameters)

Here we distinguish between subject (MRO) its action "… increments the TTT by …%", which is our predicate, while the rest (A, 10) makes a set of parameters. Note, however that if another control loop would exist that could modify Time To Trigger, then TTT would be part of the parameter set rather than of a predicate; the same applies to % - the units in which the increment is measured. The increment though cannot be part of a parameter set; when MRO chooses to decrement TTT it uses another predicate.

We shall use the following forms of a predicate: A-predicate=Predicate (*,*), which defines an abstract behaviour; P-predicate= {Predicate(S,*) | Predicate (*,P)}, which defines only partially qualified behaviour, and F-predicate=Predicate(S,P), which defines fully qualified behaviour. Predicates are used to define, establish, negotiate, etc. trust in the process of self-orchestration by control loops and between control loops inside a domain and across domains, however under any-scale-precise governance of a human. Trust predicates are to be defined at the design phase and verified at run-time, so that a self-orchestrating behaviour is safeguarded in a behaviour envelope that is constrained by achieved utility and threshold on components (goals) of this utility.

Definition of predicates can be considered as a specific step in the SON functional design that follows the step in which goal (operator) policies are designed. Such sequence of design steps assures that all SON behaviour choices targeted by goal policies are being properly externalised via the predicates.

3.2 Measurable Trust

Independent of the particular method (e.g. Bayesian Networks, Markov model) that may be applied for evaluating whether a control loop is trustworthy or not, a first important step is to define observable parameters/metrics that can be measured and monitored in a system, and that can be linked to a level of trustworthiness. A set of generic observable parameters that can be considered for the self-evaluation of control loops so as to derive a measure of their trustworthiness includes (but is not limited to) the following:

- Deviation from requested goals of a control loop (e.g. Quality of Service (QoS) levels)
- Resources involved for the enforcement of certain control loop(s) decisions/actions.
- Time required for the enforcement of control loop decisions/actions.
- Number of reconfigurations deriving from certain control loop decisions/actions.

It should be noted that different observable parameters may be considered for diverse self-management functionalities (control loops). Such observable parameters can be measured for each executed control loop to obtain an estimation of an "instantaneous trust index" [1], e.g. as a weighted sum of the observable parameters following an approach of [2] for the evaluation of an interaction between, say an agent and a user. It should also be noted that the term goal above refers to what should be achieved by a control loop. In order to achieve contractual agreement elements, self-management functions (control loops) must reach certain goals. Thus if the decision of a certain control loop deviates from these goals this decision should not be deemed as trustworthy. Consequently, the larger the measured deviation, the number/amount of resources involved, the time required and the number of reconfigurations, the higher the level of inefficiency of certain control loop decisions/actions. In general a high level of inefficiency can be mapped to a low level of trustworthiness, i.e. to a low instantaneous trust index. More specifically, if the level of inefficiency of a control loop exceeds a specific predefined threshold, its instantaneous trust index will be decreased.

In order to obtain an "overall trust index", the long term performance of governance and self-management functions should also be taken into account, considering instantaneous as well as past information. The metrics, the instantaneous and the overall trust index are combined in the below proposed model (in section model-driven trust), following a similar to Q-learning approach in order to enhance the decision making process of a control loop in terms of trust.

3.3 Domain-specific Trust

The domain of LTE SON can be characterised as a highly dynamic one: both uplink and downlink capacities for a mobile user will be increased dramatically alongside with possibility for a user to move while being connected at a speed of up to 300 km/h. Another aspect of dynamicity is a number of SON features that are not only standardised but are reported by vendors as being implemented. It appears that conventional management approaches will not work in such dynamic environment.

We consider a number of SON LTE controls co-existing in operator network(s) and jointly performing specified optimisation of various RAT parameters. These co-optimizations may lead to conflicts [9] and in general to the lack of network stability, which might cause a massive loss in operator revenue, and as a result lack of operator trust in the SON capabilities of the LTE technology. This in turn, creates a fundamental issue with the LTE architecture; in a radical move towards flat architecture the LTE embraces SON paradigm as a substitute to the total centralization (the Core Network concept) of previous mobile network architectures. This way it appears that operator trust can be based on the assured (and perhaps certified) stability of the SON, which translates into the stability of all co-existing controls in all possible operational situations.

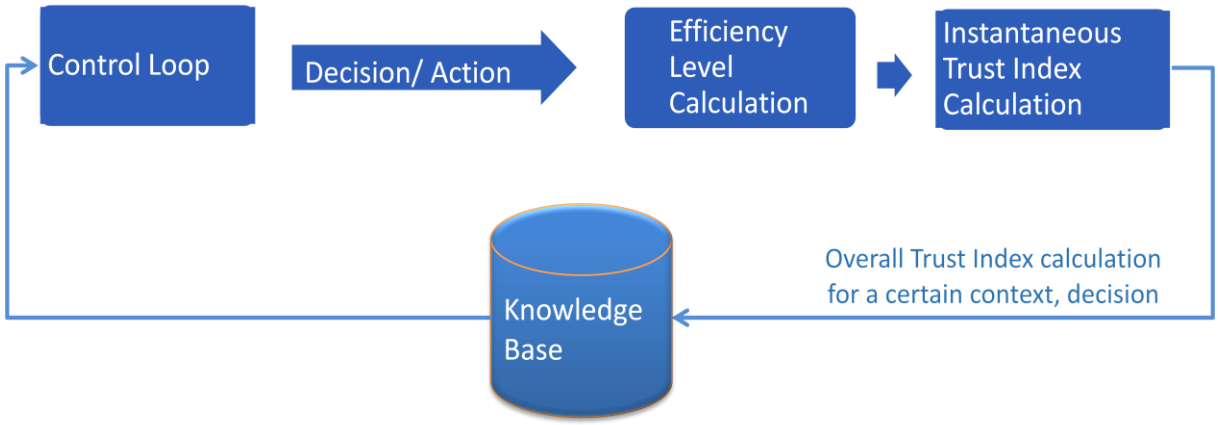
The controlled emergence in the orchestration plane appears to be an important issue. This can be explained by a very high heterogeneity of parameters controlled by SON mechanisms, and respectively by heterogeneous impact on the set of target KPI. The heterogeneity appears from the fact that almost all controls when modelled as a set of Finite State Machines will have states that are long-lived and short-lived; additionally transitions between some states will be relatively low, and quick between other states. Orchestration though will be required only in certain states and triggered only by certain transitions; thus, we conclude it is mostly impossible to build a unified (in state space and in interaction patterns) orchestration plane. Instead, we attempt to build an orchestration that is dynamic.

The emergence will follow dynamic hierarchy in which primary control loops are those that support the real world events (e.g. mobility events), while all sophistications should go on top. The framework for the

emergence can be also expressed in predicates, such as trigger, continue, speed-up, slow-down, stop, etc. which themselves can be seen as secondary control loops embedded in SON, and serve the purpose of building, controlling and dismantling the orchestration plane on demand. Orchestration on demand (or, emergence of control communication) is conjectured here as the main facilitator of SON LTE stable and safe interactions. Nevertheless, it is possible to include additional useful and generic orchestration behaviour elements, such as leader election, leader rotation, bootstrapping to a community, etc. – those will help to further optimise the SON.

3.4 Model-driven Trust

The method described here as a generic one (and applied to real SON LTE cases in the next section) targets at enforcing a control loop with the knowledge if its decisions are trustworthy enough. This knowledge may be exploited so as to improve the performance of the control loop by excluding decisions/actions that do not achieve desired trustworthiness levels. More specifically, the method stores information on situations encountered by the control loop including the corresponding decision that was applied for handling them in a knowledge base similar to the one depicted in Figure 11. This allows considering past interactions so as to allow faster and more efficient handling of problems. Thus, a control loop is enabled to reach its most trustworthy decisions given the current context of the system. Furthermore, the decision making process can be enhanced in terms of reduced time required for selecting a particular action, as a former "trustworthy" decision may be applied for the same/similar context without the need of executing an optimisation process.



State ID	Decision ID	Trust Index
201201001	201201101	60%
201201002	201201201	75%
201202003	201201202	52%
201201001	201201202	80%

Figure 11. High-level view of control loop trustworthiness assessment process and update of corresponding knowledge base

Each time a decision is made, the control loop provides information on the contextual situation (parameters of the trigger for the Control loop) and the corresponding decision made. The values of relevant metrics are retrieved after the application of the decision of the control loop. The retrieved values are used so as to calculate and update the efficiency level, the instantaneous trust index and the overall trust index, given the context of the system. The knowledge-base is updated accordingly.

The proposed model follows Q-learning approach and the flow of Figure 11. In particular, after the selection of the metrics and the collection of the measurements, the latter will be used for calculating the level of efficiency of the control loop. The way that the Efficiency Level (EL) can be calculated is given by equation (1) and may involve different functions *F* such as the weighted sum of the metrics.

$$EL(t) = F(m_1, m_2, ..., m_n) \quad (1)$$

where m_1, m_2, ..., m_n stand for the different metrics.

Additionally, a threshold for *EL* is defined with respect to equation (2), i.e. the value of function *F* when the minimum desired values of the metrics are used.

$$EL_{thres} = F(m_{1,opt}, m_{2,opt}, ..., m_{n,opt}) \quad (2)$$

Consequently, if the *EL* of the control loop decision is below the defined threshold, the control loop decision will be rated negatively i.e. the Instantaneous Trust Index (ITI) (corresponding to the payoff of Q-Learning technique) for the certain state and action (decision)/ selected control loop will be a negative real number. On the contrary, if the *EL* of the control loop decision is over the defined threshold, then the control loop decision will be rated positively, i.e. the corresponding payoff (ITI) for the certain state and action (decision) a will be a positive real number (equation (3)).

$$EL(t)\begin{cases} < EL_{thres}, & r(s(t), a(t)) < 0 \\ > EL_{thres}, & r(s(t), a(t)) > 0 \end{cases} \quad (3)$$

where represents the payoff for a particular system state and action at instance .

