Deliverable D4.16

Assessment Results of Trust in Autonomics Release 2

Grant Agreement	257513	
Date of Annex I	8 October 2013	
Dissemination Level	Public	
Nature	Report	
Work package	WP4 – Deployment and Impact	
Due delivery date	1 December 2013	
Actual delivery date	11 November 2013	
Lead beneficiary	Fraunhofer	Mikhail Smirnov mikhail.smirnov@fokus.fraunhofer.de

SEVENTH FRAMEWORK PROGRAMME

FUTURE NETWORKS

Authors	ALBLF –	L. Ciavaglia, S. Ghamri-Doudane, Benoit Ronot
	Fraunhofer –	M. Smirnov, M. Emmelmann, M. Corici
	UPRC –	P. Demestichas, V. Stavroulaki, A. Bantouna
	Orange-FT -	B. Sayrac
	ALUD –	M. Gruber, I. Karla, L. Ewe
	NKUA –	V. Kosmatos, E. Patouni
	iMinds -	S. Delaere, S. Spek, V. Gonçalves

Executive summary

In the 1st release of this report we have formulated the key requirements for operator trust in autonomics (TiA) and described the approaches to meet them; those are summarised in the current release and enhanced further.

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 autonomic features of their networks (e.g. LTE SON) 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.

The five outlined approaches were integrated in the emerging TiA methodology, which is exemplified by the creation of raw certificates for a couple of sample Network Empowerment Mechanisms also developed in the project. The methodology requires further work; we report here some fundamental issues and consider their discovery and identification as one of the main achievements of the TiA task in the project. Yet we were able to demonstrate that our approaches are viable when the problem of trust is defined with sufficient precision. First, we limit the understanding of trust only to Operator Trust in Automation, where automation is applied to certain management functions traditionally performed/controlled by humans. Second, we demonstrated that our five approaches to trust facets make trust being operationally defined[1] in Deming's sense This definition precision did allow us to successfully create a relevant Work Item within ETSI, where further work on certification proceeds.

The second major enhancement as compared to the 1st release is the inclusion of business aspect of certification. We report here our findings on possible certification scope, discuss the assumptions and requirements, and describe from a business impact point of view a number of certification scenarios such as offline and online certification, self-certification and 3rd party certification.

[1] http://en.wikipedia.org/wiki/Operational_definition#Business

Table of Content

Foreword **5**
Introduction **6**
State of the Art and Beyond **8**
1. Learning from Aviation 9
2. State of the Art in Certification 9
   i. The importance of certification 10
   ii. Certification Process 10
   iii. Models for Autonomic Assessment and Software Quality 11
Design-for-Trust Methodology **13**
3. Unified trust for deployment domains 13
4. Measurable Trust 14
5. Domain-specific Trust 15
6. Model-driven Trust 15
7. End-to-End Trust 17
8. Evaluation of Trust of policy methodology 19
9. Method for certification of autonomic systems 20
Raw Certificates **22**
10. The process 22
11. The outcome and discussion 22
Business impact of certification **24**
12. Software component certification 24
13. Offline and online certification 24
14. Self-certification and third-party certification 26
15. Analysis: four scenarios 26
Conclusion **29**
References **30**
Abbreviations **41**
Annex A: NEM SOUP Raw Certificate **30**
Annex B: NEM Handover Optimisation **34**

1. Foreword

In the domain-specific deployments of UMF (Unified Management Framework) the interests of several groups of stakeholders will be interwoven. This 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. Therefore the project addressed these two requirements together and demonstrated that design for manageability is bringing certain benefits for the design for trustworthiness and vice versa.

Such an attempt was reported in the 1st release of this deliverable. Considered successful this experiment performed in WP4 occupies an important and central place within the project: WP2 received benefits 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 received benefits 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 had an opportunity to enrich its UC4, as well as bring similar considerations to other project use cases. Last but not least, the WP5 had the opportunity to include the reported work into its trend-setting activities, because the main stakeholders addressed by the work are Mobile Network Operators (MNOs).

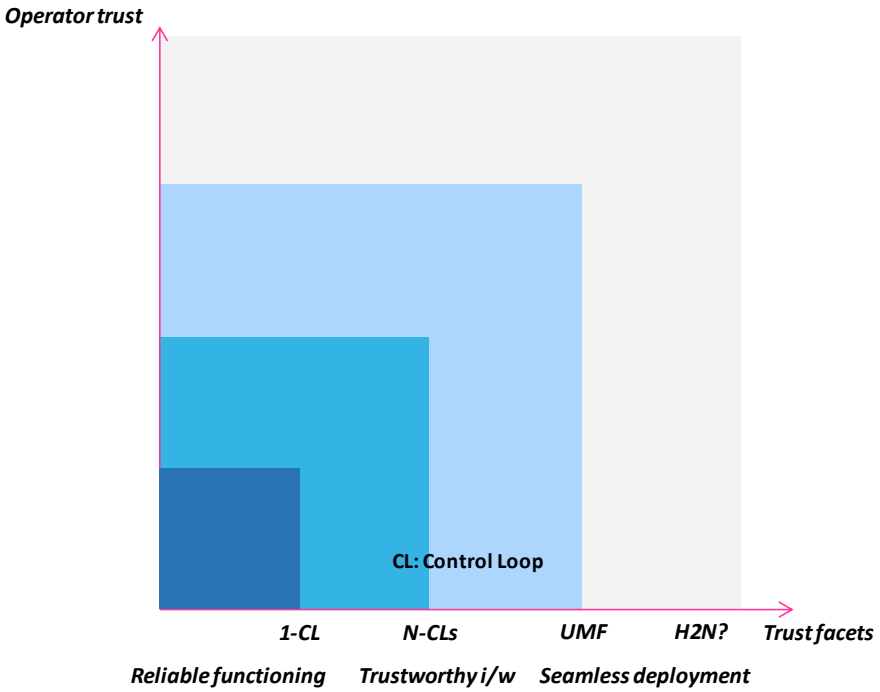


Figure 1. Facets of Operator trust in SON
(Legend: CL- control loop, H2N – human to network)

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; and, third to facilitate **seamless deployment** of any number of CLs enabled by the UMF. The Human-to-Network (H2N) aspects comprise the next step on this roadmap, hence we deliberately included the business implication section in the current release.

2. Introduction

Our goal was not to identify exploits of future technology but to design technology add-ons that shall prevent the exploits. Our agenda was to design add-ons that shall be mutually beneficial to both - management of future technology and increased operator trust in that future technology. The outcome had to contribute to the certification process of future equipment that shall cover the entire manufacturing chain from design technology to the deployment process.

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 technology 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. Of course, ideally this learning should happen during the testing of design choices, however it can also impact the scope and the type of certification mostly applicable to the said equipment,

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 five aspects of trust described in the 1st release of this document are as follows.

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 NEM-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).

These aspects were integrated into an emerging methodology that enables future certification of UMF-compliant autonomic NEMs. We discuss this methodology in the next section together with our major finding on that road: the autonomic NEMs applicable for a certification must be designed differently from conventional orientation of function-only design. The non-conventional requirements pose the main challenge nowadays.

The second major enhancement as compared to the 1st release is the inclusion of business aspect of certification. We report here our findings on possible certification scope, discuss the assumptions and requirements, and describe from a business impact point of view a number of certification scenarios such as offline and online certification, self-certification and 3rd party certification.

3. State of the Art and Beyond

Relevant papers are referenced in the main part of the report as appropriate. In this section we 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.

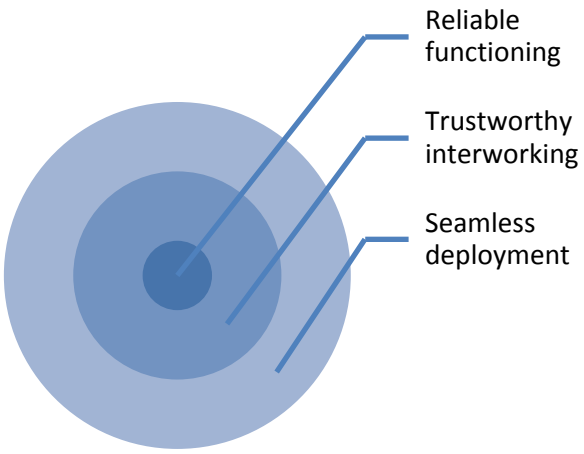


Figure 2. Hierarchy of Trust Facets

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 was addressed in the design and specification of the Unified Management Framework (UMF), in particular with the development of the Coordination block and its mechanisms.

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.

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.

3.1 Learning from Aviation

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] highlights the remaining challenges and obstacles but also the close opportunity that unmanned or pilotless airplane becomes 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 release 1 we have studied a very specific example of operator trust in automatic control borrowed from the aviation, here we report only the key findings of that study.

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 Benwee 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.

3.2 State of the Art in Certification

The autonomic research community has achieved significant progress, 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.

3.2.1. 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. 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.

3.2.2. Certification Process

Certification process must meet some conditions. However only very few researchers have identified trustworthiness as a major AC challenge and yet fewer [28], [29], [30] have actually suggested or proposed techniques.

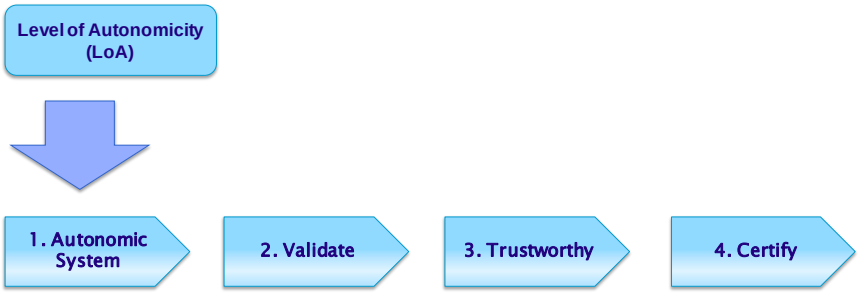


Figure 5. Certification process

Figure 5 represents phases of certification process.

