INTERNATIONAL TELECOMMUNICATION UNION

TELECOMMUNICATION STANDARDIZATION SECTOR

STUDY PERIOD 2017-2020

FOCUS GROUP ON MACHINE LEARNING FOR FUTURE NETWORKS INCLUDING 5G

ML5G-I-117-R3

Question(s): N/A

Shenzhen, China, 5, 7-8 March 2019

INPUT DOCUMENT

Source: Vishnu Ram OV- Independent Research Consultant India;
Navneet Agrawal- TU Berlin

Title: Unified architecture for ML in 5G and future networks

Purpose: Proposal

Contact:	Vishnu Ram OV Independent Research Consultant, India	Tel: +91 9844178052 E-mail: vishnu.n@ieee.org
Contact:	Navneet Agrawal TU Berlin, Germany	Tel: +49-30-31428-498 E-mail: navneet.agrawal@tu-berlin.de
Contact:	Hucheng Wang DaTang Telecommunication Technology@ Industry Holding Co,.LTD P.R.C	Tel: +86 135 5293 4952 E-mail: wanghucheng@catt.cn
Contact:	Liang Wang ZTE Corporation	Email: wang.liang12@zte.com.cn
Contact:	Liya Yuan ZTE Corporation	Email: yuan.liya@zte.com.cn
Contact:	Mostafa Essa Vodafone Plc., UK	Tel: +201009570327 E-mail: mostafa.essa@vodafone.com
Contact:	Qi Sun China Mobile	E-mail: sunqiyjy@chinamobile.com
Contact:	Yami Chen China Mobile	E-mail: chenyami@chinamobile.com
Contact:	Yan Wang Huawei	E-mail: jason.wangyan@huawei.com
Contact:	Ping Song Huawei	E-mail: songping@huawei.com
Contact:	Minsuk Kim ETRI, Korea (Republic of)	Tel: +82 42 860 5930 Email: mskim16@etri.re.kr
Contact:	Kwihoon Kim ETRI, Korea (Republic of)	Tel: +82 42 860 6746 Email: kwihooi@etri.re.kr
Contact:	Yong-Geun Hon ETRI, Korea (Republic of)	Tel: +82 42 860 6557 Email: yghong@etri.re.kr

Contact:	Masanori Miyazawa KDDI Corporation JAPAN	Tel: +81-80-5985-6331 E-mail: ma-miyazawa@kddi.com
Contact:	Taro Ogawa Hitachi, Ltd., JAPAN	Tel: +81-80-5541-1752 E-mail: taro.ogawa.tg@hitachi.com
Contact:	Hideyuki Shimonishi NEC Corporation , JAPAN	Tel: +81-50-3757-1646 E-mail: h-shimonishi@cd.jp.nec.com
Contact:	Takaya Miyazawa NICT, JAPAN	Tel: +81-42-327-7274 Fax: +81-42-327-6680 E-mail: takaya@nict.go.jp
Contact:	Shoichi Senda NICT, Japan	Tel: +81-42-327-5320 Fax: +81-42-327-5519 E-mail: s.senda@nict.go.jp
Contact:	Ved Prasd Kafle NICT, Japan	Tel: +81-42-327-5471 Fax: +81-42-327-6680 E-mail: kafle@nict.go.jp
Contact:	Kaoru Kenyoshi NICT, Japan	Tel: +81-42-327-5262 Fax: +81-42-327-5519 E-mail: kaoru.kenyoshi@nict.go.jp
Contact	Ping Du The University of Tokyo, Japan	Tel: +81-3-5841-8201 Fax: +81-3-5841-8201 E-mail: duping@iii.u-tokyo.ac.jp
Contact:	Akihiro Nakao The University of Tokyo, Japan	Tel: +81-3-5841-8201 Fax: +81-3-5841-8201 E-mail: nakao@nakao-lab.org

Keywords: Architecture, ML, patterns, use cases, 3GPP, ETSI, issues, standards.

