29.3.2020

Artificial Intelligence (AI) in Test Systems, Testing AI Models and ETSI GANA Model's Cognitive Decision Elements (DEs) via a Generic Test Framework for Testing GANA Multi-Layer Autonomics & their AI Algorithms for Closed-Loop Network Automation

White Paper No.5 *produced under the umbrella of* ***ETSI PoC on 5G Network Slices Creation, Autonomic & Cognitive Management and E2E Orchestration; with Closed-Loop(Autonomic) Service Assurance of Network Slices; using the Smart Insurance IoT Use Case***

Edited by ETSI 5G PoC Consortium Steering Committee and Contributors
ETSI TC INT AFI WG 5G POC

Table of Contents

Related White Papers:

- ***White Paper No.1:*** *C-SON Evolution for 5G, Hybrid SON Mappings to the ETSI GANA Model, and achieving E2E Autonomic (Closed-Loop) Service Assurance for 5G Network Slices by Cross-Domain Federated GANA Knowledge Planes*

- ***White Paper No.2:*** *ONAP Mappings to the ETSI GANA Model; Using ONAP Components to Implement GANA Knowledge Planes and Advancing ONAP for Implementing ETSI GANA Standard's Requirements; and C-SON – ONAP Architecture*

- ***White Paper No.3:*** *Programmable Traffic Monitoring Fabrics that enable On-Demand Monitoring and Feeding of Knowledge into the ETSI GANA Knowledge Plane for Autonomic Service Assurance of 5G Network Slices; and Orchestrated Service Monitoring in NFV/Clouds*

- ***White Paper No.4:*** *ETSI GANA as Multi-Layer Artificial Intelligence (AI) Framework for Implementing AI Models for Autonomic Management & Control (AMC) of Networks and Services; and Intent-Based Networking (IBN) via GANA Knowledge Planes*

- ***White Paper No.6:*** *Generic Framework for Multi-Domain Federated ETSI GANA Knowledge Planes (KPs) for End-to-End Autonomic (Closed-Loop) Security Management & Control for 5G Slices, Networks/Services*

Executive Summary

Artificial Intelligence Models (AI Models) are enablers for advanced intelligence in the self-management and control operations now strongly required of the evolving and future networks such as 5G Networks. AI algorithms bring benefits to diverse aspects in development and deployment of AI exhibiting systems such as Autonomic/Cognitive 5G networks and their associated Autonomic Management and Control systems. There is now a strong and urgent industry need for standards and methods for Testing AI Models that are posed to empower 5G networks in achieving ***autonomic 5G networks that are self-configuring, self-adaptive (self-diagnosing, self-healing/repairing, self-optimizing, self-protecting), and self-aware***. Recently NGMN has published the 5G E2E Architecture Framework [33] in which requirements for Autonomic Management & Control (AMC) of Networks and Services are specified.

This White Paper covers three topics that ETSI TC INT and TC MTS have started to jointly work on using a Newly Launched Work Item in ETSI: (1) *The benefits AI brings to Test Systems (e.g. in reduction of Test Suites execution time in Performance Testing complex systems);* (2) *Testing AI Models for autonomic and cognitive management & control of network resources, parameters and services—and possibility for certification of AI models by using a "Qualified Automated Test Component(s) or System" that exhibit best quality AI capabilities for testing those of AI Component(s)/System Under Test, thanks to metrics that need to be standardized and be applied for certifications*; (3) *Generic Test Framework for Testing ETSI GANA (Generic Autonomic Network Architecture) Model's Multi-Layer Autonomics and associated Multi-Layer AI Algorithms for Closed-Loop Network Automation*. The White Paper addresses the following aspects of relevance to the work that has been commenced in ETSI to produce a set of documents:

- A General Guide on the Benefits of Artificial Intelligence (AI) in Test Systems, with illustrations of Artificial Intelligence (AI) in Test Systems and the Benefits
- A General Guide for Testing AI Models in General, and the Definitions of Standardized Metrics for Measurements and Assessments in Testing and Certification of AI Models, Design-Time Testing and On-Line (Run-Time) Testing of AI Models, including Certification of AI Models of Autonomic Components/Systems
- The Metrics on what AI brings to Test Systems (e.g. Performance Test Systems and Functional Test Systems)
- Initiation of Work on Definitions and Standardization of Metrics of specific classes of AI Models so as to enable Test Systems and Certification communities to test the types of AI Models and to provide certification services
- A Methodology for Testing AI Models that are designed for autonomic (closed-loop) and cognitive management & control of network resources, parameters and services
- A Generic Test Framework for Testing the ETSI GANA Model's Multi-Layer Autonomics Decision-making-Elements (DEs) and their associated AI Algorithms for Closed-Loop Network Automation
- How Testers for Performance and Functional Testing of Complex Systems can leverage the understanding of Metrics on what AI brings to Performance and Functional Test Systems to partner with Organizations that have Products/Solutions in this space in order to benefit from such solutions/products and the overall benefits of AI in Test Systems
- How Test Solution Vendors shall benefit from both a Methodology for Testing AI Models that are designed for autonomic and cognitive management & control of network resources, parameters and services; and the Generic Test Framework for Testing ETSI GANA Model's Multi-Layer Autonomics & AI Algorithms for Closed-Loop Network Automation, in developing Test Solutions for Autonomic Management and Control Intelligence Software being introduced in emerging and future networks such as 5G.
- The Value of Federated Testbeds and Open API for Interoperable Testbed Federations in the Testing of AI-powered Federated GANA Knowledge Plane (KP) Platforms for AMC

This White Paper lays the groundwork for the newly launched Work Item (WI) in ETSI TC INT on ***AI in Testing Systems and Testing AI Models*** that is to address the various aspects linked to this topic through developing ETSI assets such as Specifications that can be used by the industry. The Work Item has now been created in ETSI (https://portal.etsi.org/webapp/WorkProgram/Report_WorkItem.asp?WKI_ID=58442), and organizations are invited to contribute to the work and deliverables of the Newly Launched Work Item in ETSI as described in Chapter 1 of this White Paper. Such Specifications should cover the definition of Metrics pertaining to selected classes of AI Models that can be targeted for Testing and Assessment since such Metrics Definitions are missing in the work being done in the various Standardization Groups today. As an example, Metrics for assessing GANA Cognitive DEs as Deployable AI Models. **NOTE:** Readers are encouraged to follow the developments on this topic in ETSI, and also to download and read complementary *White Papers* of the ETSI 5G PoC, which are available and downloadable at: https://intwiki.etsi.org/index.php?title=Accepted_PoC_proposals .

1. ETSI's Newly Launched Work Item on "**AI in Test Systems, and Testing AI Models**", and the Key Takeaways of the White Paper

1.1 ETSI TC INT and ETSI TC MTS team up on the Newly Launched Work Item: "**AI in Test Systems, and Testing AI Models**": Invitation for Contributions

A new Work Item(WI) has been launched in ETSI TC INT, and it is expected that groups in ETSI that are working in the areas of Testing are to team up in some way in order to produce the deliverables expected of this Work Item. This White Paper is providing an introduction to the aspects captured in the diagram below (Figure 1), outlining the problem statements associated with each aspect and what should be the approach to addressing the problems. The various aspects covered in this White Paper are to be used in contributing to the deliverables of the newly launched Work Item (WI) in ETS TC INT (https://portal.etsi.org/webapp/WorkProgram/Report_WorkItem.asp?WKI_ID=58442). The Scope and the ***Four Expected Deliverables*** of the Newly Launched Work Item in ETSI TC INT are as follows:

1. *A General Guide on the Benefits of Artificial Intelligence (AI) in Test Systems, with illustrations of Artificial Intelligence (AI) in Test Systems and the Benefits*
2. *A General Guide for Testing AI Models in General, and the Definitions of Standardized Metrics for Measurements and Assessments in Testing and Certification of AI Models, including Certification of AI Models of Autonomic Components/Systems*
3. *Testing ETSI GANA Model's Cognitive Decision Elements (DEs) as AI Models for Autonomic (Closed-Loop) Network Automation in the space of Autonomic Management & Control (AMC) of Networks and Services, with illustrations of AI Models for Autonomic Management & Control of 5G Network Slices*
4. *Generic Test Framework for Testing ETSI GANA Multi-Layer Autonomics & their AI Algorithms for Closed-Loop Network Automation. An ETSI Technical Report (TR) will be produced in 2020/2021 to extend the early Draft Generic Test Framework in ETSI EG 203 341 V1.1.1*

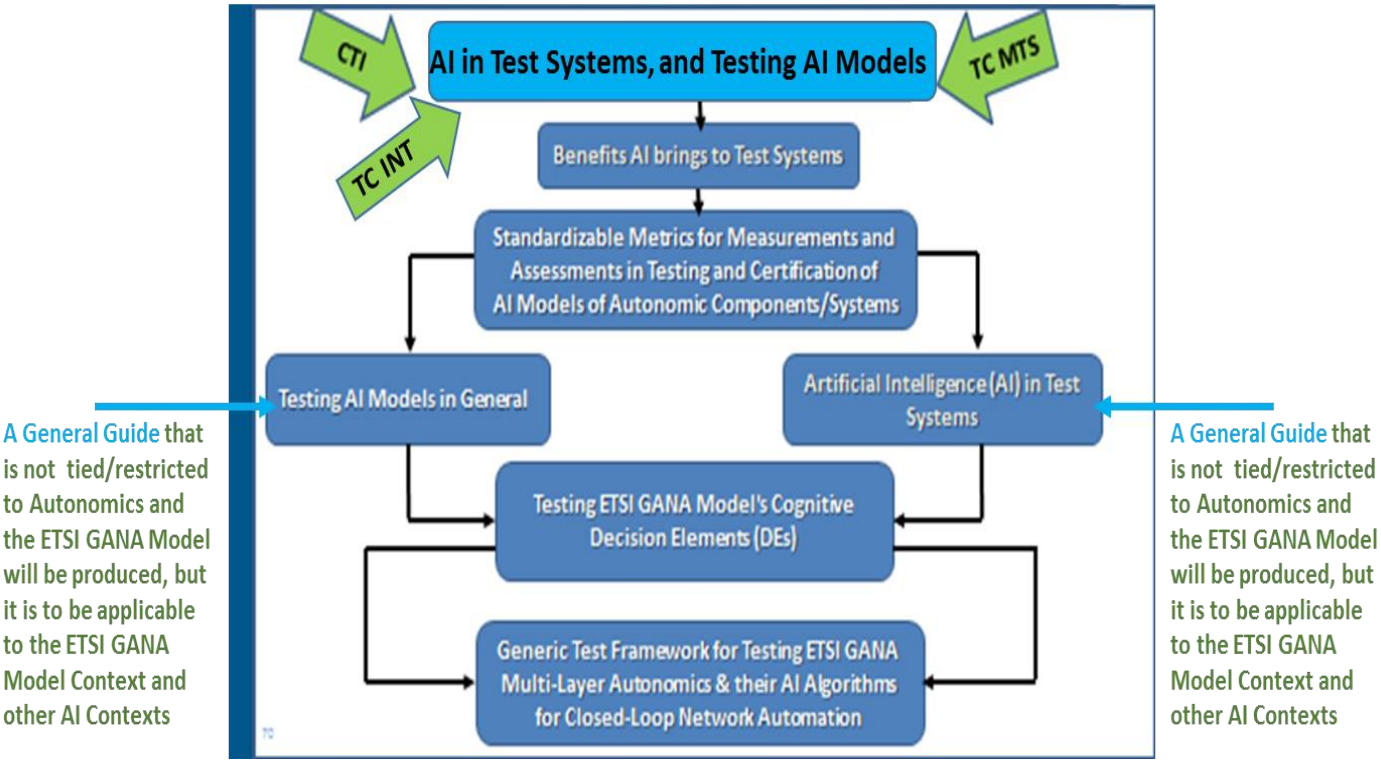

Figure 1: ***Topics and Deliverables to be produced by the Newly Launched Work Item (AI in Test Systems, and Testing AI Models)***

Summarized Perspectives from ETSI TC MTS and ETSI TC INT on what should be addressed by the following Two Guides (documents) to be produced as part of the deliverables of the Newly Launched Work Item:

1. ***A General Guide on the Benefits of Artificial Intelligence (AI) in Test Systems, with illustrations of Artificial Intelligence (AI) in Test Systems and the Benefits***

   The following are the aspects to be covered in the Guide:
   - AI in Performance Test Systems
   - AI in Functional Test Systems: For example, in regression test optimization using search-based algorithms to identify the smallest subset of test cases or prioritize regression test cases, with the benefit of reduced manual effort for execution time for regression testing—thanks to AI
   - AI in Online Test Systems (An Online Test System is one used for testing a System or Component that is integrated and actually running in a production/real environment in which it is designed to operate):
     a) Search-based algorithms for dynamic test generation and evaluation for systems with large input spaces and test goals

that can be formalized as a quality function, as demonstrated by many security testing tools, most prominent is the fuzz testing tool AFL, useful for automated exploratory testing, with the benefit that AI can help achieve fully-automated testing with reduced manual effort and optimization for each goal that can be formalized

b) model inference for non-functional testing to learn the model from execution traces (online/offline) or through active learning, with the benefit that AI helps achieve a model for further test generation with small or no manual effort

c) automated test oracles for anomaly detection, e.g. using classification algorithms, with the benefit that AI may reduce false negative rate

- Stakeholders that should consider the Benefits in their Business Models, namely: Test Solution Vendors; Testers, other stakeholders
- Identify Low-hanging fruits, e.g. test result classification and test selection
- **Use of AI technology in testing systems** along the test process, targeting testing and certification of classical systems, identification of best-practices:
  - **a)** Requirement solicitation
  - **b)** Test objective and test model specification
  - **c)** Test derivation for functional and non-functional requirements:
    - **1.** Evolutionary testing, use of genetic algorithms
    - **2.** Reinforcement learning in adaptive testing of executable test models
    - **3.** Meaning of "state" in non-functional testing
    - **4.** Learning state coverage in adaptive testing
    - **5.** Further metrics in adaptive testing, besides state coverage
  - **d)** Test result analysis and fault localization:
    - **1.** Anomaly detection in test logs, e.g. stark deviation of test execution times
    - **2.** Classification of test results, e.g. according to types of performance characteristics
  - **e)** Test selection in regression testing:
    - 1. Test selection based on code changes and probable failure rates

2. A General Guide for Testing AI Models in General, and the Definitions of Standardized Metrics for Measurements and Assessments in Testing and Certification of AI Models, including Certification of AI Models of Autonomic Components/Systems

The following are the aspects to be covered in the Guide:

- A General Methodology for Testing AI Models

- Standardization of Metrics of specific classes of AI Models so as to enable Test Systems and Certification communities to test the AI Models and to provide certification services
- Stakeholders that should make use of the Guide
- Proposed approach: Testing of systems that comprise AI technology:

  a) When considering certification (conformance, interoperability testing), the concrete AI technology used in the system might be not exposed.

  b) Consider general system properties that are realized using AI and define the test framework accordingly:

  1. Test objective and test model specification
  2. Test derivation and test execution
  3. Test result analysis

  c) Design-Time Testing and On-Line (Run-Time) Testing of AI Models
- Classes of AI-empowered systems to be considered:

  a) Classification systems:

  1. Data processing, similar to data-flow testing
  2. E.g., separation of training data and test data
  3. Quality characteristics of data

  b) Self-adaptive systems (Autonomic Systems):

  1. Testing resilience against failures, e.g. fault-tolerant and self-healing systems
  2. Testing structural adaptation, e.g. adding/deleting nodes in a distributed system
  3. Testing behavioral adaptation due to environment changes, e.g. system operation under uncertainties
  4. Etc. (systematic classification required)

  c) Other Types may be considered but there will be initial scope of focus and later other types of systems would be considered
- Generic Test Framework for Testing AI Models / Systems during their lifecycles (with a focus on selected classes of AI Models, e.g. Machine Learning (ML) and Deep Learning (DL))
  1. **Testing and Validation:** of an AI Model (or a collective bundle of interworking AI Models) based on well-defined criteria for verdict passing
  2. **Conformance Testing** (against a well-defined specification of behavioural and structural features expected of the AI Model)

3. **Interoperability Testing** (based on what is expected to be communicated at Reference Points involving the AI Model (as a Component/ System) and other entities required to interact with it); and against what is expected of the AI model behaviour by the target integration and deployment environment
4. **Integration and User Acceptance Testing** of the AI Model that has been integrated to interwork with other AI Models in the network that is also being tested as a whole
5. **Self-Testing Capability of a AI Model** as a Component or System
6. **Validation phase of an AI Model** (or collective bundle of interworking AI Models)
7. **Trustworthiness building phase** of an AI Model (or collective bundle of interworking AI Models), including fuzzing approaches with domain-specific heuristics for generating adversarial samples to perform security and robustness testing of AI models
8. **Certification Phase for the AI Model** (or collective bundle of interworking AI Models):
   a) **NOTE 1:** For Certification, the various Definitions of Standardized Metrics for Measurements and Assessments in Testing and Certification of AI Models to be defined by this guide shall be the basis for Certification of AI Models by the Industry and relevant bodies.
   b) **NOTE 2:** Certification is currently a research topic (several research project and research programs are currently active in this fields, e.g. for the aerospace sector a research project there is a project that will start this summer of 2020 and Fraunhofer is working on this, and the German Federal Office for Information Security is also working on a verification scheme for AI).
9. **Test Network Deployment Phase** (pertaining to a Test Network in which the AI Model(s) is being deployed)
10. **AI Model Deployment and Activation Phase**
11. **Test Network Operation phase** (pertaining to a Test Network in which the AI Model(s) has been deployed)
12. **Test Network Optimization Phase** (pertaining to a Test Network in which the AI Model(s) has been deployed)

- Testing Offline AI Models which need a programmable configurable (re)-Training process in the CSPs' Training environment before being exposed to new data (real data) in the Production environment
- Testing Online AI Models that are directly deployed in the CSP's Production environment, and are continuously exposed to real and new data and they have the ability to learn and modify their behavior continuously. They require a continuous testing, with faster response time.

1.2. Key Takeaways of the White Paper

Regarding contributions to the newly launched Work Item in ETSI and the outlined deliverables expected to come from the Work Item, organizations are invited to make contributions to the work item: https://portal.etsi.org/webapp/WorkProgram/Report_WorkItem.asp?WKI_ID=58442

The novelty regarding the aspect of "***Benefits Artificial Intelligence (AI) brings to Test Systems"*** lies in the benefits such as reduction of Test Suites execution time in Functional and Performance Testing of complex systems in contrast to comparable Functional and Performance Test Systems that do not employ AI; and in cutting down the execution time for Functional and Performance Test Suites. The Metrics on what AI brings to Functional and Performance Test Systems need to be standardized, and this white paper sought to lay the groundwork for that and other types of standardisable items to be covered by the newly launched Work Item. The novelty regarding "***Testing AI Models employed by components for autonomic and cognitive management & control of network resources, parameters and services***" lies in that the recently emerged ETSI GANA (Generic Autonomic Networking Architecture) Reference Model (ETSI TS 103 195-2) is being considered as the main architectural framework of focus here as it is now being adopted by various SDOs/Fora as the standard for Multi-Layer Autonomic Management & Control (AMC) of networks and services. Therefore, the novelty is also linked to the consideration of the ongoing ETSI GANA Model adoptions by various SDOs/Fora into their frameworks. Also, because the industry is now calling for a Test Framework for Testing GANA components, various assets on Test Components and Test System Architectures are now required by the industry in the context of the so-called GANA Cognitive Decision-making-Elements (DEs) as deployable AI Models for autonomic management and control of network resources and services parameters. The following aspects are discussed in this White Paper:

1. Consideration of the ETSI GANA Model Standard (ETSI TS 103 195-2) as a Multi-Layer Artificial Intelligence (AI) Framework for Implementing AI Models for Autonomic Management & Control (AMC) of Networks and Services. As such, the ETSI GANA Model enables to capture the nature of AI Models employed by components for autonomic and cognitive management & control of network resources, parameters and services. The ETSI GANA Model defines such components as Decision-making-Elements (DEs), the levels of abstractions at which to implement such DEs, and also defines the structural model of a DE concept (DE interfaces and primitives that should be supported by a DE)
2. Consideration of the TMForum efforts on Building a Training Data Repository for AI Models that can be used by AI Models Developers and Testers
3. Taking into consideration the Ecosystem on AI Models Development, Procurement, Test and Certification, and Deployment, including the roles played by the following stakeholders: *Autonomic Network Management & Control Domain Expert(s); Cognitive GANA DE Vendor (AI Model Supplier); Data Scientist; Provider of Training Data Repository*
4. The impact of the development of a *Market Place for ETSI GANA DEs*, e.g. the implication of the Market Place on the Ecosystem on AI Models Development, Procurement, Test and Certification, and Deployment

Through this White Paper, various organizations in the industry obtain knowledge and insights on the following aspects:

- The launch of work in ETSI TC INT on Testing AI Models, and the nature of the Specifications and Technical Reports expected as output of the joint effort of ETSI TC INT and ETSI TC MTS
- The Metrics on what AI brings to Performance Testing Systems for Complex Systems, i.e. Targets for AI in Performance Test Tools:
    1. *Speeding up and improving quality of evaluations of performance measurements*
    2. *Supporting or producing correct configuration of different performance measurements based on the purpose and conditions of a measurement, i.e. the user specifies what are the targets and conditions of an intended performance measurement and the test tool with AI does the rest.*
    3. *AI could also find out all additional performance characteristics and KPIs that can be produced from the captured data of a performance measurement.*
    4. *AI will also enable completely autonomous performance measurements.*

- The Metrics on what AI brings to Performance Test Systems, and Products that can be used by Testers to leverage the benefits
- Standardization of Metrics of specific classes of AI Models so as to enable Test Systems and Certification communities to test the AI Models and to provide certification services
- How Testers for Performance Testing of Complex Systems can leverage the understanding of Metrics that AI brings to Performance Test Systems to partner with Organizations that have Products/Solutions in this space in order to benefit from such solutions/products and the overall benefits of AI in Test Systems
- How Test Solution Vendors shall benefit from both a Methodology for Testing AI Models and the Generic Test Framework for Testing Multi-Layer Autonomics & AI Algorithms for Closed-Loop Network Automation in developing Test Solutions for emerging and future networks such as 5G networks (expected to embed AI and Autonomic (Closed-Loop) Management and Control capabilities)
- Establishing a Generic Test Framework that can guide Testers in testing of ETSI GANA Functional Blocks for Autonomics as individual components and as integrated components.
- Conformance Testing and Interoperability Testing for GANA Functional Blocks (DEs, ONIX, MBTS) based on their Reference Points and Characteristic Information exchange expected on the Reference Points; i.e. Conformance Testing and Interoperability Testing is required on the Reference Points for GANA Autonomics components instantiated in a target network architecture and environment
- Criteria for use in Verdicts passing when Testing Autonomic Functions (AFs), i.e. GANA DEs and in Testing AI Models in general
- The role of a GANA Meta-Model in generation of Data Types for use in Test Suites Development
- Need for Specifications to be provided to Tester by a DE vendor, regarding "assertions/claims" on what the DE strives to achieve during its operations, with indications on the metrics (e.g. KPIs) that can be measured and monitored, appearance/manifestations of new instances of objects the DE causes to be created, or change in state of certain objects impacted by the DE's autonomic operations
- Testing non-cognitive GANA DEs, taking into consideration the "Operating-Region" of a Control-Loop(s) associated with the DE
- Testing Cognitive GANA DEs as deployable AI Models, taking into consideration the "Operating-Region" of a Control-Loop(s) associated with the DE, while taking into considerations Training Data Repository for AI Models (i.e. the cognitive DEs), and AI Algorithms that can be employed by a DE logic, such as Machine Learning(ML), Deep Learning (DL), Computational Intelligence, etc.
- Integrated self-testing within a DE (i.e. embedded testing using a test component embedded within the DE
- Validation, Trustworthiness-building, and then Certification of a DE (or collective bundle of interworking DEs)
- Testing a collective group/bundle of interworking DEs as a black box (applies especially to DEs within GANA nodes (i.e. DEs in Network Elements/Functions(NEs/NFs))
- Consideration of the various Components that need to interwork with a GANA DE Under Test in Automated Test Developments and Executions
- Testing GANA Knowledge Plane (KP) Platform as an AI system of collaborating DEs (AI Components)
- Testing vertical interactions of DE stacks along the GANA Hierarchy of Decision-making-Elements (DEs)
- Test Data required for Testing DEs
- The Types of Testing and associated Test Systems and components that should be applied in Testing DEs during various phases of DE lifecycles and also phases of the lifecycle of the Network to be impacted by the DE's operations (Validation Phase of a DE, Trustworthiness building phase for a DE, Certification Phase for a DE, Network Deployment Phase, DE deployment and activation Phase, Network Operation Phase, Network Optimization Phase)
- Where Passive Testing plays a role and where combined Active and Passive Testing plays a role in Testing DEs
- Integration and User Acceptance Testing of GANA DEs

2. The Benefits AI brings to Test Systems (embedding AI in a Test System/Component)

2.1. Overview

The following are some of the benefits AI brings to Test Systems:

- Reduction of Test Suites execution time in Functional and Performance Testing of complex systems;
- Performance issues and snapshots
- Improvement in the Quality of Measurements
- There is also some value AI brings in a Test System for Offline Testing (meaning, testing a System or Component that is integrated and actually running in a production/real environment in which it is designed to operate) as found in various sources in literature.
- There is also some value AI brings in a Test System for On-Line Testing as found in various sources in literature. An example of such value is illustrated in chapter 7 that describes an AI-based Test System meant for testing a 5G network that is integrated and actually running as a production network that is delivering real network services.

There are reports in literature on Test frameworks that leverage AI in Test Systems, and also on autonomics (closed-loops) introduced in Test Systems to make them intelligent and adaptive in the way they continuously perform Testing in DevOps and Agile products development and testing environments. For example, [21] presents the concept of Autonomic Test Automation, which according to [21] is defined as "an innovative new approach that uses artificial intelligence, machine learning, and test analytics for automated test design, execution, optimization and maintenance of the automation framework". Quoting [21], "*the benefits of AI and autonomics in Test Systems include self-adaptation, increased test operational efficiency, improved test coverage and extended duration test cycles with fail-safe recovery, and that also, Artificial-Intelligence-driven test analytics can be used to enrich the test model using machine-learning techniques on large sets of historical and production data*".

2.2 AI in Performance Test Systems: A Perspective by Softwell Performance AB

2.2.1 Status for performance measurements and tools today

The pace of deliveries of software to testing increases all the time due to current software development paradigms. At the same time complexity of tested software also grows rapidly.

While function testing gradually becomes part of development cycles through test automation, performance measurement is still outside the development loop and consequently tends to be neglected.
In the connected world, however, performance requirements also increase all the time. As a matter of fact, system performance is the most visible asset of every application in the connected world since it's experienced in every interaction by the users. These trends put quality assurance of system performance under tough pressure. Code delivered to function testing should be tested for system performance at the same pace, which is required to monitor changes in system performance of the System Under Test (SUT).
This is, however, not simple to solve. Performance requirements are in many cases far from as precisely specified as function requirements, if specified at all. Furthermore, test cases for system performance measurements are very complex and time consuming to create. The test case specifications vary greatly depending on the kind of system performance characteristics that should be measured. Some examples of variations include:

- *SUT conditions that must apply when system performance measurement data are captured.*
- *Selection of SUT services that should be used in the performance measurements.*
- *The required execution time of a test case.*
- *The required number of simulated users.*

- *Requested combinations of application protocols, transport protocols, and network protocols.*
- *Selection of measurement data that should be collected to deliver requested system performance characteristics.*
- *Specification of transaction data needed to make service requests from simulated users look different.*

Enabling simple and fast test case creation and possibilities to execute performance measurements for every system build is, however, not all.
In order to monitor the changes of system performance from build to build, performance measurement results from each build must be evaluated and compared with stated performance requirements and earlier measurement results. This requires a performance measurement database where all collected measurement data from every test run are stored.
Other targets for improvements of performance measurement tools are test productivity. System performance tests are generally regarded as costly activities in terms of costs of test tools, costs of required test equipment for SUT and test tools, costs of manpower required for training, creation of test cases, test runs, and evaluation of test results.