Standardization required	1	**Autonomic System** (*defining autonomous*)	Autonomic characteristics, decision-making algorithms and policies are defined. (Architecture + self-* properties)
	2	**Classified AS – according to LoA** (*defining autonomicity*)	Autonomicity measuring metrics are specified.
	3	**Tested AS** (*Appropriate validation for identified LoA*)	Validation is defined according to system's LoA. Validation is also implemented in a layered structure
	4	**Trustworthy AS**	Trustworthy AS is dependable AS. It is not reasonable to consider other properties such as evolvability without first achieving trustworthiness. Validation is prerequisite for trustworthiness
	5	**Certifiable AS**	Certifiable AS is at the height of AC goal. It is shown at this point, beyond every reasonable doubt, that a system can be trusted

a	**Generic** (*within LoA*)	Validation approach should be reusable across LoA. Procedure/process for approach to adaptation is clearly specified
b	**Design-time**	Policies that handle design-time validation are defined
c	**Run-time**	Run-time validation policies and algorithms are defined
d	**Integrated**	Algorithms for components interactions and spontaneous (automatic) test activity call are defined

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. 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.

3.2.3. Models for Autonomic Assessment and Software Quality

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).

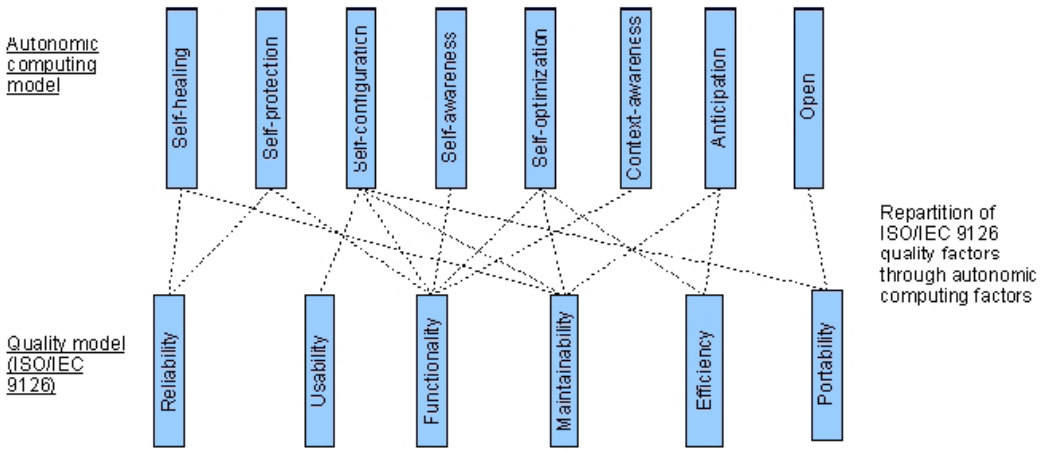


Figure 7. Organization of autonomic computing characteristics based on ISO/IEC 9126 standard quality factors

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.

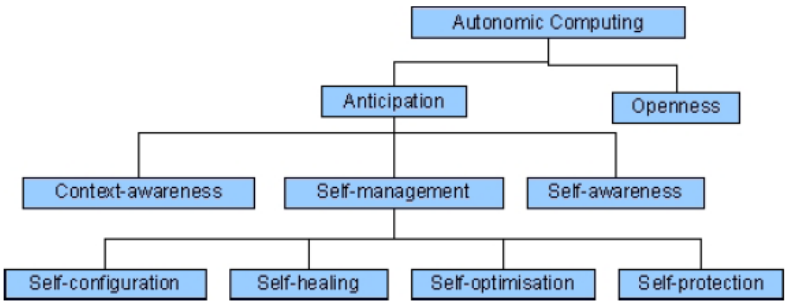


Figure 8.: Hierarchy between autonomic computing characteristics

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.

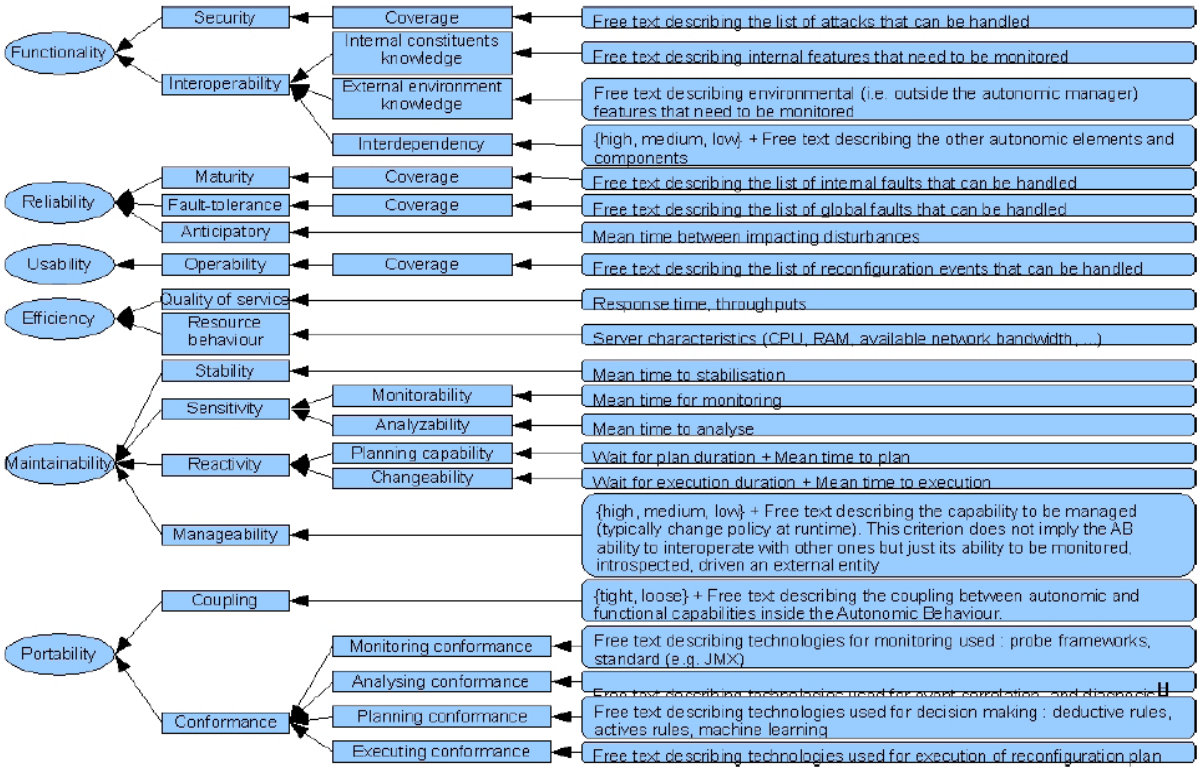


Figure 9. Quality model for autonomic computing assessment

Learning from these approaches and tailoring them to the UniverSelf project context and to the different deployment domains of the UMF has enabled us to integrate all TiA components into one emerging methodology that we report next.

4. Design-for-Trust Methodology

From a very rigorous viewpoint perhaps our methodology is not complete , however we see the skeleton of it, which can be explained by few things. i/. Trust has many facets (hence, not all facets are equally important for all NEMs under trust evaluation); ii/. Trust is always explorative (hence, it's rather a process than an object); iii/. trust in any human-made technology is always constrained by the competence envelope of the technology (hence, it must be understood prior to trusting), iv/. algorithmical trust is always biased (hence, it's important to understand the designer biases); v/. since dis-trust is absolute (while any trust is relative to certain , let's call them trust conditions) it is important during trusting to verify whether all conditions are still holding. We proceed in this section following the logic of fife facets of trust.

4.1 Unified trust for deployment domains

"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.

The above is essentially generic. How do we apply these findings to a specific deployment domain? We conjecture that this mapping is straightforward and repeat, omitting details, Figure 10 with such mapping exercise performed during the project for the LTE SON domain and reported in much detail in release 1 of this document.

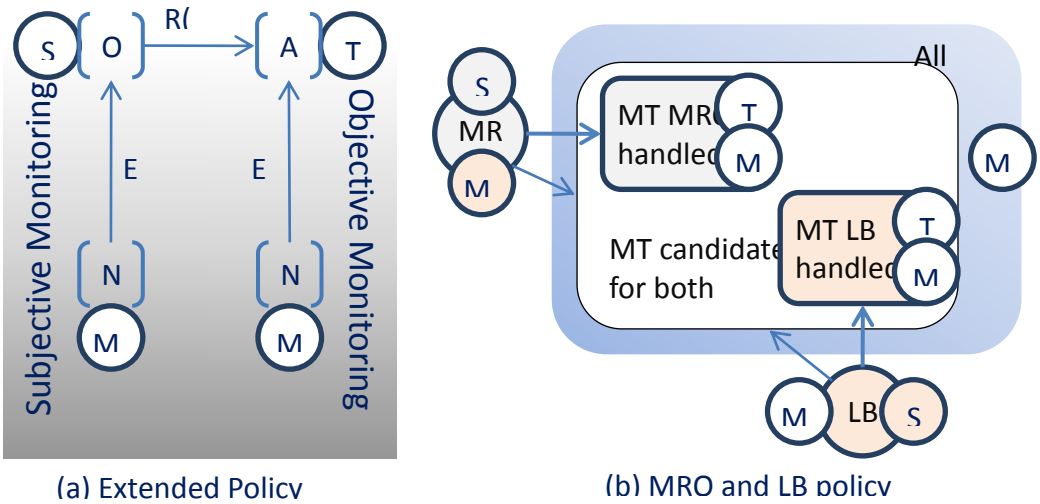


Figure 10. Extended policy domain (a) and its mapping to SON LTE (b)
Legend: S – Subject; T – target object; N – notification obligation; A- action; M – monitor; E – event: R(m) – request for a method; MT – mobile terminal; LB – Load balancing; MRO – Mobility robustness optimisation

This mapping was made easy by using the policy domain extension model outlined in [11]. 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.

