Experiential Networked Intelligence System Architecture

- Dr. John Strassner (FutureWei, Rapporteur)
Overview of ENI Work Item – System Architecture

General information:

<table>
<thead>
<tr>
<th>Creation Date:</th>
<th>2019-09-30</th>
<th>Type:</th>
<th>Group Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Work Item Reference:</td>
<td>DGS/ENI-0016</td>
<td>Latest version:</td>
<td>2.0.18</td>
</tr>
<tr>
<td>Rapporteur:</td>
<td>John Strassner</td>
<td>Technical Officer:</td>
<td>Christine Mera</td>
</tr>
</tbody>
</table>

Scope:
The purpose of this work item is to define the software functional architecture of ENI. This includes:
• defining the functions and interactions that satisfy the ENI Requirements
• defining an architecture, in terms of functional blocks, that can meet the needs specified by the ENI Use Cases
• defining Reference Points that the above functional blocks use for all communication with systems and entities
ENI System Architecture - Mode of Operation and Class

Class 1: An Assisted System that has No AI-based Capabilities

- In each case, ENI requires data from the Assisted System.
- Changes to the Assisted System are not required for any class of Assisted System in order to facilitate the use and rapid adoption of ENI.
- ENI shall use the API Broker to mediate between ENI and the Assisted System.
- ENI provides actionable decisions back to the assisted system (autonomous or recommendatory).
- ENI monitors the effect.
Architecture External Reference Points (Inputs & Outputs)

Functional Architecture with its Input Reference Points

**Imperative, Declarative, and Intent Policies** are handled within the same architecture, with no additional RP or FB needed

Source: ETSI GS ENI 005, ENI System Architecture
Architecture Internal Reference Points

Source: ETSI GS ENI 005, ENI System Architecture
Closed Loop Control in FOCALE

1. Outer Closed Control Loop for a Given Context and Long-Term Optimization
2. Inner Closed Control Loop Triggered by Context Change

- Design
- Orchestrate
- Monitor
- Offline Analytics
- Real-time Analytics
- Optimize

AI Guidance, Policy Enforcement using Model-Driven Engineering
Knowledge Management & Context-Aware Management

Functional Blocks

Knowledge Management
The purpose of the Knowledge Management Functional Block is to represent information about both the ENI System as well as the system being managed. Knowledge representation is fundamental to all disciplines of modelling and AI. It also enables machine learning and reasoning – without a formal and consensual representation of knowledge, algorithms cannot be defined that reason (e.g., perform inferencing, correct errors, and derive new knowledge) about the knowledge. Knowledge representation is a substitute for the characteristics and behaviour of the set of entities being modelled; this enables the computer system to plan actions and determine consequences by reasoning using the knowledge representation, as opposed to taking direct action on the set of entities.

There are many examples of knowledge representation formalisms, ranging in complexity from models and ontologies to semantic nets and automated reasoning engines.

Context-Aware Management
The purpose of the Context-Aware Management Functional Block is to describe the state and environment in which an entity exists or has existed. Context consists of measured and inferred knowledge, and may change over time. For example, a company may have a business rule that prevents any user from accessing the code server unless that user is connected using the company intranet. This business rule is context-dependent, and the system is required to detect the type of connection of a user, and adjust access privileges of that user dynamically.
Cognition Management

Functional Block

Cognition, as defined in the Oxford English Dictionary, is “the mental action or process of acquiring knowledge and understanding through thought, experience, and the senses”. The purpose of the Cognition Framework Functional Block is to enable the ENI system to understand ingested data and information, as well as the context that defines how those data were produced; once that understanding is achieved, the cognition framework functional block then provides the following functions:

- change existing knowledge and/or add new knowledge corresponding to those data and information
- perform inferences about the ingested information and data to generate new knowledge
- use raw data, inferences, and/or historical data to understand what is happening in a particular context, why the data were generated, and which entities could be affected
- determine if any new actions should be taken to ensure that the goals and objectives of the system will be met.

A cognition framework uses existing knowledge and generates new knowledge.