Abstract: The goal of this document is to analyse and unify various contributions to FG ML5G related to ML-aware network architectures. As a result, a comprehensive set of (architectural) requirements are derived from each contribution, which in turn leads to specific architecture constructs needed to satisfy these requirements. Based on these constructs, a logical ML pipeline along with the above said requirements and its realizations in various types of architectures are presented. Finally the key architectural issues facing the integration of such ML Pipeline in continuously evolving future networks are listed.

References

[1] 3GPP TS 23501 System Architecture for the 5G System (Release 15)
[2] ML5G-86R2 WG2 deliverable for MPP use case
[3] ML5G-I-49R2 Mobility Pattern Prediction based on ML
[4] ML5G-I-56R1 High Level ML-aware Network Architecture for ML5G
[5] ML5G-I-85 Data Driven ML empowered Network Architecture for 5G & Future Networks
[6] ML5G-I-81 Requirements of using MPP to enable mobility management customization in 5G network
[7] ML5G-I-72 Applications and optimizations in IoT edge computing using ML
[8] ML5G-I-079R4 Generalization of architecture patterns from use cases
[9] ML5G-I-069 Cognitive Het-Net Use Cases for consideration in WG1
[10] ML5G-I-055R3 Requirements, framework and gaps for Edge Analytics in 5G
[11] Broadband: Acronyms, Abbreviations & Industry Terms https://www.itu.int/osg/spu/ni/broadband/glossary.html
[12] Edgex Wiki https://wiki.edgexfoundry.org/display/FA/Introduction+to+EdgeX+Foundry
[13] IEC whitepaper on Edge Intelligence http://www.iec.ch/whitepaper/pdf/IEC_WP_Edge_Intelligence.pdf
[14] ETSI GS NFV-IFA 014 V2.3.1 (2017-08) Network Functions Virtualisation (NFV) Release 2; Management and Orchestration; Network Service Templates Specification
[15] Intent NBI – Definition and Principles, Open Networking Foundation, ONF TR-523
[16] ETSI GS MEC 003 V1.1.1 (2016-03)
[17] ML5G-I-95R1 Work in progress: Gaps in standards and opensource related to ML for future networks.
[18] https://github.com/cncf
[19] ML5G-I-100 5G Network slicing End to End Resource Allocation Orchestrator Node
[20] Homing and Allocation Service (HAS) https://wiki.onap.org
[21] ETSI SOL002 Network Functions Virtualisation (NFV) Release 2;Protocols and Data Models; RESTful protocols specification for the Ve-Vnfm Reference Point
[22] ETSI GS ZSM 001 V0.4.0 (2018-11) Requirements on the zero-touch end-to-end network and service management
[23] ITU-T Recommendation Q.5001 (ex. Q.IEC-REQ) “Signalling requirements and architecture of intelligent edge computing”
[24] ML5G-I-109-R2 Requirements and use cases of AI/ML for end-to-end network operation automation
[25] ML5G-I-108 AI and ML on ICT environment
[26] 3GPP TS 28.554 Management and orchestration of 5G networks; 5G End to end Key Performance Indicators (KPI) (Release 15)
[27] ML5G-I-087 Grading method for intelligent capability of mobile networks in WG3
[28] ML5G-I-105 Application-Specific Network Slicing through In-Network Deep Learning

Abbreviations and Acronyms

Note: This list includes abbreviations and acronyms not otherwise mentioned in the glossary [11]. The list aims to cover the main terms used in this report, but is not exhaustive.

5GC	5G Core
AF	Application Function
API	Application Programmer Interface
AR/VR	Augmented Reality/Virtual Reality
C	collector (ML pipeline)
CN	Core Network
CNCF	Cloud Native Computing Foundation
CNF	Cloud native network function