Thus, we conclude our methodology starts with the unified expression of both operator policies and NEM internal policies in the deployment domain. The next step is to define how trust can be measured.

4.2 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.

Now, when a designer has a uniform expression of all involved policies and has metrics for measuring trust – both being uniform and generic, it is time to concentrate on domain-specific aspects of trust.

4.3 Domain-specific Trust

Our sample domain - 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.

As usual, when we face the need to address heterogeneity within the design we create models that help to map domain specifities into a scale that is instrumental to make decisions, hence the next section elaborates on a model-driven trust.

4.4 Model-driven Trust

The method described in the release 1 of this document was a generic one (and applied to real SON LTE cases in the next section) and targets at enforcing a control loop with the knowledge if its decisions are trustworthy enough. 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.

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 detailed in the 1st release. 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) r (corresponding to the payoff of Q-Learning technique) for the certain state s and action (decision)/ selected control loop a 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) r for the certain state s 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 $r(s(t), a(t))$ represents the payoff for a particular system state s and action a at instance t.

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 $Q(s(t), a(t))$ corresponds to the OTI when the control loop is triggered by a situation (system state) $s(t)$ and reaches decision $a(t)$ and symbol $\langle \ \rangle_{s,r}$ refers to the average value. The discount factor $0 < \gamma < 1$ stands for the weight of the payoff and is closely related to the time that has elapsed from the payoff, i.e. the larger γ designates that the more distant payoffs are more important. Moreover, parameter ε denotes the learning rate of the system.

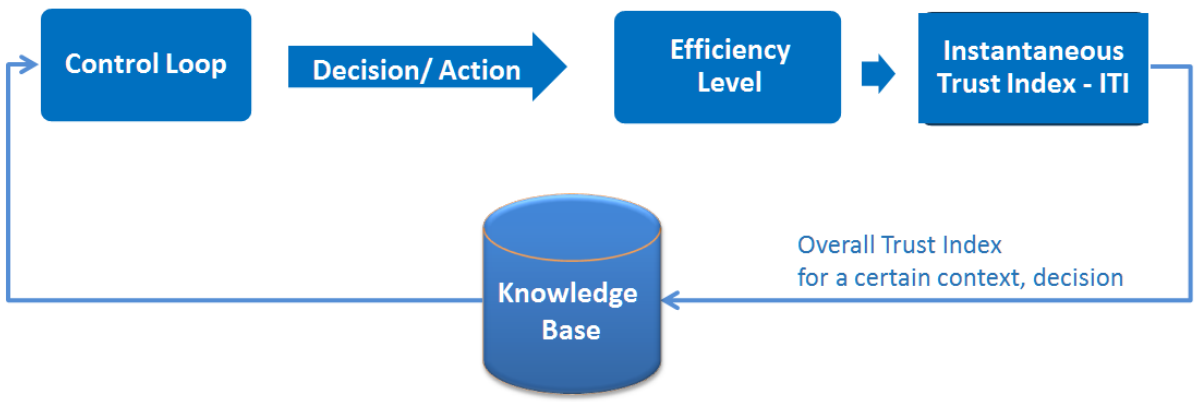


Figure 11. High-level view of control loop trustworthiness assessment process and update of corresponding knowledge base

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 (Figure 11). 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. Next, how to apply the above to multiple concurrent control loops?

4.5 End-to-End Trust

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.

Our concept mapping experience was reported in the 1st release and has revealed 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.

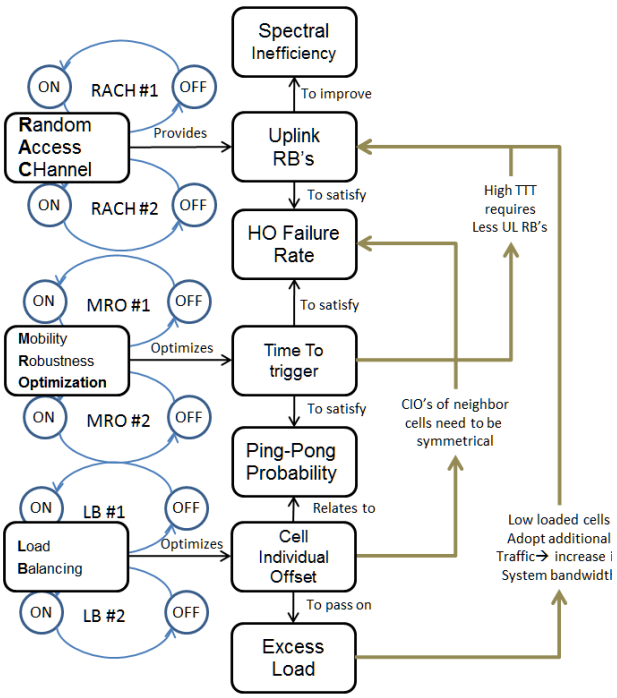


Figure 13. Three interacting SON control loops
Legend: RB – Resource Block, CIO – Cell Initial offset, TTT – Time to trigger

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.

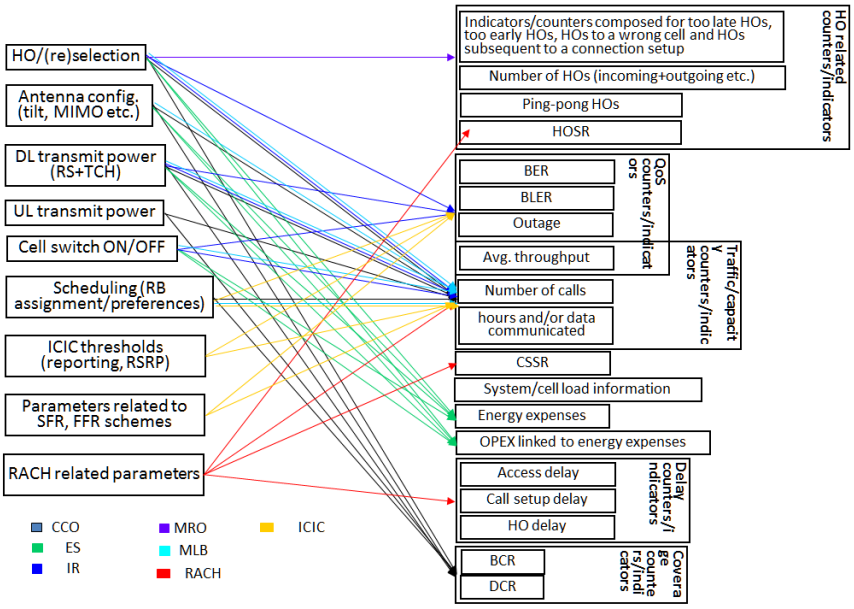


Figure 14. Seven interacting SON control loops
Legend: CCO- Coverage and Capacity Optimisation; ES – Energy Savings; IR - Interference reduction; MRO – Mobility Robustness Optimisation, MLB – Mobility Load balancing; RACH- Random Access Channel; ICIC – Inter-cell Interference Coordination

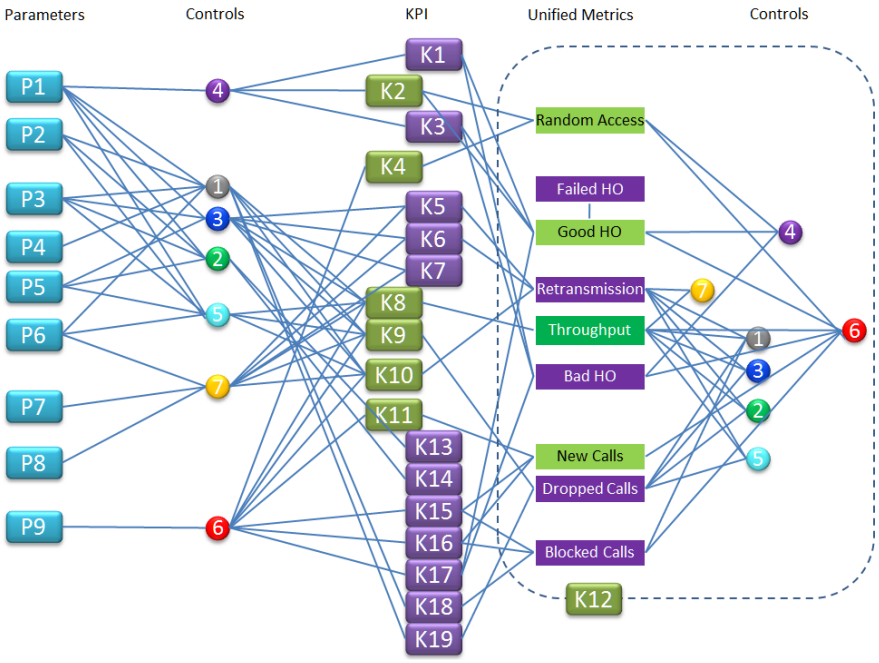


Figure 15. Towards self-orchestration model of SON
Legend: (see Table 1), Unified metrics*: A* – Random Access, *Gho* – Good Handover, *R*- Retransmission, *T* – Throughput, *Bho* – Bad Handover, *Nn* – New Calls,*Nd* – Dropped calls, *Nb* – Blocked calls

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.