2.2.2 What is required?

What is required from quality assurance of system performance measurements under these circumstances?
Here follow some examples of requirements on performance test tools:

1. *Performance test tools must have the capability to act on signals from development cycles such as system builds.*
2. *Performance test tools must have the capability to start and run system performance tests autonomously.*
3. *Performance test tools must have access to a database for performance measurement data with a granularity in terms of performance of single services*
4. *Performance test tools must have the ability to quickly perform capacity measurements of individual services. This is a requirement to run performance measurements in parallel with functional tests of system builds.*
5. *Performance test tools must be capable of translating specifications of performance test cases expressed in terms of what characteristics shall be measured and for what SUT services into traditional specifications of performance test cases.*
6. *Performance test tools must have the ability to evaluate system performance results autonomously.*
7. *Performance test tools must have the ability to deliver system performance results autonomously.*
8. *Performance test tools must have the capability to monitor test runs regarding bad performance trends and abort the job if such situations are detected. This would save valuable test time when running performance measurements of reliability characteristics, such as stability or availability figures. Such test cases can be configured for several days of continuous test execution.*
9. *Performance test tools must be capable of drawing conclusions from earlier test run in order to focus coming test runs on the most critical performance requirements of the SUT.*

Other requirements, outside the performance test tools, should be a generally accepted syntax for all aspects of performance requirement specifications that enable automated compilation into test cases.

2.2.3 What are the solutions?

The following four requirements from the list above are prerequisites to enable system performance measurements at the same pace as functional tests and thus don't require AI per se, but they are also prerequisites for building AI into performance test tools:

1. *Performance test tools must have the capability to act on signals from development cycles such as system builds. This is a requirement that can be resolved simply with event messages sent from the system build process at the end of a build to the performance measurement system.*
2. *Performance test tools must have the capability to start and run system performance tests autonomously. This requirement does not require AI, but is needed for activities initiated by AI.*
3. *Performance test tools must have access to a database for performance measurement data with a granularity in terms of performance of single services. This is another prerequisite to AI in performance measurement tools.*
4. *Performance test tools must have the ability to quickly perform capacity measurements of individual services. This is a requirement to run performance measurements in parallel with functional tests of system builds.*

Requirement 4 also enables performance measurements of all permutations of two services to detect hidden dependencies that have an impact on system capacity.

2.2.4 What values will AI in performance test tools add?

The following five requirements from the list of what is required above will enable high level of automation of performance measurements with help of AI built into the test tools:

1. *Performance test tools must be capable of translating specifications of performance test cases expressed in terms of what characteristics shall be measured and for what SUT services into traditional specifications of performance test cases. This requirement will reduce time and manpower to a minimum and lower the requirements on testers.*
2. *Performance test tools must have the ability to evaluate system performance results autonomously. This requirement will further reduce time and manpower required for performance measurements. A generally accepted syntax for all aspects of performance requirements is, however, prerequisite for high quality performance evaluations.*
3. *Performance test tools must have the ability to deliver system performance results autonomously. This requirement will also reduce time and manpower required for performance measurements. Together with requirements 1 -6 this requirement is the last piece in what is required fully autonomous performance measurements.*
4. *Performance test tools must have the capability to monitor test runs regarding bad performance trends and abort the job if such situations are detected. This is case where AI in performance tools will pay off and save valuable time and test resources for other test tasks.*
5. *Performance test tools must be capable of drawing conclusions from earlier test run in order to focus coming test runs on the most critical performance requirements of the SUT. Measuring system performance is a learning process of the tested systems behaviour in different situations and the impact of different changes on the system performance. Over 95% of all performance measurements are regression tests that will over time generate very large amounts of collected measurement data. This is therefore an ideal for computer learning that will eliminate large amount of manpower. With help of a computer learning system and earlier measurement results will a performance test tool also be able to focus future performance measurements for every build on the most performance critical parts of the system.*

2.2.5 New challenges for performance measurement tools

In addition to this, system performance measurements face new challenges with introduction of AI as described in the ETSI GANA model [1]. The targets here are to measure the quality and performance of AI based decision making, which in many cases require a performance measurement tool. This is described later in the paper.

2.2.6 What targets shall be achieved by this?

We are convinced that proposed solutions to the requirements stated above will lower the cost of performance test to a level below the cost of functional tests and yet deliver measurement results with much better quality than today. Furthermore, measurement results will enable automated configurations of systems with high precision based on customer requirements to be fulfilled by a particular demanded system.

3. Testing AI Models for autonomic and cognitive management & control of network resources, parameters and services

3.1. The Main Aspects in Testing AI Models/Systems in General

In general, Testing AI Systems involves the following aspects [22]:

- **Data Validation**: In general, Testing of AI systems, in contrast to the testing of traditional non-AI systems that normally involves the **validation of the conformance of the outputs** of the system against specific inputs, testing of AI system rather involves **validation of the inputs themselves** (i.e. **Data Validation**) in order to verify the robustness of the AI system in terms of its ability to make effective and appropriate decisions in its outputs. The reason is because the effectiveness of AI systems in their decision making

capability mainly depends on the quality of the training data (and the training data has to also include aspects such as bias and variety/diversity) [22].

- **Algorithms Testing and Model Validation**: The core part of AI systems is built on various types of algorithms that process data and generate actionable insights from the data. Model validation, the ability to learn, efficiency of a particular algorithm and empathy belong to the various key features that have to be considered in testing AI systems. The ability of the AI system to learn is the ability of the system to learn and modify its behavior with time [22]. The broader picture of algorithms at the core of AI systems are called *cognitive algorithms* and as defined in ETSI TS 103 195-2, "cognition" involves "learning" and "reasoning". The various algorithms and techniques for learning help build the AI Model that is at the core of an AI system, and is what needs to be subjected to validation by rigorously testing it under various inputs. Other algorithms that may be employed by an AI system include *optimization algorithms* aimed at optimizing some parameters that determine the AI system's behavior and some corresponding actions the system has to take in order to enforce the parameters to attain the optimized values.
- **Non-functional Testing (e.g. Performance Testing and Security Testing)**: Performance Testing of the AI systems needs to be carried out in order to determine the effectiveness of the system in delivering its services and timeliness in its responses to changing inputs, as well as measuring the Key Performance Indicators (KPIs) of the system against various factors such as the resources required by the system. Security testing of the AI system is done in order to ensure that the system does not violate certain security requirements and also to determine the extent to which the system is vulnerable to attacks or can pose a security risk when put into operation.
- **Integration Testing:** The Integration Testing of the AI system involves testing the ability of the system to communicate with the various other systems and components the system is required to interact with during its operation, and it also involves ability of the system to accept any loadable components of which the system may be required to support the loading by the user such that the system uses the on-boarded components in its operation.

3.2. ETSI GANA as Multi-Layer AI Reference Model for Implementing Autonomic Management & Control (AMC) of Networks and Services (including 5G Network Slices)

White Paper No.4 [23] covers in much more detail the topic of ETSI GANA as Multi-Layer AI Reference Model for Implementing Autonomic Management & Control (AMC) of Networks and Services (including 5G Network Slices). This section is an extract from the descriptions provided in the White Paper No.4 [23] and is for the purpose of providing insights on the nature of AI Models associated with the GANA framework before delving into the generic framework for testing such AI models as described later.

The ETSI TS 103 195-2 [2], a standard of a Generic Autonomic Networking Architecture (GANA)—an Architectural Reference Model for Autonomic Networking, Cognitive Networking and Self-Management of Networks and Services, defines the concept of what is called an "**Autonomic Manager** element" (called a "**Decision-making-Element" (DE)** in the GANA terminology) as a functional entity that drives a control-loop meant to configure and adapt (i.e. regulate) the behaviour or state of a Managed Entity (i.e. a resource)—usually multiple Managed Entities(MEs). Figure 2 presents a snapshot of the GANA Framework and the abstraction levels at which DEs for implementing control-loops and the interworking the DEs should be considered in order to implement multi-layer autonomics.

ETSI TC INT AFI WG's work on E2E autonomic networking involves introducing self-manageability (autonomics) properties (*e.g. auto-discovery, self-configuration, self-diagnosis, self-repair, self-healing, self-protection, self-defence, self-awareness, etc*.) within network nodes/functions themselves and also enabling distributed "*in-network*" self-management within the data plane network architectures (and their embedment of "thin control planes"). This low level intelligence (autonomics) achievable by so-called "GANA DEs" that should be instantiated to drive fast control-loops within network nodes/elements and to drive horizontal, self-adaptive, and collaborative "in-network" behaviour involving the collaboration of certain autonomic nodes at certain points in a network topology is also called

"*Micro level"* autonomics ("fast control-loops"). The low level autonomics shall be complemented and policy-controlled (governed) by higher level autonomics ("slow control loops") (at "*Macro level*"). *Macro level autonomics* is achievable and driven by higher level "GANA DEs" responsible for network-wide and logically centralized autonomic management and control of networks and services. At "Macro level", the autonomics paradigm (control loops) is introduced outside of network elements, in the outer, logically centralized, management and control planes architectures of a particular target network. This outer "realm" for implementing the much more complex, cognitive and analytics algorithms (including Artificial Intelligence (AI) Algorithms) for autonomics that operate on network-wide views is called the GANA Knowledge Plane (GANA KP). The three key Functional Blocks of the GANA KP are summarized below (more details are found in the ETSI GANA standard itself (ETSI TS 103 195-2)):

- **GANA Network-Level DEs**: *Decision-making-Elements (DEs)* whose scope of input is network wide in implementing "slower control-loops" that perform policy control of lower level GANA DEs (for fast control-loops) instantiated in network nodes/elements. The Network Level DE are meant to be designed to operate the outer closed control loops on the basis of network wide views or state as input to the DEs' algorithms and logics for autonomic management and control (the "Macro-Level" autonomics). The Network-Level-DEs (Knowledge Plane DEs) can designed to run as a "micro service".
- **ONIX (Overlay Network for Information eXchange)** is a distributed scalable overlay system of federated information servers). The ONIX is useful for enabling auto-discovery of information/resources of an autonomic network via "publish/subscribe/query and find" mechanisms. DEs can make use of ONIX to discover information/context and entities (e.g. other DEs) in the network to enhance their decision making capability. The ONIX itself does not have network management and control decision logic (as DEs are the ones that exhibit decision logic for Autonomic Management & Control (AMC)).
- **MBTS (Model-Based Translation Service)** which is an intermediation layer between the GANA KP DEs and the NEs ((Network Elements)—physical or virtual)) for translating technology specific and/or vendors' specific raw data onto a common data model for use by network level DEs, based on an accepted and shared information/data model. KP DEs can be programmed to communicate commands to NEs and process NE responses in a language that is agnostic to vendor specific management protocols and technology specific management protocols that can be used to manage NEs and also policy-control their embedded "micro-level" autonomics. The MBTS translates DE commands and NE responses to the appropriate data model and communication methods understood on either side. The value the MBTS brings to network programmability is that it enables KP DEs designers to design DEs to talk a language that is agnostic to vendor specific management protocols, technology specific management protocols, and/or vendor specific data-models that can be used to manage and control NEs.

The GANA reference model combines perspectives on NE/NF embedded GANA DEs ("Micro-Level" autonomics (defined by the so-called GANA levels-1 to Level-3 illustrated in Figure 2)) and their interworking with GANA KP DEs (GANA level-4)—i.e. the "Macro-Level" autonomics realized by the GANA Knowledge Plane). The reference model also defines the responsible Functional Blocks and Reference Points that enable developers to implement autonomics software, with all perspectives combined together so as to capture the holistic picture of autonomic networking, cognitive networking and self-management design and operational principles. This ETSI GANA Framework is illustrated in Figure 2. **NOTE:** The *Four GANA Levels of abstraction of self-management functionality* at which control-loops can be introduced are defined to great detail in ETSI White Paper No.16 [1] and ETSI TS 103 195-2 [2], with arguments as to why the *three GANA Levels 2 to 4* should be the most important levels to consider when introducing autonomics in network architectures and their associated management and control architectures.

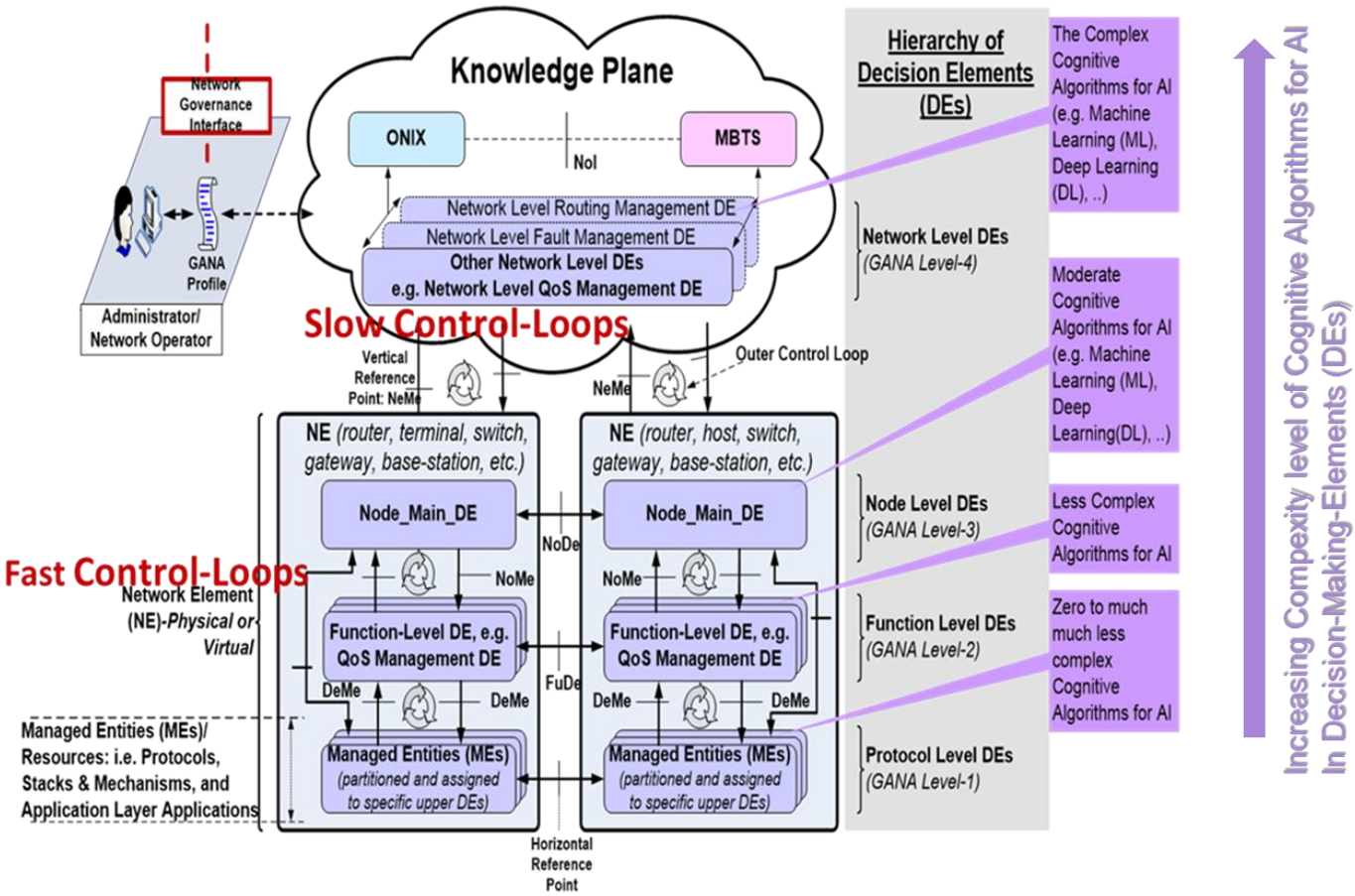

Figure 2: ***Snapshot of the GANA Reference Model and Autonomics Cognitive Algorithms for Artificial Intelligence (AI), and illustration of the notion of increasingly varying complexity of AI from within an NE up into the Knowledge Plane level***

As such (in reference to Figure 2) the ETSI GANA Standardized Framework for AMC (ETSI TS 103 195-2) defines an Intelligent Management and Control Functional Block called GANA Knowledge Plane (KP) Platform that is an integral part of Management and Control Systems for the network—as that part that provides for the space to implement complex network analytics functions performed by interworking Modularized and specialized DEs. The KP DEs run as software in the Knowledge Plane and drive *self-* operations for AMC such as self-adaptation, self-optimization, self-monitoring, self-protection, self-defense* objectives for the network and services by programmatically (re)-configuring Managed Entities (MEs) such as protocols and other configurable network resources and parameters of the network infrastructure through various means possible: e.g. through the NorthBound Interfaces available at the OSS (Operations Support Systems), Service Orchestrator, Domain Orchestrator, SDN (Software-Defined Networking) controller, EMS/NMS (Element Management System/Network Management System), NFV (Network Functions Virtualization) Orchestrator and/or MANO (Management and Orchestration) stack in general, etc. ETSI White Paper No.16 [1] discusses the details on the stakeholders impacted by GANA, e.g. the kinds of players that should play the role of suppliers of specific GANA software components.

The GANA KP consists of multiple modularized DEs. In contrast to non-modularized management systems, each DE is expected to be a module (as atomic block) and that it should address a very specific "management & control domain (scope of management/control aspects/problems)" such that it can run as a "micro service". Examples of autonomic manager elements (i.e. DEs) are: *QoS-management-DE, Security-management-DE, Mobility-management-DE, Fault-*

management-DE, Resilience & Survivability-DE, Service & Application management-DE, Forwarding-management-DE, Routing-management-DE, Monitoring-management-DE, Generalized Control Plane management-DE.

DE components of the GANA KP are "macro" autonomic managers (atomic and modular) that are logically centralized and operate on network-wide views in driving slow control loops that adaptively program and policy-control the behavior of Network Elements/Functions (NEs/NFs) while operating in "slower timescale" than similar control-loops introduced to run in NEs/NFs and operating as "fast control-loops". Macro autonomic managers (GANA KP DEs) should be complemented by "micro" Autonomic Manager components (DEs injected into NEs) that can be introduced in the Network Elements (physical or virtualized) for driving local intelligence within individual network elements to realize "fast control-loops" in network elements. Macro autonomic managers (GANA KP DEs) policy-control the "micro" autonomic managers (GANA DEs in NEs—i.e. the so-called GANA Level-2 and Level-3 in the ETSI TS 103 195-2).
The following Figure (Figure 3) provides an illustration of how a GANA Knowledge Plane Level DE can be implemented (more details on approaches to designing GANA DEs are found in ETSI TS 103 195-2), using the Security-Management-DE and its associated Managed Entities (MEs) types as illustration.

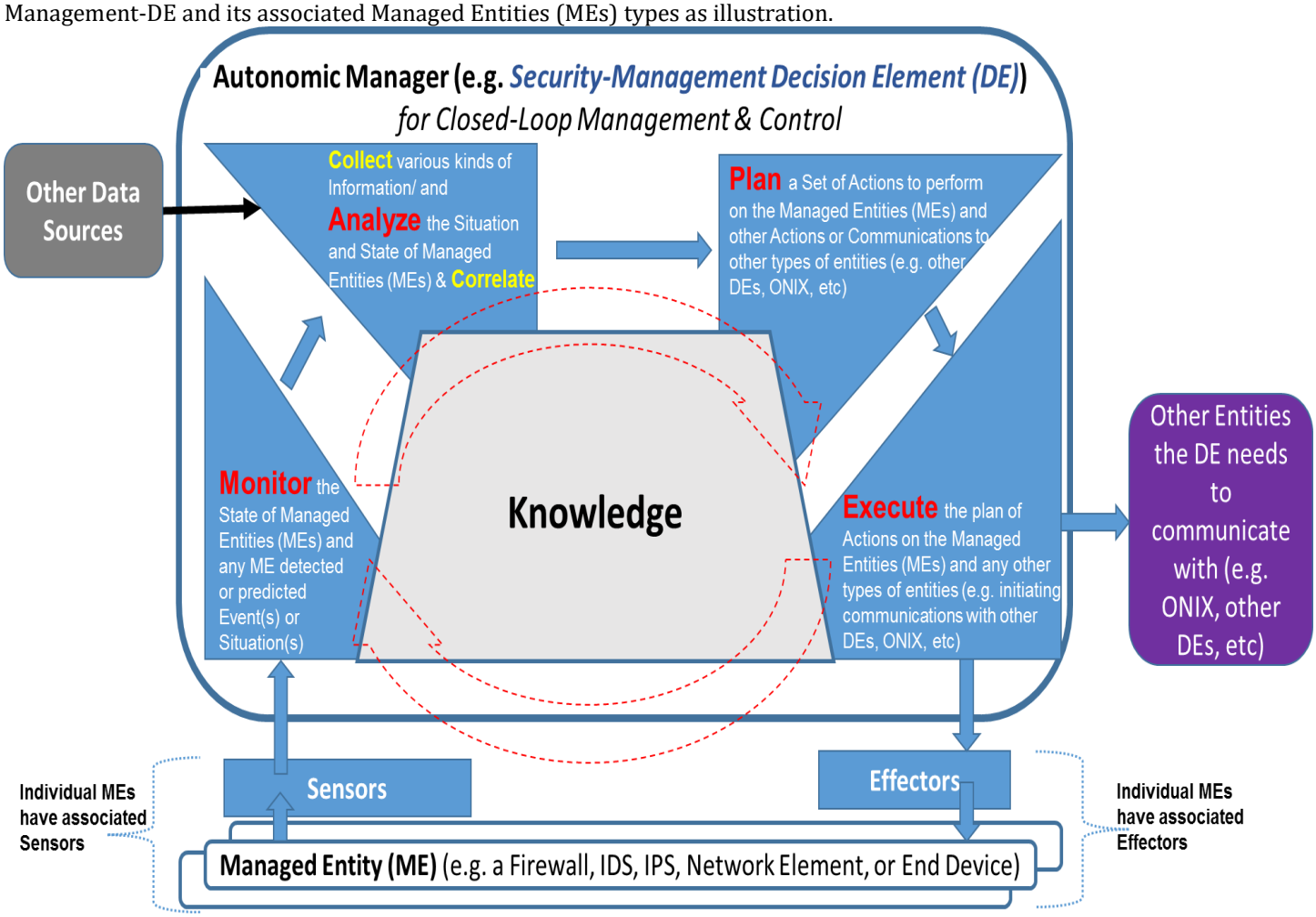

Figure 3: ***Illustration of how a GANA DE can be implemented (more details on approaches to designing GANA DEs are found in ETSI TS 103 195-2), using the Security-Management-DE and its associated Managed Entities (MEs) types as illustration***

NOTE: Today in the industry, there are already various kinds of implementations of autonomic (closed-loop) autonomic management and control components that are either integrated as part of network element/function products or as part of management and control systems associated with specific network architectures. A lot of such

components can be mapped to specific types of GANA DEs and levels of abstractions defined by ETSI TS 103 195-2 standard. Some implementations of such components fulfil the requirements on the desirable features of the specific GANA DEs they can be mapped to as implementations of GANA DEs, while some components would be a good starting point to implementing GANA DEs and can easily be further developed to fully implement desirable features of the specific GANA DEs they can be mapped to. To give a few examples of DEs implementations and also pointers to studies on how AI can be used to implement network and service management & control intelligence software such as *Cognitive GANA DEs*:

- White Paper No.6 [35] provides illustration of implementation of Security Management DEs (expected to be powered by AI algorithms)
- White Paper No.1 [5] and White Paper No.2 [19] provide detailed implementations of GANA DEs for the RAN, as SON (Self-Organizing Network) Functions, including the role of AI in the SON/DE Modules. There are more sources such as products implementations datasheets and also in the literature on AI application in SON functions (DEs for RAN)
- Example Implementation of a Routing Management DE that dynamically adapts routing protocols and mechanisms as its Managed Entities (MEs) and can be powered by AI is demonstrated in [37] [38].
- There are now various sources in literature and products implementations that demonstrate the way AI can be used to implement network and service management & control intelligence software such as *Cognitive GANA DEs*. For example, [36] and [39] provide such insights.

3.3. Stakeholders that should play certain roles in the context of Test and Certification of GANA Cognitive DEs and GANA Knowledge Planes for AMC

White Paper No.4 [23] (downloadable at https://intwiki.etsi.org/index.php?title=Accepted_PoC_proposals) provides insights on *Test and Certification, Regulation, Legislation and Auditing of Cognitive DEs (AI / ML Models)*. Therefore, White Paper No.4 [23] defines the Stakeholders that should play certain roles in the context of Test and Certification of GANA Cognitive DEs and GANA Knowledge Planes for AMC. Other stakeholders defined in White Paper No.4 [23] include the following types (as well as related aspects covered in chapter 4 and subsequent chapters of the White Paper No.4 [23]):

- *Design, Development and Production of Cognitive AMC_DE(s) (AI / ML Models);*
- *CSP's internal Stakeholders involved in AI Strategy, AMC_DE (AI/ML Model) Procurement and Deployment Management.*

The following diagrams are extracts from White Paper No.4[23] that illustrate the aspect of Testing and Certification for Cognitive GANA DEs (for both distributed DEs (dDEs) and centralized DEs (cDEs)). "dDEs" are the DEs implemented to operate in Network Elements/Functions (NEs/NFs) to realize the "*fast control-loops*" and also to implement distributed control loops within the network infrastructure through horizontal interactions with other dDEs across a network scope in form of "in-network management" in the case when implementer(s) of autonomics chose to implement the autonomics through a distributed algorithm than using a centralized approach. The "cDEs" are the centralized DEs implemented in the GANA Knowledge Plane (KP) level to run and operate within the KP platform, and are meant to implement autonomic management and control algorithms in a centralized approach while at the same time being responsible of policy-controlling the dDEs in NEs/NFs of the network segment(s) the KP DEs are responsible for.