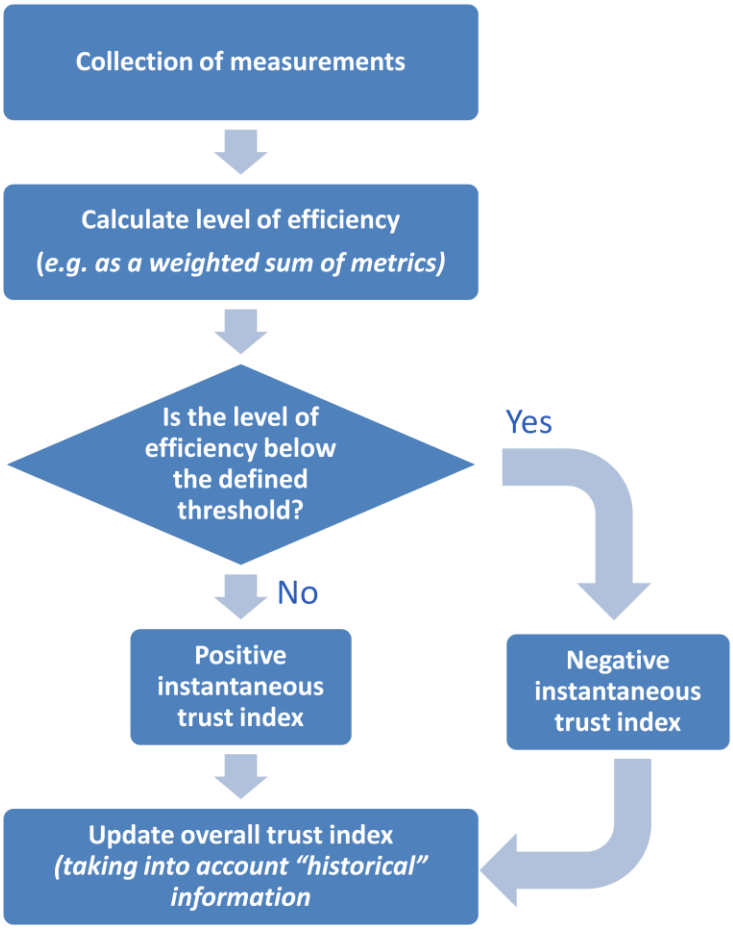


Figure 12.Overview of process for updating the Overall Trust Index

Accordingly, the ITIs will then be used to update the Overall Trust Index (OTI), which provides a more aggregated view of the trustworthiness of a particular decision/control loop over time (taking into account past trust estimations). In other words, it comprises both instantaneous information as well as "historical" information. Following the Q-learning approach [3], the OTI will first be calculated and then, during the next iterations, be updated until it reaches its maximum value according to equations (4) and (5):

$$Q(s(t),a(t)) = \left\langle \sum_{t=0}^{\infty} \gamma^t r(s(t),a(t)) \right\rangle_{s,r} \quad (4)$$

$$Q(s(t),a(t)) \rightarrow Q(s(t),a(t)) + \varepsilon[r(t) + \gamma \max_{a(t+1)} Q(s(t+1),a(t+1)) - Q(s(t),a(t))] \quad (5)$$

where corresponds to the OTI when the control loop is triggered by a situation (system state) and reaches decision and symbol $\langle \ \rangle_{s,r}$ refers to the average value. The discount factor stands for the weight of the payoff and is closely related to the time that has elapsed from the payoff, i.e. the larger designates that the more distant payoffs are more important. Moreover, parameter denotes the learning rate of the system.

As already mentioned, the derived knowledge on situations encountered and corresponding decision made by the control loop from the above described method will then be stored in a knowledge base. According to this knowledge base the control loop will be enabled to select the most trustworthy decision given its inputs and the trigger. It should be noted that the focus of the presented method is more on assessing the performance of a control loop (or potentially a set of control loops) and consequently derive how much it can be trusted to operate autonomously and is not relevant to security. Nevertheless, the use of the described method and its outcomes may help to identify situations when conventional isolation mechanisms need to be applied.

3.5 End-to-End Trust

The Control Loops (CL) comprising LTE SON pose a challenge on a MNO. On one side the self-optimization of RAT parameters that SON brings is much desired especially facing massive deployment of LTE technology soon to happen. On the other side the very limited experience with LTE and absolute absence of any experience with SON are very good reasons for operators to switch off the SON functionality in their early LTE deployments.

We know from a large European MNO that many vendors are offering multiple SON solutions, yet nobody can prove and/or demonstrate that those multiple solutions when deployed in the same network and operated concurrently will not ruin the network. Operators' trust in SON may increase with experience based on particular deployment; however this will be operator-specific part of the overall trust.

In general, all operators are interested to see either a rigorous proof or a convincing demonstration of the durable, safe and stable operation of a SON in the highly dynamic LTE environment. SON-wise the LTE dynamics can be defined as a flow of heterogeneous optimization tasks, which flows very possibly can be self-similar – with no particular regular pattern at any time scale. Each optimization task (e.g. in mobility, coverage, energy efficiency, etc) will have its own time range (window of opportunity to solve the task), cost and utility that the optimization brings. We can also easily imagine that cost and utility will be non-linear functions in the time range, and we know that those functions are different for different CLs. We try to report in the below our achievements in the area as a chronology.

We have asked ourselves a question: whether it is possible that SON CL's being highly heterogeneous from the Time, Cost, and Utility viewpoint exhibit certain similarity from some other viewpoint? This question became important as soon as we have realised the high complexity of the interaction of multiple CLs. Following the pioneering work reported in [9] we started our analysis with the detection and resolution of conflicts between co-existing CLs. During this work the graphical representation of interactions between control loops by means of concept maps appeared to be not only fruitful but also revealed the importance of the above question.

Good news from our concept mapping experience was the fact that the internal structure of all CLs is almost the same for all controls. It turned out that each CL can be decomposed into two sub-loops – one increasing the values of certain control parameters, another – for decreasing these values, both in response to the observed (measured, sensed) changes in the target Key Performance Indicators. As the concept map in Figure 13 demonstrates the three co-existing CLs, namely MRO, RACH, and LB are each consisting of the two sub-loops and are tightly interconnected through their influence to LTE cell parameters and through their joint but

not yet orchestrated response to the changes in the overlapping KPIs. The former opens the opportunity to orchestrate co-existing controls by manipulating their on-off processes, but the latter was really the source of the bad news known as the state explosion problem.

A model (concept map or state machine) of only three co-existing control loops already is so complex that does not facilitate required reasoning about the orchestration of the interactions. Such reasoning appeared to us as the necessary prerequisite towards the design of a management approach for LTE SON. We must clarify here what we really mean by SON management, since it must depart from a conventional paradigm "Manager – Managed Object" because objects (i.e. control loops) in SON are designed to be self-managed in certain respect.

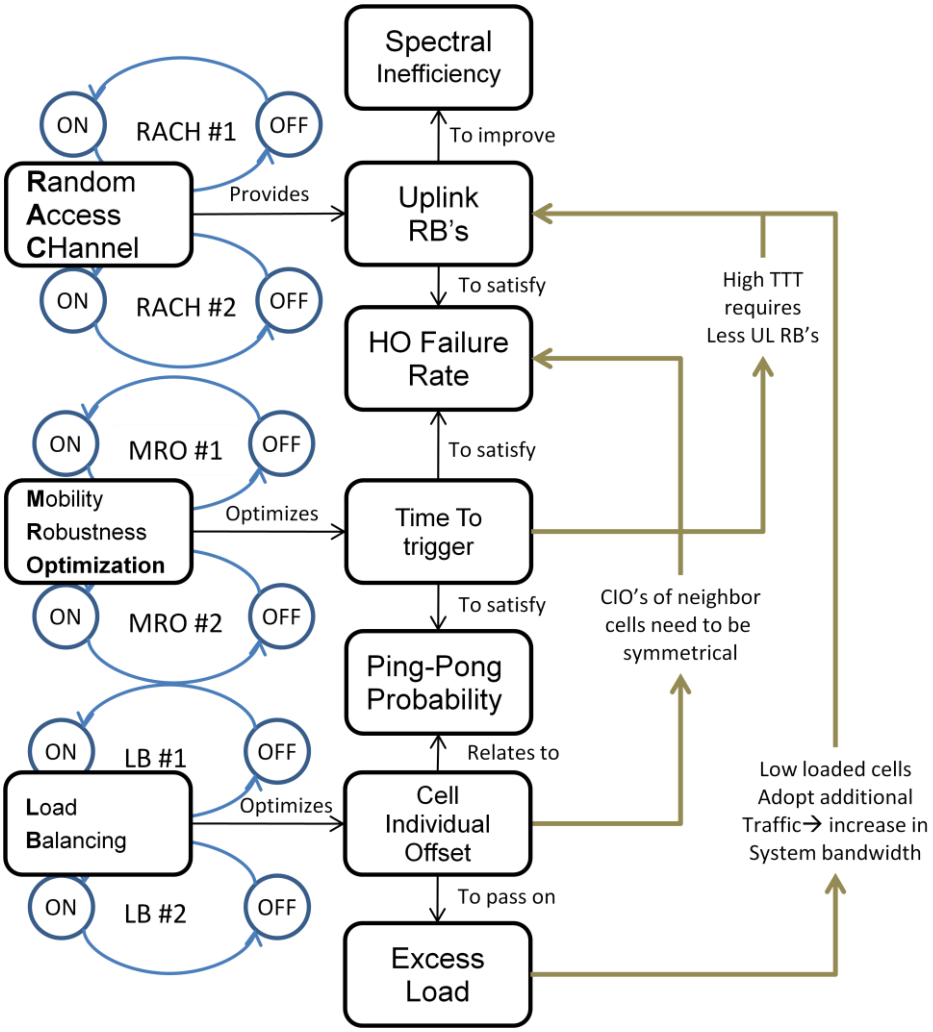


Figure 13. Three interacting SON control loops
Legend: RB – Resource Block, CIO – Cell Initial offset, TTT – Time to trigger

The self-organizing capabilities of control loops must be carefully orchestrated, and the orchestration must be as much as possible automated, so that the role of a human in this novel management (we term it governance, perhaps unconventionally but with the clear goal to differentiate it from conventional management) shall become more creative and more forward-looking than currently. This is of course impossible without trust that, we claim will be facilitated by *any-scale-precise* understanding by human of what exactly and how is happening in the SON, of course when needed, on demand. This brings us to the idea of an orchestration model, i.e. a model that when facing a human is capable to provide detailed explanation of SON operation, predictions of future behaviour. That orchestration model hands to a human not only controls (e.g. via goal policies) by which SON operation can be changed in the desired direction, but as well guides the human intuition and suggests the desired directions.