Finally, as reported in the deliverable D3.9 the achieved self-orchestration greatly improves stability, as Figure 16 is demonstrating the joint utility of seven concurrent LTE SON control loops has almost five times smaller standard deviation on mean, with the mean joint utility value being almost the same.

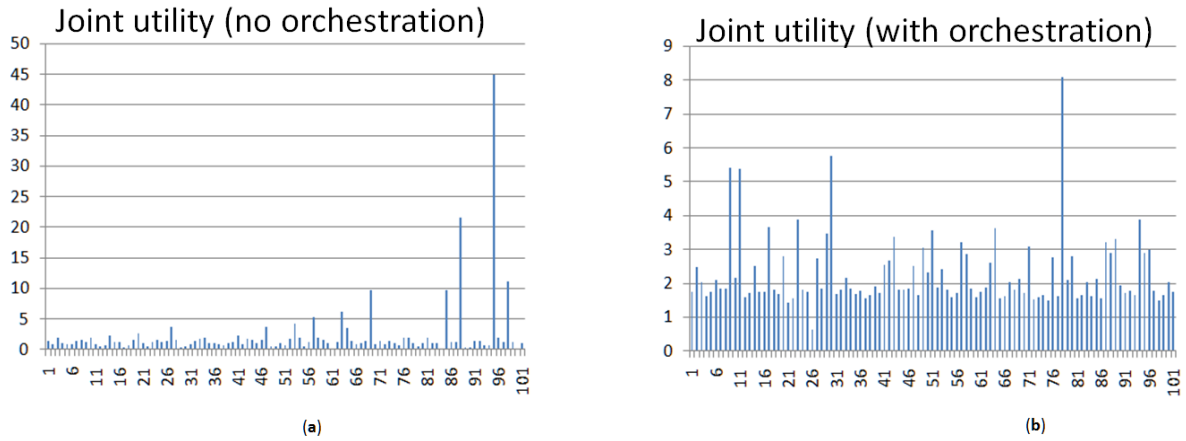


Figure 16. Standard deviation of mean joint utility (a) without self-orchestration (b) with self orchestration

4.6 Evaluation of Trust of policy methodology

In the release 1 of this document we have reported set of simulations has been realised in order to evaluate the proposed methodology for Trust of policy estimation, we summarise here the main findings. The simulation scenarios are executed in the OPNET simulation tool [38].

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.

The end-to-end delay values of streaming service experienced by the Gold user in all scenarios are illustrated in Figure , left while end-to-end delay values experienced by the Bronze user are illustrated to the right. 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.

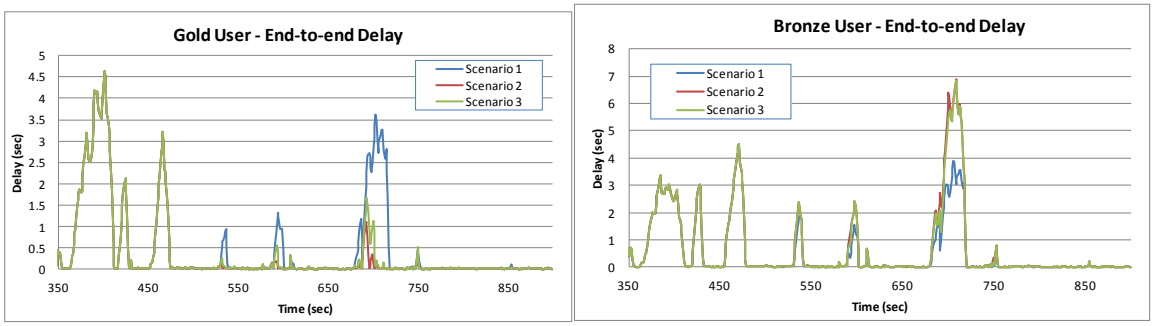


Figure 18. End-to-end delay of (left) Gold user, and (right) Bronze user

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.

4.7 Method for certification of autonomic systems

The starting point of certification is to classify a NEM 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.

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 LoA 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.

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.

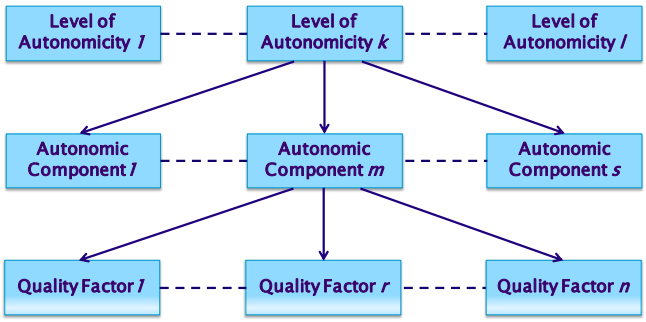


Figure 21. Hierarchical autonomic evaluation model

In the 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.

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:

$$T = \sum_{i=1}^{S} S_i \, S_{il}$$

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:

$$S_m = \sum_{i=1}^{n} N_i \, N_{im}$$

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.

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.

However, before the rigorous models could be applied within the certification we have to make certain reality check and make sure that all designers and testers and evaluators of NEM have comparable understanding of key features and capability of a system (NEM) under certification. We did find that this is often not the case and report our findings in the next section.

5. Raw Certificates

5.1 The process

The NEM Evaluation (aiming at producing raw certificates for sample NEMs) structure of this deliverable was designed to contain two parts: Part 1 with results of individual assessment of a representative set of NEMs by providing the evaluation of their reliable operation, of their trustworthy interworking and of their seamless deployment. This evaluation in turn could have two parts. Part 1.a with numerical evaluation (by simulation, by prototyping, by formal analysis, etc.) that was done elsewhere in the project and which evaluates NEM's performance envelope. Part 1.b with expert evaluation of trust facets per NEM by dedicated experts based on structured questionnaire and on the procedure of consensus (using subjective logic) building, which evaluates NEM's competence envelope. Part 2 was to use a common methodology. Because 1.b attempts to be NEM agnostic it opens an opportunity to facilitate the common generalised assessment methodology of UMF conformant NEMs, which is at the foundation of certification process.

This common expert evaluation methodology was given by the nine generic questions and associated assessments. Each NEM is to be considered from the three viewpoints:

1. reliable operation of NEM's algorithm; (algorithm per se)
2. trustworthy interworking of this NEM with other NEMs, and (algorithm in a group)
3. seamless deployment of the NEM. (algorithm's life cycle management)

Consequently, operator trust in a NEM will have three facets.

At each viewpoint a NEM is to be evaluated at three sub-levels, which have the same structure for all viewpoints:

x.1 basic NEM operation in noisy environment;

x.2 NEM operation with context sensitivity,

x.3 NEM operation with context sensitivity and cognition.

Consequently, operator trust in a NEM for each facet will have three values.

Each x-th evaluation is formulated as a value Ex and its confidence level CEx:

Both values are within ranges - Ex is within the [Tx,100] % and CEx within the [Cx,100] %, where 100>Tx,Cx>0 are reasonable thresholds for particular evaluation. It seems reasonable to have Cx>50%.

5.2 The outcome and discussion

From discussions with some partners it appeared that the most hard part of trust evaluation is in the right definition of noise. We define noise here in the control-theoretic way as the metric that measures the distance between current input of a control loop and the ideal input. Of course the terms *distance*, *input*, and *ideal input* can be correctly defined only when the essence of the control loop is clear (hence, the first version of a trust certificate is obtained by expert evaluation), however something can be explained even in a NEM-independent way. First, the noise defined as *distance = ideal – current* can be both 'positive' (lack of information) and 'negative' (redundant information), with the former being due e.g. insufficient measurements or lack of response, and the latter being due to measurement of wrong KPI's or non-pertinent responses. Second, the evaluation of thus defined noise requires knowing a *model* that is being used by the decision process inside a control loop. Note that according to the three viewpoints each NEM exists in the three control loops concurrently - algorithm, group, deployment (lifecycle) – and consequently will have three different models. The project has already demonstrated in Y2 review that deployment model can be one and the same for all NEMs; the hope behind this evaluation is that group model can be also one and the same for all heterogeneous NEMs, while the – unique for each NEM – model used in a NEM's algorithm can be fully (100%) tested.

Yet another difficulty was coming from context. Since the advances of context-based research It became common to consider that context always helps. We were asking : Is it really always true? Is it true for your NEM? Consider the case when context acquisition is also subject to noise (not even mentioning that context acquisition and processing cost time and effort), then suddenly context becomes a subject of evaluation (Is it the right context? Is it timely, accurate, etc.?). That means that you have to take into account yet another KPI,

related to context, and the resulting impact on the original KPI set might be negative. As illustration of this point we refer to the SOUP raw certificate (Annex A of this deliverable).

Our two guesses on what might be difficult in the expert evaluation turned out to strongly correct. As the demonstration of that we placed the two raw certificates as Annex A and Annex B of this document : their comparison and expert's rationale included in the certificates are very much illuminating.

As a positive outcome of this otherwise discouraging exercise we derive the following comment on method evaluation: it seems that we NEM designers and engineers are at certain turning point where they must be prepared to depart from traditional in-lab evaluation of methods and face the real-life challenges, which their methods will face when embedded in the network equipment. For example, when in lab we are interested in e.g. convergence time of an algorithm, while for the same algorithm (method) - if it passes convergence threshold (but perhaps more algorithms would pass) - when embedded we shall be mostly interested in its **ability to predict,** which is of course rather different from convergence time. Necessarily, all algorithms that will be useful must be equipped with certain machine learning part, when tested in lab these machine learning parts can produce acceptable results due to known over-fitting issue. Again, to combat over fitting we need to evaluate the ability to predict. The above appears to be in-line with the general technological trend, which is expressed by Roger Barga of Microsoft as "*Looking up the stack, faster technologies such as stream processing engines, in-database and in-memory analytics are progressively being coupled with "agile" analytical methods and machine learning techniques to produce insights at a much faster rate. Analytical models are being embedded into operational and decision processes, dramatically increasing their speed and impact*[2].". It is essential, says R. Barga earlier that the above to be applied to situations "***with real-time changing conditions such as fraud detection***".