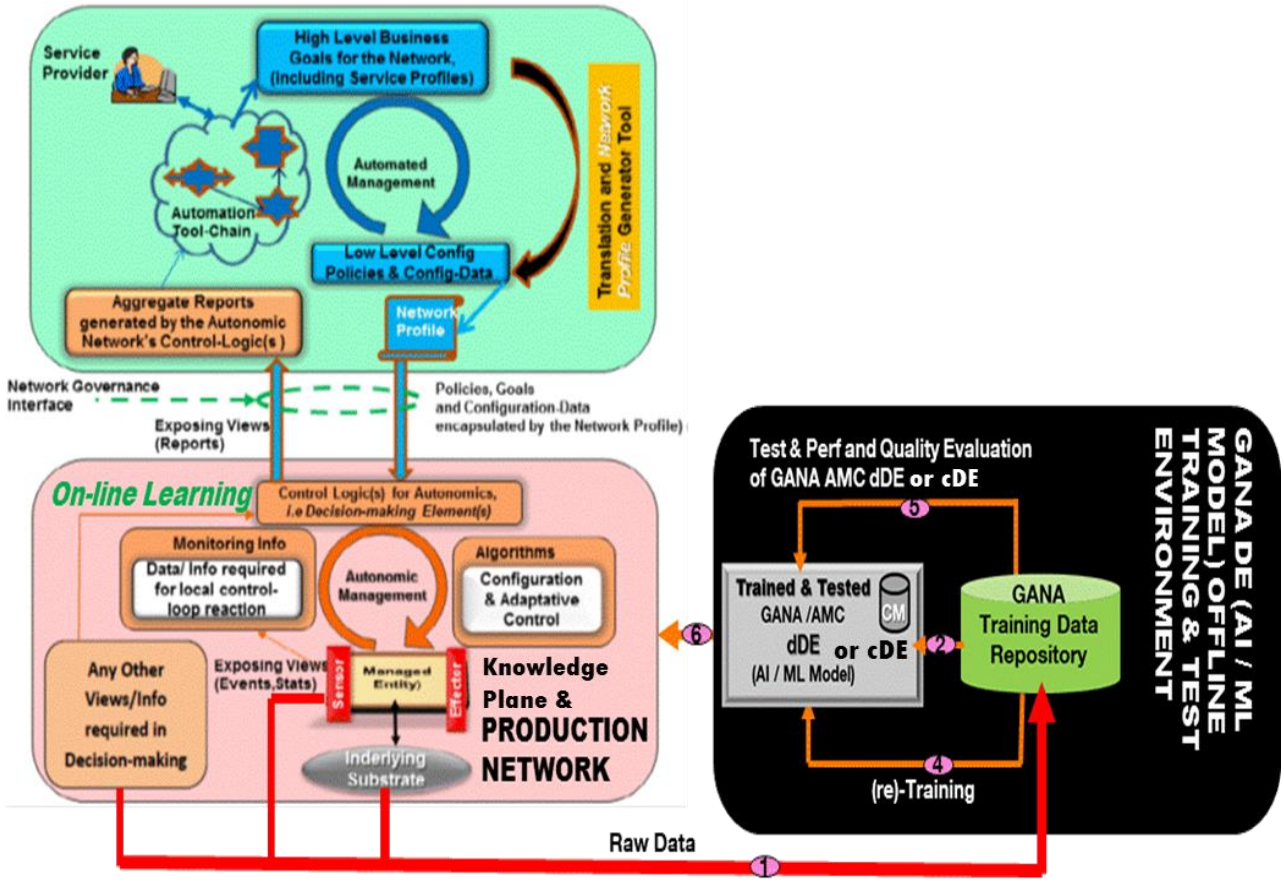

Figure 4: ***ETSI GANA AMC paradigm powered by AI / ML capabilities***

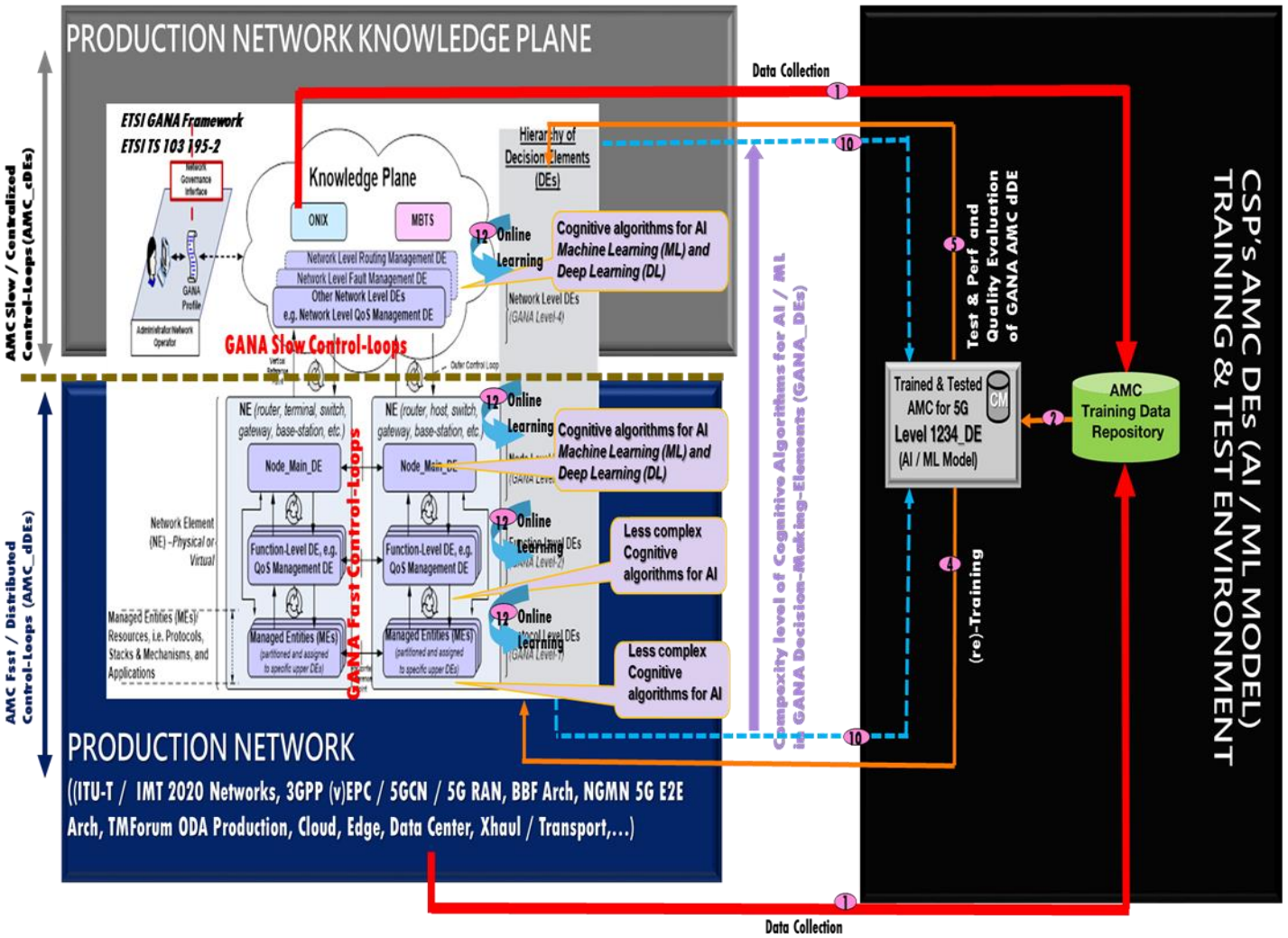

Figure 5: ***ETSI GANA as Multi-Layer Artificial (AI) Framework for Implementing AI Models for AMC of Networks and Services***

NOTE: While the testing aspects covered in White Paper No.4 [23] are general aspects, such aspects are to be complemented by other testing aspects that are specific to testing GANA Functional Blocks for autonomics (such as Conformance Testing aspects described later in chapter 9).

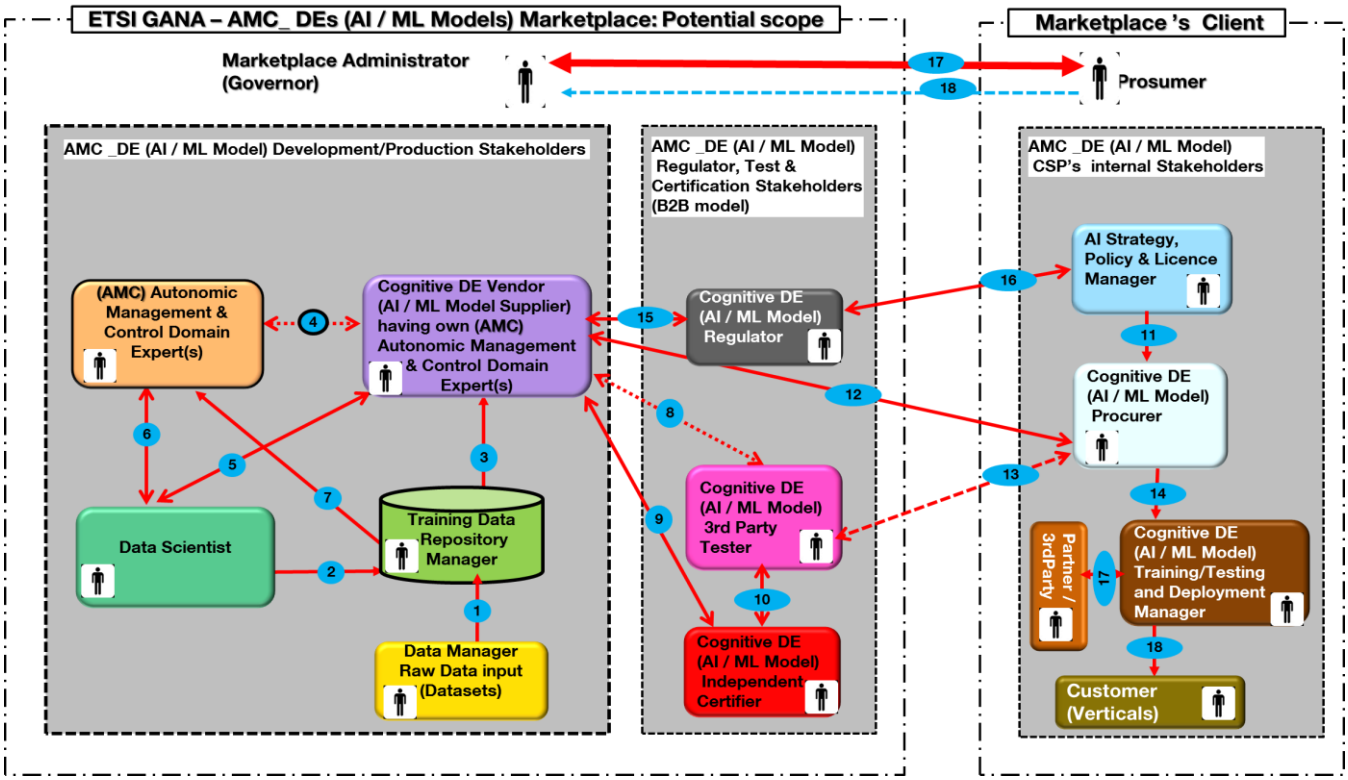

Figure 6: ***Concept of GANA DEs (AI / ML Models) Marketplace in the context of AI Models Test and Certification and associated stakeholders (more details in White Paper No.4)***

There are *Technical, Operational and Regulation challenges* linked to the various environments involved in the lifecycle of an AI Model in the form of a cognitive DE: *Development, Deployment, Production of AI exhibiting Systems for Autonomic and Cognitive Management and Control of Networks and Services* as Testing of AI Models is becoming crucial. Various challenges need to be addressed in the lifecycle of such AI exhibiting systems with respect to the following systems engineering aspects "***Development – (re)Training – Testing –Verification / Certification – Deployment- Execution***", including the roles played by the following stakeholders: *Autonomic Network Management & Control Domain Expert(s); AI Models Vendor (AI Models Supplier); Data Scientist; Data Owner; Data Manager, 3rd Party Tester, Regulator, Auditor, Independent Certifier.*

Figure 6 depicts the interactions between involved stakeholders, their respective roles and responsibility demarcation cross different environments: AI Models' Vendor / Supplier one, CSP's ones (Training & Testing, Production). Other stakeholders (CSP's internal Stakeholders) may be involved in AI Strategy, (AI/ML Model) Procurement and Deployment Management to mention a few but not shown in Figure 7.

Associated to the three environments, Figure 7 indicates respective status of an AI / ML Model, namely: 1) "Procurable" AI / ML Model; 2) "Trained & Tested" AI / ML Model; 3), "Deployable" AI / ML Model. Concerning an Online "Procurable" AI / ML Model, which is exposed to real data, and having the ability to learn, Figure 6 indicates the inherent continuous Testing and Continuous Validation /Certification which is very challenging compared to an Offline AI / ML Model which is the main illustration the Figure 7 proposes.

Regulation is mainly dealing with Accountability, Auditability and Ethical related aspects of the AI /ML Model whatever its status is.

Introducing the AI-Support System (AI-SS) concept using GANA concepts for AI-powered AMC:

- Figure 7 is an updated version of the corresponding figure in White Paper No.4 [23] that describes AI Model Life Cycle Management process (*Development, Training, Testing, Certification, Deployment*) with associated three main stakeholders: *AI Regulator / Auditor, 3rd Party AI Model Tester, AI Model-dependent Certifier*. Each of the three mentioned stakeholder provides a related support (Methodology, Function, Process, assessment Metrics, etc.) as a part of what we named an **AI Support System (AI-SS)** as depicted in the new Figure 7 below. The doted rectangle shows the demarcation of the AI-SS's thee parts. Each of the three components of the AI-SS could be provided /offered as a service (as we name below):
  - *Regulation as A Service (**Reg**aaS)*
  - *Testing as a Service (**Tes**aaS)*
  - *Certification as a Service (**Cert**aaS)*

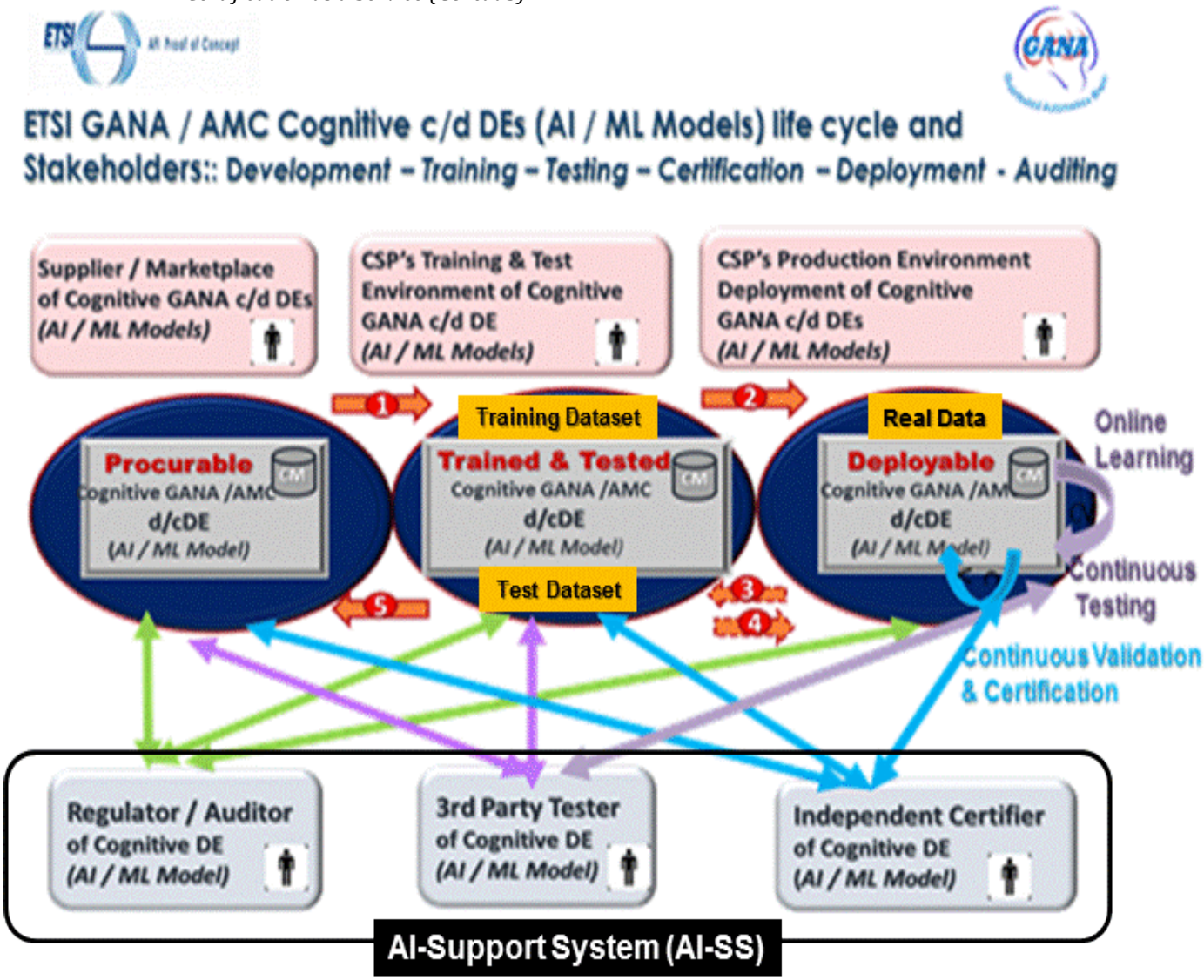

Figure 7: ***GANA AMC Cognitive c/d DEs (AI / ML Models) life cycle and Stakeholders: Development – Training – Testing – Certification – Deployment - Auditing (more details in White Paper No.4 [23])***

NOTE: Figure 7 is considered as ETSI TC INT AI Model Life Cycle Management and AI Support System (AI-SS).

3.4. General Approach to Designing the Test Systems/Components for Testing Autonomics AI Models, and Challenges in AI Models' ability to cope with 5G Network Dynamics that need to be taken into consideration

Some challenges in AI Models pertain to how they can cope with the dynamics of the network in which they are designed to operate, and such challenges need to be taken into consideration when designing the Test Systems/Components for Testing Autonomics AI Models. The following are some general aspects that need to be taken into consideration by both, the developer of an AI Model and the Tester of an AI Model:

- *Time it may take for an AI Model for autonomic management and control to meaningfully be applicable and be able to keep pace with dynamics of the network*
- *Time it may take for an AI Model embedded in a Test Component/System to meaningfully be applicable and be able to keep pace with dynamics of the network*
- *Verdicts Passing in Testing AI Models, and How Suppliers of AI Models (e.g. Cognitive GANA DEs) to be Tested and Certified can produce "Assertions/Claims Specifications of Measurable Metrics/KPIs and certain observable and verifiable outputs" on what the AI Model can achieve under certain conditions during its operation as "assertions/claims" by the supplier of the AI Model*
- *Idea of using the concept of a "Qualified Automated Test Component(s) or System" that exhibits best quality AI capabilities, in testing comparable capabilities of AI Component(s)/System Under Test;*

NOTE: While the testing aspects covered in this section are general aspects, such aspects are to be complemented by other testing aspects that are specific to testing GANA Functional Blocks for autonomics (such those aspects in chapter 5 and chapter 6 and Conformance Testing aspects described later in chapter 9).

3.5. Illustrations of Types of Standardizable Metrics that should be target for Measurements and Assessments in Testing and Certification of AI Models of Autonomic Components/Systems

There are various metrics that can be used for assessment and differentiation of AI Models such as Machine Learning Models as described in [31]. There is the need to leverage such metrics and expand them with other metrics that can be standardized across comparable AI Models for a specific domain of AI application. For example, the following are some examples of the Standardizable Metrics for Measurements and Assessments in Testing and Certification of AI Models of Autonomic Components/Systems (and such metrics need to be complemented with other relevant metrics that can also be standardized):

- *Stability of the AI Model*
- *Speed of Learning of the AI Model*
- *Speed of Decision-making cycle of the AI Model after receiving triggering inputs*
- *Speed of Convergence of multiple interacting AI Models/components in a larger AI System*
- *Quality of Decision-Making of the AI Model*

This list of *Standardizable Metrics for Measurements and Assessments in Testing and Certification of AI Models* of Autonomic Components/Systems is subject to possible expansion by the community and the Metrics are expected to be further detailed in one of the deliverables of the newly launched work item in ETSI on AI in Test Systems and Testing AI Models (https://portal.etsi.org/webapp/WorkProgram/Report_WorkItem.asp?WKI_ID=58442).

4. European Commission White Paper COM (2020) Key Recommendations on AI, and the Mapping of TC INT's *AI Support System "AI-SS" concept* with EC's Recommendations

The European Commission White Paper published February 19th 2020 COM (2020) [26] mentions many times the need for: ***1)*** *A Regulatory Framework;* ***2)*** *Creation of an AI Testing Center;* ***3)*** *Creation of A Certification Center*. We tried to translate it in the proposed Figure 9 in such way that it could be easily mapped with the TC INT proposed AI-Support System (AI-SS) concept.

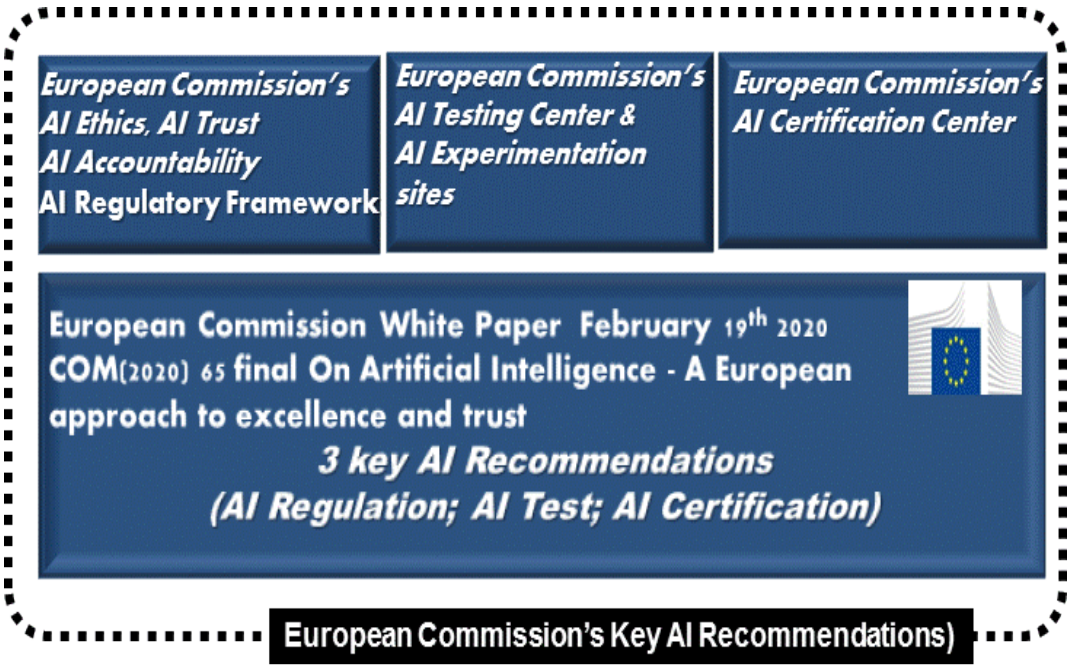

Figure 8: ***ETSI TC INT translation of the European Commission White Paper of February 19th 2020 COM (2020) Key Recommendations: 1) A Regulatory Framework, 2) Creation of an AI Testing Center; 3) Creation of A Certification Center into a diagram***

When looking at the mapping of the TC INT's *AI Support System "AI-SS" concept* to the *EC's AI Regulatory, Testing & Certification Recommendations*, we can notice there is a good mapping between Figure 7 and Figure 8 as shown in the Table 1 below (which is a first mapping attempt). Therefore, the Testing work launched in ETSI TC INT should be further aligned with the European Commission's white paper and made to operationalize those three main recommendations.

Table 1: TC INT AI Support System "AI-SS" mapping with EC's AI Regulatory, Testing & Certification Recommendations

TC INT AI Support System (AI_SS)	
TC INT AI Model LCM process (Development, Training, Testing, Certification, Deployment) and associated Stakeholders delivering a sort of AI Support System (AI-SS)	**European Commission White Paper published on February 19th 2020 COM (2020) 65 final On Artificial Intelligence- A European approach to excellence and trust**

Regulator / Auditor of AI Models	*AI Ethics, AI Trust* *AI Accountability* ***AI Regulatory Framework*** *1) A risk-based approach is important to help ensure that the regulatory intervention is proportionate* *2) Could prescribe accurate records regarding the data set used to train and test the AI systems (C. Scope of Regulatory Framework; D) Requirements)*
3rd Party AI Models Tester	***Create AI Testing Center*** *&* *Create AI Experimentation sites to support the development and subsequent deployment of novel AI applications* *Make sure that Europe equips itself progressively with the capacity needed for testing and certification of AI-enabled products and services (H. Governance)* *The Commission will facilitate the creation of excellence and testing centers that can combine European, national and private investments, possibly including a new legal instrument (4/B/Article 2)*
Independent Certifier of AI Models	***Create AI Certification Center:*** *to support the development and subsequent deployment of novel AI applications* *Make sure that Europe equips itself progressively with the capacity needed for testing and certification of AI-enabled products and services* *to avoid fragmentation of responsibilities, increase capacity in Member States* *Advising on standardization activity as well as on certification*

Figure 9 shows how ETSI TC INT AI-Support System (AI-SS) as depicted in Figure 7 maps with the three EC's AI recommendations categories as depicted in Figure 8. Figure 9 is the result of superposing Figure 7 and Figure 8.

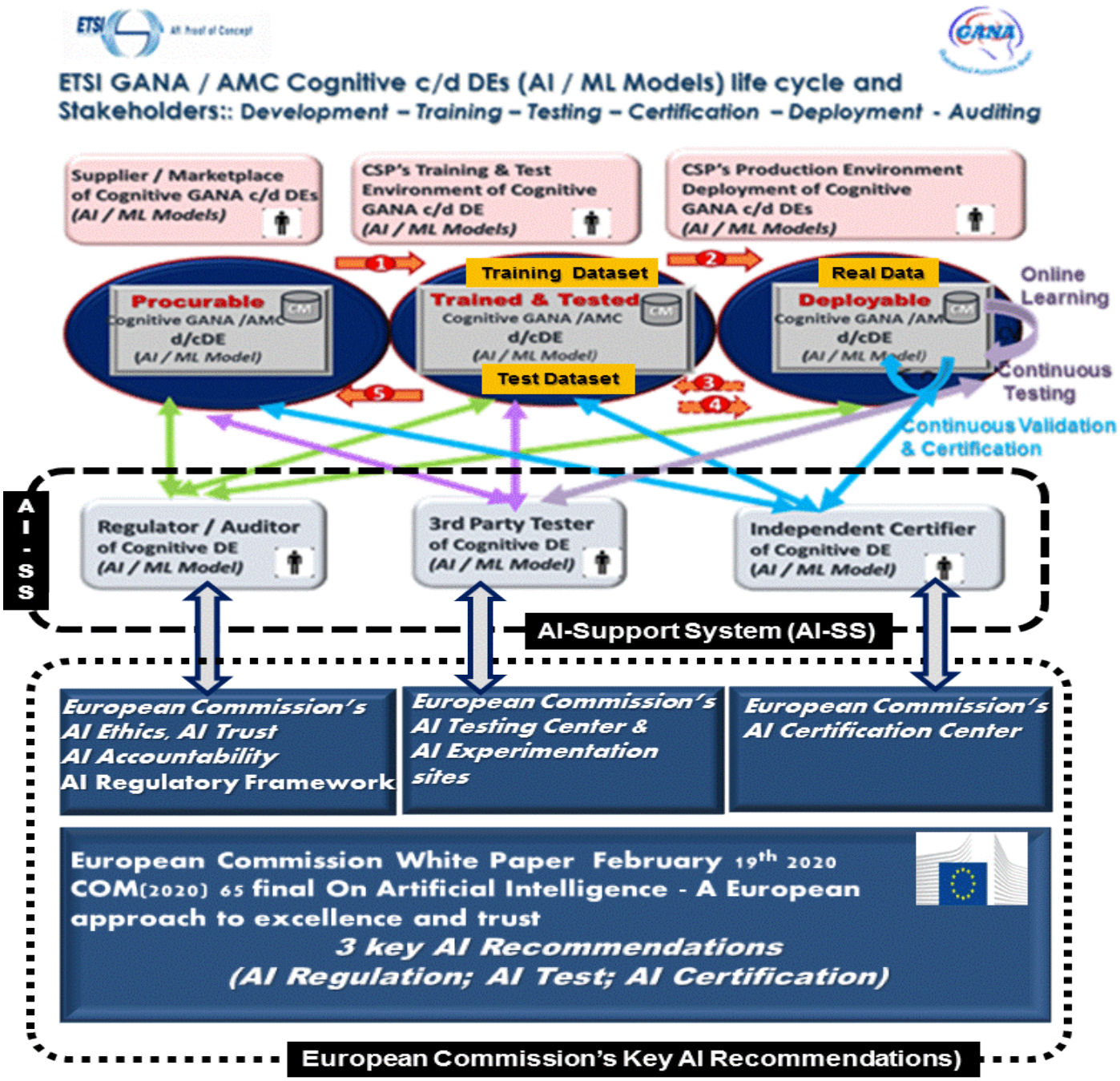

Figure 9: Mapping of TC INT AI Support System "AI-SS" with EC's AI Regulatory, Testing and Certification Recommendations

5. Capabilities of Softwell Performance AB's AI-empowered Performance Test Solutions that help address Challenges in Performance Testing of 5G Networks, Products and Slice Services

This chapter provides perspectives on the benefits of AI in Performance Test Systems. Softwell Performance AB has developed and continues to develop solutions that address the various aspects highlighted in this chapter—aspects that are very much relevant to AI-empowered Performance Test Solutions that help address Challenges in Performance Testing of 5G Networks, Products and Slice Services.

5.1. Why does AI fit especially well in performance testing?

Over 95% of all performance measurements are regression tests! Implications:

- Regression tests of system performance is a learning process in the behavior of a tested system! *An ideal case of AI for computer learning in a performance test tool.*
- Frequent performance tests produce huge amounts of measurement data! *This is also an ideal case for AI, to analyze collected measurement data, to draw long term conclusions of performance characteristics.*

5.2 What can be improved by AI in performance measurement tools?

The following aspects provide insights on the value of AI in performance measurement tools. Figure 10 presents the value of AI in Performance Test Systems.