CSI	Channel State Information
CU	Centralized Unit
CUDA	CU Data Analytics
DB	Database
DNN	Deep Neural Networks
DU	Distributed Unit
DUDA	DU Data Analytics
EMS	Element Management System
ETSI	European Telecommunication Standards Institute
FMC	Fixed Mobile Convergence
GPU	Graphic Processor Unit
HAS	Homing Allocation Service
M	Model (ML pipeline)
MEC	Mobile Edge Computing
mIoT	massive Internet of Things
MIMO	Multiple Input Multiple Output
ML	Machine Learning
MLFO	Machine Learning Function Orchestrator
ML-ML	Machine Learning Meta-Language
MPP	Mobility Pattern Prediction
MnS	Management Service
NF	Network Function
NFV	Network Function Virtualization
NFVO	Network Function Virtualization Orchestrator
NWDAF	Network Data Analytics Function
NMS	Network Management Subsystem
NOP	Network Operator
NSI	Network Slice Instance
ONAP	Open Networking Automation Platform
OTT	Over The Top
P	Policy (ML pipeline)
P-GW, PGW	Packet Gateway
PP	Preprocessor (ML pipeline)
RAN	Radio Access Network
RCA	Root Cause Analysis
RDA	RAN Data Analytics
RRC	Radio Resource Control
SBA	Service Based Architecture
Sim	Simulator
SO	Service Orchestration
SON	Self-Optimizing Network
src	source (ML pipeline)
UE	User Equipment
V2X	Vehicle-to-everything
VIM	Virtualization Infrastructure Manager
VNF	Virtual Network Function
ZSM	Zero Touch Network and Service Management

1. Terms defined in this document

1.1. ML Pipeline: is defined as a set of logical entities (each with specific functionalities) which can be combined to form analytics function. ML Pipeline node: Each functionality in the ML pipeline is defined as a node (eg. Src, collector, preprocessor, model, policy, distributor and sink).

- src (source): This node is the source of data which could be used as input for ML function. Examples of src could be UE, SMF [1], AMF[1] or any other entity in the network including AF [1].
- C (Collector): This node is responsible for collecting data from the src. It may use specific control protocols to configure the src. Example: it may use 3GPP RRC protocol [1] to configure a UE acting as src. It may use vendor specific OAM protocols to configure a AMF acting a src.
- PP (Preprocessor): This node is responsible for cleaning, aggregating data or performing any other preprocessing needed for the data so that it is fit for ML model to consume it.
- M (Model): This is an ML model. Example could be a prediction function.
- P (Policy): This node provides a control for operator to put in place mechanism to minimise impacts on live network, so that the operation is not impacted. Specific rules may be put in place by operator to safeguard the sanity of the network. Eg: major upgrades may be done only at night times or when traffic is less.
- D (Distributor): This node is responsible for identifying the sinks and distributing the ML output to the corresponding sinks. it may use 3GPP RRC protocol [1] to configure a UE acting as sink.
- Sink: This node is the target of ML output. It takes actions on the ML output. Example: It could be a UE adjusting the measurement periodicity based on ML output.

Note: The nodes are defined in [8] Sec 4.1 and are logical entities which are proposed to be managed in a standard manner (refer MLFO below)) and hosted in a variety of Network Functions.

The realization of such an ML pipeline (in, say, 3GPP R16 or R17 networks) will result in a standard method of introducing and managing ML functionality in a 5G network.

The above symbol is used to denote the ML pipeline in general. When this symbol is used, it denotes a subset (including proper subset) of nodes [8] in the pipeline.

1.2. Interface 8: is defined (see also [8] Fig. 3) as a multi-level, multi-domain collaboration interface, between nodes of an ML pipeline which allows the ML pipeline to be disaggregated and distributed across domains, e.g. between Edge and Core Cloud.

Note: This is a flexible, logical interface, whose realization may depend on extending some of the existing interfaces, say in 3GPP, MEC, Edgex or other specific plaforms.

1.3. MLFO: ML function orchestrator: is a logical orchestrator which may be used for monitoring and managing the ML pipeline nodes in the system. MLFO understands the use case needs of the NOP from the Intent, it understands the resource status from the NFVO, and it monitors the model performance. MLFO selects and reselects the ML model based on the model's performance. The placement of various ML pipeline nodes, based on the use case specifics, and the corresponding capabilities and constraints of the use case is the responsibility of the MLFO. Lifecycle management of the model, in close coordination with the Sandbox is one of the responsibility of MLFO.