The orchestration model facing SON controls is necessarily a dynamic one, somewhat similar to scripts handling procedural knowledge. As M. Burges puts it, "... to start from these basic atoms, with individual intentions that can be combined into a larger picture ... with their own copy of the script, much like the players in an orchestra" [12]. It is important to take into account these “individual intentions”, which for SON controls are their optimisation tasks. Without orchestration of control loops their individual intentions might become counter-productive from the holistic viewpoint.

The two faces of our orchestration model are so different that must be implemented by different means. The human facing model might be a well-known and convenient tool routinely used in human practice, so that there are no learning barriers. The SON orchestration model facing a human operator must reflect differences in common attributes of different SON mechanisms such as Time, Utility, and Cost of operation as well as the impact of each mechanism operation but must be mechanism agnostic. This facilitates certain modularity of SON mechanisms from the orchestration viewpoint; SON from different vendors and/or with different capability sets may coexist and can be orchestrated using the same adjustable model. Any point in time a human operator must be able to zoom into on-going orchestration by requesting model update with actual network data; at the same time another version of model can be used for the off-line generation of predictions, for the study of various What If scenarios. The SON facing part must be control-specific, and at the same time easily represented to a human. The dynamic linkage between the two parts was another, besides orchestration challenge that we were facing.

The starting point of our current understanding of SON orchestration model was an attempt [6] to put all seven currently considered LTE SON controls in one picture. Figure 14 shows the relations between nine groups of control parameters and 26 KPI’s that are to be monitored in LTE; the relation arrow between parameter and KPI is colour-coded by seven SON controls; basically Figure 14 in which controls are hidden in the relations, is a bipartite graph. It appeared to us that despite the high complexity of this graph it was a useful step towards the understanding of the required orchestration model.

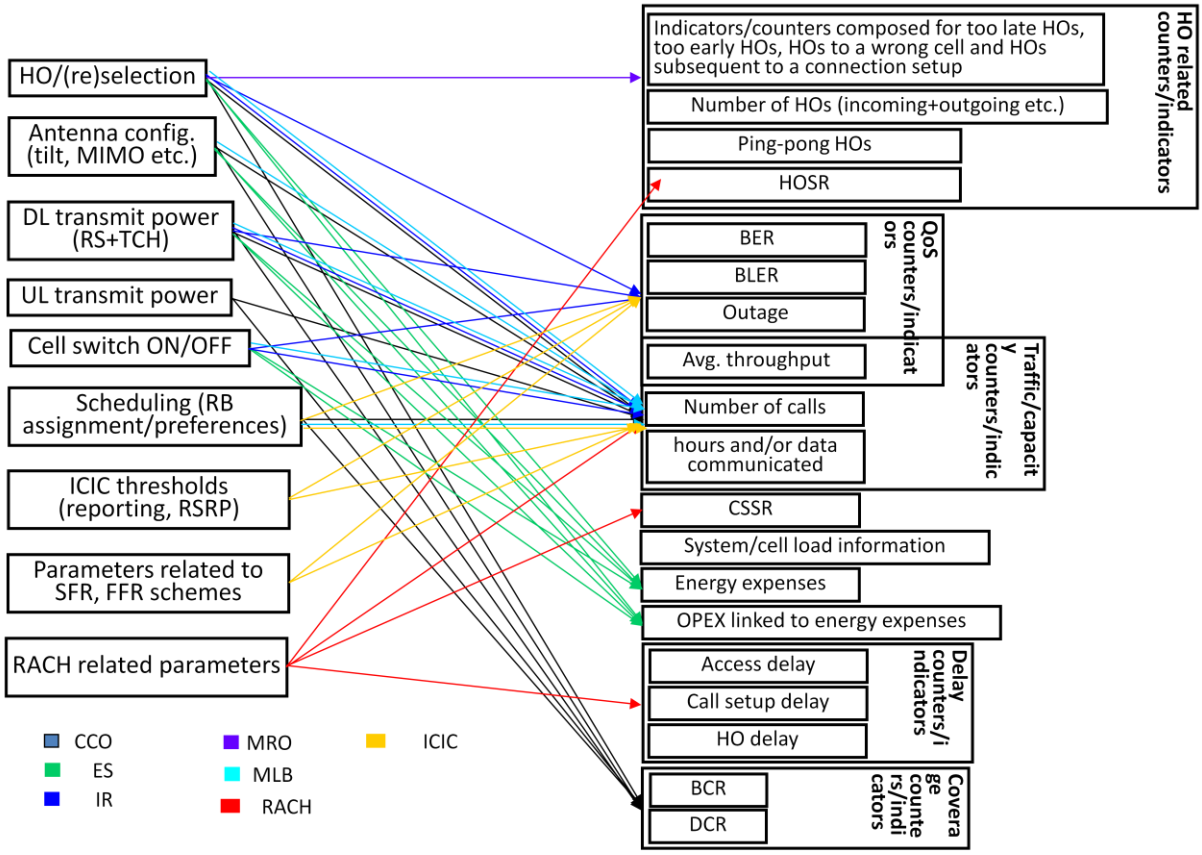


Figure 14. Seven interacting SON control loops

Legend: CCO- Coverage and Capacity Optimisation; ES – Energy Savings; IR - Interference reduction; MRO –

Mobility Robustness Optimisation, MLB – Mobility Load balancing; RACH- Random Access Channel; ICIC – Inter-cell Interference Coordination

We conjectured that this graph can help us to build such a viewpoint at interacting SON controls, from which all controls can be handled in a similar fashion. In certain sense we have looked at Fig.9 as at some kind of neural network that can be tuned into a self-orchestration mechanism.

To reveal the essential structural properties (traceable connectivity, degree of connectivity) of our graph we did redraw it to explicitly show SON controls, and clustered KPI's in the two groups – positive and negative respectively. The resulting graph (Figure 15) turned out to be very instrumental in the unification of metrics. But also it reveals important structure interconnecting our seven SON controls with our nine unified metrics, which is shown in the right part of Figure 15.

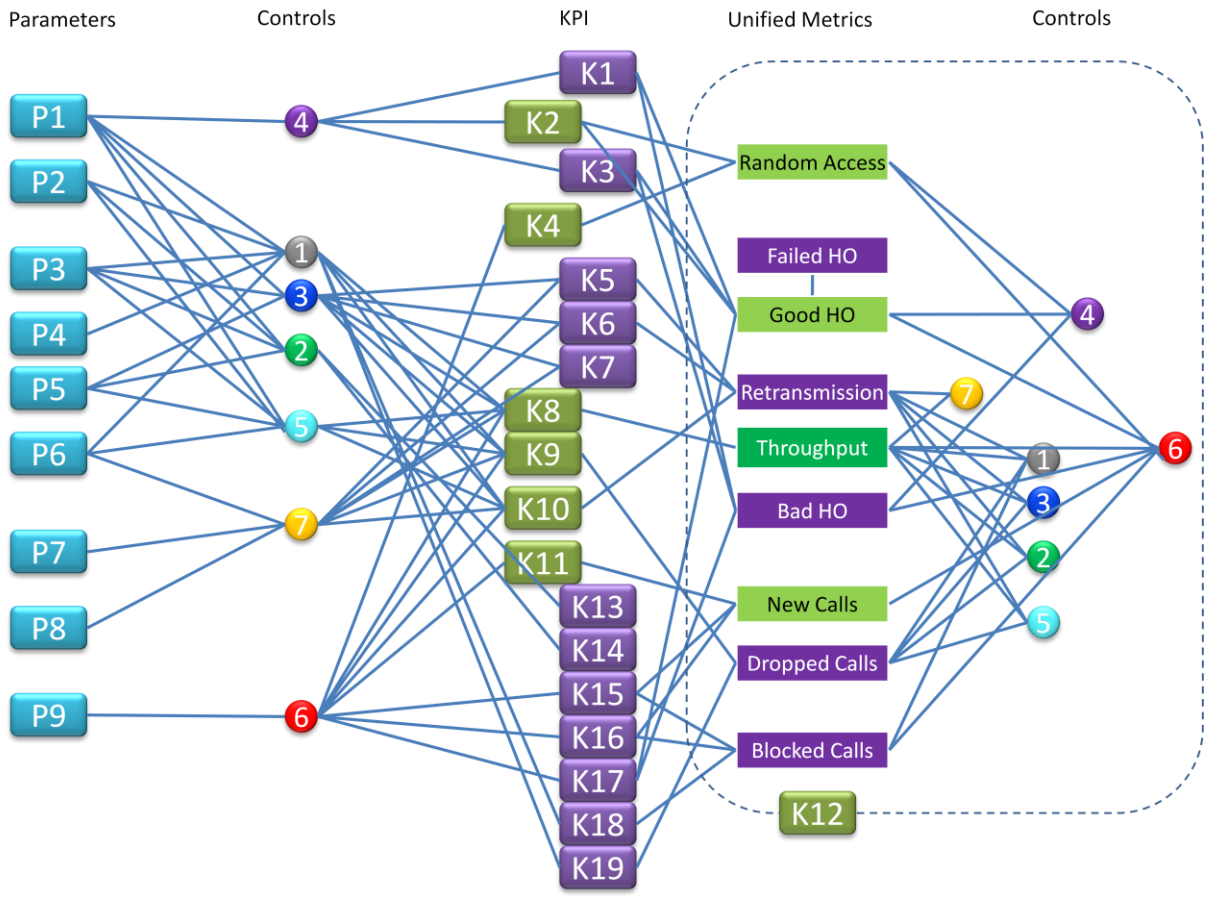


Figure 15. Towards self-orchestration model of SON

Legend: (see Table 1), Unified metrics*: A* – Random Access, *Gho* – Good Handover, *R*- Retransmission, *T* – Throughput, *Bho* – Bad Handover, *Nn* – New Calls,*Nd* – Dropped calls, *Nb* – Blocked calls