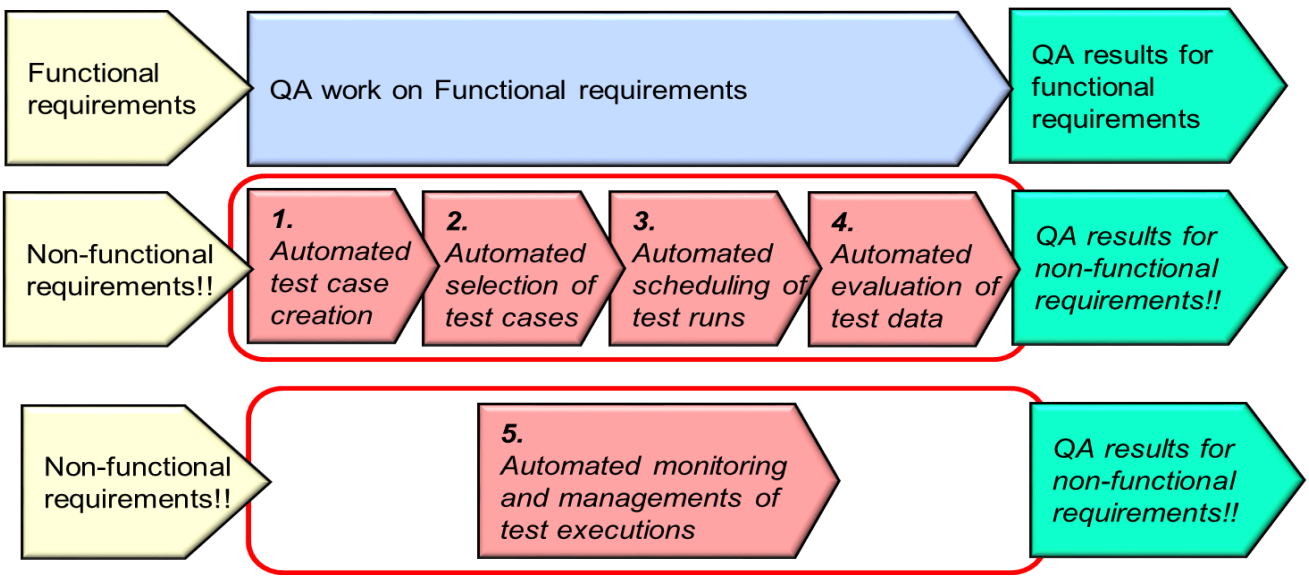

Figure 10: ***The value of AI in Performance Test Systems***

1. Creating performance test cases must be simplified. Creation of test cases is a complex operation that takes time and manpower. Test cases can be created directly from performance requirements. AI in performance test tools could solve this.
2. Selection of test cases for a test run should be automated. This is manual work today that can be eliminated. Automated selection of the most important test cases can be done with AI in performance test tools.

3. Automated performance tests should be executed for every build. This is not done today. Only the most important performance test cases are selected and executed. AI in performance test tools could solve this.
4. Automated evaluation of performance measurement results: This is manual work today that can be eliminated. Evaluation of measurement results can be done fast and precise based on performance requirements with AI in performance test tools.
5. Automated monitoring of performance tests that run for long time. Trend analysis of captured data can be done during execution. Test execution can be aborted when it is pointless to continue—thanks to using AI in performance test tools.

6. Convergence of Performance Testing with Functional Testing in Testing AI Models: Proposal on a Performance Test Framework for the GANA DEs, a perspective by Softwell Performance AB

6.1 How does Adaptive Load Control work?

The Adaptive Load Control is a process that manages the actual load by dynamically changing the traffic rates based on measured conditions reported by probes on the SUT. The technique enables the performance measurement tool to execute an explorative performance tests fully autonomously. Based on a requested condition on the SUT stated in the test case Adaptive Load Control will adjust the load until it the requested condition is established.

Adaptive Load Control is based on probes managed by MBC and running on the SUT that report measured conditions to the Adaptive Load Control process that in turn adjusts the generated load until the requested conditions apply. Figure 11 presents the approach on *How Adaptive Load Control works in Performance Testing*.

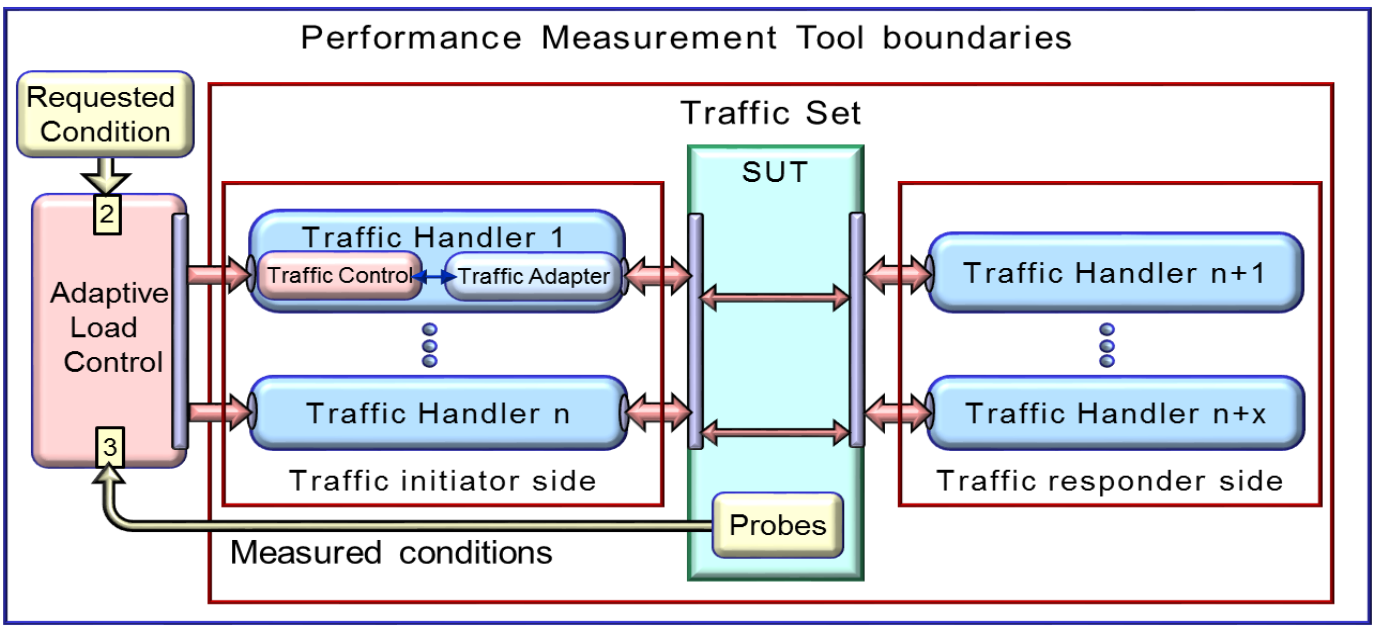

Figure 11: ***How Adaptive Load Control works in Performance Testing***

Here is an example of a performance test with Adaptive Load Control where the task is to find the processing capacity of the SUT at 80% CPU load or 20% CPU idle. The traffic rate is changed every 10th second based on current CPU load

reported by a probe, automatically downloaded by the test tool. The target is found when the CPU load is in the range 79 – 81 percent or CPU idle is in the range 19 – 21 percent. Figure 12 presents an example of performance tests with Adaptive Load Control.

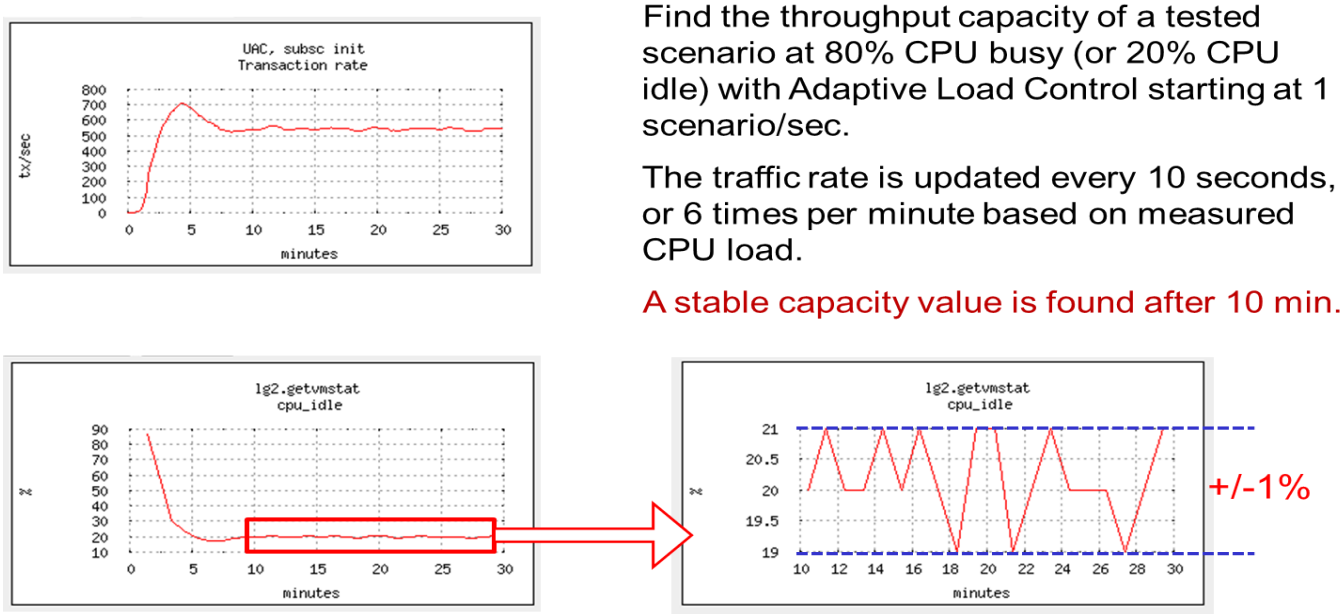

Figure 12: ***An example of performance tests with Adaptive Load Control***

In reference to the figure below (Figure 13), there are several advantages with this technique, such as the tested conditions are not related to the actual testbed of the SUT. This means that the test case can be reused on differently sized SUTs without changing the test case scripts, i.e. 80% CPU load is the same condition on any SUT regardless of the size.

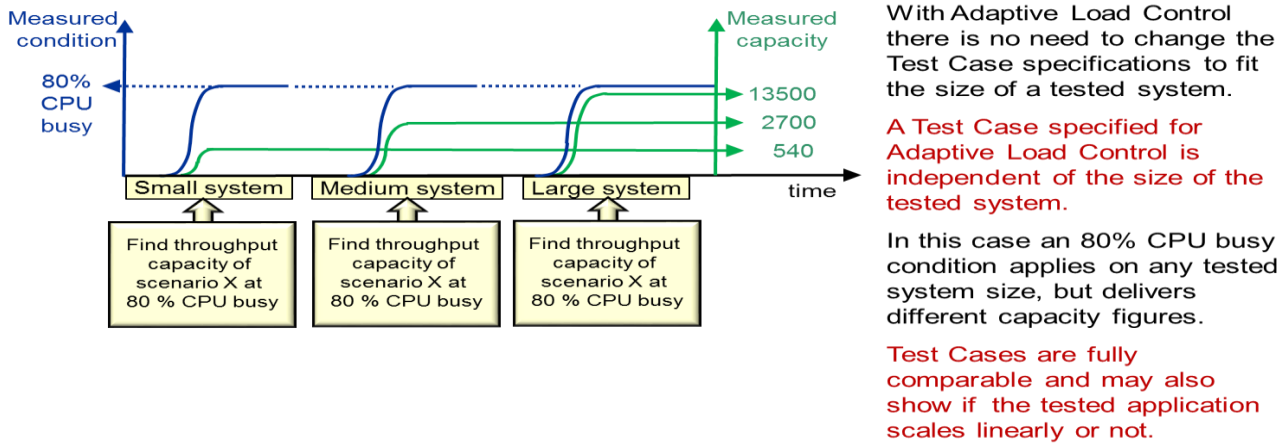

Figure 13: A Capacity Test Case can be re-used on differently sized SUTs

Another advantage with Adaptive Load Control, is the test tool can maintain the requested system condition for any amount of time. This means that a test case searching the capacity to deliver the SUT's services can be seamlessly changed to a stability and availability measurement by just extending the test time. Furthermore, running a stability test with Adaptive Load Control improves the visibility of stability and availability problems. For example a resource

problem in the tested SUT that decreases the processing capacity will be visible as a decresing trend. Figure 14 illustrates how *Adaptive Load Control improves visibility of stability problems*.

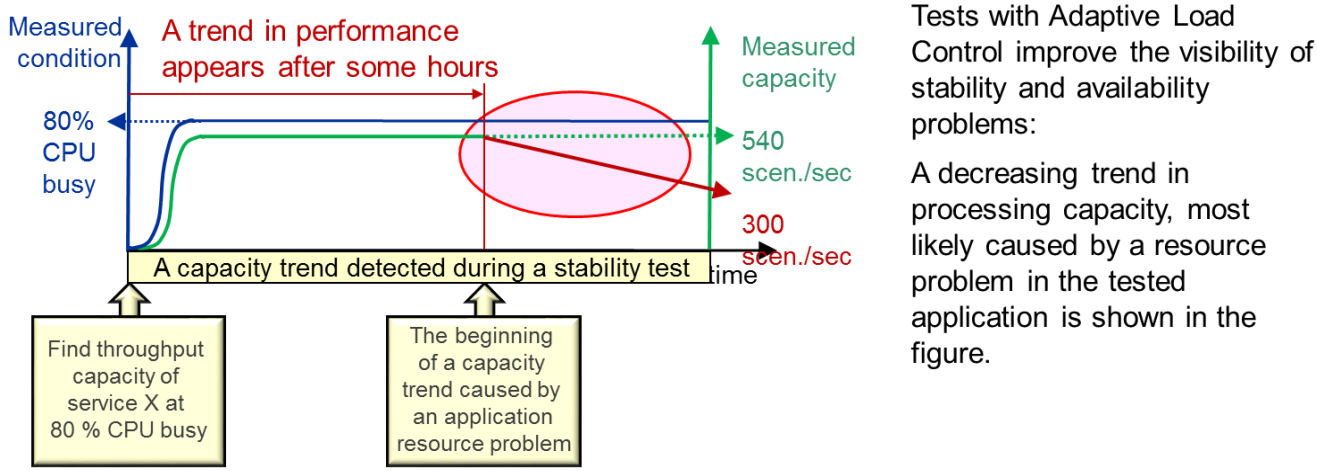

Figure 14: ***Adaptive Load Control improves visibility of stability problems***

6.2 A Proposal for a Performance Measurement Model for GANA

6.2.1 Overview

NOTE: The Performance Measurement Model for GANA and the ideas described in this whole chapter 6 of this White Paper should be considered as part of the desired Generic Test Framework for GANA Autonomics described later on chapter 9.

The whole idea expressed in this chapter 6 of this White Paper is in our opinion the first time execution of function test cases require a system performance test tool.

The proposed idea to use Adaptive Load Control to trigger decisions made by a Function DE to an ME and the Node DE follows the same principles. Conditions that will trigger a DE to act are specified in a script called DE trigger conditions. **NOTE:** Trigger conditions of a DE shall be derived from the following interfaces of a DE implementation: *DE's E_I Interface; DE's ME2DE_SIR_I Interface; DE's Other_Int_I Interface; DE2DE_I Interface*(refer to ETSI TS 103 195-2 and Section 9.2.3 of this White Paper for the structural model of a DE). Depending on what conditions will trigger a DE action, such as a load condition, a traffic mix condition or other the Adaptive Load Control will change the conditions on the SUT, until an expected DE action is triggered.
A DE action can be captured in three different ways:

1. In some cases the triggered action can be measured or confirmed in the Traffic Handlers (External events).
2. In some cases the triggered action can be confirmed by a Probe on the SUT.
3. In some cases the triggered action can be captured by tracing messages sent between an ME and its associated Function-Level DE by accessing the "*Effector*" Interface of the ME.

A DE test case may have several trigger conditions that shall be confirmed, where for instance the Node DE or the Function DE will be triggered several times. Probes measure some conditions that need to be measured as required by the test case(s) and provide results of the measurements. Figure 15 presents an *example of an approach on How Adaptive Load Control can be applied on GANA*. **NOTE**: In Figure 15, the "*Probes*" indicated as part of the ME correspond to "***Sensors***" exposed to the associated DE of the ME on the model of a Control-Loop structure in Figure 3, though Figure 15 is refering to a Control-Loop structure within an NE/NF. That means that the "*Probes*" (sensors) may be implemented outside of the ME and fed with monitoring data and events by the ME. In Figure 15, the "*DE Trace*"

indicated as part of the ME correspond to "***Effectors***" exposed to the associated DE of the ME on the model of a Control-Loop structure in Figure 3, though Figure 15 is refering to a Control-Loop structure within an NE/NF.

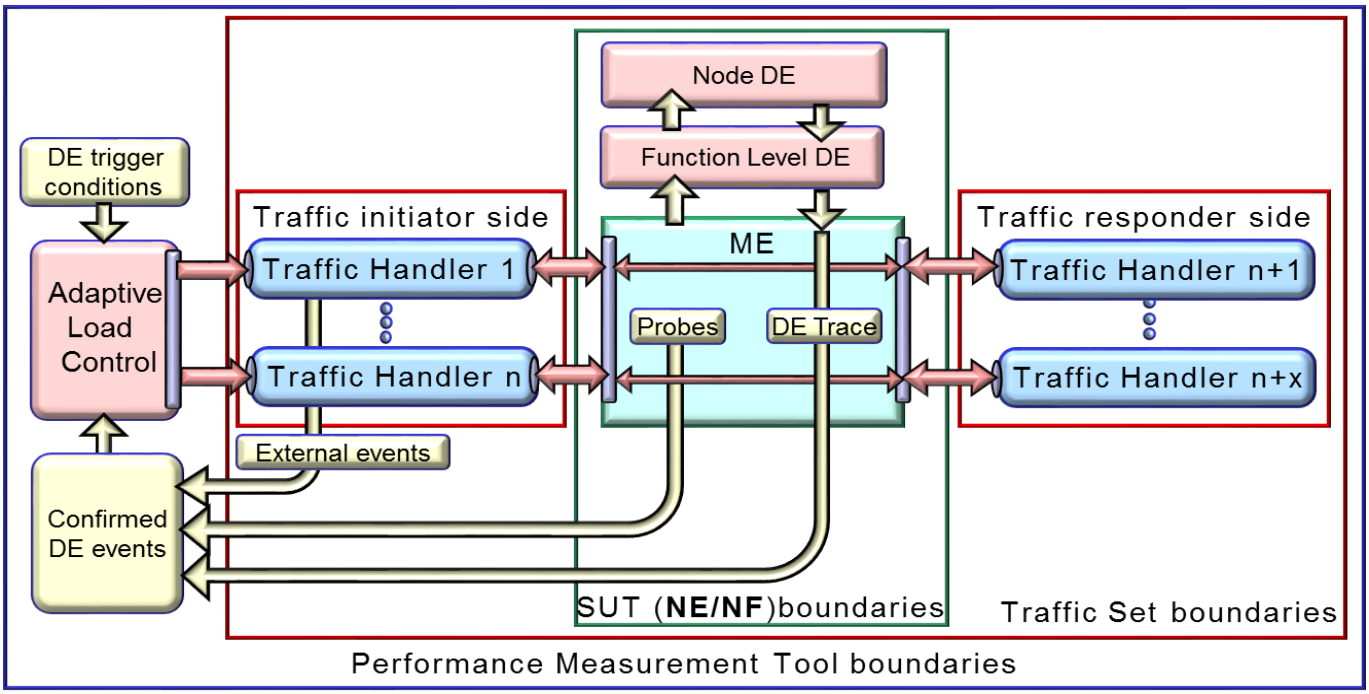

Figure 15: ***How Adaptive Load Control can be applied on GANA***

NOTE: The following aspects should be considered as part of the performance testing framework for the GANA model DEs:

- The Functional Tests performed for DEs (e.g. conformance tests described later in Chapter 9 of this White Paper) can be re-used in Performance Testing of DEs or it may be possible to perform Combined Functional and Performance Testing of the DEs. Testing AI models such as cognitive DEs, is a new area for performance measurements. This is because the test object is not the traditional SUT (in this case an ME), but the DE managing an ME to maintain specified goals. Seen from the ME there are both performance requirements and functional requirements:
    1. Performance requirements must be specified for a DE, such as expected reaction time to a triggering condition in the associated ME.
    2. Functional requirements must be specified for a DE, such as quality of decisions related to a stated SLA (Service Level Agreement) or other goals managed by a DE.
- As explained earlier, a way to intercept DE Outputs by the Test System is required during the Testing. DE inputs (triggers) are mostly synthetically generated by the Test System but some inputs to DE under test could be passively accessed by the Test System if they are generated or originating from components that the Test System leverages for its Test Cases for testing a DE(s). However, there are several ways of measuring any improvement in service delivery related to a DE action.

- There are two possibilities that the Tester may be faced with:
    1. The Tester relies on Assertions/Claims specified by the DE supplier/vendor concerning the DE's AMC targets (targets for *self-* operations*, such as dynamic self-optimization of ME configurations to meet certain objectives) when deployed in its target environment of operation (e.g. as a loadable module or "agent"), so that the Tester, in designing Test Cases for the DE(s), is driven by the consideration of DE(s)' AMC targets in its operation in the Performance Testing since there are conditions under which the DE must strive to find optimal configurations for its MEs towards achieving a particular AMC target, as this is what is expected to be carried out continuously by every

DE on any level. The assumption is that when optimizing the configuration of their associated ME(s) the DE implementations followed the framework for achieving stability of control-loops prescribed for the GANA framework in ETSI TS 103 195-2.

2. The DE supplier/vendor may be interested in requesting the Tester to simply consider the DE under Test as a "black box" whose behavior during operation is to be learned and characterized by a Test system that has the capability to learn the DE behavior and characterize it.

- There is an advantage in considering to re-use an environment under which the NE/NF has been tested without DEs loaded into it, in the testing of the DEs that are then loaded into the NE/NF to empower it with the intelligence (while considering the specific NE/NF system capability and flexibility for DE logics to be loadable into it to drive the NE/NF).
- In Testing a DE it is necessary to consider all forms of probes or information/data coming from the local environment in which the DE takes into consideration in its decision-making process or algorithms, and that may include Cross Layer Information as discussed in ETSI TS 103 195-2 and ETSI GS AFI 002.
- The proposed approach can be applied to performance testing of a single DE or multiple DEs at the same time. Because it depends on what is the tested system in a performance test case. The tested system or the SUT (System Under test) is what the performance test tool connects to. That may in the smallest case be a single function block. From there the tested system may include more and more function blocks until a full performance test of end-to-end communication is possible. In the GANA model the tested system may in the simple case include one level 2 DE controlling one or more ME's. On the next level it may include several level 2 DE controlling multiple NE to measure "sideways" activities between level 2 DE. Next step could include vertical communication between level 2 DE and level 3 DE and finally all levels from level 4 DE via level 3 DE down to level 2 DE and down to level 1. The tested system could also contain multiple types of level 2 to level 4 DEs controlling a more complex system. The test methodology in the complex cases must be based on earlier measurements from the simple cases. Performance is always built and tested from inside out, i.e. from component level to more and more aggregated levels. This is because a system can't perform better than the weakest component and the only way to find what components set the limits is by testing aggregated levels knowing their stand-alone performance.

6.2.2 What the DE Test Developer is to rely upon, i.e. What an independent (third party) Test Case Developer is to know and rely on

- Either the DE vendor or supplier has provided the "Assertions/Claims Specification of Measurable Metrics/KPIs and certain observable and verifiable outputs" on what the DE can achieve under certain conditions (inputs) during its operation" OR the Test system is there to help the Tester "Study" the behavior of the DE by stimulating it with legitimate inputs in one case and illegitimate inputs in another case, and observe Log the DE actions over varying inputs such that the Log can then be analyzed (even by AI??) to characterize the DE
- Availability of methods by which the Test System can intercept DE events that occurred/occur on its interfaces

6.2.3 The Performance Test Case model for GANA

The Performance Test Case model must be analyzed regarding test cases usually excluded in system performance testing such as negative function test cases.

In order to create the script "DE trigger conditions" the decision model in the Function DE and the Node DE should be analyzed by a DE trigger analyzer process. The process will deliver two DE trigger condition files:

1. *The Node DE trigger conditions*
2. *The Function-Level DE trigger conditions*

The two DE trigger condition files are merged into one file, the "DE trigger conditions", by the DE trigger merger process. The the "DE trigger conditions" is the final test case description file, used by the Performance measurement/test system. **NOTE:** Trigger conditions of a DE shall be derived from the following interfaces of a DE implementation: *DE's E_I Interface; DE's ME2DE_SIR_I Interface; DE's Other_Int_I Interface; DE2DE_I Interface*(refer to

ETSI TS 103 195-2 and Section 9.2.3 of this White Paper for the structural model of a DE). Figure 16 presents the process required for delivering DE trigger condition files.

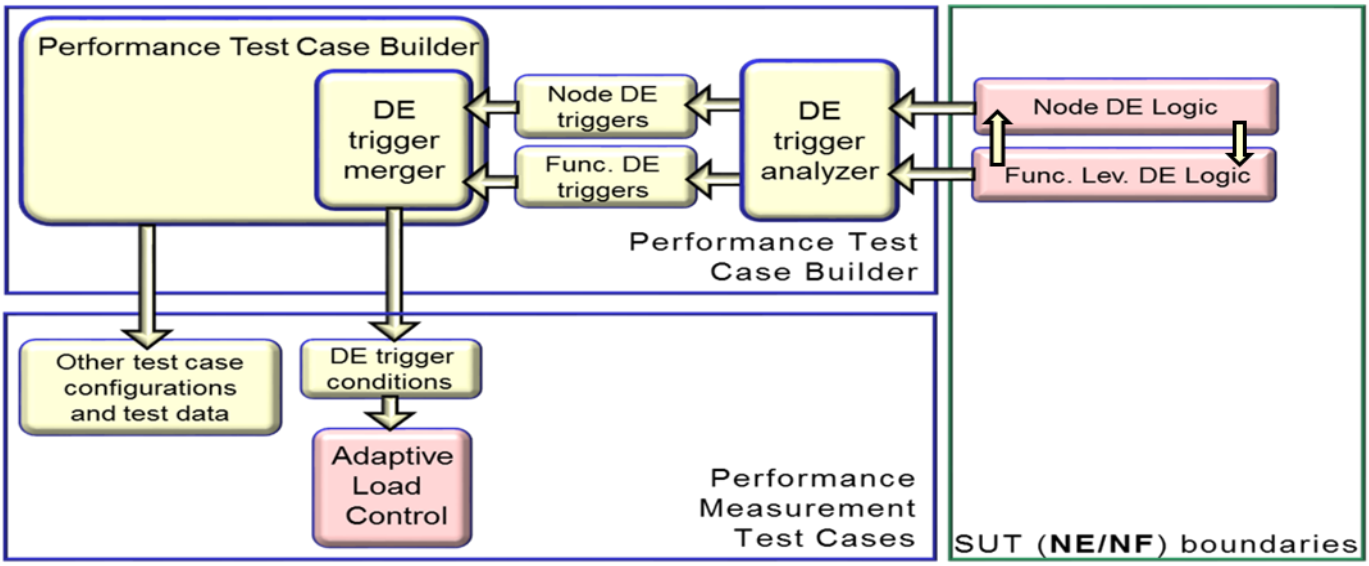

Figure 16: ***The process required for delivering DE trigger condition files***

Figure 17 presents the Scenario 1, namely: Testing a KP DE designed to autonomically manage certain Managed Entities (MEs) in a legacy NE/NF that is like a "black/closed box" and is a legacy NE/NF that does not embed GANA Levels 2&3 DE.