The above challenges discovered in purely technical evaluations of purely technical components appear to be strongly related to major values and assets of networking business, therefore we are prepared to consider the business impact of certification next.

[2] keynote at DEBS2013 (http://www.orgs.ttu.edu/debs2013/index.php?goto=keynotes)

6. Business impact of certification

This chapter deals with certification issues from a business perspective. While literature on the specific business consequences of different software certification methods is scarce, it is an important issue since choices made from a technical perspective might have significant impact on business issues. For instance, certain business model parameters might have been decided by past technical choices, or the type of certification requires changes in the value network, by introducing new actors or aggregating roles with existing actors.

In subsection 6.1, the theoretical aspects of software component certification are being discussed, since they might provide parallels with the UniverSelf NEM model. In subsection 6.2 and 6.3, two scenario parameters are considered: respectively on **online and offline certification** and **self vs. third-party certification**. Finally, in subsection 6.4 an analysis is made by four scenarios based on these options, while subsection 5 provides conclusions.

6.1 Software component certification

Several types of software certification exist, but with using the UMF framework and autonomous mechanism, the UniverSelf model is not very generic. The autonomous NEMs in certain ways resemble software components: they are not designed for re-use, but they are designed to be developed by different parties and to work in a collaborative, or at least interoperable, fashion; similar to the definition of software components provided by Szyperski: “a unit of composition with contractually specified interfaces and explicit context dependencies only. A software component can be independently deployed and is subject to third-party composition” [1] via [2]. It is therefore natural to consider software-component certification, be it with the consideration that software components are designed for re-use rather than instantiation and execution.

In software components, certification is an issue. In a 2000 SEI report [3], a survey states that lack of independent certification is considered the third biggest problem, after a lack of available components and a lack of stable standards. These results are not recent, but the UniverSelf NEM ‘marketplace’, especially in its infant state, will most likely face similar challenges as software components.

An overview of the history of software component certification is provided by Alvaro *et al.* [2]. Their timeline is divided into two ‘ages’: an early age of mathematical and test-based models from 1993 to 2001, and a second age of quality requirement prediction after 2001. It also describes two failed initiatives for third-party certification in this field. The Trusted Computer Security Evaluation Criteria (TCSEC) by the National Institute of Standards and Technology (NIAS) and the National Security Agency (NSA) “at least partially because it had defined no means of composing criteria (features) across classes of components and the support for compositional reasoning, but only for a restricted set of behavioural assembly properties” [2]. An IEEE initiative from 1997 was suspended the same year since the committee was unable to formulate a strong candidate for a software component quality standard. [4] These examples show that providing a third-party certification system is far from trivial, even when it is backed by respected organizations.

6.2 Offline and online certification

In this subsection and the next, two scenario choices that are relevant for the UniverSelf model and discuss their business impacts will be discussed. The first focuses on the difference between offline and online certification.

Offline certification is a traditional model, in which a NEM would be tested in isolation (only with the UMF) in a standard configuration. This is to ensure that the NEM behaves normal in a standard situation and conforms to set standards. However, when employed on an operator network, many alternative configurations and exceptional situations might occur that cannot be simulated in an offline setting. Therefore, a model that considers both offline and online certification should be considered.

The distinction between offline and online certification was made as well in the E2R II project where it was applied to the market introduction of radio configuration software, although there the terms ‘pre-market’ and ‘post-market entry’ were used. [5] While this does not necessarily apply to offline and online certification, in this case it does: pre-market certification is based on self-declaration based on conformity to harmonized standards, while the post-market entry certification was based on post monitoring of conformance recording

and analysis of configuration history. The second takes into account that "equipment may change their functionality, as well as their intended use, depending on the configuration implemented." [5] The responsibility chain specifies that upon market introduction, a description of its functionality and its hardware/software combination, a declaration of intended use and a statement or declaration to a harmonized standard has to be made, as well as responsibility needs to be taken by the declarer. This responsibility is then only for a certain tested configuration.

When translating this to UMF and NEMs, one could design a protocol in which, apart from the offline certification, a NEM developer assumes responsibility for several online configurations of its NEM. These can be hardware configurations, as well as combinations of the NEM with other mechanisms. Obviously, the harmonious collaboration between a NEM and the UMF is always assumed and falls directly under the responsibility of the NEM developer.

A new entity, comparable to E2R II's *Reconfiguration Support Service Provider*, can be created to monitor existing online configurations of different NEMs. They can provide to their partners an overview of online configurations together with their performance and evaluation. Network operators looking to acquire or license a certain NEM could consult this overview for similar configurations, thus achieving trust with a certain probability for their own configuration.

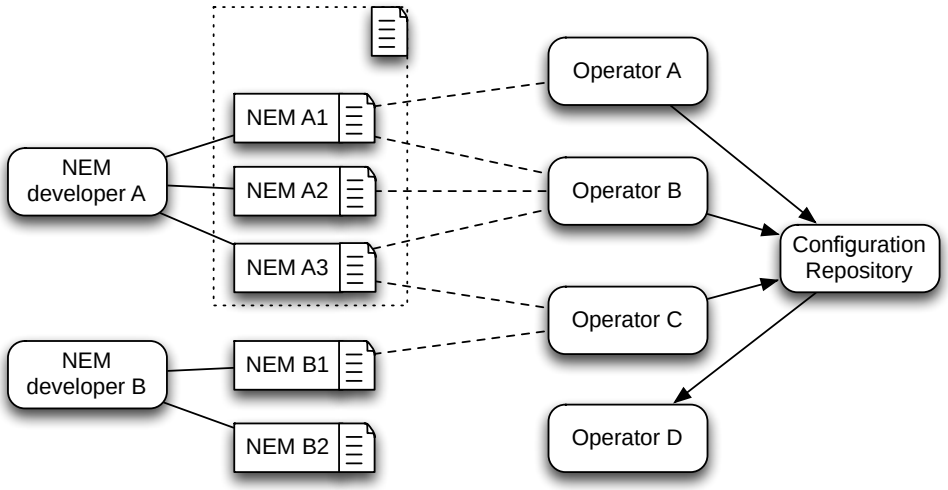


Figure 1: Suggestion for offline and online certification system

Figure 1 provides a suggestion for a NEM certification system that combines both offline and online certification. The model contains two NEM developers, A and B, who have developed three and two NEMs, respectively. These NEMs are named A1 to B2. Each of these NEMs are offline certified, which means that their functioning in a stand-alone situation (one NEM plus UMF) in a standard configuration is guaranteed. In addition, NEM developer A also provides a separate certification for all its NEMs together (NEM A1, A2 and A3 plus UMF) in a standard configuration. Operator A only makes use of NEM A1, and thus can rely on the offline certification. Operator B employs all NEMs of developer A, having received confidence from the combined certificate of developer A's NEMs. Finally, operator C has decided to employ NEMs from both developers. All these configurations are being monitored and stored by the Configuration Repository, which, as suggested by Voas [6], contains information on the software's behaviour as well as on the ways it is being used in the field. When operator D wants to employ certain NEMs, it can access the repository and study performance certificates for different configurations, deducting a trust level for its own desired configuration.

From an acquisition perspective this provides new opportunities that need to be taken into account. Operators might want to perform online testing, in a controlled environment and on their operational networks, before acquiring a license to use a certain NEM. Developers could anticipate this by offering leasing schemes that will lead to a license acquisition as soon as a satisfactory amount of trust has been gathered from online testing.

6.3 Self-certification and third-party certification

A second scenario deals with self-certification and third-party certification. As in the previous case, these options are not exclusive and can be used in a hybrid form.

Voas [6], [7] discusses the consequences of independent third-party certification as compared to self-certification. There is a strong case for third-party certifications from both the user community as the software publishers. Users benefit from third-party certification since it provides unbiased assessments. The reason why publishers prefer third-party certification is to shift responsibility and/or liability. However, benefits are not the same for all publishers. In a situation with self-certification, customers most likely prefer established players since these have build up trust over the years. A third-party certification provides a level playing field, in which new players can easily build up trust, backed by a third-party certification.

The main reason, however, why third-party certification is not as popular as it should be from the above reasons, is that it's hard to find a party willing to take up the role of the certifier. These independent examiners are considered experts, which in the legal sense, burdens them with a great liability. Voas compares this with the case of the surgeon: one can perform thousands of brilliant operations, but after one negligent mistake, also non-medical consequences can be severe, leading to bankruptcy of the surgeon. Regardless of the low failure rate achieved in the past, one mistake or miscalculation can cost both surgeon and software certifier dearly.

So, as for the third-party certifier the risks are high and the benefits unclear, it might prove very difficult finding candidates to take this role upon themselves. If publishers or developers want to push for a third-party certification system, one solution would be to jointly invest in an organization that takes care of this task. However, then it almost becomes a covert self-certification system again, where publishers might exercise control within the certification body, and the organization might favour software products published by their members rather than those of other publishers. If, to compensate for this, all NEM publishers would be obliged to participate in the certification body, this would damage the open character that the UniverSelf project hoped to achieve for its NEM ecosystem.