Not only it seemed hard to work with 19 monitored KPIs but also it did not help in orchestration because the KPIs are not fully independent. We have derived unified metrics in such a way that their total number is relatively small, and that all monitored KPIs are accounted in our metrics. Our nine unified metrics are listed in Table 2 together with the metric interpretation, whether it is productive or non-productive.

Our reasoning in defining the productivity of a metric follows a simple taxonomy of types of load that an LTE cell can experience: the load can be productive if it directly contributes to the throughput (paid and delivered load), or if it is a control load that is not wasted due to e.g. radio failures of a handover to a wrong cell, which is then an overhead. So defined metrics sum up to a total cell load as shown in the below expression, which together with the expression for energy expenses covers all 19 monitored KPIs.

Traditionally, in telecommunications, the load is measured in Erlang units however for the sake of self-orchestration it appears beneficial to consider slotted time, so that each KPI that contributes to the cell load above can be expressed as an absolute value number rather than as a rate. Assuming that LTE access time is in the range 20..30 ms, we would suggest to setup the time slot value to 240 ms, which value will then cover all important timing parameters such as call setup time and handover time.

The slotted time together with the unification of metrics did allow us to define utilities of invocations of control loops as a function of overall cell situation. Cell situation is given by the three components: 1. the cell current load – productive and non-productive; 2. the difference in load components in the previous time slot; and 3. the predictions for the next slot. Each SON control makes such predictions locally based on its local decision process, where the most important parameter is the threshold value of an observed metric.

These threshold values are given to SON controls as goal (governance) policies and can be expressed as a share of the cell load (*L*). Having nine metrics we can speculate that a fair share of the Throughput metric could be 50% of overall load, while it would be fair to share uniformly remaining 50% of the load between all other metrics, each accounting for 6,25% of the load. Since some load is productive and some load is non-productive we can express the overall utility of a cell *c* in a given time slot *t* as:

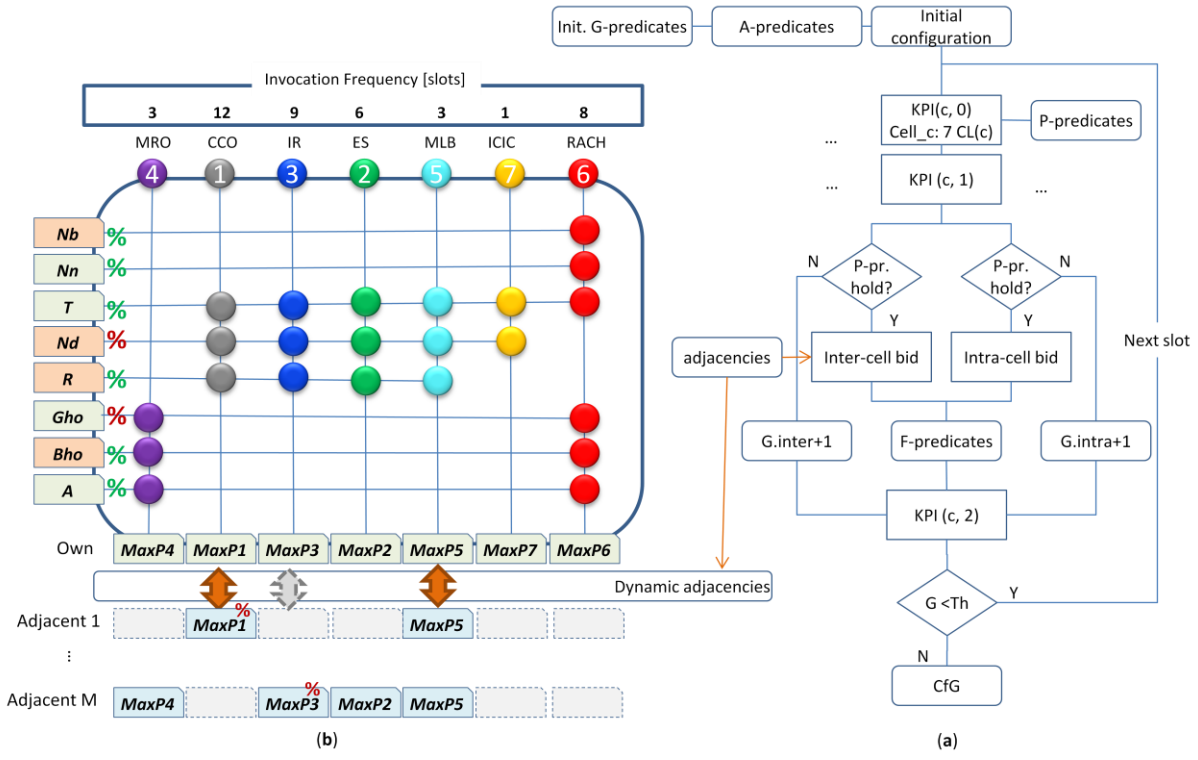


Figure 16. Self-orchestration logic of SON controls (a) operation in a time slot, (b) decision matrix
Legend: G – goal policy (and goal policy violation counters - intra-cell and inter-cell; Predicates; A-abstract; P-partially qualified; F – fully qualified, CfG – Call for Governance, Unified metrics: *A* – Random Access, *Gho* – Good Handover, *R*- Retransmission, *T* – Throughput, *Bho* – Bad Handover, *Nn* – New Calls, *Nd* – Dropped calls, *Nb* – Blocked calls

The proposed self-orchestration logic is detailed in Figure 16: we assume that all seven SON controls are permanently active in each cell – this is given by the initial configuration of a cell that includes abstract predicates for each control loop as well as settings for goal policies set by the governance. Before time-slotted operation starts certain bootstrapping must be made; abstract predicates are being partially qualified by each

control – now these are behaviours in particular cell parameterised accordingly (depending on equipment type, topology, cell adjacencies, antenna configuration, etc.). Those parameters of P-predicates that depend on measurements of KPI values (unified metrics in our case), which are not available yet are substituted by default values denoted as KPI(c,0) in Figure 16.a. The so qualified P-predicates are used by the controls in time-slot=0 to compute (basically, to guess) which of their behaviours must be performed and what are the expected utility increases.

We assume that all unified metrics can be evaluated (if not measured) during one time-slot, so when the values KPI(c,1) are available the controls will need to check whether all triggering conditions of actions selected at time-slot=0 still hold, or easily can be adjusted. Indeed the logic of each control is relatively simple, and even complete re-computations should be possible. However we reserve the option for each control to arrive in a situation, in which triggering conditions are met but a control cannot unambiguously select any action that promises to increase the utility. In this case a control experiencing this type of internal conflict must increment the goal policy violation counter.

Obviously, some SON controls in a cell are more others are less strongly interacting (explicitly and/or implicitly) with SON controls in adjacent cells. Our decision matrix (Figure 16.b) is able to determine the need of the interaction for each SON control. Depending on whether a control needs to interact with controls only in its own cell or with controls in adjacent cells it increments in the case of above described internal conflict either intra-cell counter (G.intra+1), or inter-call counter (G.inter+1). When the goal violation counter is above certain threshold (which can be set to 1) the cell must issue the Call for Governance (CfG). The CfG shall arrive at human-facing part of the orchestration model as a detailed warning enriched with all relevant information.

In a conflict-free operation mode each control that identified a possibility to increase the overall utility places a bid – its promise to increase the utility by certain value via the invocation of certain action in the next time-slot: intra-cell bid if sent to other six controls in the cell ; inter-cell bid is sent to dynamically identified set of adjacent cells.

The self-orchestration decision matrix (Figure 16.b) shows that time-slotted values of unified metrics are first compared to respective thresholds (shown as a %-sign next to a metric); if a non-productive metric is beyond the threshold this should have a higher priority than threshold violation for a productive metric – simply because productive metrics might reflect actual situation of low mobility or low traffic. Each SON control defines its action, if any for the next time-slot and informs other controls on expected utility increase. The action that promises the largest increase is selected for the next time slot, the same happens at inter-call level if the dynamic adjacencies set is not empty. This way we postulate that at most only one action (sub-loop) of only one control loop is performed in a cell in every time-slot. This effectively eliminates conflicts between controls; however, it does not help to avoid the effect of thrashing, i.e. resource contention.