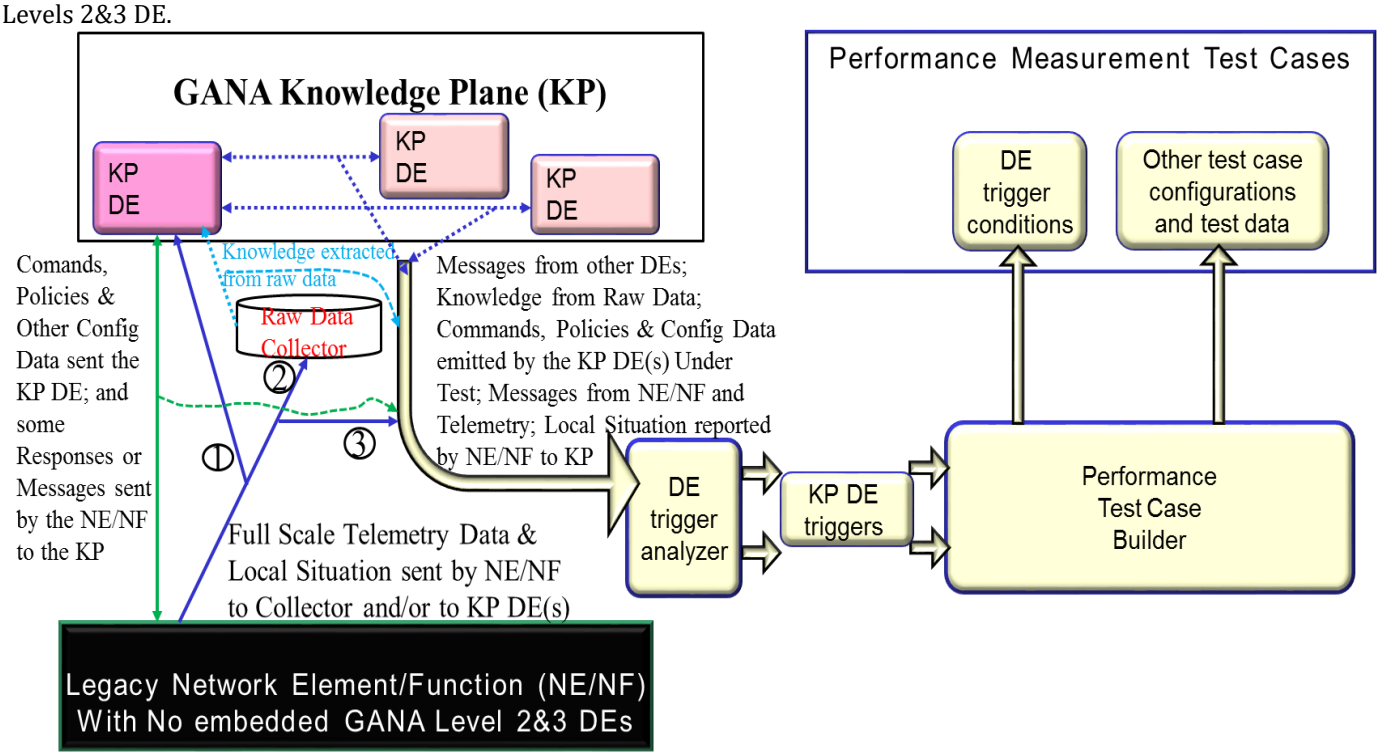

Figure 17: Scenario 1: Testing a KP DE designed to autonomically manage certain Managed Entities (ME) in a legacy NE/NF that is like a "black/closed box" and is a legacy NE/NF that does not embed GANA Levels 2&3 DE

The Scenario below considers a GANA Node implemented by a NE/NF Vendor as embedding GANA Levels 2&3 DEs and with flexibility for Tester to access the internal interactions between DEs and between DEs and MEs, or implemented as a White Box embedding Levels 2&3 DEs. Figure 18 presents Scenario 2, namely: Testing a GANA Node implemented by a NE/NF Vendor as embedding GANA Levels 2&3 DEs and with flexibility for Tester to access the internal interactions between DEs and between DEs and MEs, or implemented as a White Box embedding Levels 2&3 DEs.

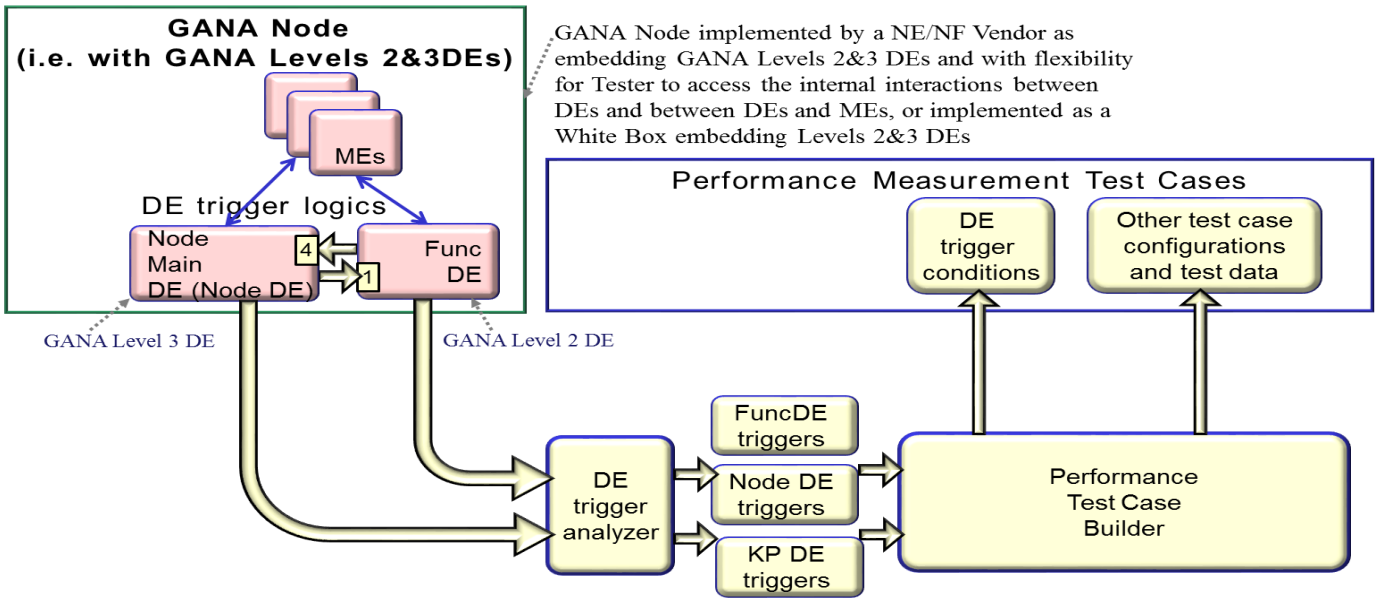

Figure 18: Scenario 2: Testing a GANA Node implemented by a NE/NF Vendor as embedding GANA Levels 2&3 DEs and with flexibility for Tester to access the internal interactions between DEs and between DEs and MEs, or implemented as a White Box embedding Levels 2&3 DEs

Figure 19 presents the Scenario 3, namely: Testing a GANA Node implemented by a NE/NF Vendor as embedding GANA Levels 2&3 DEs but as "Black Box" with NO flexibility for Tester to access the internal interactions between DEs and between DEs and MEs.

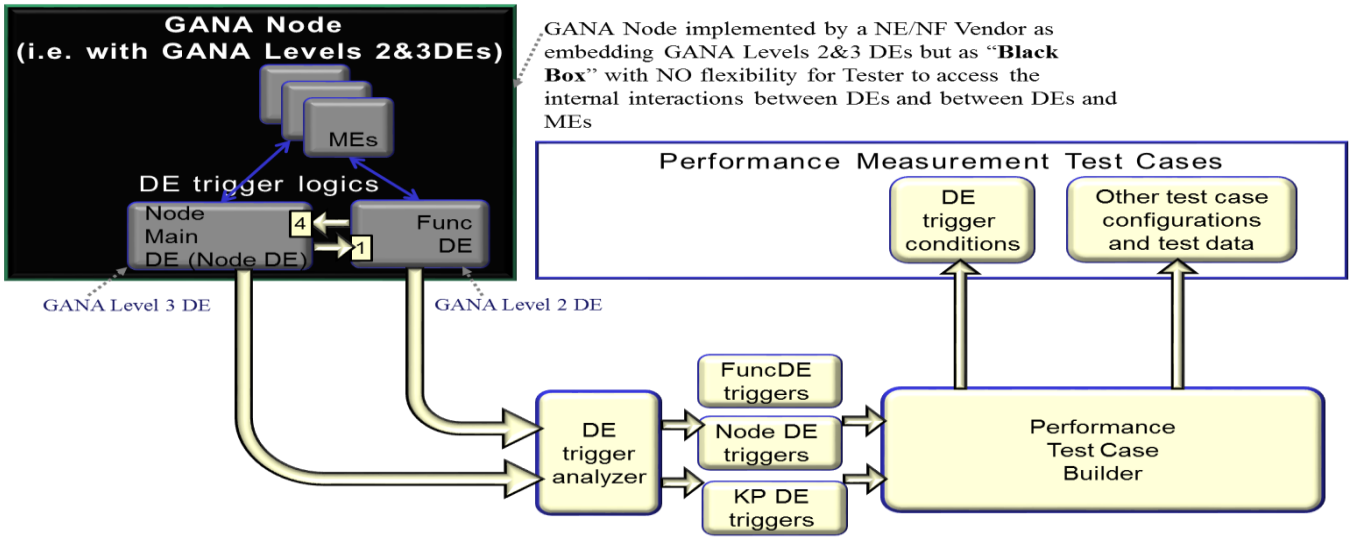

Figure 19: Scenario 3: Testing a GANA Node implemented by a NE/NF Vendor as embedding GANA Levels 2&3 DEs but as "Black Box" with NO flexibility for Tester to access the internal interactions between DEs and between DEs and MEs

The Scenario below (Figure 20) considers a GANA Node implemented by a NE/NF Vendor as embedding GANA Levels 2&3 DEs and with flexibility for Tester to access the internal interactions between DEs and between DEs and MEs, or implemented as a White Box embedding Levels 2&3 DEs.

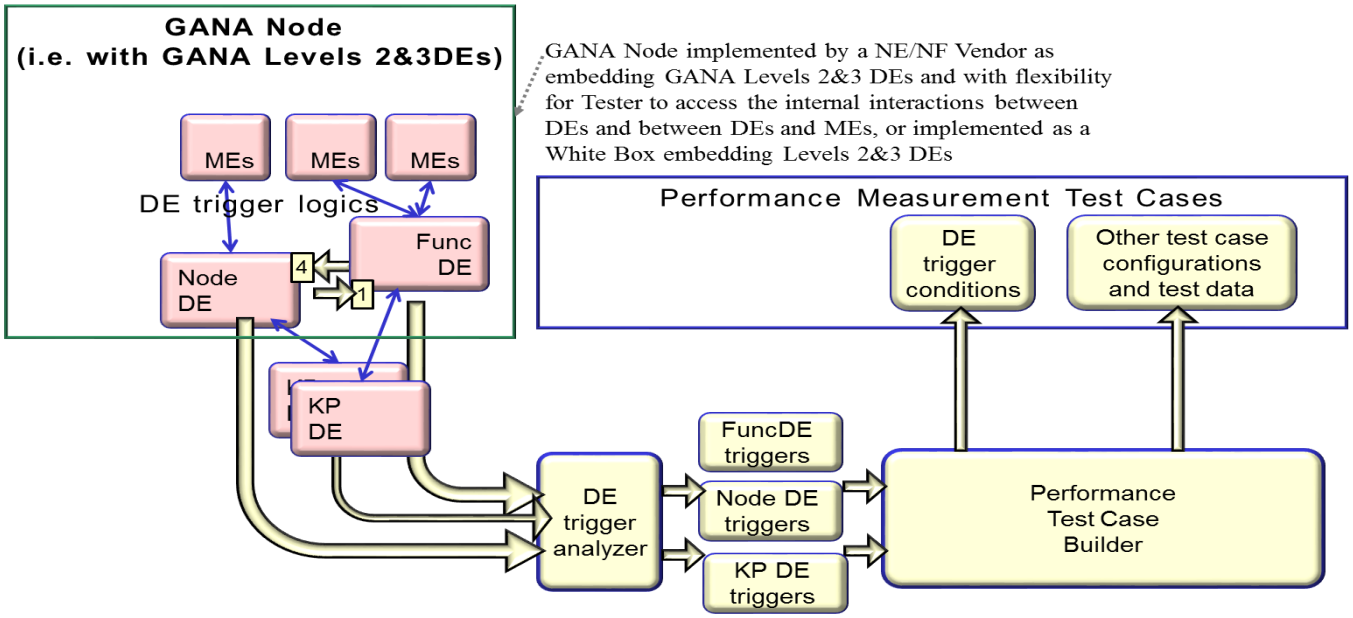

Figure 20: Scenario 4: Testing a GANA Node implemented by a NE/NF Vendor as embedding GANA Levels 2&3 DEs with the flexibility indicated, or implemented as a White Box embedding Levels 2&3 DEs, combined with simultaneously Testing KP DEs in the Test

Figure 21 presents Scenario 5, namely: Testing a KP DE designed to autonomically manage a GANA Node as an NE/NF that embeds GANA Levels 2&3 DEs.

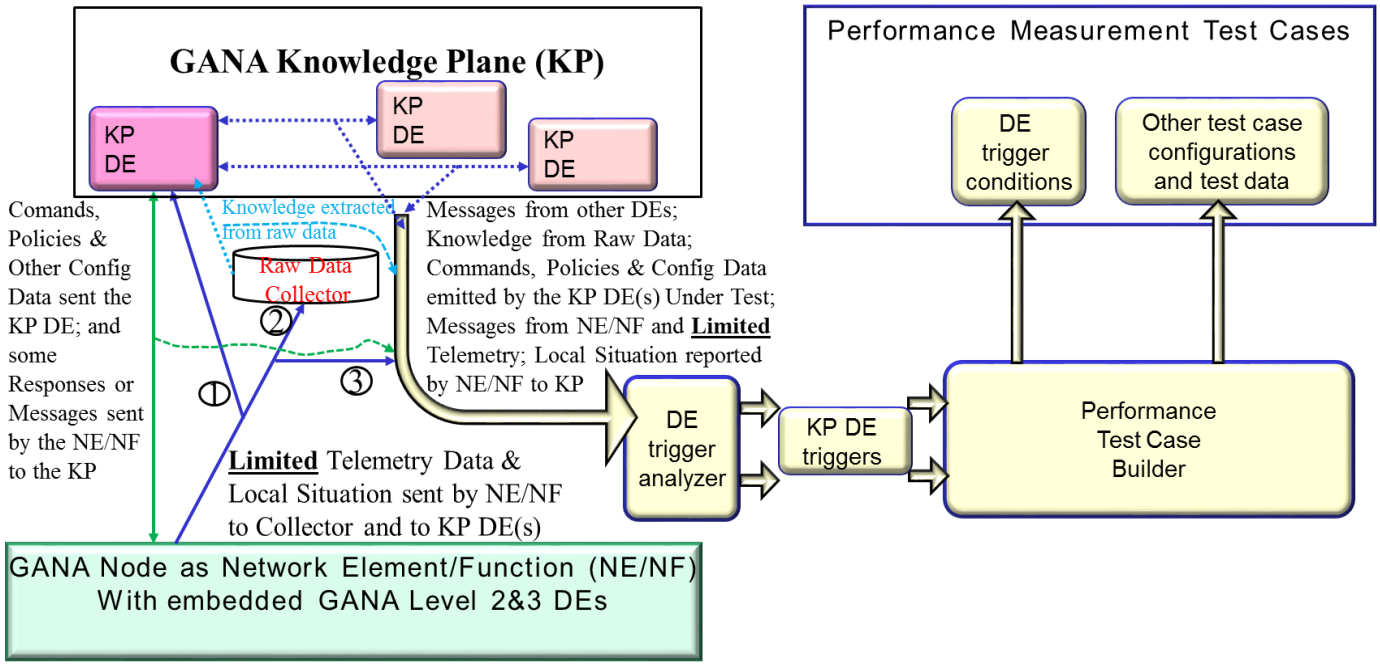

Figure 21: Scenario 5: Testing a KP DE designed to autonomically manage a GANA Node as an NE/NF that embeds GANA Levels 2&3 DEs

Figure 22 presents Scenario 6, namely: Testing a KP DEs Collectively under the scenario that the KP DEs are designed to autonomically manage a GANA Node(s) as an NE(s)/NF(s) that embeds GANA Levels 2&3 DEs.

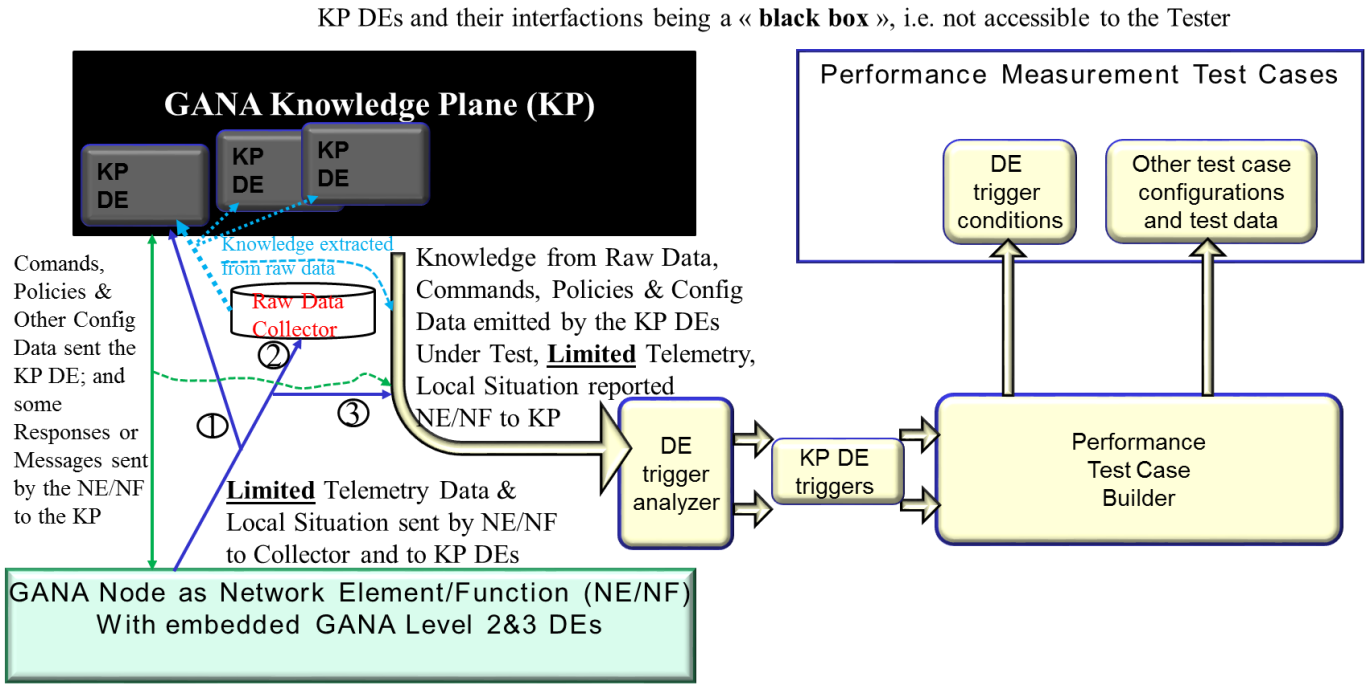

Figure 22: Scenario 6: Testing a KP DEs Collectively under the scenario that the KP DEs are designed to autonomically manage a GANA Node(s) as an NE(s)/NF(s) that embeds GANA Levels 2&3 DEs

6.2.4 Advantages of proposed model for tests of GANA

The proposed performance measurement model for GANA will enable:

1. Measurements of reaction time for Function Level DE actions.
2. Measurements of reaction time for Function Level DE requests to Node DE followed by Node DE actions and Function Level DE actions.
3. Measurements of correctness of Function Level DE decisions.
4. Measurements of correctness of Node DE decisions.
5. Measurements of sensitivity to conditions reported by the ME
6. Measuring the speed of learning for a Cognitive DE

The model enables any number of DE layer logics to be tested.

7. Capabilities of Rohde & Schwarz Test Solutions that play a role in Enabling Autonomic Management & Control of 5G Slices

The contents of this Chapter illustrate an example of the value AI brings in a Test System for Offline Testing (a Test System meant for testing a network (e.g. a 5G network) that is integrated and actually running as a production network that is delivering real network services).

Voice call stability is one of the main targets for mobile network optimization. The Call Drop Rate (CDR) is one of the KPIs used to measure the stability of calls in a particular area. The CDR is the ratio of drop calls over the total number

of calls in a test campaign. The value of the CDR normally ranges from 1% to 3%, as a drop call is a very unlikely event. As a result, a high number of calls must be performed in order to measure the CDR with statistical significance. This results in time-consuming measurement campaigns.

Call Stability Score (CSS) is new KPI to measure call stability using machine learning. A neural network is trained with a large dataset of real call tests, to output a normalized value from 0 to 1 that represents how far a call is from the drop calls the model has seen. The machine learning training process is depicted in the following figure (Figure 23):

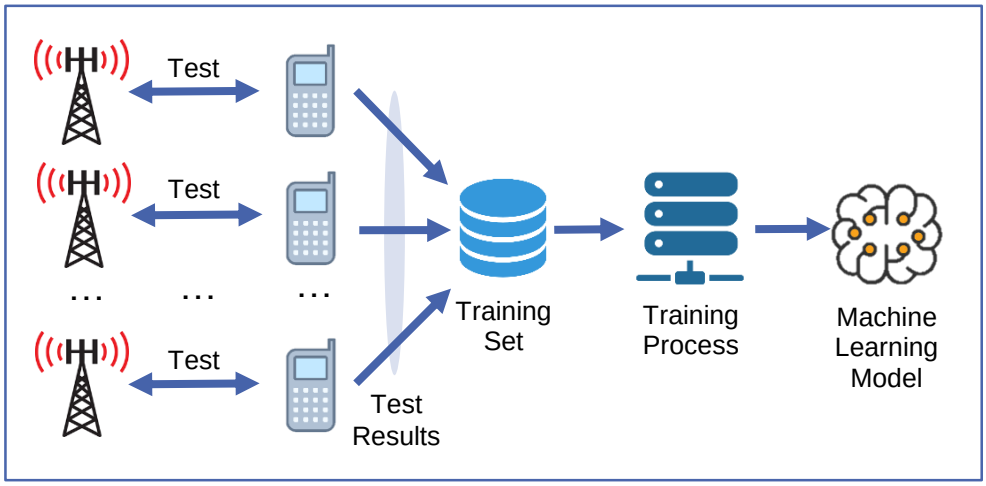

Figure 23: ***Machine Learning (ML) training process***

Once the model is trained, the CSS of each call test can be independently obtained in almost real-time, giving an indication of the call stability of the serving cell at the UE location. The following figure (Figure 24) shows the machine learning process taking place in the production environment:

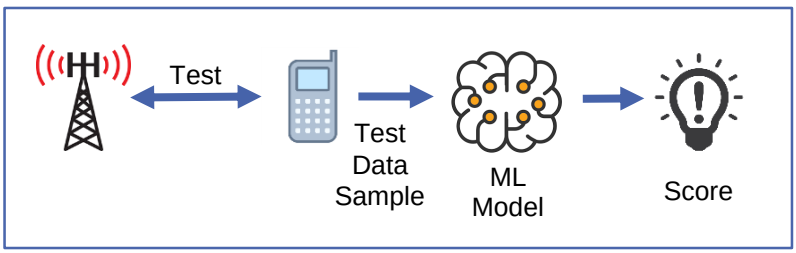

Figure 24: ***Output of Call Stability Score (CSS) by the Machine Learning (ML) Model***

The key benefit that enables CSS to be used in the decision making process of a knowledge plane is the fast response time achieved by machine learning. Probes can be placed in the network to have a continuous coverage of feedback in a defined grid or they can be deployed in a drive test setup. As the CSS can be derived down from a single call, the impact of the network performance by the probes is minimized. In fact, the CSS reflects the quality of a voice service. However, in general other KPIs for different slices can be developed and optimized by the use of machine learning. Figure 25 presents the Rohde & Schwarz Online AI-based Test System for CSS Measurements that play a role in Enabling Autonomic Management & Control of 5G Slices.

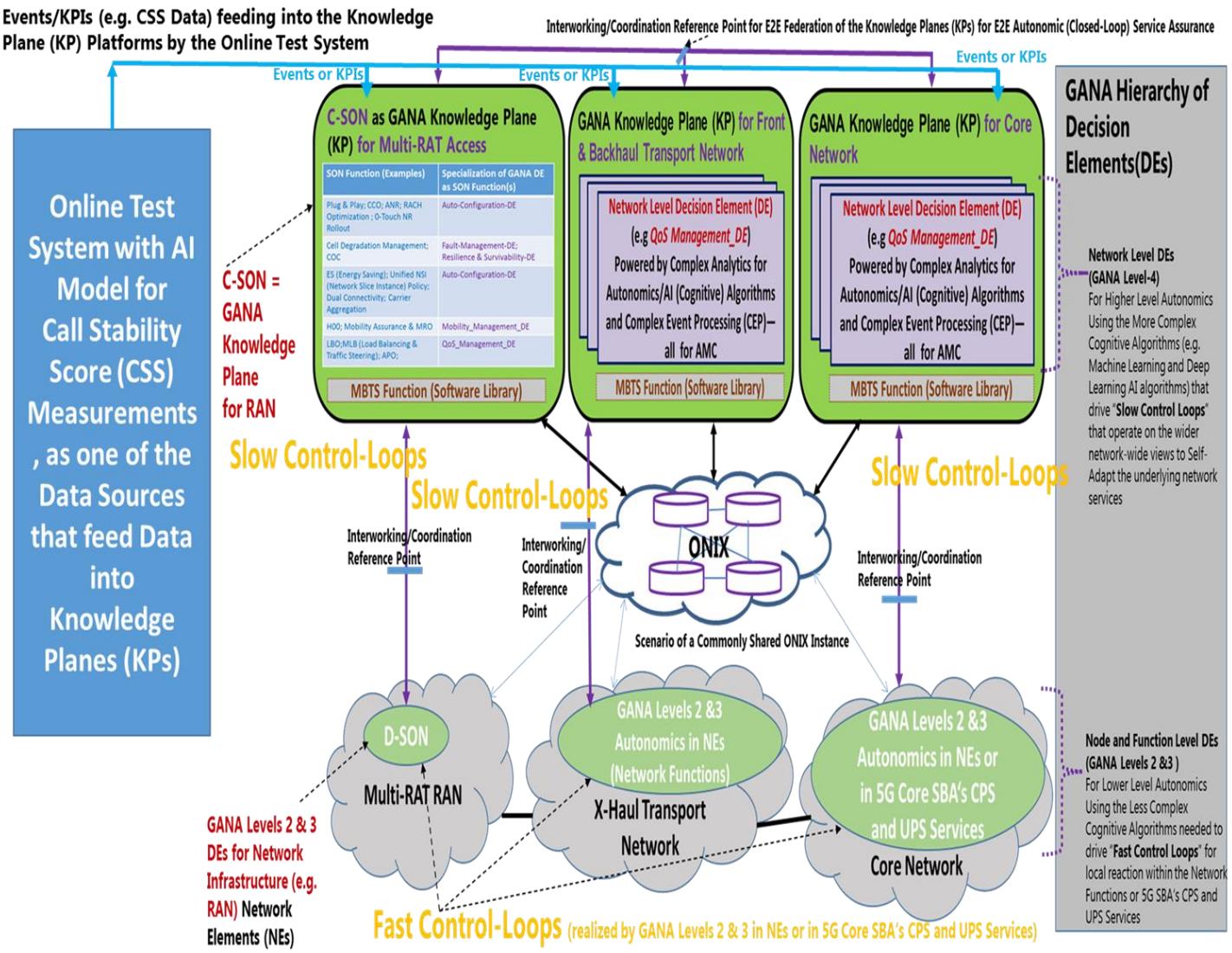

Figure 25: ***Rohde & Schwarz Online AI-based Test System for CSS Measurements that play a role in Enabling Autonomic Management & Control of 5G Slices***

In future, machine learning algorithms will help to identify and analyse trends in those KPIs to be able to execute preventive actions before a failure happens. This is in particular essential in the Industry 4.0 environment where very high availability and low latency are required.

8. Capabilities of Spirent and DATAKOM Solutions that can play a role in Testing AI Models for Autonomic (Closed-Loop) Management & Control of 5G Slices

There exist already some test systems that can be used in testing networks that are powered by autonomic management and control software for closed-loop adaptation of networks and services to deliver on SLAs (Service Level Agreements) as strongly required of 5G networks. Autonomic Management & Control (AMC) is a strong

requirement for E2E 5G Architectures as described by NGMN's 5G E2E Architecture Framework *v3.0.8* [33]. Both Active Testing of Services and Passive Testing of Services (including use of Passive Probing and Traffic Analytics) can play complementary roles in Network Testing and E2E Services Testing. Spirent has solutions in this space (see [34] on various solutions available). For example, "*Spirent VisionWorks provides intelligent and automated active test and assurance across the lifecycle of the evolving 4G/5G hybrid network*" [34]. The figure below (Figure 26) shows how such an Active Testing System can be used to Test networks that are powered by GANA Autonomic Decision-making-Elements (DEs) in the network segments infrastructures' NEs/NFs themselves and in the GANA Knowledge Plane (KP) Platforms realm. The KP level DEs are powered by AI and so are called "Cognitive DEs" while some DEs embedded in the NEs/NFs may have been designed as non-cognitive DEs or possibly with some level of non-complex AI. By focusing on the role Active Testing can play, the following ordered Test Scenarios can be executed in Testing and Measuring and Analysing the impact of the Multi-Layer GANA Autonomic Functions (Decision Elements) and their AI Models when the DEs are activated to run in a "closed-loop" mode to dynamically adapt the network(s) resources, parameters and services.