6.4 Analysis: four scenarios

In the previous two subsections, two scenario parameters for a certification system have been discussed. Using those parameters, one arrives at four possible scenarios, as presented in Table 1.

Table 1: Certification scenarios.

	Offline certification	Offline and online certification
Self-certification	Basic	Catalogue
Third-party certification	External lab	Observatory

In the *basic* configuration, NEM developers or publishers themselves will do all certification, and all will happen offline. This is the most basic configuration, since it will only provide a certification that guarantees the NEM (plus UMF) behaves well in isolation in a simulated environment. This might be the easiest option for the NEM developers and publishers, but at the same time, this provides the least valuable situation for the customers. This is not the preferred way when one wants to stimulate the use of the UMF framework and its NEMs. Also, a strict formalisation of the certification process will be required. This scenario is depicted in Figure 2.

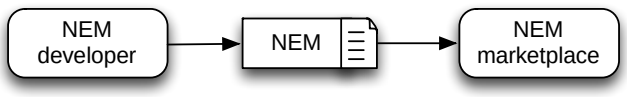


Figure 2: The basic scenario.

In the scenario titled *External lab*, NEMs will again only be tested offline, but this time it is performed by a third-party certification institute. This means that before publishing, NEM developers have to submit their product to the institute and wait for third-party approval. If the basic scenario is similar to Google's Play Store,

this scenario resembles the iTunes App Store. It will create overhead and a longer time-to-market, while at the same time the benefits over the basic scenario will be minimal since it only covers offline testing. So a neutral party will perform the testing, but it will be limited to a simulated standard environment. Moreover, this scenario comes with the disadvantages of the third-party certification, namely that there is hardly a business case for this role, as discussed in section 6.3. In that sense, this does not have strong advantages over the basic scenario.

Figure 3: The external lab scenario.

The *catalogue* scenario implements the scenario options of self-certification with online testing. This means that NEM developers and publishers first self-certify their software as in the basic scenario, but also receive feedback from real instantiations on operator networks. They can use this feedback not only for their internal NEM maintenance, but also for creating a (best-practice) catalogue for their NEM with the real-life configurations in which it is employed. The operators would be references to these 'testimonials', preventing fraud from the developer's side and thus creating more confidence in this self-certification system.

Moreover, in this scenario, developers and publishers of NEMs could generate extra business by developing consultancy services aiding the operators to smoothly implement their NEMs. This would allow them to gain even more insights into the building of this catalogue.

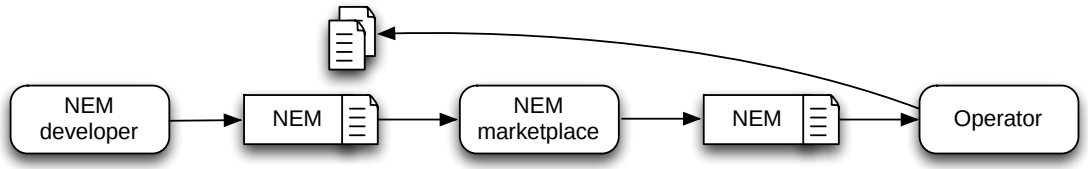


Figure 4: The catalogue scenario.

The final scenario, the *observatory*, displayed in Figure 5, is a combination of the external lab and the catalogue scenario. A third-party certifies a NEM with an offline certificate before it enters the marketplace. After that, the third-party collects usage information in order to compose a catalogue of configurations and usage situations. This catalogue will be accessible for other operators seeking to license a specific NEM.

The advantage for the operator is that this scenario provides them with the most information, while it is all compiled by a third-party and thus considered objective. For the third-party certifier, this has the advantage that their task is more complex than in the external lab scenario, and therefore it could provide more options for revenue generation. For instance, the offline NEM certificates could be a basic service, whereas access to the catalogue could be sold as a premium. This might make it more interesting for a business actor to take upon them the role of the third-party certifier.

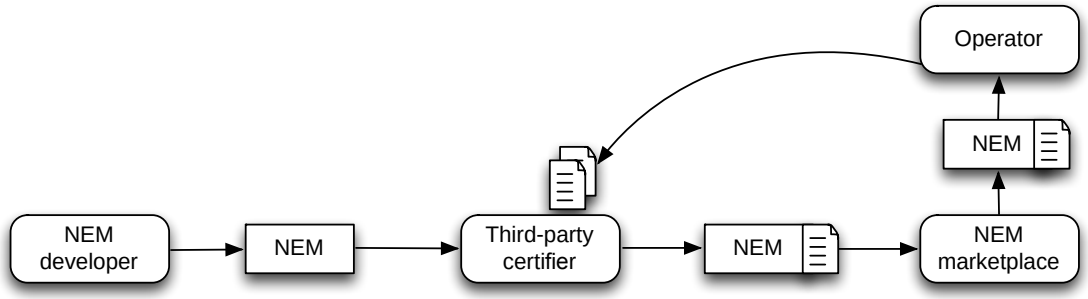


Figure 5: The observatory scenario.

Apart from the external lab, the scenarios in the order they have been discussed provide increasing advantages. Therefore, it might be possible to not consider them as a choice that has to be made beforehand, but rather three phases of growth: from the basic scenario to the catalogue to the observatory scenario. In this

way, both complexity and trust will increase, while at the same time the NEM market can prove itself while potential third-party certifier can consider their options before joining in the last phase. The certifier can build a new catalogue based on the ones already being provided by the NEM developers in the previous phase.

7. Conclusion

The reported work is limited to a single project only, and within this project to a couple of domains, 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 bring to the design of a certification process, which we outlined from the technical and from the business perspective.

The business aspects of certification have been discussed: first by a literature review about component certification, second by an analysis of two scenario parameters leading to four scenarios. The analysis deemed three out of the four scenarios viable options: the basic scenario, the catalogue scenario, and the observatory scenario. While these scenarios can be considered as exclusive options, they can also be seen as a three-phase roadmap to a certification system of increasing complexity and trust, with the observatory scenario as the final state. This scenario contains a third-party certification body that subjects NEMs to offline certification before they enter the marketplace and constructs a catalogue of operational NEM instantiations and their configurations and performances.

Annex A: NEM SOUP Raw Certificate

NEM Title: Self-Orchestration via Utility Policy

NEM #: 8

NEM Designer: Mikhail Smirnov <mikhail.smirnov@fokus.fraunhofer.de>

NEM WP2 Expert: Dana Satriya < dwianto.dana.satriya@fokus.fraunhofer.de>

Date of completion: 27.05.2013 Deadline 17.05.2013 EoB

Processing of Evaluations

To build a consensus of the two evaluations we use the rules of Subjective Logic[3] that operates with <**b**elief, **d**isbelief, **u**ncertainty> measures, such that d+b+u=1 . For this we must interpret the two values <E,C> as the three values <b, d, u>. The structure of this questionnaire was intentionally designed so that such interpretation is obvious. First, since both[4] Designer and Expert are mostly uncertain about the tolerable level of noise it is clear that E maps to u. Similar since confidence is naturally mapping to b the missing disbelief is obtained as d=1-b-u. Second, we evaluate the same mechanism from the three different viewpoints – RO, TI, SD – each having three estimates, thus the normalisation of the SL measures is again naturally performed for the first viewpoint as below:

u(k)=E1(k)/[E(1) +E(2)+E(3)]; b(k)=C(1)/[C(1)+C(2)+C(3)]; d(k)=1 – b(k) – u(k); k=1,2,3

The above is to be computed two times – for the **D**esigner and **E**xpert separately that produces the two sets of values <b(D),d(D),u(D)> and <b(E),d(E),u(E)> respectively.

Finally, we apply consensus operator of SL to derive a common opinion on the mechanism under evaluation. Let us denote x(D,E)=u(D)+u(E) - u(D)·u(E) then the consensus values are given by

cons.b = [b(D) · u(E) + b(E) · u(D)] / x(D,E);

cons.d = [d(D) · u(E) + d(E) · u(D)] / x(D,E);

cons.u = u(D) · u(E) / x(D,E).

The results for NEM SOUP evaluation are given in the table

Designer		**Expert**		**Consensus**		
Evaluation(D)	**C**onfidence(D)	**E**valuation(E)	**C**onfidence(E)	Belief	Disbelief	Uncertainty
0,10	0,85	0,30	0,60	0,415	0,409	0,175
0,12	0,65	0,20	0,60	0,360	0,488	0,152
0,15	0,80	0,40	0,60	0,433	0,298	0,269
0,10	0,80	0,60	0,60	0,416	0,426	0,158
0,20	0,70	0,40	0,50	0,357	0,461	0,182
0,15	0,70	0,70	0,60	0,409	0,365	0,226
0,05	0,75	0,70	0,70	0,409	0,366	0,226
0,05	0,75	0,40	0,70	0,397	0,413	0,190
0,05	0,75	0,60	0,70	0,409	0,366	0,226

Evaluation = Tolerable level of noise

Trust in NEM's Reliable Operation

[3] See references under Links and Sources at http://wiki.univerself-project.eu/t44-trust-autonomics

[4] However the essense of their uncertainty can be different therefor it is highly recommended to provide the two rationales as below.

Expert's Rationale: As NEM consist of multiple sub components that builds the overall architecture, having a reliable entity will always be challenging, if not difficult. This condition however is expected to be worse with additional external entity (KPI) that acts as an input towards the NEM. This is because another layer of complexity needs to be added to sustain the external components' reliability factor. An introduction of cognition technique is helpful to push the reliability of the NPI, however it is ensured that the difference between the three will not differ that much.