It appears however that the very structure of our decision matrix (repeated in each cell) helps to avoid thrashing as well, and surprisingly, concurrently with the detection of the need of inter-call interaction of control loops. We noticed that if the relation structure between controls and the unified metrics (shown inside the dashed frame at Figure 15) is used to compute control to control dependency index, and to use it as a frequency (measured in time slots) of a control loop invocation then the following conditions hold.

First, the invocation frequency sets for a control a time range - a number of time-slots for computing the trend in values of unified metrics; intuitively it appears that thus computed frequency is in a very good alignment with the robustness of a control. For example, actions of CCO are those providing the most dramatic changes in the cell, mostly affecting adjacent ones, and according to CCO frequency it should be invoked only once in 12 time-slots. On contrary the actions of ICIC (e.g. switching an active mobile user from a direct connection to LTE base station to a relay and back) can bring very fine-grained optimizations in a cell load, absolutely invisible in adjacent cells (due to intra-cell frequency time division between relays and base station), and correspondingly ICIC invocation frequency is the highest – once per time-slot.

Second, the invocation frequency is an estimation only; if respective threshold is violated the control must act (or, at least to bid for an action). Consider MLB and CCO: these two controls are very likely candidates to inter-cell interaction. When it appears that their thresholds are violated more often than it would be estimated based on suggested frequencies then it makes sense to include these two controls in dynamic adjacencies. If it then happens that suggested frequencies are obeyed then the respective controls can be removed from the dynamic adjacencies. We observe in this dynamic behaviour that self-orchestration based on promises and on sharing expected utility values not only works but helps to propagate trust.

3.6 Trust of policy in Governance

The successful translation of high level to low level policies is of high importance from the operator's point of view. Term "successful" is used to express the sufficiency of the derived policies to accomplish the goals described by the operator in the high level policies. A successful policy will lead to well controlled and efficient network operations, while an unsuccessful policy may lead to misconfigurations, QoS / QoE degradation and network instabilities. Thus, a mechanism able to evaluate the policy translation process and measure the gains from the policy application is necessary. The success of a policy in accomplishing the goals described by the operator is in strong relation with the trustworthiness of this specific policy. Trust of policy is be defined as a comparison between the reference behaviour (the behaviour implied in high level policies) and the actual behaviour (based on measurements) of the network after the execution of the policy.

In order to estimate Trust, the Entropy-based trust model [36] can be used which uses uncertainty as measure of trust. In the proposed method of trust estimation, which is in line with the methods described previously in section 3.4, the concept of trust describes the certainty of whether the implemented policy will fulfil the objectives described in the high level goals. Information theory states that entropy is a natural measure for uncertainty [37][37]. Thus, entropy-based trust value is defined as:

$$T\{subject : policy, successful\} \begin{cases} 1 - H(p) & 0.5 \le p < 1 \\ H(p) - 1 & 0 \le p < 0.5 \end{cases}$$

where $H(p) = -p \log_2(p) - (1 - p) \log_2(1 - p)$ is the entropy function and p=P{policy,successful}.

According to this formula, the trust value is a continuous real number in [-1,1]. This definition satisfies the following properties. When p=1, the subject trusts the policy the most and the trust value is 1. When p=0, the subject distrusts the policy the most and the trust value is -1. When p=0.5, the subject has no trust in the policy and the trust value is 0. In general, trust value is negative for $0 \le p < 0.5$ and positive for $0.5 \le p < 1$. Trust value is an increasing function of p.

According to the aforementioned trust methodology, the policy assessment function calculates p after the implementation of a policy, based on network measurements collected by agents (reflecting a set of KQIs and KPIs according to TM Forum SLA Management Handbook) and assigns to this specific policy a value of trust. The estimated values of trust for specific policies can also be presented to the operator through the H2N Governance GUI in order to be able to supervise and control the underlying autonomic functionalities.

3.6.1 Evaluation of Trust of policy methodology

A set of simulations has been realised in order to evaluate the proposed methodology for Trust of policy estimation. The simulation scenarios are executed in the OPNET simulation tool [38].

The simulation environment is illustrated in Figure . The network infrastructure is composed of a DiffServ core domain, an access network and the servers' network. The DiffServ domain consists of three routers which support QoS based on the Per Hop Behaviour (PHB) paradigm. Towards this approach the Ingress and Egress Routers are supporting Weight Fair Queuing (WFQ) schemes in order to differentiate packets based on their DSCP (DiffServ Code Point) values.

The access network includes two users with different Type of Classes (Gold and Bronze) and consequently different SLAs. The Gold user's SLA has stricter QoS conditions, while the Bronze user's SLA is more elastic to QoS degradations. The service packets (coming from and destined to users) are marked with the appropriate DSCP code, to identify the WFQ PHB in the core routers. In more detail, according to the WFQ scheme, each DiffServ router has two queues with different WFQ weights. Each packet coming from or destined to a different type of user (different DSCP code) is placed to the appropriate queue and consequently is served with the appropriate weight.

It is assumed that both users are using Streaming service. In order to execute the simulations under a realistic environment trace files have been used for the simulation of streaming traffic (H.264 VBR Video) imported into the network.

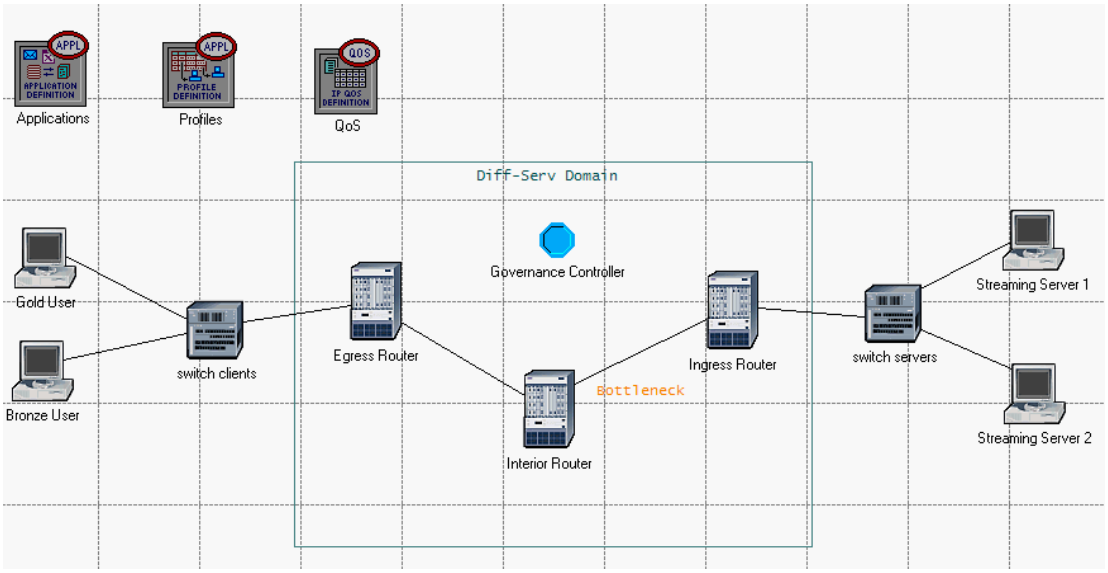


Figure 17. Simulation environment

A set of simulation scenarios have been defined, depicted in the following table in order to illustrate the policy translation and enforcement and evaluate the proposed methods of policy assessment and trust computation.

Scenario 1	Absence of high level policies.
Scenario 2	The operator defines a business goal which is translated to specific low level actions
Scenario 3	The operator defines the same business goal, but this time it is translated to a different set of low level actions.

In Scenario 1, the operator does not define and activate any business goal. In this case the network elements retain their initial configuration, which in our case means that same WFQ weight = 20 are assigned to all queues).

In Scenario 2, the operator defines the business goal “Gold users using the Streaming service should experience Good level of Speed”. According to the proposed methodology, this high level goal will be translated to a configuration action on routers which assigns a value of 40 in the WFQ weight of TOC=4 queue (queue of gold TOS).

In Scenario 3, the operator defines the same business goal with the previous scenario. It is assumed that in this scenario, the policy translation process results into a different policy representation based on slightly different models and model mapping. The initial business goal is translated to a device command of changing the WFQ weight to 30.

For all simulation scenarios, the total simulation time is 900 sec, the streaming services for both users start at around 100 sec, while the created policy is activated at 500 sec.

The end-to-end delay values of streaming service experienced by the Gold user in all scenarios are illustrated in Figure while end-to-end delay values experienced by the Bronze user are illustrated in Figure . From the first graphs it becomes obvious that in scenarios 2 and 3 after the activation of the policy (500 sec) the end-to-end delay during congestion times is suppressed from high values (around 3 sec) to low values, below 1.8 sec and 1.2 sec respectively. Thus, the behaviour of the network changes in order to fulfil the new goal set by the operator. The end-to-end delay of Bronze user increases to high values in scenarios 2 and 3. However this fact has no effect on total SLA degradation, as the SLA of Bronze user is elastic to high delays.