1. **Test Scenario 1**: If DEs are required to perform the initial configuration of the network (i.e. the configuration of the respective Managed Entities (MEs) of the various DEs) in order to put the network in a state of being ready to deliver services (e.g. eMBB services in 5G networks), then enable the DEs to do the initial configurations only and to switch to ***"open-loop" mode*** so that the DEs do not perform any adaptive autonomic change to the initial configuration of Managed Entities (MEs). Then ensure that the E2E network(s) required to deliver E2E services is ready to deliver services that can now be targeted for Testing and Gathering of Performance KPIs by the Test System(s), i.e. that the E2E Test Service(s) has been provisioned and can now be tested by the Test VisionWorks Test System. Then Perform the E2E Service Tests and gather Performance KPIs of the Tested E2E Services (for later comparison with the Test run results to be obtained in the subsequent test scenarios). **NOTE:** Various Test Load Scenarios for Service Performance Testing under varying Loads should be considered.
1. **Test Scenario 2**: Configure the GANA DEs (which include AI-powered/Cognitive DEs of the KP platforms and "possibly Non-Cognitive DEs" embedded in the NEs/NFs as well) to now operate in ***"closed-loop" mode***, and this may be done for all the DEs responsible for autonomic network adaptation of their specific network segment, i.e. activating DE autonomics for each network segment "silo" involved in the E2E service delivery. Then Perform the E2E Service Tests and gather Performance KPIs of the Tested E2E Service(s) (for later comparison with the Test run results obtained in the various test scenarios). Allow the test to run over a reasonable length of time to ensure the DEs are reacting to changes such as varied Load Scenarios the Test Cases can create and even impairments or challenges that can be emulated and injected in the network. This Test Scenario is aimed at indirectly Measuring and Analysing the impact of the Multi-Layer GANA Autonomic Functions (Decision Elements) and their AI Models when the DEs are activated to run in a "closed-loop" mode to dynamically adapt the network(s). The activations of the DEs can be done per network segment at a time (as sub-scenario) and then re-run tests and collect Service Performance KPIs data, and repeat the process until all network segments' associated DEs have been covered (activated) in some segment autonomics scenario based Tests. Later, by performing the Test results analysis, the impact of autonomics for a specific network segment on Service Performance can be inferred and characterized by indirect measurements of KPIs of service performance and other observable objects the DEs may have forced to be created (by virtue of their intelligence), e.g. new backup network functions (including virtual instances) or forwarding paths created or reconfigured to achieve certain resilience and QoS targets. **NOTE:** Various Test Load Scenarios for Service Performance Testing under varying Loads should be considered.
2. **Test Results Comparisons for the various Test Scenarios:** Gather and compare the various Service Performance KPIs data in order to ***indirectly Measure and Analyse the impact of the Multi-Layer GANA Autonomic Functions (Decision Elements) and their AI Models*** when the DEs are activated to run in a "closed-loop" mode to dynamically adapt the network(s). Other observable objects the DEs may have forced to be created (by virtue of their intelligence) need to be considered as well in such analysis.

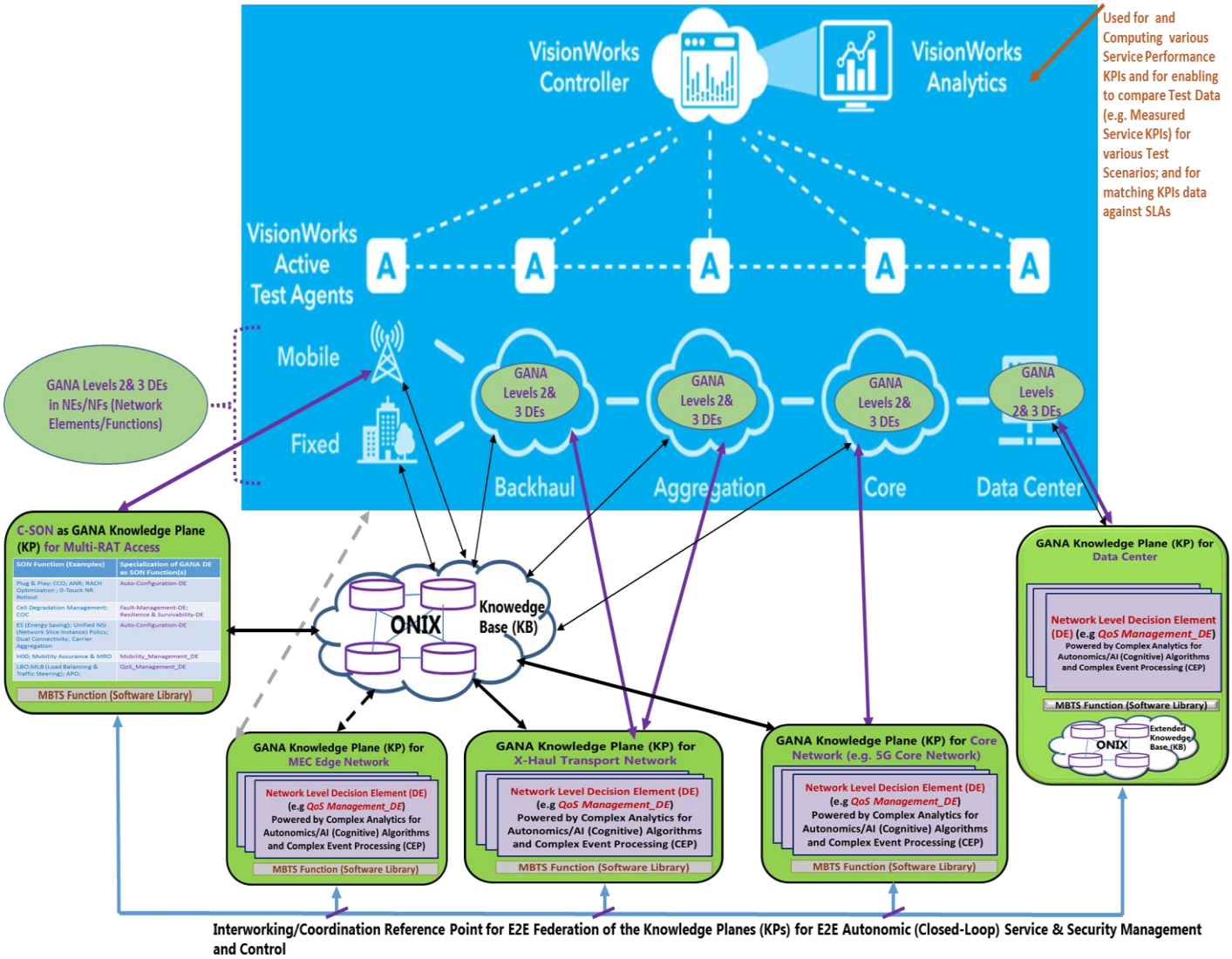

Figure 26: ***An Example Spirent Solution that can be used in E2E Service Testing of the E2E Network to Measure and Analyse the impact of the Multi-Layer GANA Autonomic Functions (Decision Elements) when the DEs are activated to run in a "closed-loop" mode to dynamically adapt the network(s) parameters, resources or services***

9. Generic Test Framework for Testing ETSI GANA Model's Multi-Layer Autonomics & AI Algorithms for Closed-Loop Network Automation

9.1. Overview of the High-Level Requirements for the desired Generic Test Framework for GANA Autonomics

NOTE: The Performance Measurement Model for GANA and the ideas described in the whole chapter 6 on "Convergence Performance Testing with Functional Testing in Testing AI Models: Proposal of a Performance Test Framework for the GANA DEs" should be considered as part of the desired Generic Test Framework for GANA Autonomics.

As described earlier, the ETSI TS 103 195-2 [2] defines the concept of Autonomic Manager element (called a "**Decision-making-Element" (DE)** in the GANA terminology) as a functional entity that drives a control-loop meant to configure and adapt (i.e. regulate) the behaviour or state of a Managed Entity (i.e. a resource)—usually multiple Managed Entities(MEs). The ETSI GANA Standardized Framework for AMC (ETSI TS 103 195-2) defines an Intelligent Management and Control Functional Block called GANA KP that is an integral part of AMC Systems that provides for the space to implement complex network analytics functions performed by interworking Modularized and specialized DEs. The KP DEs run as software in the Knowledge Plane and drive *self-* operations such as self-adaptation, self-optimization, self-monitoring* objectives for the network and services by programmatically (re)-configuring Managed Entities (MEs) in the network infrastructure through various means possible: e.g. through the NorthBound Interfaces available at the OSS, Service Orchestrator, Domain Orchestrator, SDN controller, EMS/NMS, NFV Orchestrator, etc. While a Cognitive GANA DE is the concept that must be considered with respect to Testing AI Models in GANA as Component Under Test (CUT), this Chapter provides a Generic Test Framework for Testing ETSI GANA Model's Multi-Layer Autonomics & AI Algorithms for Closed-Loop Network Automation. The general principles outlined earlier in chapter 6 (which include the principles outlined in ETSI EG 203 341 V1.1.1 (2016-10) [27]) should also be considered by testers as they also apply to testing Cognitive DEs. The Generic Test Framework for Testing GANA Autonomics is an elaboration of the early draft framework presented in Annex of ETSI EG 203 341 V1.1.1 (2016-10) [27] and has been extracted from ETSI EG 203 341 V1.1.1 (2016-10) and elaborated as illustrated in this Chapter. **NOTE:** This Generic Test Framework for GANA is expected to be further developed by one of the deliverables of the newly launched Work Item in ETSI TC INT [25] and readers are encouraged to follow the further developments on the framework by the Work Item. The Generic Test Framework is aimed at providing guidance on various aspects such as the following aspects:

- Conformance Testing and Interoperability Testing for GANA Functional Blocks (DEs, ONIX, MBTS) based on their Reference Points and Characteristic Information exchange expected on the Reference Points; i.e. Conformance Testing and Interoperability Testing is required on the Reference Points for Autonomics instantiated in a target architecture and environment
- Criteria for use in Verdicts passing when Testing Autonomic Functions (AFs), i.e. GANA DEs
- The role of a GANA Meta-Model in generation of Data Types for use in Test Suites Development
- Need for Specifications to be provided to Tester by a DE vendor, regarding "claims" on what the DE strives to achieve during its operations, with indications on the metrics (e.g. KPIs) that can be measured and monitored, appearance/manifestations of new instances of objects the DE causes to be created, or change in state of certain objects impacted by the DE's autonomic operations
- Testing non-cognitive GANA DEs, taking into consideration the "Operating-Region" of a Control-Loop(s) associated with the DE
- Testing Cognitive GANA DEs as deployable AI Models, taking into consideration the "Operating-Region" of a Control-Loop(s) associated with the DE, while taking into considerations Training Data Repository for AI Models (i.e. the cognitive DEs), and AI Algorithms that can be employed by a DE logic, such as machine Learning(ML), Deep Learning (DL), Computational Intelligence, etc.
- Integrated self-testing within a DE (i.e. embedded testing using a test component embedded within the DE
- Validation, Trustworthiness-building, and then Certification of a DE (or collective bundle of interworking DEs)
- Testing a collective group/bundle of interworking DEs as a black box (applies especially to DEs within GANA nodes)
- Consideration of the various Components that need to interwork with a GANA DE Under Test in Automated Test Developments and Executions
- Testing GANA Knowledge Plane as an AI system of collaborating DEs (AI Components)
- Testing vertical interactions of DE stacks along the GANA Hierarchy of Decision-making-Elements (DEs)
- Test Data required for Testing DEs

- The Types of Testing and associated Test Systems and components that should be applied in Testing DEs during various phases of DE lifecycles and also phases of the lifecycle of the Network to be impacted by the DE's operations (Validation Phase of a DE, Trustworthiness building phase for a DE, Certification Phase for a DE, Network Deployment Phase, DE deployment and activation Phase, Network Operation Phase, Network Optimization Phase)
- Where Passive Testing plays a role and where combined Active and Passive Testing plays a role in Testing DEs
- Integration and User Acceptance Testing of GANA DEs

There are various Types of Testing that need to be more detailed by the Generic Test Framework, and need to be considered by Test systems developers and testers for GANA autonomics, namely:

- *Conformance Testing of the GANA Knowledge Plane (KP) DEs and their Reference Points, and Test Data*
- *Conformance Testing of the GANA Knowledge Plane (KP) ONIX system and its associated Reference Points, and Test Data*
- *Conformance Testing of the GANA Knowledge Plane (KP) MBTS and its associated Reference Points, and Test Data*
- *Conformance Testing of the GANA Levels 2 and 3 DEs and their Reference Points, and Test Data*
- *Integration Testing of GANA Functional Blocks (KP DEs, ONIX, MBTS, GANA Levels 2 and 3 DEs in NEs/NFs), and Test Data*
- *Performance Testing of individual GANA Functional Blocks (KP DEs, ONIX, MBTS, GANA Levels 2 and 3 DEs in NEs/NFs), and Test Data Gathering and the need to ensure Quality for the Test Data through appropriate Data Validation techniques and methods to improve the quality through generation of synthetic data where necessary to complement real data gathered from real network operation environments*

9.2. The Draft of the for the desired Generic Test Framework for GANA Autonomics

9.2.1 Overview

The Generic Test Framework identifies different types of test systems that could be employed to the problem space of testing Autonomic Functions (AFs), i.e. GANA DEs, and are to be applied in phased testing starting at design time up to the point when a network consisting of trusted and certified AFs is tested as a whole (for integration and user acceptance testing). **NOTE**: Throughout the content in this Chapter/Section, ***Autonomic Function (AF)*** means a *GANA DE (a cognitive DE or non-cognitive DE)*. The following topics are addressed aimed to be addressed by the Generic Test Framework for GANA Autonomics Components:

- What type of testing is performed for an AF (GANA DE) during design time and who performs the testing and owns test components used?
- What role Testing plays in the following phases of an AF (GANA DE) lifecycle?
  - Validation of an AF (or collective interworking AFs);
  - Trustworthiness building on an AF (or collective bundle of interworking AFs);
  - Certification of an AF (or collective bundle of interworking AFs).
- Where Conformance Testing comes into play, and where Interoperability Testing comes into play?
- Where Integration and User Acceptance Testing of an Adaptive Network comes into play?
- Criteria/basis for assigning verdicts in Test cases employed at various phases of Testing AFs?
- The need for the Test Systems or Test Components that test AFs to be intelligent (i.e. themselves being autonomic-like) as to mimic AFs themselves?
- Where Passive Testing plays a role and where combined Active and Passive Testing plays a role?

Using knowledge from reference models for adaptive networks, such as the GANA as well as research efforts in literature on Testing and Validation for Autonomic and Self-Managing Systems, the following aspects can be deduced:

1) According to Reference Models such as the GANA: Autonomics introduce Functional Blocks (FBs) and their associated Reference Points that are specific to enabling to implement autonomics (AFs and the enabling components) in a target network architecture such as the 3GPP network architecture and its management and control architecture. The implication of this is that Conformance Testing and Interoperability Testing is required on the Reference Points of the Autonomics Functional Blocks (e.g. GANA KP DEs, ONIX, MBTS, GANA Levels 2 and 3 DEs in NEs/NFs) instantiated in a target architecture and operational environment. The reason is that the various FBs for autonomics may come from different vendors/suppliers of those autonomics specific FBs. The GANA defines various reference points whose instantiation in target architecture and environment calls for conformance and interoperability testing.

2) Individual AFs or their composition into Autonomic Systems need to undergo the following processes in the lifecycle: Validation**,** Trustworthiness-building, and then Certification**.** Providers/suppliers of AFs are responsible for performing these processes. Deployability of an AF should be based on the condition that the AF passed all the processes up to having been certified. Figure 27 illustrates the processes.

Figure 27: ***Potential AF certification process before inclusion in ANs (Adaptive Networks)***

3) As discussed in ETSI TS 103 195-2 and White Paper No.4 [23], AFs (i.e. GANA DEs) may be designed as run-time loadable or replaceable software modules (better AFs in terms of quality of decision-making capability may be used to replace low quality AFs), and may be deactivated and activated during operation time.

4) The Network Scope (Domain) for which a supplier of AFs and their embedment in network equipment or some host platforms should be clarified by the supplier of the AFs. The scope could be the whole network segment and its management and control architecture (e.g. core network and the associated management and control architecture, backhaul and it management and control architecture, RAN and its management and control architecture, or a larger edge to edge or end-to-end (E2E) network and its management and control architecture).

5) The Self-* features realized by a particular AF, as well as "Assertions(claims)" on what the AF strives to achieve during its operations, with indications on the metrics (e.g. KPIs) that can be measured and monitored, appearance/manifestations of new instances of objects the AF causes to be created, or change in state of certain objects impacted (e.g. parameters of its Managed Entity(ME) or MEs), should all be used to verify/test the claim and should be described by the AF provider and made known to the tester.

6) Testing of AFs involves various techniques and approaches, ranging from:

   a) Integrated self-testing within an AF (i.e. embedded testing using a test component embedded within the AF) as shown in Figure 28.

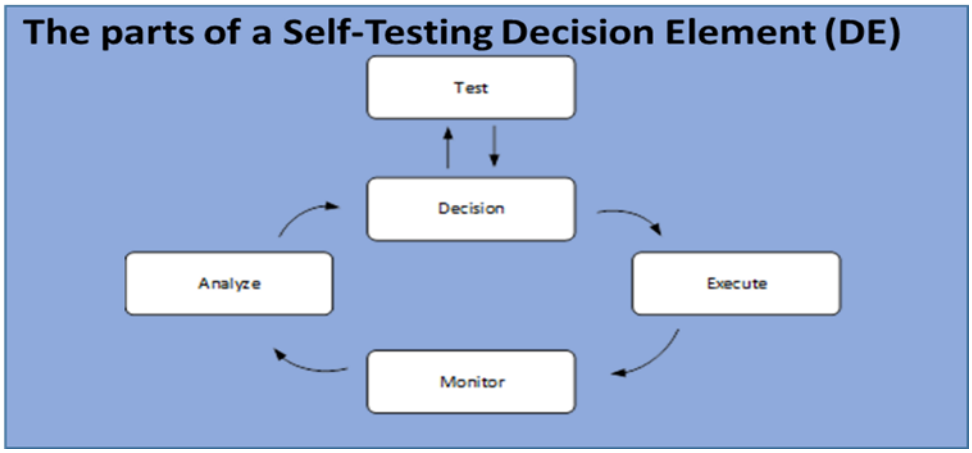

Figure 28: ***Self-Testing concept for AFs (GANA DEs)***

b) Testing a collective group/bundle of interworking AFs as a black box (applies especially to AFs within nodes (NEs/NFs)).

c) Testing system/component may intercept and observe actions of the AF under test that are performed in response to stimuli (mainly based on the operating region of the AF's control-loop) and use the actions in inferring correctness of the action (depends on the intelligence and correctness of the test component's algorithms it employs in the testing). This applies to environment in which AF actions can be intercepted during active testing (with injection of stimuli data to the AF under test), e.g. in tests conducted by the AF owner. This could be considered as a foundation for establishing a "Qualified Automated Test Component(s) or System" that exhibit best quality AI capabilities or Decision-making Capabilities for testing those of AI Component(s)/System Under Test or simply a Decision-making Element (DE) Under Test

d) Passive testing may be used by the test system to observe the metrics (KPIs) and the objects that may be instantiated or intentionally impacted by the AF, using monitoring techniques and inferring whether the changes are desirable for meeting the objectives of the network and claims made about the AF's impact on the monitored metrics and/or objects instantiated or whose state gets modified intentionally by the AF's actions. The approach could be as follows: a set of metrics (e.g. KPIs) determined to be critical to be observed is derived based on impacts the AF is claimed to positively have on the metrics or observable objects (such as services or service nodes),and base acceptable values for the metrics or state of objects are first established, and then the test system passively monitors the metrics and objects over time while the environment and the workload around the AF is known to be changing over time, and then the test system/component keeps tracing if and how the KPIs or objects are impacted over an observation window. Verdicts are then assigned after the sampling. The test objective may seek to determine whether the measured values improve, remain close to acceptable values.

e) Combination of active and passive testing may be employed.

7) Test and Validation Verdicts for an AF or a bundle of AFs (treated as black box) should be based on the following, depending on the testing approach used:

a) Verdict passing may be based on determining whether an intercepted action performed by an AF within a certain acceptable time that is measured relative to some event of

interest to the test system/component, has significant impact on meeting the objectives required of the node (NE/NF) or network, depending on the claims attached to the AF on what it does with respect to the objectives. The impact factor can be used in determining correctness of the action during validation of the AF. This applies to environment in which AF actions can be intercepted.

b) Verdict passing may be based on observing the impacts (metrics and/or objects instantiated or modified as a result of the adaptive behaviour of the AF under test) and assessing whether the impacts support the claims concerning what the AF is meant to achieve in its operation. This applies in environment and testing in which it may even not be possible to intercept AF actions.

c) Verdicts may be based on various criteria, such as a combination of actions, timing and impacts observed on metrics and objects of interest to the test case.

d) Because each AF may be designed to realize multiple Self-* features/objectives of a NE/NF or Network as a whole (i.e. by AF in the GANA Knowledge Plane level), such as *auto-discovery and self-configuration, self-optimization, self-healing, etc.*, verdicts may be defined that specifically target the individual Self-* features of an AF.

8) Test Systems or Test Components that test AFs should be intelligent (i.e. themselves being autonomic-like) as to mimic AFs they are meant to test, meaning that confidence in the test system also needs to be built up over a certain time and application to various test scenarios. This could be considered as a foundation for establishing a "Qualified Automated Test Component(s) or System" that exhibit best quality AI capabilities or Decision-making Capabilities for testing those of AI Component(s)/System Under Test or simply a Decision-making Element (DE) Under Test.

9) Testing of Adaptive Networks (ANs) is to be *considered as decomposed into various testing needs* and associated test systems and components, from component level testing of an AF as individual software modules up to the highest level Integration and User Acceptance Testing of an Adaptive Network as a whole, which can only be done under the conditions that AFs passed all testing phases and types of testing and validation to the point of having been certified. AFs that are trusted and ideally certified should be the ones that can be made to participate in the overall Integration and User Acceptance Testing of an Adaptive Network as a whole, implying dependencies on the tests conducted in various phases of an AF lifecycle.

10) Integration and User Acceptance Testing of an Adaptive Network as a whole comes into play when individual AFs have undergone as complete as possible the whole chain of Validation, Trustworthiness-building, and then Certification. At such a stage the Test system needs to access the system boundary that is defined by all the open interfaces for control and observation exposed mainly by the AFs and their interfaces with other Functional Blocks that enable the AF to operate. Some test cases on this higher level testing may depend on passive testing.

11) Input to deriving test cases for an AF should be based on the following items:

a) Reference Points that apply to the AFs or bundled AFs to be tested. ETSI TS 103 195-2 defines the various Reference Points of DEs and other GANA Functional Blocks (FBs). What needs to be considered with respect of the Reference Points of an AF in developing Test Cases is the Characteristic Information that needs to be exchanged on the Reference Point and the means by which the characteristic information is conveyed (e.g. by means of protocols or APIs methods). The Reference Points implementations (complete operational information exchange and protocols or APIs used in real-implementation) need to be obtained by Testers developing Test Cases from the documents on GANA instantiations onto particular target network architectures and their associated management and control architectures such as such as BroadBand Forum (BBF)

architectures (ETSI TR 103 473 V1.1.2), 3GPP Backhaul and Core Network (ETSI TR 103 404)) and other GANA instantiations documents produced by ETSI (e.g. ETSI TR 103 626 and ETSI TR 103 495).

b) The "Operating-Region" of a Control-Loop(s) associated with an AF.

c) The provider/supplier of an AF specifies what the AF is designed to achieve when running in the network (even without having to disclose the algorithms in the AF), specifying the network metrics (e.g. KPIs) that get improved by the AF or kept to a certain threshold by virtue of optimizations operations by the AF.

12) Test System for an AF can evolve in its test capabilities along with the need to test evolved AF algorithms.

13) Test Data may be synthetic or include in-service data involving a real environment in which AFs are being tested.

Table 2 categorizes the types of testing and associated test Systems and components that should be applied in Testing AFs during their lifecycles.

Table 2: Types of testing and associated deployment phases

Type of Testing	Validation phase of an AF	Trustworthiness building phase	Certification Phase for the AF	Test Network Deployment Phase	AF deployment and activation Phase	Test Network Operation phase	Test Network Optimization Phase
AF Testing and Validation	x	x	x				
Conformance Testing			x	x	x		
Interoperability Testing			x	x	x		
Integration and User Acceptance Testing of an Adaptive Network as a whole				x	x	x	x

9.2.2 The value of the GANA Meta-Model in the Generation Test Case Skeletons and Data Structures and Data Types Definitions that can be used by Test Case Developers in Test Case Developments for the various GANA Components

To facilitate for software development and testing of GANA components and their interactions, especially considering model driven development and testing approaches for GANA components and their interactions, the GANA Reference Model described in ETSI TS 103 195-2 needs to described using a formal representation like in the form of a UML (Unified Modeling Language) Model. The formal model representing the GANA Reference Model (e.g. as a UML Model) needs to then be further enriched with more details that are specific to a concrete GANA instantiation onto a target implementation-oriented network architecture and its associated management and control architecture, and even further enriched with implementation environment specific details that may be attached to meta-data that describes a particular GANA component targeted for implementation (including concrete communication methods and data it communicates or receives on its interfaces). The formal description of the GANA Reference Model is called the GANA Meta-Model (i.e. an Information Model) and its enrichment by a concrete GANA instantiation and some implementation environment oriented details makes it an "enriched GANA Meta Model" that is scoped(tailored) to a

target implementation domain that is determined by the network architecture and associated management and control architecture for which GANA has been instantiated (e.g. a BBF architecture (see ETSI TR 103 473 V1.1.2), a 3GPP network architecture (see ETSI TR 103 404 V1.1.1), or other network architectures). A DE Model presented in ETSI TS 103 195-2 (also on the Figure 30 below), would have a representation model in the desired GANA Meta-Model (not available yet), such that Model-Driven Software Development approaches such as MDA (Model Driven Architecture) [28] and other approaches in [29] can be applied in design and implementation of DEs instantiated to operate in a specific network node (network element/function) environment and/or in the associated management and control space/environment for the specific network (i.e. in the Knowledge Plane Platform). That means that the meta data of a DE model, with its structural model, interfaces and their primitives/procedures and parameters would need to be represented in a GANA Meta-Model. Actual instances of DEs required to be implemented to operate in specific networks and their associated management and control architectures can get instantiated in a modeling environment that relies on the GANA Meta-Model, such that environment specific features are tailored and added to the instances of the DEs for the targeted operational environment.

The diagram below illustrates the transformation through which a GANA Metal Model may undergo and at each stage it can be used by various tools that may be used by different stakeholders depending on the level of details they require of the GANA Meta Model. Figure 29 presents the processes that need to be executed in producing a formal model of the ETSI GANA Meta-Model of varying depth of details required for implementing GANA components for a specific environment.