Designer's rationale: Context helps but not really much because a good design will convert all likely useful context into the "text", what remains are domain- and deployment-specific pieces of context that could not be taken into account during the design. However I am largely uncertain about these unknown to me pieces of context. Nearly the same for cognition: there is not much room for cognition in the self-orchestration because orchestration is just scheduling, while cognition according to Mitola's CR architecture is mainly in sensing (of radio) – there it is perfectly suited and the result will be brought to SOUP as more accurate input data, i.e. instant values of KPIs.

1. The ability of NEM's algorithm **to solve its problem under noise**

Designer		Expert	
E1=10%	CE1=85%	E1= 30%	CE1= 60%

2. The ability of NEM's algorithm to solve its problem under noise **with the acquisition of relevant context**

Designer		Expert	
E2=12%	CE2=65%	E2= 20%	CE2= 60%

3. NEM's ability to solve its problem under noise with the acquisition of relevant context and **with the use of cognition techniques**

Designer		Expert	
E3=15%	CE3=80%	E3= 40%	CE3= 60%

Trust in NEM Interworking

Expert's Rationale: Multiple NEM is useful to solve a group problem, as long as the system is capable of handling overlapping decision. This eventually depends on many factors including machine learning algorithm that act as the primary decision maker. The same reason as within trust for Reliable Operation, an acquisition of relevant context will complicate the process and will effect in overall algorithm solving. Nevertheless, the supporting group will be able to cope with the additional overhead caused by relevant context acquisition. The cognition technique will for sure benefit the overall system hence the higher trust is obtained by the system group.

Designer's Rationale: Because self-orchestration is essentially the group problem the evaluation of reliable operation is the same, however it appears to me that the role of context in this case is significant because in the typical deployment the parameters of groups of LTE SON control loops will not change very rapidly and therefore an ability to access some kind of directory service where the group information is being periodically updated will largely replace possibly noisy group communication channel. Learning may worsen the situation because of the over-fitting, which (as it intuitively appears to me but was not really studied yet) is a typical situation in group learning.

4. NEM's ability to solve its group problem **under noise in group communication**

Designer		Expert	
E4=10%	CE4=80%	E4= 60%	CE4= 60%

5. The ability of NEM's algorithm to solve its problem under noise **with the acquisition of relevant context**

Designer		Expert	
E5=20%	CE5=70%	E5= 40 %	CE5= 50%

6. NEM's ability to solve its group problem under noise in group communication with context acquisition and **cognition**

Designer		Expert	
E6=15%	CE6=70%	E6= 70%	CE6= 60%

Trust in Seamless Deployment

Expert's Rationale: An introduction of UMF is useful to increase the trust level as by using a single governance entity will elevate the one decision point system. In a group communication, the UMF will be capable as a lead decision maker that decides based on multiple results that might be provided. Compared to overall system, a trust in seamless deployment should show the significant value opposed to the other trust system.

Designer's Rationale: The deployment problem of SOUP is to to launch (also to stop, to modify parameters, to modify rate, …} the scheduling between already deployed LTE SON mechanisms, which mechanisms by themselves are much more robust in solving their deployment problems. But not SOUP. Because of the nature of scheduling SOUP has almost no means to decide on its deployment problem other than configured by GOV. I put flat 5% (not zero) for all three evaluations because there are perhaps scenarios involving interaction of SOUP with KNOW.

6. NEM's ability to solve its deployment problem **under noise**

Designer		Expert	
E7=5%	CE7=75%	E7= 70%	CE7= 70%

7. The ability of NEM's algorithm to solve its deployment problem under noise **with the acquisition of relevant context**

Designer		Expert	
E8=5%	CE8=75%	E8= 40%	CE8= 70%

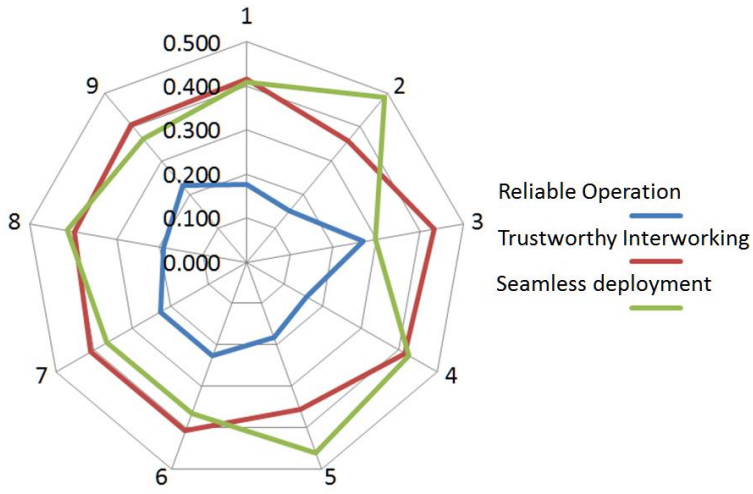
8. NEM's ability to solve its deployment problem under noise in governance with acquisition of relevant context and **cognition**

Designer		Expert	
E9=5%	CE9=75%	E9= 60%	CE9= 70%

Consensus

To assist the computation of NEM raw certificates by multiple experts we have created a spreadsheet termed "consensus calculator", which implements the computation of a consensus operator from subjective logic applied to individual assessments of experts.

The below diagram demonstrates the final consensus between the two experts, who took part in the evaluation of SOUP NEM.


1
2
3
4
5
6
7
8
9
0.500
0.400
0.300
0.200
0.100
0.000
Reliable Operation
Trustworthy Interworking
Seamless deployment

Annex B: NEM Handover Optimisation

NEM Title: Handover Optimization
NEM Designer: Lutz Ewe <Lutz.Ewe@alcatel-lucent.com>
NEM WP2 Expert: Ingo Karla <Ingo.Karla@alcatel-lucent.com>
Date of completion: 16.05.2013 ; (Updates: V2: 03.07.2013, V3: 10.07.2013)

Processing of Evaluations

The NEM Handover Optimization optimizes autonomously the handover parameters in wireless cellular networks, such as LTE, as it had been described and outlined in detail e.g. in [Reference: D3.5, chapter 2.4]. Since then, this NEM has already been developed further to become mature for commercial product deployment. As a part of its functionality, this NEM does also contain a large set of additional security, stability and robustness controlling, ensuring and enforcing functions, control-loops and self-correcting functions, fall-back mechanisms, special case handlings, as well as techniques to safely handle maybe occurring erroneous behaviour e.g. from other network nodes or arising from other external influences. Furthermore, the interactions with other NEMs and other SON functions is addressed and handled. This NEM has extensively been tested and validated at in demonstrators, in the commercial product division, and then finally in a commercial field trial with a commercial LTE network operator on the running LTE system with real users. These tests and trials do also include the interactions with other SON functions / control loops within the system. All these tests have confirmed that this NEM is very reliable, trustworthy and is safely able to perform according to its specified features.

The evaluations of the sections below are summarized the table below, where “E=” denotes the trust level, and “CE=” denotes the confidence level in this trust evaluation. I.e. E=100% means fully trustworthy, and CE=100% means that there is full confidence that this evaluation is correct.

Expert 1		Expert 2		Consensus Values	
E1=100%	CE1=100%	E1=100%	CE1=100%	E1=100%	CE1=100%
E2=100%	CE2=100%	E2=100%	CE2=100%	E2=100%	CE2=100%
E3=100%	CE3=100%	E3=100%	CE3=100%	E3=100%	CE3=100%
E4=100%	CE4=100%	E4=100%	CE4=100%	E4=100%	CE4=100%
E5=100%	CE5=100%	E5=100%	CE5=100%	E5=100%	CE5=100%
E6=100%	CE6=100%	E6=100%	CE6=100%	E6=100%	CE6=100%
E7=100%	CE7=100%	E7=100%	CE7=100%	E7=100%	CE7=100%
E8=100%	CE8=100%	E8=100%	CE8=100%	E8=100%	CE8=100%
E9=100%	CE9=100%	E9=100%	CE9=100%	E9=100%	CE9=100%

In all cases, the NEM-Designer and the NEM expert do agree, that is NEM is 100% trustworthy, and they have 100% confidence in their evaluation, as based on the assumptions as stated in the sections below.

Trust in NEM’s Reliable Operation

Rationale: In order to address this assessment, at first the precise trust scope needs to be defined, the scenario against which the 'trust in reliable operation' is being evaluated. For the trust evaluation of this NEM here, we take the following assumptions:

1) The NEM contains as an integral part all the above described set of stability, reliability and robustness ensuring functions, special case handlings, fall-back procedures, as well as mechanisms to detect and resolve possibly occurring erroneous external situations.
2) The NEM's operation is judged according to what is specified, according to what has been promised that the NEM shall be able to perform and within its specified operation conditions and parameter ranges.
3) The NEM is embedded within the whole system which does also contain several control loops, such as e.g. power adaptations, and pretty robust handover procedures itself, as well as other techniques to ensure that the whole telecommunication system remains stable and reliable. This NEMs operation is also judged for its reliable operation, embedded within the whole telecommunication system.
4) In order to assign 'trust-percentage-numbers' it needs to be defined which number describes which level of trust. We define here that 100% trust assigns that the NEM is fully mature for a commercial product without any trust-reservations at all.
5) This trust evaluation here does address the NEM and the system, but this NEM evaluation does not include the low background risk of hardware failure, such as fire within the equipment-house, and not implementation mistakes or other human errors.

Based on the above assumptions of the trust-scope, we do not see any indications which may doubt in the trust of the reliable operation for this NEM. The extensive testing in the business division and in the field trial in a running commercial network has not shown any aspects which undermine the trust in this NEM. This NEM is – including all its embedded reliability features etc. – a fully mature product for commercial deployment in real systems.