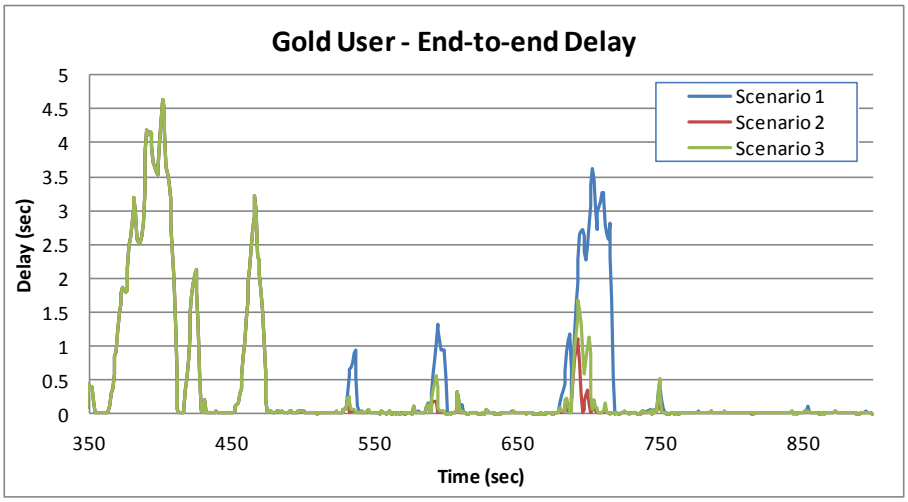


Figure 18. End-to-end delay of Gold user

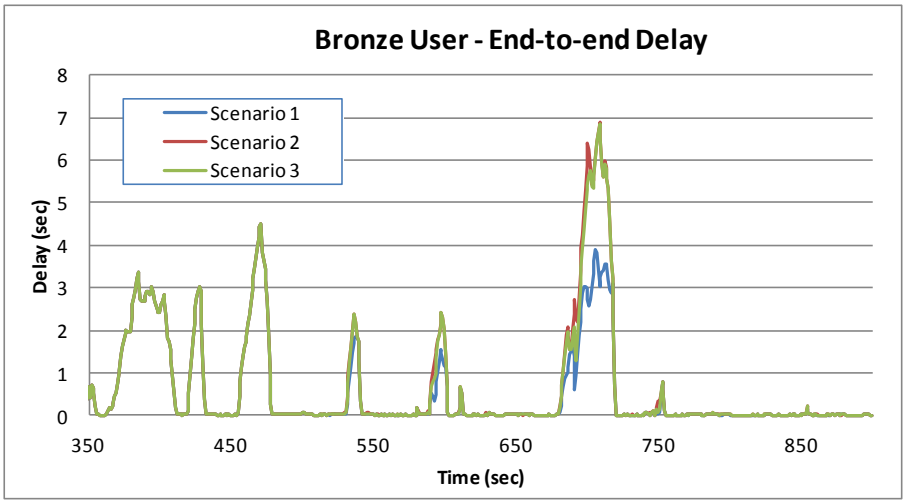


Figure 19. End-to-end delay of Bronze user

In order to estimate Trust according to the proposed methodology, the possibility p is calculated. The possibility p=P{policy, successful} express the possibility that the specific policy will be successful in fulfilling the business goal. The value of P is calculated based on measurements of end-to-end delay values collected from network monitoring after the implementation of the policy and information from upper layers of the policy continuum. In our case, based on Network view, it is assumed that a policy is successful if at least 90% of the end-to-end packet delay values are below 200msec. The following table summarises the estimated values of p, H(p) and policy trustworthiness according to the methodology described in the previous subsection.

Policy	p	H(p)	Trust
Scenario 2 Policy	0.8341	0.648	0.3519
Scenario 3 Policy	0.6708	0.914	0.086

The assessment of policy translation indicates that both policies are successful. In addition, as far as Trust of policy is concerned, it becomes obvious that policy of scenario 2 is more trustworthy than policy of scenario 3. In reality, policy of scenario 3 is untrustworthy at all as its value is near 0. In a real implementation, assuming that the Trust threshold for policy evaluation process is set to 0.3, the policy of simulation scenario 3 will be rejected, having policy of scenario 2 as the only candidate solution.

3.7 Method for certification of autonomic systems

The ultimate goal of autonomic systems should be to become certified. The proposed methodology illustrated in Figure represents the roadmap toward certification.

At point (1) of Figure the autonomic system is classified based on its Level of Autonomicity (LoA). In other studies it is referred to as Autonomic Control Levels (ACL) [32] or Degree of Autonomicity [33]. Still other papers term it the Autonomic Adoption Model (AAM) [13] or the Autonomic Computing Maturity Index (AMI) [34]. In this work, the preferred term will be LoA.

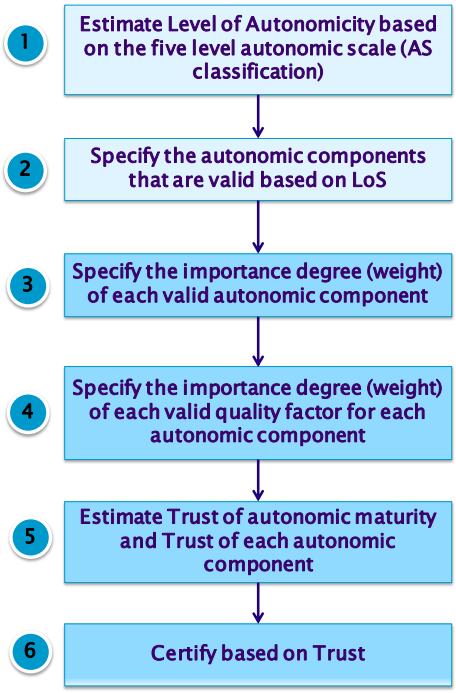


Figure 20. Certification process

LoA is a way of categorising ASs according to degrees. In our study we adopt the LoA classification proposed in [13]. According to this approach, the LoS is characterized by what parts of the system's autonomic management activities are automated versus those that are manually implemented; The resulting five level autonomic scale is as delineated below;

- Manual Level: At this level all autonomic management activities are handled by the human operator.
- Instrument and monitor: Here, the autonomic system is responsible for the collection of information: This collected/aggregated information is analyzed by the human operator and guides future actions of the operator.
- Analysis Level: On this level, information is collected and analyzed by the system. This analyzed data is passed to the human administrator for further action(s).
- Closed loop Level: This works in the same way as the Analysis level, only this time the system's dependence on the human is minimized i.e., the system is allowed to action certain policies.
- Closed loop with business processes Level: At this level, the input of the administrator is restricted to creating and altering business policies and objectives. The system will operate independently using these objectives and policies as guides.

The classification of the autonomic system to one of the above categories will give an idea of their full capabilities and also identifies what conditions they must meet for certification. The classification of AS based on LoA at this point is prerequisite to the next steps.

At points (2-4) of Figure 20 resides the system validation (both across design-time and run-time). The proposed validation and assessment method is based on the Analytic Hierarchy Process (AHP) which is a hierarchy weight

decision-making analysis method applying network system theory and multi-objective comprehensive evaluation method. By dividing decision-making factors into goal, rule and scheme, AHP can implement qualitative and quantitative analysis [31]. Adopting AHP, we construct a three-level autonomic evaluation model, including Level of Autonomicity (LoA), autonomic elements and quality factors. The hierarchy structure of the proposed methodology is showed in Figure . The factors in lower hierarchy will be evaluated firstly, and then the factors in higher hierarchy will be evaluated until the integrated evaluation result is got in the highest hierarchy. A similar approach is adopted by [20] in order to evaluate complex software.

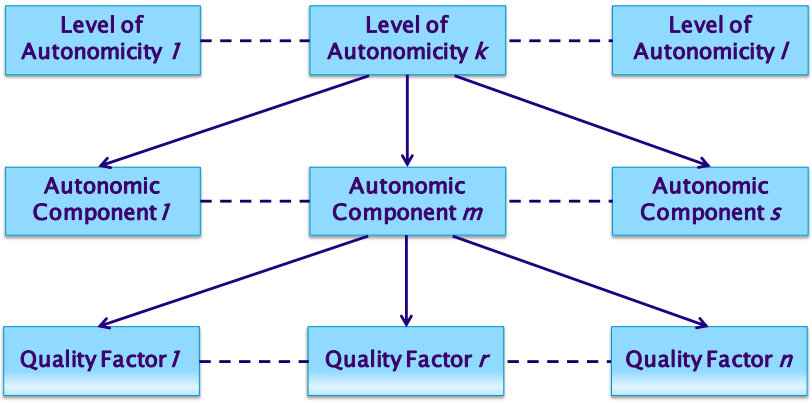


Figure 21. Hierarchical autonomic evaluation model

In the upper layer of the hierarchical autonomic evaluation model resides the Level of Autonomicity (LoA). LoA explains what level or extent of validation is needed. Therefore to autonomic systems with different LoA, the definitions of autonomic characteristics are different. For example, if the LoA of a system is higher, autonomic characteristics defined will be more and the autonomic maturity will be higher.

In the next layer a set of autonomic characteristic components are defined. The set of components is in line with the aforementioned autonomic computing model. Therefore autonomic characteristics include four major characteristics: self-configure, self-heal, self-optimize and self-protect; and four minor characteristics namely: self-awareness, open, context-aware and anticipatory. The characteristics can be described according to [35] as follows:

- Self-configuring: it should implement self reconfiguration dynamically according to the changing of internal and external environment, and keep robust and efficient continuously.
- Self-healing: it should detect the errors during the running and correct them automatically without influencing the normal running, greatly improving its availability.
- Self-optimizing: it can reassign resources according to the requirement of users or itself, ensuring the reliability and availability of service.
- Self-protecting: it must discover the intrusion or virus in time and adopt corresponding safeguard to maintain the normal state.
- Self-awareness: it should be aware of its state and control its behaviour, cooperating with other systems.
- Open: it must operate in a heterogeneous environment and is portable across multiple platforms.
- Context-awareness: it should be aware of execution environment and react to environment changes.
- Anticipatory: it must anticipate the optimized resources needed while keeping its complexity hidden.