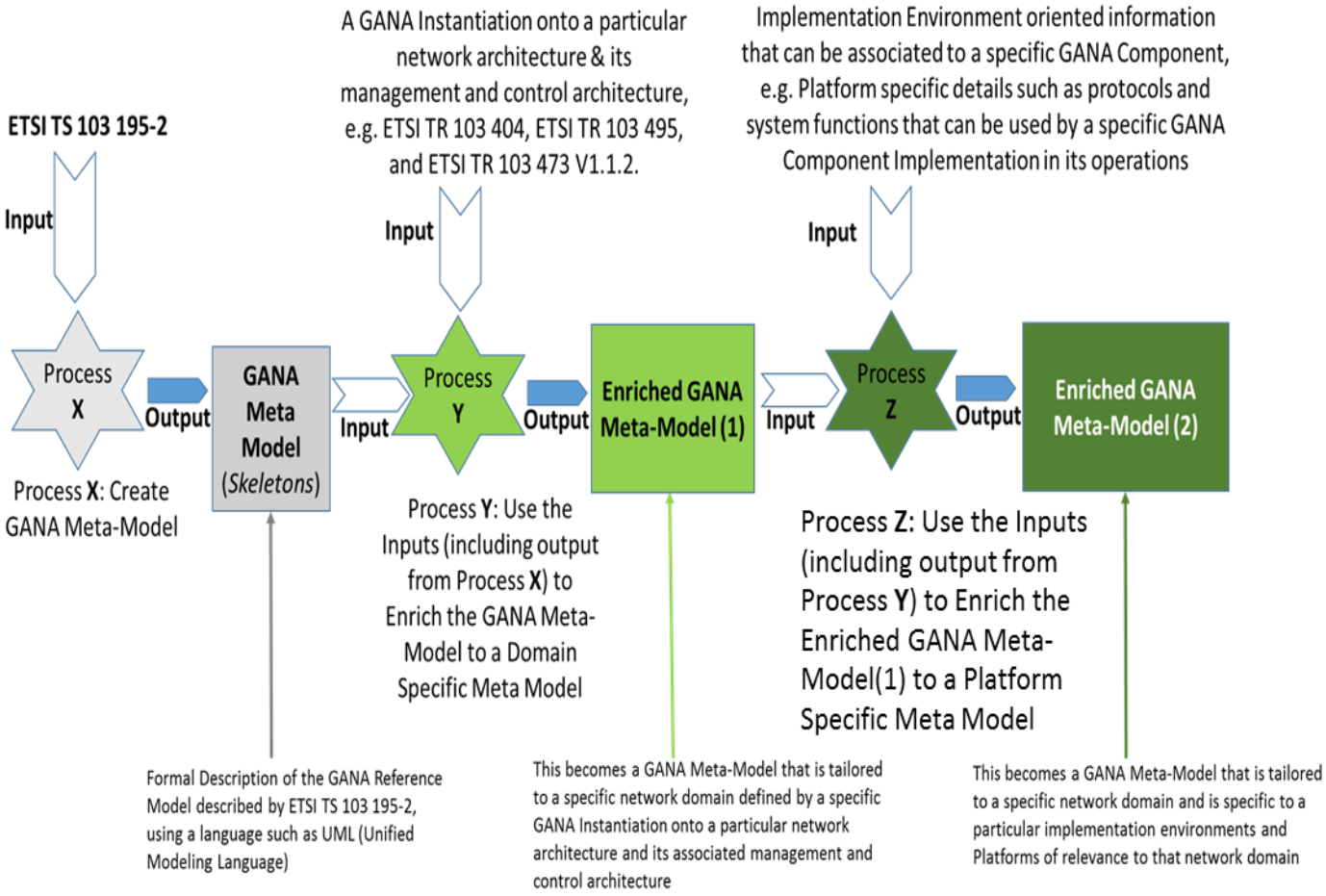

Figure 29: ***The processes that need to be executed in producing a formal model of the ETSI GANA Meta-Model of varying depth of details required for implementing GANA components for a specific environment***

Figure 32 provides an Illustration of how a GANA DE Model described in ETSI TS 103 195-2 can be modelled using UML as part of the overall GANA Meta-Model (an Asset to DE Implementers and Test Developers), while taking into account the *model of a DE and its interfaces and primitives* as described in section 9.2.3.

In [4] there is some work that had commenced on creating a GANA Meta-Model from the earlier version of the GANA Reference Model description in ETSI GS AFI002 document, but that GANA Meta-Model needs to be adapted to the updated and evolved GANA Reference Model as described in the Further Work part of ETSI TS 103 195-2.

The importance of the GANA Meta-Model (the skeleton created out of the GANA Reference Model described by the ETSI document).

9.2.3 Testing the Interfaces and Primitives/Operations that Enable Programmability (Manageability) of a Cognitive GANA DE (a Deployable "AI Model" software component for specific AMC targets)

The figure below (Figure 30), is an extract of the Model of GANA Decision making Element (DE) as Autonomic Management & Control AI Software Component that has Interfaces and shall support some Primitives on the Interfaces defined in ETSI TS 103 195-2. Such Primitives should be considered in developing Test Cases for Testing GANA DEs. **NOTE:** The means by which the primitives are implemented then need to be considered when developing test cases to test a DE (e.g. for conformance Testing). As described earlier, the value of the GANA Meta-Model is that it enables to automatically generate, using software, the meta-data and data types that describe the GANA specific concepts and even instances in specific languages of choice for GANA software implementers and testers, e.g. in C++, C, Java, or other programming languages (refer to [30][4][32] on the value Metamodeling brings to software development and automated code generation and test cases generation). As mentioned earlier, the meta data of a DE model, with its structural model, interfaces and their primitives/procedures and parameters would be represented in a GANA Meta-Model, and actual instances of DEs required to be implemented to operate in specific networks and their associated management and control architectures can get instantiated in a modeling environment that relies on the GANA Meta-Model, such that environment specific features are tailored and added to the instances of the DEs for the targeted operational environment.

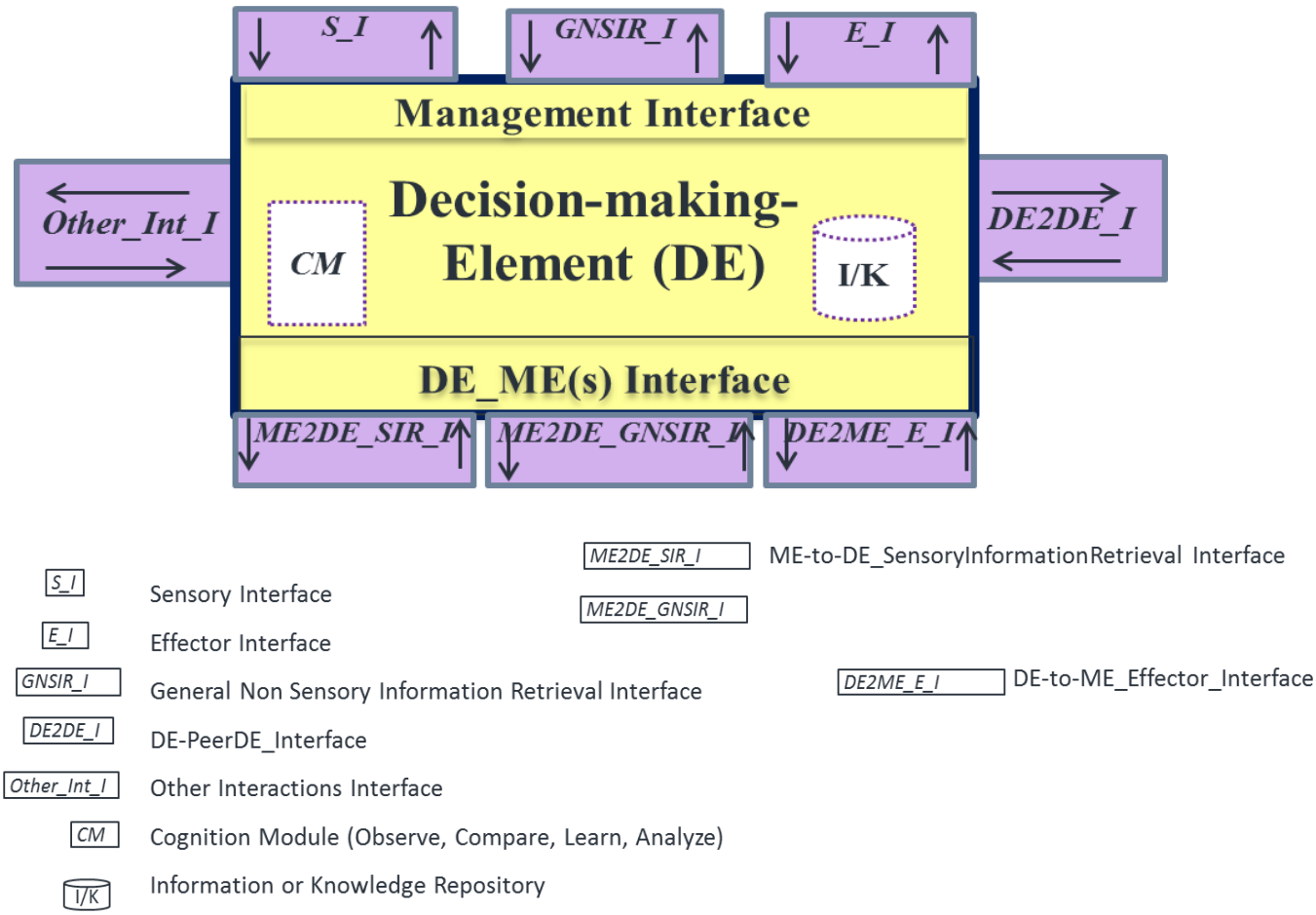

Figure 30: ***Model of a Cognitive GANA DE, and Primitives on the Interfaces Model of a Cognitive GANA DE***

The following figure (Figure 31) illustrates the primitives that can be introduced in extending those already defined in ETSI TS 103 195-2. **NOTE:** These extensions to the primitives are inspired by work in TMForum on AI Models and the Open Digital Architecture (ODA) [6] [18].

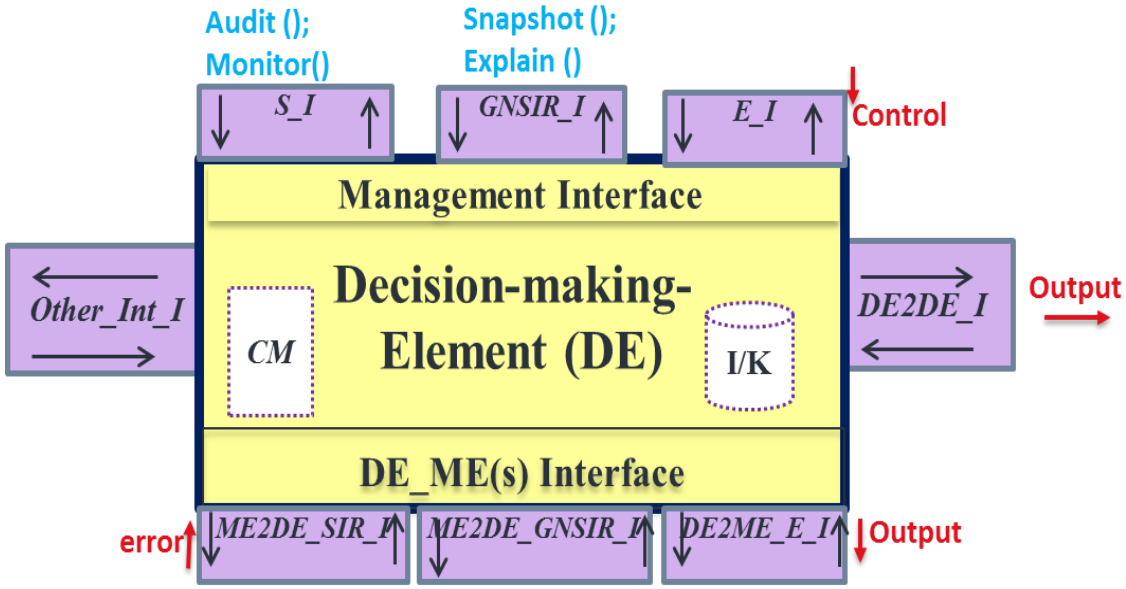

Figure 31: Extending the Primitives on Interfaces *Model of a Cognitive GANA DE defined in ETSI TS 103 195-2*

Descriptions of the Primitives that can be added to those already defined in ETSI TS 103 195-2 for a Cognitive Interface Model:

- The **Audit ()** Primitive should be supported on the **DE's S_I Interface** and can be implemented primarily by the use of the generic **Pull ()** Primitive defined for a DE Model (i.e. **Audit ()** is a **specialized Pull ()** Primitive) as presumed to be supported on this interface. From the DE's perspective, a Pull() is implemented by the DE and an entity that is allowed by a policy can make a call to the specific Pull() operations (e.g. Audit()) to retrieve the information of interest. *[The **Callee** is the DE]*
- **Monitor ()** Primitive should be supported on the **DE's S_I Interface** and can be implemented primarily by the generic **Push ()** Primitive defined for a DE Model (i.e. **Monitor ()** is a **specialized Push ()** Primitive that when called by the DE (as a skeleton, not the actual implementation of the Primitive) it results in a call on the recipient entity to receive the monitoring data/information the DE is „pushing" to the receiver entity that must have been registered as listener entity for the DE to „push" data to. *[The **Callee** is the Receiver Entity of the Monitoring Data].* However, a **Pull ()** Primitive may also be supported in implementing the **Monitor()** Primitive such that any entity (allowed by policy) can invoke(call) a **Pull()** operation on the DE to obtain the intended DE state meant to be always conveyable by **Monitor()** Primitive. When Monitor() is called by an entity (not the DE itself) a Pull() operation is called locally that returns the intended results of the **Monitor ()** call. *[The **Callee** is the DE]*
- **Snapshot()** Primitive should be supported on the **DE's GNSIR_I Interface** and can be implemented primarily by the generic **Get ()** Primitive defined for a DE Model (i.e. **Snapshot ()** is a **specialized Get ()** Primitive). *[The **Callee** is the DE]*
- **Explain ()** Primitive Primitive should be supported on the **DE's GNSIR_I Interface** and can be implemented primarily by the use of the generic **Get ()** Primitive defined for a DE Model (i.e. **Snapshot ()** is a **specialized Get ()** Primitive) as presumed to be supported on this interface. *[The **Callee** is the DE]*

- **data** (as input to the DE) is received or consumed by the DE primarily through either of the following DE Interfaces: **DE's ME2DE_SIR_I Interface**; **DE's ME2DE_SIR_I Interface** and various methods of receiving data can be supported by either **Push()** or **Pull()** models supported by the DE and the data source. "data" **is NOT a primitive (Operation/Method).**

- **control** is implemented by the Primitives already defined on the **DE's E_I Interface,** such as ***Start()*; *Enforce_Policy()*; *Apply_Control_Strategy()*, *Set()*, and the other primitives defined on the DE's E_I Interface**. "control" **is NOT a Primitive (Operation/Method)**.
- **output** is implemented by the Primitives already defined on the **DE's DE2ME_E_I and DE2DE_I Interfaces primarily,** such as ***Enforce Policy()*; *De-activate-Policy()*; *Set()*; *Test()*; *Set-Filter()*; *Apply-Control-Strategy()*, etc. on DE2ME_E_I Interface**; and ***ConveyMessage()*** on **DE2DE_I Interface**. output is communicated mainly as a result of a *"decision"* computed by a DE that results in a "*change request*" by the DE. However, for certain output that requires to be communicated up the DEs Hierarchy, e.g. in escalations of situations the DE is not able to handle but requires the approval to execute certain decisions by its upper DE, the **DE's Other_Int_I Interface or S_I Interface** may be used. "output" **is NOT a Primitive (Operation/Method)**.
- **Error** is implemented by the Primitive already defined on the **DE's ME2DE_SIR_I,** i.e. the Primitives such as ***Push()*** that can be used by an ME to communicate an *error (or erroneous situation)* to the DE responsible for managing the ME. "error" **is NOT a Primitive (Operation/Method)**.

Therefore, the additional Primitives that can be added to those already defined on DE Management Interface defined in ETSI TS 103 195-2 are summarized in the Table 3 below.

Table 3: Additional Primitives that can be added to those already defined on DE Management Interface defined in ETSI TS 103 195-2

DE Interface	Sub-interface	Full name of the sub-interface	Caller and Callee/Consumer	Primitives
Management interface	S_I	Sensory Interface	Upper DE as caller of the *Get() and Pull()* primitives and "this" particular DE as callee; Upper DE as callee of the *Push()* primitive and "this" particular DE as caller;	*Get(),* *Pull(),* *Push()* Audit (); Monitor()
	GNSIR_I	General Non Sensory Information Retrieval Interface	Upper DE as caller and "this" particular DE as callee for these primitives	*GetCapabilityDescription();* *GetFiniteStateMachine();* *GetFaultErrorFailureAlarmCausalityModel();* *GetFailureModesDescription();* *Snapshot ();* *Explain ()*
	E_I	Effector Interface	Upper DE as caller and "this" particular DE as callee	*Start();* *Pause();* *Resume();* *Terminate();* *Enforce_Policy();* *De_activate_Policy();* *Set();* *Apply_Control_Strategy()*

Figure 32 provides an Illustration of how a GANA DE Model described in ETSI TS 103 195-2 can be modelled using UML as part of the overall GANA Meta-Model (an Asset to DE Implementers and Test Developers).

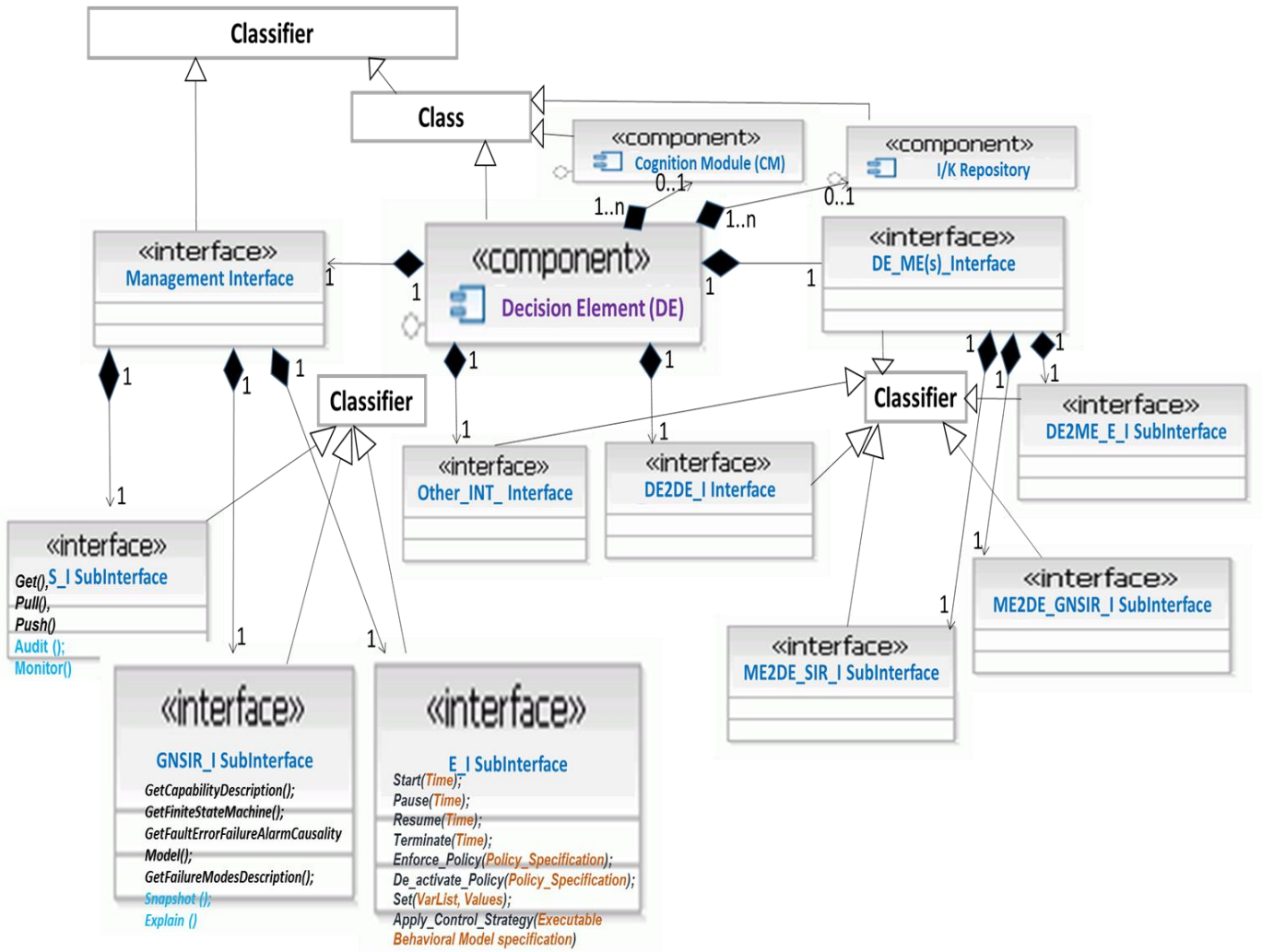

Figure 32: ***Illustration of how a GANA DE Model described in ETSI TS 103 195-2 can be modelled using UML as part of the overall the GANA Meta-Model (an Asset to DE Implementers and Test Developers)***

9.2.4 Testing the Algorithms of a DE, and Testing a Cognitive DE as an AI Model

While taking into consideration the other aspects covered in the previous chapters and sections regarding testing GANA components for self-adaptive networks, the algorithms of a DE (whether the DE is cognitive (embedding AI) or not cognitive) need to be tested by appropriate means. For example, the Optimization algorithms the DE employs to compute the values of certain parameters of its MEs that need to be (re)-set to new values (in programming ((re-)configuring) the ME parameters), need to be tested.

For a Cognitive DE, as described earlier in section 3.1, the general aspects concerning testing an AI system have to be considered in testing a cognitive DE, and a cognitive DE (as an AI exhibiting Component) should undergo Algorithms Testing and Model Validation. Approaches to testing a cognitive DE as an AI Model have been outlined in section 3.3 on "Stakeholders that should play certain roles in the context of Test and Certification of GANA Cognitive DEs and GANA Knowledge Planes for AMC".

9.2.5 Conformance Testing of the GANA Knowledge Plane (KP) DEs and their associated Reference Points, and Test Data

Figure 33 presents a picture on a GANA KP's data sources diversity, events visibility and consumption into KP DEs, and KP Integration Options. More details on KP platform characteristics are found in ETSI TS 103 195-2 and in [5] [17] [19] [24]. [19] presents an approach to implementing a KP Platform using the ONAP open source software, and [19] also discusses how other open source products such as [9] [10] [11] [12] [13] [14] [15] [16] can be used in implementing a KP Platform that integrates with other management and control systems depicted on Figure 33, such as SDN controllers, NFV MANO stacks, etc.

While taking into consideration the aspects covered in the previous chapters and sections regarding testing GANA components for self-adaptive networks, testers are hereby further guided on how to carry out Conformance of a GANA KP Platform's DEs. Conformance Testing a GANA Knowledge Plane (KP) Platform's individual DEs involves the following aspects:

1. Testing the communications involving the Reference Points (Rfps) defined in ETSI TS 103 195-2 and further detailed in specific network architecture (environment) in which the GANA DEs have been instantiated and implemented to operate (e.g. specific GANA instantiations such as ETSI TR 103 473 V1.1.2, ETSI TR 103 404 and other GANA instantiations), i.e. the Rfps through which a particular Knowledge Plane DE communicates with other Functional Blocks (FBs). Inputs that Test Developers should consider in developing Test Cases for DE conformance to the Rfps and the data models that apply for the specific Rfp are based on the complete implementation of the Rfp in the environment in which the DE is to operate. The Component Under Test (CUT) is a KP DE (such as the Network Level Fault Management-DE) and its interfaces with its MEs and other entities it is required to interact with. The implementer (supplier) of the DE Under Test should provide to the Test Developer the details of Rfps from ETSI TS 103 195-2 that have been implemented for the DE, and the Test Developer should also use assets such as DE related data structures and data types definitions that can be generated from the GANA Meta-Model (represented in a formal modeling language such as UML as discussed earlier).
2. Taking into consideration standardized interfaces and data models employed by the particular KP DE(s) under test in communicating with another entity such as OSS, SDN controller, or any other component that the DE may be required to communicate with. The diagram below (Figure 33) provides insights on the interfaces (NorthBound Interfaces (NBIs)) of various systems or components that may be integrated with the Knowledge Plane Platform to be used by the individual DEs of the KP.
3. Taking into consideration, in the Test Cases for the DE, the required interaction between the DE and any lower level DE implemented in at the level of a Network Element/Function (NE/NF) that the KP level DE policy controls and from which the KP level DE listens for escalations messages for actions synchronizations.
4. Obtaining Quality Test Data required to test the DE. This involves Test Data Gathering and the need to ensure Quality for the Test Data through appropriate Data Validation techniques and methods to improve the quality through generation of synthetic data where necessary to complement real data gathered from real network operation environments. The Test Data for Conformance Testing of the DE primarily comes from the standardized data models that are employed in the communications between the DE and its MEs and other input data sources required to feed data into the DE, and in the communication between the DE and other DEs, MBTS and ONIX, i.e. the standardized data models that are employed on all the interfaces of a DE as illustrated by the model of a DE defined in ETSI TS 103 195-2 and as shown earlier in this paper(particularly in the instantiation and implementation of the DE).

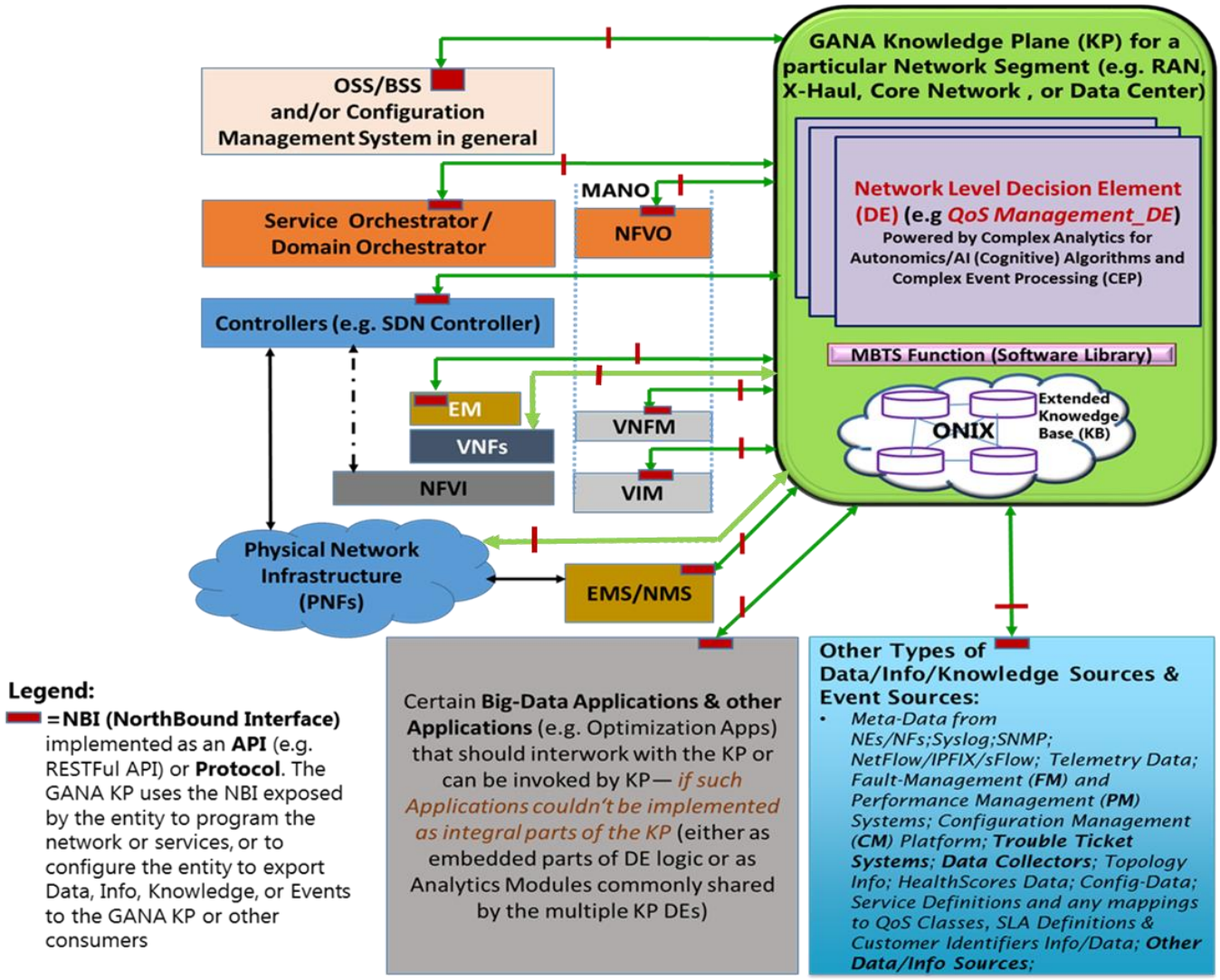

Figure 33: ***GANA KP's data sources diversity, events visibility and consumption into KP DEs; and KP Integration Options***

9.2.6 Conformance Testing of the GANA Knowledge Plane (KP) ONIX system and its associated Reference Points, and Test Data

While taking into consideration the aspects covered in the previous chapters and sections regarding testing GANA components for self-adaptive networks, testers are hereby further guided on how to carry out Conformance of a GANA KP Platform's ONIX part. Conformance Testing a GANA Knowledge Plane (KP) Platform's ONIX involves the following aspects:

1. Testing the communications involving the Reference Points (Rfps) defined in ETSI TS 103 195-2 and further detailed in specific network architecture (environment) in which the ONIX system has been instantiated and implemented to operate (e.g. in specific GANA instantiations such as ETSI TR 103 473 V1.1.2, ETSI TR 103 404 and other GANA instantiations), i.e. the Rfps through which the ONIX communicates with other Functional Blocks (FBs). Inputs that Test Developers should consider in developing Test Cases for ONIX system conformance to the Rfps and the data models that apply for the specific Rfp are based on the complete implementation of the Rfp in the environment in which the ONIX system is to operate. The System Under Test

(SUT) is the ONIX and its interfaces for *publish/subscribe/query&find services*. The implementer (supplier) of the ONIX (the SUT) should provide to the Test Developer the details of Rfps from ETSI TS 103 195-2 that have been implemented for the ONIX, and the Test Developer should also use assets such as any ONIX interfaces related data structures and data types definitions that can be generated from the GANA Meta-Model (represented in a formal modeling language such as UML as discussed earlier).