A cognition framework uses multiple diverse processes and technologies, including linguistics, computer science, AI, formal logic, neuroscience, psychology, and philosophy, along with others, to analyse existing knowledge and synthesise new knowledge.
The purpose of the Situation Awareness Functional Block is for the ENI system to be aware of events and behaviour that are relevant to the environment of the system that it is managing or assisting. This includes the ability to understand how information, events, and recommended commands given by the ENI system will impact the orchestration and operational goals and behavior, both immediately and in the near future. Situation awareness is especially important in environments where the information flow is high, and poor decisions may lead to serious consequences (e.g., violation of SLAs).

The working definition of situation awareness for ENI is:

*The perception of data and behaviour that pertain to the relevant circumstances and/or conditions of a system or process (“the situation”), the comprehension of the meaning and significance of these data and behaviours, and how processes, actions, and new situations inferred from these data and processes are likely to evolve in the near future to enable more accurate and fruitful decision-making.*
The Model Driven Engineering (MDE) Functional Block is responsible for enabling software development to be accomplished using models that generate code. MDE represents an approach to software development where models are used in the understanding, design, implementation, deployment, operation, maintenance and modification of software systems. The advantage of MDE is that models are, by definition, machine-readable. Hence, they can be used to specify Functional Blocks, programs, and applications.

A set of models may be defined based on different viewpoints. Formally, a viewpoint is an abstraction of the function and behaviour of a system using a selected set of architectural concepts; this facilitates focusing on a particular aspect or set of responsibilities of the system. ENI defines a number of viewpoints, including business, system, and implementation. Model transformation tools and services are used to align the different models (e.g. deriving a set of data models from an information model), and for generating code.

The function of the MDE Functional Block is to decide how to implement the selected actions from the Situational Awareness Functional Block. It uses model-driven engineering mechanisms to convert the actions into a form that enables imperative, declarative, and/or intent policies to be constructed (by the Policy Management Functional Block).
Policy-based Management

Functional Block

The purpose of the Policy Management Functional Block is to provide decisions to ensure that the system goals and objectives are met. Formally, the definition of policy is:

*Policy is a set of rules that is used to manage and control the changing and/or maintaining of the state of one or more managed objects.*

There are three different types of policies that are defined for an ENI system:

- **Imperative policy**: a type of policy that uses statements to explicitly change the state of a set of targeted objects. Hence, the order of statements that make up the policy is explicitly defined. *In this document, Imperative Policy will refer to policies that are made up of Events, Conditions, and Actions.*

- **Declarative policy**: a type of policy that uses statements to express the goals of the policy, but not how to accomplish those goals. Hence, state is not explicitly manipulated, and the order of statements that make up the policy is irrelevant. *In this document, Declarative Policy will refer to policies that execute as theories of a formal logic.*

- **Intent policy**: a type of policy that uses statements to express the goals of the policy, but not how to accomplish those goals. Each statement in an Intent Policy may require the translation of one or more of its terms to a form that another managed functional entity can understand. *In this document, Intent Policy will refer to policies that do not execute as theories of a formal logic. They typically are expressed in a restricted natural language, and require a mapping to a form understandable by other managed functional entities.*

An ENI system MAY use any combination of imperative, declarative, and intent policy to express recommendations and commands to be issued to the system that it is assisting.
Exemplary Policy Elements

High-level functional block diagram of a Policy Domain

Policy Domain

1. Policy Applications
2. Policy Language Translation
3. Policy Validation (Local Conflict Resolution)
4. Policy Repository
5. Policy Broker
6. Policy Decisions
7. Policy Execution
8. Policy Verification

Event Bus

Other Policy Domains
Future Collaboration

NFV IFA Feature 26
- 4 options for cooperation ENI & NFV IFA
  1. Between OSS and NFV
  2. Embedded in MANO
  3. Closely coupled with MANO
  4. Hybrid

Mano presentation
- OSM Architecture
  - Simplified modules of grouping FBs
  - Degrees of Automation
  - In Future, next release AI machine learning internal functions
  - Open loop automation – calls to an assistant system
  - Today’s MANO is KPI driven
  - Map AI 4 stage decision to OSM
    - AI agent could be resource heavy flexible deployment
    - AI in policy in Release 8, AI decision making in Release 9, full AI may be later

ONAP has a Intent policy that follows an assistance model:
- Step by step or an open framework.

ORAN preliminary thoughts on Intent policy modelling re-use of the network work of ETSI.