Each type of aforementioned autonomic characteristic component is defined as an s-dimension vector according to AHP methodology. Each autonomic characteristic component is decided by n relative quality factors. We suppose that T_z expresses autonomic maturity of the l_{th} LoA category, S_{il} expresses the important degree (weight) of the il autonomic characteristic component in the l_{th} LoA category, and S_i expresses autonomic value of the i_{th} autonomic characteristic component. We can calculate T by using formula:

In the same way, we suppose that N_i expresses autonomic value of the i_{th} quality factor, and N_{im} expresses the important degree of the i_{th} factor in the m_{th} autonomic characteristic component. The autonomic value S_m of the m_{th} autonomic characteristic component can be expressed as in formula:

Therefore, the aforementioned methodology estimates the autonomic maturity T of the autonomic system based on the LoA, the values of the quality factors for all autonomic characteristics included and the selected weights of each level, which expresses the important degree of each characteristic/quality factor.

At point(4) of Figure the Trust of the autonomic system is estimated based on autonomic maturity T and the autonomic values S_m of the previous steps. Here, alternatives methods for Trust estimation can be adopted (some of them are already defined in this document). Using the preferable approach on Trust estimation the trustworthiness of AS is estimated based on maturity T, while the trustworthiness of each autonomic characteristic is estimated based on S_m values.

The final step toward the AS certification (point 6 of Figure) includes the examination of whether the Trust calculated in previous steps are above a set of predefined thresholds. In this step we discriminate between the following two approaches:

- The Trust value of autonomic maturity should be below a predefined value
- The Trust values for each one of the autonomic characteristic S_m should be below a set of predefined values

The second approach, which is stricter than the first, testifies that the autonomic system is certified for all its valid autonomic characteristic, namely self-configure, self-heal, self-optimize, self-protect, self-awareness, open, context-aware and anticipatory.

4 Conclusion

Clearly, the reported work is limited in many respects, and we plan to apply the reasoning outlined in this paper to more use cases than SON LTE only, and to continue this work in both simulation and prototyping directions. However we believe that this is a promising direction due to the following achieved benefits.

We demonstrated that trust can be achieved by means of technological (non-functional) add-ons embedded into the fabric of network functionality concurrently with the management (governance) of that functional fabric; in that we follow the opinion that “a general theory of trust in networks of humans and computers must be build on both a theory of behavioural trust and a theory of computational trust” [4].

We claim that trust (more precisely Operator trust in SON) must be supported by certain technology, and this technology must be equally human- and network-friendly, thus the self-orchestration model we are building has two faces, and is formalised as extended policy domain – traditional mediator between human goals and network capabilities and “intentions”.

Metric-wise, we propose to measure trust based on the dynamics of the cognition process, with several levels of trust indicators that help altogether to form the needed pyramid of situation awareness to make robust and informed decisions even in the situations of uncertainty.

Complexity-wise, we actually propose to combat the complexity associated problems with complexity itself as a weapon: careful and detailed analysis of local interactions that result in the emergence of complex and dynamic behaviours is indeed possible and can serve as the basis of embedded decision processes that will require much less human attention than currently.

These achievements we plan to bring to the design of a certification process that shall guide the entire life-cycle of future network equipment, and similar to ISO quality standards shall concentrate on process and on their unified descriptions rather than on properties of the final result. These properties should grow in the process of future network design and evolution.

References

[1] H. Chan, A. Segal. B. Arnold, I. Whalley, "How Can We Trust an Autonomic System to Make the Best Decision?," Autonomic Computing, 2005. ICAC 2005. Proceedings. Second International Conference on , vol., no., pp.351-352, 13-16 June 2005

[2] Y. Wang, J. Vassileva, "Bayesian Network-Based Trust Model", The Int. Conf. On Web Intelligence (WI'03), Halifax, Canada, 2003

[3] R. S. Sutton, A. G. Barto, "Reinforcement Learning An Introduction", Trends in Cognitive Sciences , Volume 9, Issue 5, MIT Press, 1998, DOI: 10.1109/TNN.1998.712192

[4] Virgil Gligor, Towards a Theory of Trust in Networks of Humans and Computers (announcement of the talk in Columbia University Department of Computer Science 12.DEC.02011 (accessed 10.DEC.02011 at 17:46 CET https://lists.cs.columbia.edu/pipermail/colloquium/2011q4/001416.htm)

[5] Wikipedia contributors, "Promise theory," *Wikipedia, The Free Encyclopedia,* http://en.wikipedia.org/w/index.php?title=Promise_theory&oldid=436133917 (accessed December 13, 2011).

[6] Berna SAYRAC, Simplifying SON interactions, Orange Labs, Research & Development, UniverSelf project, 23 Aug 2011

[7] 3GPP TS 36.902, V9.2.0, "Evolved Universal Terrestrial Radio Access Network (E-UTRAN); self-configuring and self-optimizing network (SON) use cases and solutions", 2010

[8] M. Sloman, home page at http://www.imperial.ac.uk/people/m.sloman

[9] Socrates deliverable D2.4: "Framework for self-organizing networks", EU STREP Socrates (INFSO-ICT-216284), 2008

[10] "SON and SON collaboration according to operator policies", Z. Altman (Ed.), UniverSelf Project, Use Case 4, 2011

[11] Mikhail Smirnov, *Cognitive Radio Control: The Disappearing Policy*, book chapter in"Cognitive Radio: Terminology, Technology and Techniques", NOVA Science, USA, 2010.

[12] Mark Burges, *Striking The Balance Between Man And Machine In IT Management*, A keynote presented in Paris at CNSM 2011 on 27th October 2011 on-line at http://cfengine.com/markburgess/blog_manmachine.html

[13] IBM, "An architectural blueprint for autonomic computing," IBM Whitepaper, June 2006.

[14] Haffiz Shuaib, Richard J. Anthony, "Towards Certifiable Autonomic Computing Systems – Background", Technical Report I, October 2010

[15] Haffiz Shuaib, Richard J. Anthony, "Measuring and Rating Autonomic Computing Systems", Technical Report III, February 2011.

[16] E. Xavier, C. Thierry, V. Guy, "Experiences in Benchmarking of Autonomic Systems", Autonomic Computing and Communications Systems, Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, Volume 23, 2010, p. 48.

[17] R. Zhang, R. Whang, R. Zheng, "An Autonomic Evaluation Model of Complex Software", ICICSE '08 Proceedings of the 2008 International Conference on Internet Computing in Science and Engineering.

[18] T. Eze, R. J. Anthony, C. Walshaw, A. Soper, "The Challenge of Validation for Autonomic and Self-Managing Systems", ICAS 2011, The Seventh International Conference on Autonomic and Autonomous Systems.

[19] Salehie, M., Tahvildari, L.: Autonomic Computing: Emerging Trends and Open Problems. ACM SIGSOFT Software Engineering Notes 4, 1–4 (2005)

[20] Zhang, H., Whang, H., Zheng, R.: An Autonomic Evaluation Model of Complex Software. In: International Conference on Internet Computing in Science and Engineering, pp. 343–348 (2008)

[21] Milicic, D.: Software Quality Models and Philosophies. In: Lundberg, L., Mattsson, M., Wohlin, C. (eds.) Software Quality Attributes and Trade-offs, p. 100. Blekinge Institute of Technology (2005)

[22] Lin, P., MacArthur, A., Leaney, J.: Defining Autonomic Computing: A Software Engineering Perspective. In: 16th Australian Software Engineering Conference, pp. 88–97. IEEE Computer Society, New York (2005)

[23] McCann, J.A., Huebscher, M.C.: Evaluation Issues in Autonomic Computing. In: Grid and Cooperative Computing - GCC 2004 Workshops: GCC 2004 International Workshops, IGKG, SGT, GISS, AAC-GEVO, and VVS, pp. 597–608. Springer, Heidelberg (2004)

[24] Chen, H., Hariri, S.: An Evaluation Scheme of Adaptive Configuration Techniques. In: 22nd IEEE/ACM International Conference on Automated Software Engineering, pp. 493–496. ACM Press, New York (2007)
[25] De Wolf, T., Holvoet, T.: Evaluation and Comparison of Decentralized Autonomic Computing Systems. Technical report. Department of Computer Science, K.U. Leuven, Leuven, Belgium (2006)
[26] Zeiss, B., Vega, D., Schieferdecker, I., Neukirchen, H., Grabowski, J.: Applying the ISO 9126 Quality Model to Test Specifications - Exemplified for TTCN-3 Test Specifications. In: Software Engineering 2007, Fachtagung des GI-Fachbereichs Softwaretechnik, pp. 231–244. GI (2007)
[27] Salehie, M., Tahvildari, L.: Autonomic Computing: Emerging Trends and Open Problems. ACM SIGSOFT Software Engineering Notes 4, 1–4 (2005)
[28] Tariq King, Djuradj Babich, Jonatan Alava, Peter Clarke and Ronald Stevens, Towards Self-Testing in Autonomic Computing Systems, Proceedings of the Eighth International Symposium on Autonomous Decentralized Systems (ISADS'07), Arizona, USA, 2007
[29] Andrew Diniz, Viviane Torres and Carlos José, A Self-adaptive Process that Incorporates a Self-test Activity, Monografias em Ciência da Computação, No. 32/09, Rio – Brasil, Nov. 2009
[30] Tariq M. King, Alain Ramirez, Peter J. Clarke, Barbara QuinonesMorales, A Reusable ObjectOriented Design to Support SelfTestable Autonomic Software, Proceedings of the 2008 ACM symposium on Applied computing, Fortaleza, Ceara, Brazil, 2008
[31] F.Z. Li, GD. Hu, "Model of Network Security Comprehensive Evaluation Based on Analytic Hierarchy Process and Fuzzing Mathematics," Ningxia Engineering Technology, 2006, Dec.
[32] B. T. Clough, “Metrics, schmetrics! how the heck do you determine a uavs autonomy anyway?,” Proceedings of the Performance Metrics for Intelligent Systems Workshop, Gaithersburg, Maryland, 2002.
[33] M. C. Huebscher and J. A. McCann, “A survey of autonomic computing degrees, models, and applications,” ACM Computing Surveys, vol. 40(3), August 2008.
[34] IBM, “An architectural blueprint for autonomic computing,” IBM Whitepaper, 2004.
[35] M. Salehie, L. Tahvildari, "Autonomic Computing: Emerging Trends and Open Problems," DEAS 2005
[36] Yan Lindsay Sun, Wei Yu, Zhu Han, Liu, K.J.R., Information Theoretic Framework of Trust Modeling and Evaluation for Ad Hoc Networks, Journal on Selected Areas in Communications, IEEE, Feb. 2006, Volume: 24 Issue:2, p: 305 – 317.
[37] T. M. Cover and J. A. Thomas, Elements of Information Theory. New York: Wiley, 1991.
[38] OPNET, Website: http://www.opnet.com/
[39] IEEE Spectrum Magazine, December 2011, p38-43, http://spectrum.ieee.org/magazine/2011/December

.