2. Taking into consideration standardized interfaces and data models employed by the particular ONIX system under test in communicating with another entity such as a DE, OSS, SDN controller, or any other component that the ONIX may be required to communicate with. The diagram on Figure 33 provides insights on the interfaces (NorthBound Interfaces (NBIs)) of various systems or components that may be integrated with the Knowledge Plane Platform and may be required to interact with the ONIX part of the KP.
3. Obtaining Quality Test Data required to test the ONIX. This involves Test Data Gathering and the need to ensure Quality for the Test Data through appropriate Data Validation techniques and methods to improve the quality through generation of synthetic data as necessary (this may be complemented by real data gathered from real network operation environments in case real data is also to be used in testing). The Test Data for Conformance Testing of the ONIX primarily comes from the standardized data models that are employed in the communications between the ONIX and other Functional Blocks defined by ETSI TS 103 195-2, such as DEs and MBTS, i.e. the standardized data models that are employed on all the interfaces of an ONIX system as defined in ETSI TS 103 195-2 and in the instantiation and implementation of the ONIX system.

9.2.7 Conformance Testing of the GANA Knowledge Plane (KP) MBTS and its associated Reference Points, and Test Data

While taking into consideration the aspects covered in the previous chapters and sections regarding testing GANA components for self-adaptive networks, testers are hereby further guided on how to carry out Conformance of a GANA KP Platform's MBTS Software Library. Conformance Testing a GANA Knowledge Plane (KP) Platform's MBTS involves the following aspects:

1. Testing the communications involving the Reference Points (Rfps) defined in ETSI TS 103 195-2 and further detailed in specific network architecture (environment) in which the MBTS Software Library has been instantiated and implemented to operate (e.g. in specific GANA instantiations such as ETSI TR 103 473 V1.1.2, ETSI TR 103 404 and other GANA instantiations), i.e. the Rfps through which the MBTS communicates with other Functional Blocks (FBs). Inputs that Test Developers should consider in developing Test Cases for the MBTS Library conformance to the Rfps and the data models that apply for the specific Rfp are based on the complete implementation of the Rfp in the environment in which the MBTS Library is to operate. The System Under Test (SUT) is the MBTS and its interfaces for *Translation Services*. The implementer (supplier) of the MBTS (the SUT) should provide to the Test Developer the details of Rfps from ETSI TS 103 195-2 that have been implemented for the MBTS Library, and the Test Developer should also use any assets such as MBTS interfaces related data structures and data types definitions that can be generated from the GANA Meta-Model (represented in a formal modeling language such as UML as discussed earlier).
2. Taking into consideration standardized interfaces and data models employed by the particular MBTS Software Library under test in communicating with another entity such as a DE, OSS, SDN controller components/modules, or any other component that the MBTS may be required to communicate with. The diagram on Figure 33 provides insights on the interfaces (NorthBound Interfaces (NBIs)) of various systems or components that may be integrated with the Knowledge Plane Platform and may be required to interact with the MBTS part of the KP.
3. Obtaining Quality Test Data required to test the MBTS Software Library. This involves Test Data Gathering and the need to ensure Quality for the Test Data through appropriate Data Validation techniques and methods to improve the quality through generation of synthetic data as necessary (this may be complemented by real data gathered from real network operation environments in case real data is also to be used in testing). The Test Data for Conformance Testing of the MBTS Library primarily comes from the standardized data models that are employed in the communications between the MBTS and other Functional Blocks defined by ETSI TS 103 195-2, such as DEs and ONIX, i.e. the standardized data models that are employed on all the interfaces of the MBTS Library as defined in ETSI TS 103 195-2 and in the instantiation and implementation of the MBTS Library.

9.2.8 Conformance Testing of the GANA Levels 2 and 3 DEs and their associated Reference Points, and Test Data

While taking into consideration the aspects covered in the previous chapters and sections regarding testing GANA components for self-adaptive networks, testers are hereby further guided on how to carry out Conformance of GANA Levels 2 and 3 DEs. Conformance Testing GANA Levels 2 and 3 DEs involves the following aspects:

1. Testing the communications involving the Reference Points (Rfps) defined in ETSI TS 103 195-2 and further detailed in specific network architecture (environment) in which the GANA Level-2 DE or Level-3 DE has been instantiated and implemented to operate in a Network Element/Function (NE/NF) in a specific GANA instantiation such as ETSI TR 103 473 V1.1.2, ETSI TR 103 404 or other GANA instantiation, i.e. the Rfps through which the DE communicates with other Functional Blocks (FBs). Inputs that Test Developers should consider in developing Test Cases for the DE conformance to the Rfps and the data models that apply for the specific Rfp are based on the complete implementation of the Rfp in the environment in which the DE is to operate. The Component Under Test (SUT) is the DE such as Node-Level (GANA Level-3) Fault Management-DE or a Function-Level (GANA Level-2) such as a Routing-Management-DE, and its interfaces with its MEs and other entities it is required to interact with. The implementer (supplier) of the DE (the CUT) should provide to the Test Developer the details of Rfps from ETSI TS 103 195-2 that have been implemented for the DE, and the Test Developer should also use any assets such as DE interfaces related data structures and data types definitions that can be generated from the GANA Meta-Model (represented in a formal modeling language such as UML as discussed earlier).
2. Taking into consideration standardized interfaces and data models employed by the particular DE under test in communicating with another entity such as a DE or ME or any other component that the DE may be required to communicate with.
3. Taking into consideration, in the Test Cases for the DE, the required interaction between the DE and DE in the Knowledge Plane Level that policy controls it.
4. Obtaining Quality Test Data required to test the DE. This involves Test Data Gathering and the need to ensure Quality for the Test Data through appropriate Data Validation techniques and methods to improve the quality through generation of synthetic data (this may be complemented by real data gathered from real network operation environments in case real data is also to be used in testing). The Test Data for Conformance Testing of the DE primarily comes from the standardized data models that are employed in the communications between the DE and its MEs, between the DE and other input data sources required to feed data into the DE, and in the communication between the DE and other DEs, MBTS and ONIX, i.e. the standardized data models that are employed on all the interfaces of a DE as illustrated by the model of a DE defined in ETSI TS 103 195-2 and as shown earlier in this paper (particularly in the instantiation and implementation of the DE).

9.2.9 Integration Testing of GANA Knowledge Plane (KP) as a whole, and Test Data

There is a need to perform an integration Testing of the Knowledge Plane (KP) Platform as a whole. Inputs that Test Developers should consider in developing Test Cases for Testing the KP Platform's behavior as a whole (i.e. holistic verification of the KP Platform) involve all the Reference Points (Rfps) implementations for the KP platform, and this includes the various types of interfaces that may be implemented for a particular KP Platform implementation. Test Data that must be used to test the KP platform involve all the kinds of data and communication messages that the KP consumes or communicates on each of the various interfaces of the KP Platform. **NOTE:** The Functional Blocks of a KP Platform (i.e. DEs, MBTS, ONIX) may be produced by different players but the KP Platform Supplier is the one fundamentally responsible for the Integration Testing of the KP Platform.

9.2.10 Integration Testing of GANA Functional Blocks (KP DEs, ONIX, MBTS, GANA Levels 2 and 3 DEs in NEs/NFs), and Test Data

After performing an integration Testing of the Knowledge Plane (KP) Platform as a whole, then there is a need to perform an Integration Testing of GANA Functional Blocks (KP DEs, ONIX, MBTS, GANA Levels 2 and 3 DEs in

NEs/NFs) working together as a whole. Aspects that Test Developers should consider in developing Test Cases for Testing the integrated Functional Blocks as a whole are: focusing on specific integration phases and steps by which a set of Functional Blocks are integrated and tested first and then more Functional Blocks are added to the picture based on the results of previous step of integration testing. Test Data that must be used to test the integrated Functional Blocks (as a "bundle") involve all the kinds of data and communication messages that the group of integrated Functional Blocks (the "bundle") of a specific step of integration testing consume through the consolidated boundary interface encompassing the integrated Functional Blocks (as a "bundle") or communicate out on the consolidated boundary interface. This continues until a complete integration has been achieved. **NOTE:** When the Functional Blocks (KP DEs, ONIX, MBTS, GANA Levels 2 and 3 DEs in NEs/NFs) are all coming from the same supplier of these components, a case that may be possible for a specific network segment like the RAN (Radio Access Network (RAN), X-Haul Transport Network, Backhaul Transport Network or Core Network, the Integration Testing may be easier to accomplish.

9.2.11 Performance Testing of individual GANA Functional Blocks (KP DEs, ONIX, MBTS, GANA Levels 2 and 3 DEs in NEs/NFs), and Test Data

There is a need to perform Performance Testing of individual GANA Functional Blocks (KP DEs, ONIX, MBTS, GANA Levels 2 and 3 DEs in NEs/NFs) by applying well-established approaches to performance testing of a system/component. The Performance Measurement Model for GANA and the ideas described earlier in this paper in the whole chapter on "*Convergence Performance Testing with Functional Testing in Testing AI Models: Proposal on Performance Test Framework for the GANA DEs*" should be considered in Performance Testing of DEs, while performance testing of other GANA Components such as ONIX and MBTS should be achieved through the well-established approaches to performance testing of a component/system. Test Data generation and usage in performing performance testing of each component should follow the approaches of adaptive load control as described earlier in the paper. **NOTE:** Performance Testing of the individual GANA Functional Blocks should be performed by the individual suppliers before they are integrated together.

10. Complementary aspects of Design Time Testing (Offline Testing) of AI Models and Run-Time Testing (Online Testing)

An AI-powered Autonomic System or AI system in general, deployed in a production environment, is in dynamic changes itself over time while in operations, and should be able to continuously perform Self-Validation of its own behaviors during its operation lifetime. What this means is that Design-Time Testing (Offline Testing) that is done for an AI Model (or an integrated system with multiple AI Models interworking) must be capable of anticipating and taking into consideration as much as possible, what may happen during the actual Run-Time of the AI Model in real production environment (i.e. during the operation of the AI Model or system in real-life). It also means that the Design-Time Testing that was done at Design Time (the offline testing) of the AI Model must be complemented by Online Testing (Run-Time Testing) of the AI Model when the AI Model is now declared "operational" in the real production/operations environment. The Testing may be done at certain snapshot windows of the operation of the AI Model to take into account various "situations" that would be interesting that the AI Model encounters during its operation. Chapter 8 illustrated certain aspects of Online Testing of AI powered components as "black boxes".

11. The Value of Federated Testbeds and Open API for Interoperable Testbed Federations in the Testing of AI-powered Federated AMC GANA Knowledge Plane (KP) Platforms

The previous chapters covered various aspects on the required standards for testing AI Models, the case of a Generic Test Framework for testing the GANA Model as a Multi-Layer AI Framework for AMC, and also the value of introducing AI in Test systems. But there is also a need to look into requirements for Testing E2E Services such as E2E Network Slices that require the collaboration of multiple AMC platforms to deliver E2E autonomic(closed-loop) service and security assurance across network domains that belong to the same CSP or require multiple CPSs' partners' AMC Platforms (Autonomics/AMC Domains) to collaborate/federate in delivering E2E autonomic service and security assurance. A framework for federated testing of federated AMC using federated testbeds is required.

Testbeds Federation Objectives address the following questions: How to use Components or Capabilities hosted in distributed testbeds to compose a Service or Functionality and then Test the Service or Functionality. In other words, each Testbed hosts capabilities that are complementary to those hosted in other Testbeds. The question is how to create a federation API that can be used by a Test Scenario Composer (Test Team) to identify and instantiate the capabilities in individual testbeds that are part of the federated testbeds ecosystem, as required for an E2E Service Composition or a Functionality for example, and then test the Service or Functionality. Therefore, there is a need for "Open API (Application Programming Interface) for interoperable testbed federations". ITU-T Study Group 11 has launched a new Work Item (WI) on this topic, and ETSI TC INT seeks to support this initiative because Federated Testbeds are key to enabling to test Federated AMC Components and their associated AI Models. Federation of AI-powered Autonomic Systems is extensively addressed in the ETSI TS 103 195-2, ETSI TR 103 473 V1.1.2, and ETSI TR 103 404. Internal federation of AMC (particularly through network segment specific GANA Knowledge Plane (KP) Platforms federations across multiple network segments) can be named as "domestic or local" federation (concatenation of AMC/Autonomics domains that all belong to the same CSP (Communications Service Provider)). But CSP internal AMC domains federation for various network segments results in limited value creation, whereas the current trend of CSPs for creating new value streams can be achieved only through what can be named as "external" federation that involves partnering/peering with other CSPs' multiple Autonomic Systems (AMC Platforms) that are required for E2E Service Delivery Models. One relevant use case is the autonomic creation, orchestration, management, delivery and E2E service and security assurance of E2E 5G Network Slices across partners 5G Networks as described in the 5G PoC White paper #4 [23]. Another use case of high interest to the CSP's (MNOs) community is 5G Roaming. In order to operationalize the Federated Autonomic Systems (i.e. GANA Knowledge Plane (KP) Platforms for AMC) belonging to different CSPs and monetize them, the pre-requisite expected by the Industry is the availability of Federated Testing Framework and Federated Testbeds for Interoperability testing of the federated AMC platforms (GANA Knowledge Plane platforms).

12. Conclusions

The contents presented in this paper are the basis for the work of the newly launched Work Item in ETSI on ***AI in Testing Systems and Testing AI Models*** that is now developing ETSI assets such as Specifications that can be used by the industry. The Work Item created in ETSI is accessible here: https://portal.etsi.org/webapp/WorkProgram/Report_WorkItem.asp?WKI_ID=58442, and organizations are invited to contribute to the work and deliverables of the Newly Launched Work Item in ETSI as described in Chapter 1 of this White Paper. The newly launched WI will produce a set of four documents as described in Chapter 1, and **ETSI TC INT, ETSI TC MTS and ETSI CTI** are the main committees joining efforts to produce the four documents, while **other ETSI TCs and ISGs** are also being invited to collaborate on this work as they and the industry at large shall immensely benefit from this work and the expected deliverables. Other Standardization Bodies or Fora working on AI shall also benefit from this work ETSI has launched and the expected deliverables.

13. References

[1] ETSI White Paper no. 16: The *Generic Autonomic Networking Architecture Reference Model for Autonomic Networking, Cognitive Networking and Self-Management of Networks and Services*: http://www.etsi.org/images/files/ETSIWhitePapers/etsi_wp16_gana_Ed1_20161011.pdf

[2] ETSI TS 103 195-2 (published by ETSI in May 2018): Autonomic network engineering for the self-managing Future Internet (AFI); Generic Autonomic Network Architecture; **Part 2: *An Architectural Reference Model for Autonomic Networking, Cognitive Networking and Self-Management***: https://portal.etsi.org/webapp/WorkProgram/Report_WorkItem.asp?WKI_ID=50970

[3] ETSI 5G PoC on 5G Network Slices Creation, Autonomic & Cognitive Management & E2E Orchestration—with Closed-Loop (Autonomic) Service Assurance for the IoT (Smart Insurance) Use Case: https://intwiki.etsi.org/index.php?title=Accepted_PoC_proposals

[4] A. Prakash, Z. Theisz, and R. Chaparadza: "Formal Methods for Modeling, Refining and Verifying Autonomic Components of Computer Networks", Springer Transactions on Computational Science (TCS) - Advances in Autonomic Computing: Formal Engineering Methods for Nature-Inspired Computing Systems, 2012.

[5] White Paper No.1 of the ETSI 5G PoC: C-SON Evolution for 5G, Hybrid SON Mappings to the ETSI GANA Model, and achieving E2E Autonomic (Closed-Loop) Service Assurance for 5G Network Slices by Cross-Domain Federated GANA Knowledge Planes: https://intwiki.etsi.org/index.php?title=Accepted_PoC_proposals

[6] ODA TM Forum's Open Digital Architecture (ODA): IG1167 ODA Functional Architecture Vision R18.0.0 (Intelligence Management Function Block)

[7] ETSI TR 103 473 V1.1.2: Autonomicity and Self-Management in the Broadband Forum (BBF) Architectures: GANA Autonomics in BBF Architecture Scenarios

[8] ETSI TR 103 404: GANA instantiation onto the 3GPP Backhaul and Core Network architectures

[9] ONAP Open Source Project: ONAP Architecture Overview: https://www.onap.org/

[10] TIP Open Source Project: https://telecominfraproject.com/

[11] BBF CloudCO Open Source Project: https://www.broadband-forum.org/cloudco

[12] OPNFV Open Source Project: https://www.opnfv.org/

[13] ONOS Open Source Project: https://onosproject.org/

[14] OpenDayLight Open Source Project: https://www.opendaylight.org/

[15] ETSI OSM (Open Source MANO): https://osm.etsi.org/

[16] ACUMOS: An Open Source AI Machine Learning Platform: https://www.acumos.org/

[17] White Paper No.3 of the ETSI 5G PoC: Programmable Traffic Monitoring Fabrics that enable On-Demand Monitoring and Feeding of Knowledge into the ETSI GANA Knowledge Plane for Autonomic Service Assurance of 5G Network Slices; and Orchestrated Service Monitoring in NFV/Clouds: https://intwiki.etsi.org/index.php?title=Accepted_PoC_proposals

[18] ODA TM Forum's Open Digital Architecture (ODA): IG1177 ODA Intelligence Management Implementation Guide: R18.5.0 (IG1177 Release 18.5, December 2018)

[19] White Paper No.2 of the ETSI 5G PoC: ONAP Mappings to the ETSI GANA Model; Using ONAP Components to Implement GANA Knowledge Planes and Advancing ONAP for Implementing ETSI GANA Standard's Requirements; and C-SON – ONAP Architecture: https://intwiki.etsi.org/index.php?title=Accepted_PoC_proposals

[20] James Crawshaw: Network Automation Roadmap: Where to Start & What to Aim for: A Heavy Reading white paper produced for Juniper Networks Inc.

[21] Altran White Paper, 2019 by Subhankar Pal, Atul Jadhav, Subhash Chopra: Autonomic Test Automation For Total Testing in the Digital Era: Extending Test Automation Approaches To Newer Technology Domains

[22] Infosys Online Article by Manjunatha G Kukkuru: AI/AUTOMATION: Testing Imperatives for AI Systems: https://www.infosys.com/insights/ai-automation/testing-imperative-for-ai-systems.html
[23] White Paper No.4 of the ETSI 5G PoC: ETSI GANA as Multi-Layer Artificial Intelligence (AI) Framework for Implementing AI Models for Autonomic Management & Control (AMC) of Networks and Services; and Intent-Based Networking (IBN) via GANA Knowledge Planes: https://intwiki.etsi.org/index.php?title=Accepted_PoC_proposals
[24] ETSI 5G PoC Report on Specifications of Integration APIs for the ETSI GANA Knowledge Plane Platform with Other Types of Management & Control Systems, and with Info/Data/Event Sources in general: https://intwiki.etsi.org/index.php?title=Accepted_PoC_proposals
[25] The Work Item created in ETSI on AI in Test Systems and Testing AI Models: (https://portal.etsi.org/webapp/WorkProgram/Report_WorkItem.asp?WKI_ID=58442
[26] EUROPEAN COMMISSION WHITE PAPER (Brussels, 19.2.2020, COM (2020) 65 final): White Paper on Artificial Intelligence—A European approach to excellence and trust,
[27] ETSI EG 203 341: Core Network and Interoperability Testing (INT); Approaches for Testing Adaptive Networks
[28] Object Management Group (OMG)'s Model Driven Architecture (MDA): https://www.omg.org/mda/
[29] The International Journal on Advances in Intelligent Systems is Published by IARIA (ISSN: 1942-2679), vol 5 no 1 & 2, year 2012, http://www.iariajournals.org/intelligent_systems/
[30] Anders Sandven, Master's Thesis in Informatics – Program Development: Metamodel based Code Generation in DPF Editor, Department of Computer Engineering, Bergen University College, 2012
[31] Definitions of various Examples of Metrics for use in assessing and differentiating AI Models: https://towardsdatascience.com/20-popular-machine-learning-metrics-part-1-classification-regression-evaluation-metrics-1ca3e282a2ce ; https://becominghuman.ai/understand-classification-performance-metrics-cad56f2da3aa
[32] Piotr Rygielski, Steffen Zschaler, Samuel Kounev: A Meta-Model for Performance Modeling of Dynamic Virtualized Network Infrastructures: In proceedings of Proceedings of the 4th ACM/SPEC International Conference on Performance Engineering (ICPE'13, Czech), April 2013, DOI: 10.1145/2479871.2479918
[33] 5G End-to-End Architecture Framework by NGMN Alliance: P1-Requirements and Architecture: NGMN 5G E2E Architecture Framework v3.0.8: https://www.ngmn.org/publications/5g-end-to-end-architecture-framework-v3-0-8.html
[34] Spirent White Paper: Assuring the Promise of 5G: A New Approach to Assurance and Testing to Accelerate 5G Innovation, 2018: Rev A | 08/18: https://www.spirent.com/products/service-assurance-5g-lte-sd-wan-ethernet-virtual-networks-visionworks
[35] White Paper No.6 of the ETSI 5G PoC: *Generic Framework for Multi-Domain Federated ETSI GANA Knowledge Planes (KPs) for End-to-End Autonomic (Closed-Loop) Security Management & Control for 5G Slices, Networks/Services:* *https://intwiki.etsi.org/index.php?title=Accepted_PoC_proposals*
[36] Hao Bai: A Survey on Artificial Intelligence for Network Routing Problems: https://pdfs.semanticscholar.org/c5b9/e102c4c058a84f5078d786f857b0df5123af.pdf
[37] Rétvári G., Németh F., Chaparadza R., Szabó R. (2009) OSPF for Implementing Self-adaptive Routing in Autonomic Networks: A Case Study. In: Strassner J.C., Ghamri-Doudane Y.M. (eds) Modelling Autonomic Communications Environments. MACE 2009. Lecture Notes in Computer Science, vol 5844. Springer, Berlin, Heidelberg
[38] Bart Puype, Dimitri Papadimitriou, Goutam Das, Didier Colle, Mario Pickavet, Piet Demeester: OSPF failure reconvergence through SRG inference and prediction of link state advertisements: SIGCOMM'11, August 15–19, 2011, Toronto, Ontario, Canada: ACM 978-1-4503-0797-0/11/08.
[39] Tomoyuki Otani, Hideki Toube, Tatsuya Kimura, Masanori Furutani: APPLICATION OF AI TO MOBILE NETWORK OPERATION: In ITU Journal: ICT Discoveries, Special Issue No. 1, 13 Oct. 2017.

Editors of the White Paper, and Main Contact

ETSI World Class Standards

ETSI 5G PoC Consortium Steering Committee and Contributors:

- *Tayeb Ben Meriem, PhD: Orange: Senior Standardization Manager & Technical Expert: ETSI TC-INT/AFI WG Chair; PoC Steering Committee Member; France*
- *Ranganai Chaparadza, PhD: Altran Germany: Technical Expert & Senior Consultant/Vodafone Consultant; IPv6 Forum; PoC Steering Committee Member; Germany*
- *Michael Mild: Testing Expert & Standardization Expert; Softwell Performance AB; Sweden*
- *Ralf Karbstein: Testing Expert; Senior Account Manager at DATAKOM; Germany*
- *Uwe Baeder: Testing Expert & Standardization Expert; Rohde & Schwarz; Germany*
- *Miguel Roman: Testing and AI Expert; Rohde & Schwarz; Germany*
- *Benoit Radier, PhD: Orange: Standardization & Technical Expert; PoC Steering Committee Member; France*
- *Alain Vouffo, PhD: Testing Expert & Standardization Expert; Spirent; Germany*
- *Dirk Tepelmann: Testing Expert & Standardization Expert; Spirent; Germany*
- *Said Soulhi, PhD: Verizon: Technical Expert and Solutions Design Architect & Standardization Expert; PoC Steering Committee Member*
- *Muslim Elkotob, PhD: Vodafone: Technical Expert and Solutions Design Architect & Standardization Expert; Germany*
- *Takayuki Nakamura: NTT: Technical Expert and Solutions Design Architect & Standardization Expert; Japan*
- *Giulio Maggiore: Telecom Italia: Core, Transport & Service Platforms Engineering & Development, Fixed & Mobile Core Network: Standardization Expert: ETSI TC-INT Chair; Italy*

The Contributors from ETSI TC MTS:

- *Dirk Tepelmann: MTS Chair; Testing Expert& Standardization Expert; Spirent; Germany*
- *Andreas Ulrich: Testing Expert & Standardization Expert; Siemens; Germany*
- *Martin Schneider: Testing Exper t& Standardization Expert; Fraunhofer Fokus; Germany*
- *Gyorgy Rethy, PhD: Testing Expert& Standardization Expert; Ericsson; Hungary*
- *Philip Makedonski: Testing Exper& Standardization Expert t; University of Göttingen; Germany*
- *Kristoffersen Finn: Testing Expert& Standardization Expert; Cinderella; Denmark*

ETSI INT 5G PoC wiki: http://ntechwiki.etsi.org/;
https://intwiki.etsi.org/index.php?title=Accepted_PoC_proposals:

5G PoC Leader and INT AFI WG Chairman: Tayeb Ben Meriem (Orange)
tayeb.benmeriem@orange.com

Acknowledgements: Acknowledgements go to the contributing Organizations and also to the European Commission (EC)-supported *StandICT.eu* Project for supporting these industry activities aimed at progressing Standardization activities linked to 5G and its enablers—under sub-grantee contract number CALL04/20 (by Trust-IT Services Ltd, UK) for one of the contributing technical experts who is also Co-Editor of this paper (and is also contributing to the overall ETSI 5G PoC).

The ETSI 5G PoC Consortium as a Whole:

- Orange
- Verizon
- NTT
- Telecom Italia
- Vodafone
- Altran
- Cellwize
- Huawei
- Incelligent
- QualyCloud
- IPv6 Forum
- Big Switch Networks
- Asocs Networks
- Softwell Performance AB
- Rohde & Schwarz
- DATAKOM
- Check Point

The Contributors from ETSI TC MTS:

- Spirent
- Siemens
- Fraunhofer Fokus
- Ericsson
- Cinderella
- University of Göttingen

Disclaimer: This White Paper expresses the opinion of the ETSI TC INT/AFI WG 5G PoC Consortium Steering Committee and the other contributors.

"This AFI Proof of Concept has been developed according to the ETSI NTECH AFI Proof of Concept Framework. AFI Proofs of Concept are intended to demonstrate AFI as a viable technology. Results are fed back to the Technical Committee on Network Technologies.

Neither ETSI, its Technical Committee NTECH, nor their members make any endorsement of any product or implementation claiming to demonstrate or conform to AFI. No verification or test has been performed by ETSI on any part of this AFI Proof of Concept."