Within this NEM certificate, there are three cases to assess the NEMs "trust in reliable operation":

1) The ability of NEM's algorithm to solve its problem under noise
2) The ability of NEM's algorithm to solve its problem under noise with the acquisition of relevant context.
3) NEM's ability to solve its problem under noise with the acquisition of relevant context and with the use of cognition techniques

The performance of the Handover Optimization NEM can be influenced by the level of noise and of the one of obtained knowledge and by the use of cognition techniques. Obtained knowledge and cognition can be exploited to derive already better matching starting parameters which then potentially result in fewer iteration steps before the final handover parameter set is being reached. The noise influences the needed length for collecting measurement data for one iteration step, as e.g. the noise is averaged out; thus noise impacts the duration of the whole process.

However, these noise, knowledge and cognition aspects do not impact reliability of this Handover Optimization NEM. This NEM is designed to fully guarantee a reliable operation, as specified, under any possibly occurring situation.

9. The ability of NEM's algorithm to solve its problem under noise

Designer		Expert	
E1=100%	CE1=100%	E1=100%	CE1=100%

10. The ability of NEM's algorithm to solve its problem under noise with the acquisition of relevant context

Designer	Expert

E2=100%	CE2=100%	E2=100%	CE2=100%

11. NEM's ability to solve its problem under noise with the acquisition of relevant context and with the use of cognition techniques

Designer		Expert	
E3=100%	CE3=100%	E3=100%	CE3=100%

Trust in NEM Interworking

Rationale: In order to evaluate the trust in interworking of this Handover Optimization NEM with other NEMs, the particular case of the NEMs need to be considered as well as their interworking and conflict resolution strategies.

There are several NEMs, which are orthogonal to the Handover Optimization NEM, so that there is no direct mutual influence. Thus their interworking aspects can be assumed to be non-relevant and there should be no doubts in trustfully simultaneous operation.

The Handover Optimization NEM can potentially interact with some other NEMs, in particular with that group of NEMs which operate on the same handover parameters. In this case, it is necessary to coordinate these involved NEMs and possibly apply conflict resolving approaches.

In particular well suited are those UniverSelf conflict coordination approaches which separate the NEMs in time and/or space, i.e. that only one NEM at one time operates on one particular parameter set.

Similarly as before, the trust-scope needs first to be defined, based on which this "trust in NEM interworking" can then be assessed. The following additional assumptions are taken:

6) The interworking between NEMs is clearly specified, and it part of the NEM to apply the suited inter-NEM strategies and procedures. This includes backup strategies how to handle non-specified interworking with not explicitly defined other NEMs.

Based on these assumptions, it has been shown that the Handover Optimization NEM can trustworthy and reliably interwork with other NEMs, and there were no indications of any doubts in trust. Depending on the particular other NEM and on the employed conflict resolution-, or separation- strategy, the NEM performance may vary, but this does not impact the NEM robustness and reliability. The NEM does still interwork very trustfully according to its promised functionality and performance. During the commercial field trials, there was no indication which could lead to and doubts on the trustworthy interworking with the other simultaneously active SON procedures and system control loops.

Within this NEM certificate, there are three cases to assess the NEMs "trust in NEM interworking":

4) NEM's ability to solve its group problem under noise in group communication
5) The ability of NEM's algorithm to solve its problem under noise with the acquisition of relevant context
6) NEM's ability to solve its group problem under noise in group communication with context acquisition and cognition

The performance of the Handover Optimization NEM could be influenced by the level of obtained knowledge and by the level of noise. However, these noise-, knowledge- and cognition aspects do not impact the reliability in this NEM interworking. This NEM is designed to fully guarantee a reliable and trustworthy interworking, as specified, under any possibly occurring situation.

12. NEM's ability to solve its group problem under noise in group communication

Designer		Expert	
E4=100%	CE4=100%	E4=100%	CE4=100%

13. The ability of NEM's algorithm to solve its problem under noise with the acquisition of relevant context

Designer		Expert	
E5=100%	CE5=100%	E5=100%	CE5=100%

14. NEM's ability to solve its group problem under noise in group communication with context acquisition and cognition

Designer		Expert	
E6=100%	CE6=100%	E6=100%	CE6=100%

Trust in Seamless Deployment

Rationale: This NEM is designed and developed to be very robust, reliable and trustworthy, and to be able to interwork safely within a complex environment with several other NEM, SON and other autonomous control functions. From already at the beginning of the design, the NEM development was targeting towards being suitable for deployment in commercial networks.

This NEM has been implemented by the product business unit into the commercial LTE product and has been tested in a field trial in a running operator's wireless network. Thus this NEM has already been deployed and in this test case it has shown to work always safely, reliably and trustworthy. Thus the trust is very high, that it can without any problems also be deployed in other networks.

In this evaluation, there is a distinction between the three cases:

7) NEM's ability to solve its group problem under noise in group communication
8) The ability of NEM's algorithm to solve its problem under noise with the acquisition of relevant context
9) NEM's ability to solve its deployment problem under noise in governance with acquisition of relevant context and cognition

While a different level of noise, context and cognition could alter the NEM performance, these aspects do not alter the trust in deployment. The NEM can be trustworthy be deployed, according to its specification, for any of these cases.

15. NEM's ability to solve its group problem under noise in group communication

Designer		Expert	
E7=100%	CE7=100%	E7=100%	CE7=100%

16. The ability of NEM's algorithm to solve its problem under noise with the acquisition of relevant context

Designer		Expert	
E8=100%	CE8=100%	E8=100%	CE8=100%

17. NEM's ability to solve its deployment problem under noise in governance with acquisition of relevant context and cognition

Designer		Expert	
E9=100%	CE9=100%	E9=100%	CE9=100%

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
[40] [C. Szyperski, *Component Software: Beyond Object-Oriented Programming*. USA: Addison-Wesley, 2002.
[41] A. Alvaro, E. S. de Almeida, and S. R. de Lemos Meira, “Software component certification: a survey,” in *Software Engineering and Advanced Applications, 2005. 31st EUROMICRO Conference on*, 2005, pp. 106–113.
[42] L. Bass, C. Buhman, S. Comella-Dorda, F. Long, J. Robert, R. Seacord, and K. Wallnau, “Volume I: Market Assessment of Component-Based Software Engineering Assessments,” Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania, Technical Note C CMU/SEI-2001-TN-007, 2000.
[43] M. Goulão and F. Brito e Abreu, “The quest for software components quality,” in *Computer Software and Applications Conference, 2002. COMPSAC 2002. Proceedings. 26th Annual International*, 2002, pp. 313–318.
[44] P. Bender, D. Bourse, H. Butscheidt, B. Deschamps, K. Moessner, and B. Smith, “The E2R II Approach to Reconfiguration Dependability: The Responsibility Chain Concept,” 2007.
[45] J. Voas, “User Participation-based Software Certification,” in *European Symposium on Validation and Verification of Knowledge Based Systems - EUROVAV*, 1999, pp. 267–276.
[46] J. Voas, “Software Certification Laboratories?,” *CrossTalk*, vol. 11, no. 4, pp. 21–23, 1998.

Abbreviations

3GPP	3rd Generation Partnership Project
3GPP LTE	3GPP Long Term Evolution
3GPP SAE	3GPP Service Architecture Evolution
AFI	Autonomic network engineering for the self-managing Future Internet
AP	Access Point
API	Application Programming Interface
BoF	Birds-of-a-Feather
BSS	Business Support System
CAPEX	Capital Expenditures
CCO	Coverage and Capacity Optimization
CL	Control Loop
DiffServ	Differentiated services
DoW	Description of Work
E2E	End-to-End
EL	Efficiency Level
EMS	Element Management System
eNodeB	Evolved NodeB
ETSI	European Telecommunications Standards Institute
FG-FN	Focus Group – Future Networks
FMC	Fix Mobile Convergence
FTTH	Fibre To The Home
GUI	Graphical User Interface
GW	Gateway
H2N	Human-to-Network
HO	Hnd over
ICIC	Intra-Cell Interference Coordination
ICT	Information and Communication Technologies
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
IRTF	Internet Research Task Force
IMS	IP Multimedia Subsystem
IP	Internet Protocol
IRTF	Internet Research Task Force
IS	Information System
IT	Information Technology
ITI	Instantaneous Trust Index
ITU	International Telecommunication Union
ITU-T	International Telecommunication Union – Telecommunications standardization sector
KPI	Key Performance Indicator
LB	Load Balancing
LCCN	Learning-Capable Communication Networks
LE	Large Enterprises
LSP	Label Switched Path
LTE	Long Term Evolution
LTE-A	LTE Advanced
MLB	Mobility Load Balancing

MNO	Mobile Network Operator
MPLS	Multi Protocol Label Switching
MRO	Mobility Robustness Optimisation
MT	Mobile Terminal
NaaS	Network as a Service
NMRG	Network Management Research Group
NMS	Network Management System
OAM	Operations Administration and Maintenance
OFDM	Orthogonal Frequency-division Multiplexing
OFDMA	Orthogonal Frequency-Division Multiple Access
OPEX	Operational Expenditures
OSS	Operations Support System
OTI	Overall Trust Index
PDN-GW	Packet Data Network Gateway
QoE	Quality of Experience
QoS	Quality of Service
RACH	Random Access Channel
RAT	Radio Access Technology
ROI	Return of Investment
RAN	Radio Access Network
RRM	Radio Resource Management
SGW	Serving Gateway
SME	Small and Medium Enterprises
SLA	Service Level Agreement
SON	Self Organized Networks
TCO	Total Cost of Ownership
TMF	TeleManagement Forum
TTT	Time To Trigger
UC	Use case
UE	User Equipment
UMF	Unified Management Framework
VoIP	VoIP - Voice over IP
VPN	Virtual Private Network