Abbreviations

CCO	Coverage and Capacity Optimization
CL	Control Loop
EL	Efficiency Level
ICIC	Intra-Cell Interference Coordination
ITI	Instantaneous Trust Index
HO	Handover
KPI	Key Performance Indicators
LB	Load balancing
LTE	Long Term Evolution
MLB	Mobility Load Balancing
MNO	Mobile Network Operator
MRO	Mobility Robustness Optimization
MT	Mobile Terminal
OTI	Overall Trust Index
SON	Self-Organizing Network
QoS	Quality of Service
RACH	Random Access Channel
RAT	Radio Access Technology
TTT	Time To Trigger
UE	User Equipment

Definitions

Definitions of SON LTE :

- Parameters (P1..P9),
- Key Performance Indicators (K1..K19),
- Mechanisms (M1..M7)

P1	Handover/ (re) selection
P2	Antenna configuration (tilt, MIMO, etc.)
P3	Downlink transmit power RS+TCH (Received signal) (Traffic Channel)
P4	Uplink transmit power
P5	Cell switch ON/OFF
P6	Scheduling (RB assignment / preferences)
P7	Inter-cell Interference Coordination thresholds (reporting, RSRP –Reference Signal received Power)
P8	Parameters related to SFR- Syndicated Feed Reception, FFR-Fractional Frequency Reuse schemes
P9	RACH- Random Access Channel related parameters
K1	Indicators/counters composed for too late HOs, too early HOs, HOs to a wrong cell and HOs subsequent to a connection setup (HO- related)
K2	Number of HOs (incoming+outgoing etc.)
K3	Ping-pong Handovers
K4	HOSR – Handover Success Rate
K5	BER- Bit Error rate
K6	BLER-Block Error Rate
K7	Outage
K8	Average Throughput
K9	Number of Calls
K10	Hours and/or data communicated
K11	CSSR – Call Set-up Success Rate
K12	System / Cell Load Information
K13	Energy expenses
K14	OPEX linked to Energy Expenses
K15	Access Delay
K16	Call setup delay
K17	Handover delay
K18	BCR – Block Call Rate
K19	DCR – Drop Call Rate
M1	CCO – Capacity and Coverage Optimization
M2	ES – Energy savings
M3	IR – Interference reduction
M4	MRO – Mobility Robustness Optimization
M5	MLB – Mobility Load balancing
M6	RACH- Random Access Channel
M7	ICIC – Inter-cell Interference Coordination

Definitions of unified metrics, components of cell load (P – productive, N – non-productive)				
Metric	KPI Formula		Interpretation	P/N
T	K8	Throughput	Paid and delivered load	P
Nn	K11·(K11+K16)	NewSuccessCalls·(SetupDelay+AccessDelay)	Control load that brings new throughput	P
Nb	K18·(K11+K16)	NewBlockedCalls·(SetupDelay+AccessDelay)	Control load that brings no new throughput	N
Nd	K9·K19	Dropped Calls	Control load that reduces the throughput	N
R	K10·max(K5,K6)	Retransmission due to BER, BLER	Control load that reduces the throughput	N
Gho	K17·(K2 –K1 -K3)	HO control load	Successful, useful	P
Bho	K17·K1·K3	HO control load	Successful, not useful	N
Fho	K2·(1-K4)	HO control load	Failed	N

Annex A

Here we report the detailed trust facets in a form of a questionnaire distributed to a number of relevant groups and experts in trust. The questionnaire is structured in such a way that simultaneously defines the domain of interest and the phases of the system design and applicable trust mechanisms.

4.1 AUTONOMOUS (Self) Features

1. In your domain, which functions are automatized or rendered autonomous (i.e. functions where the "human operator" is removed from decision loop, control loop or both)?
   Importance: Gaining knowledge from the experiences in other fields, through understanding and identifying what, how and why autonomicity was achieved in other industry domains.

2. What were the difficulties and the roadblocks encountered from regulatory, standard, technology, business and market, psychological and societal perspective in deploying autonomous features? What were/are the main enablers and technology leaps introduced to overcome these issues?
   Importance: Understanding the general difficulties and the process of achieving solutions in the development of the autonomicity in other industry domains.

3. In your domain, does a certification organization, system or process exist? If yes, what have been its initial goal(s) and definition? Which aspects of the system/infrastructure/process/... is the certification organization supposed to guarantee (i.e. the technical objective)? What models or approaches have been selected and developed to implement the certification process (and, if possible to answer, why has this model in particular been selected)?
   Importance: Understanding the process (success story?) for the definition and the deployment of the certification process in order to guide our own development and possibly avoid pitfalls and dead-ends.

4.2 DESIGN and Modelling

4. From your perspective, which is the attack model that should be considered in designing an autonomous system containing “trust” functionality?
   Importance: "In the history of computing there has often been a 10 or more year gap between the use of technology and the addressing of security issues that arise from it" (Virgil Gligor, University of Maryland, National Security Award 2006, Invited talk at The 3rd Annual VoIP Security Workshop, Berlin, Fraunhofer FOKUS, 01.JUN.2006

5. How is trust modelled, designed and which trust mechanisms are applied in your domain – e.g. in aviation for autopilots – which is analogue to the operator control in a telecommunications system?

Importance: Understanding the analogies with other industry domains (as far as trust is concerned) could be the key for convincing telecommunication operators of the reliability of our autonomous solutions.

6. Which main functional requirements of the management systems in your domain can be further fulfilled by self-functionalities? For each of them can we set up a methodology in the same way traditional security or safety methodologies do (such as common criteria for example) to attribute them a given level of trust? What would be the changes otherwise?

7. How to build "trust and confidence" in autonomics (self) features from the outset e.g. at deployment phase as well as optimization phase? Which process should be embedded within the management architecture from standardization and design principles allowing the operator to keep control if needed?
   Importance: Dedicated methodologies exist to measure the assurance that a system satisfies certain objectives when it is question of safety or of information security. We are looking for a method to adapt the assurance mechanisms in order to build a scale of trust for what concerns the objectives specific to our project.

4.3 Approach in Trust Design - Measurement and Metrics

8. From your domain perspective, how is trust modelled and measured currently? What metrics do you consider as most important for the computation and the evaluation/assessment of trust in autonomics?
   Importance: Gaining experience from other domains on the design of correct metrics for our trust model. A set of not carefully selected metrics may results on unsuccessful trust computation and consequently to an unstable system.

9. If one or multiple autonomic functionality levels are added; how is the trust model and measures going to be affected e.g. how trust is going to be measured between the network providers and the service providers?
   Importance: Trust in-between operators or service providers is fundamental for business. The introduction of autonomic functionality may affect it since part of the management functions will be in the network.

10. What types of strategies do you envisage for trust in autonomics? How can metrics be exploited or mapped to certain levels of trust? Are different weights considered for various metrics (potentially diverse in various situations)? How can the translation of business goals to network requirements be evaluated in the scope of trust mechanisms?
    Importance: For being able to assess the level of "trustworthiness" of an autonomic node/entity it is crucial to have a process of mapping trust metrics to levels/values of trust. Furthermore, depending on the autonomic function/entity addressed certain metrics may

be more important than others, thus weighting of metrics/parameters for deriving the level of "trustworthiness" may be required.

11. How to ensure global optimum whilst catering for local optimum, as per autonomics (self) feature, each activated through the same optimization parameters in order to ensure stable behaviour of the system? Which coordination process and associated design principles should be provided in order to make the system beneficiary comfortable with optimization phase alongside large scale deployment?
    Importance: Current operator networks evolve towards a more “flat” architecture and the introduction of the autonomics paradigm raising at the same time major issues and challenges to be solved. These issues are pertaining to "trust & confidence" of these autonomics or self -features, as well as to the coordination of various autonomics (Self) features running simultaneously from network stability perspective.

4.4 Trust Mechanisms

12. What kind of trust mechanisms are or could be embedded in the autonomic elements in order to secure the cooperation between the different elements of the system?
    Importance: A framework for an autonomic network, able to dynamically manage itself without human intervention, needs to address this question to convince the operators it can be safely deployed.

13. What kind of trust mechanisms is required during the federation formation of a system and what level of trust are required to support this formation?
    Importance: In the current Internet there is a trend for services to be both provided and consumed by loosely coupled networks of consumers, providers and combined consumer/providers, forming federations of networks. Trust mechanisms are valuable during forming, joining and maintaining a federation.

14. What kind of trust mechanisms is necessary for building trust between different system providers? Can the same mechanisms as the one for the cooperation between the different elements of the system apply to a multi-system environment?
    Importance: Trust should be addressed as a 3-faceted issue: trust between autonomic nodes (since they need to safely cooperate between them), trust of the operator on the autonomic network (can be built by extracting information from the feedback loops to demonstrate the correct functioning of the system), and trust between different networks (that should cooperate for the management of services deployed on networks of different providers).