1.4. Chaining: Chaining of ML functions or nodes is the process of connecting them together to form the complete ML Pipeline. E.g. A src instantiated in the DU may need to be connected to a Collector and PP in the CU and they in turn, to a Model in the CN to implement the MPP use case. The chain itself is declared by the NOP in the use case specification in the Intent and its technology-specific implementation in the network is done by the MLFO. MLFO utilizes the contraints (e.g. timing constraints for prediction) defined in the Intent to decide the placement and chaining.

1.5. ML-ML: is defined as Machine learning Meta Language. This language is used to specify the constructs needed to add the ML use case and ML pipeline in a declarative fashion into the service design defined in [14].

1.6. Intent: Intent is defined as a declarative mechanism. Intent does not specify any technology implementation methods, such as 3GPP, MEC Network functions to be used in the ML use case. Hence the use cases are technology-independent and provide a basis for mapping ML use cases to diverse technology specific avatars. Intents may use ML-ML as a language for defining use case constructs.

1.7. Sandbox domain: This is a NOP-internal domain where ML models can be trained, verified and its effects on the network can be studied. This may host the monitor-optimize loop, also called closed-loop, and may use a simulator to generate data needed for training or testing (in addition to utilizing data derived from the network). Sandbox could be used by the NOP to train, test, monitor and update models during their lifecycle.

2. Terms defined elsewhere and used in this document

2.1. MEC: Mobile Edge Computing defined in [16] describes a mobile edge system that enables mobile edge applications to run efficiently and seamlessly in a mobile network.

2.2. EdgeX: EdgeX Foundry [12] is a vendor-neutral open source project hosted by The Linux Foundation, which builds a common open framework for IoT edge computing.

2.3. Capability exposure and corresponding mechanisms: The Network Exposure Function (NEF) [1] supports external exposure of capabilities of network functions. External exposure can be categorized as Monitoring capability, Provisioning capability, and Policy/Charging capability.

3. Executive Summary

There is a variety of ways in which ML could be incorporated into 5G and future networks. Any of the layers or strata or NF (network functions) could act as touch points (either source of data which could aid ML or as target for control mechanisms which implement ML decisions) for ML. This was reflected in various architecture contributions to Working Group 3 of FG ML5G.

The analysis of various architecture contributions led us to a unified, logical architecture which represents the technology-agnostic requirements and combines the strong points of all the contributions. This logical architecture could be instantiated using technology-specific realizations: 3GPP is a technology-specific realization, others could be MEC, EdgeX, etc.

The unified logical architecture allows us to share a common vocabulary and nomenclature for the ML functions and their interfaces. By applying (or superimposing) this logical architecture on specific technology like 3GPP, MEC, EdgeX or transport networks, we derive the corresponding technology-specific realization. This gives us the unique ability to analyse both tech-agnostic and specific issues, arrive at general solutions which can be standardised (in ITU) and reused elsewhere (3GPP, MEC, EdgeX).

The scope of this document is to give a brief summary of the requirements presented in detail in various contributions and analysed in [8]. Based on this summary, a unified logical architecture is introduced. A detailed derivation of this unification and ML pipeline is described in [8].

The high-level requirements, listed in section 4 of this document, provide specifications for various aspects of proposed logical architecture. The corresponding aspect of each requirement is listed under "Req applicant" row in the table. These requirements are neither exhaustive nor mutually exclusive. The reader must read these requirements as a set of specifications for corresponding aspect of the architecture.

An ML application can be realized by instantiating logical entities of the ML-pipeline with specific roles (such as src, collector, sink etc.), and distributing these entities among NFs specific to the technology (e.g. 3GPP VNFs), based on corresponding requirements of the logical entities (e.g. a traffic classifier requires to be fed with data summaries every X milliseconds) and capabilities of the node (e.g. computing power at edge).

The flow of information in an ML-based use case can be represented by a ML-pipeline [8] Let us take an example of Mobility Pattern Prediction (MPP) [3]. Reader can refer to Fig.1 for a pictorial overview. The ML algorithms for MPP requires data that are correlated to the location and data-usage patterns of the user. This data might be obtained from various levels of the network. For example, at RAN level, the MIMO CSI measurements gives location bearing of the user w.r.t. the base station, while at transport level, the data-usage patterns of user can be obtained. The data collected at various collection points (source) needs to be gathered (collector) and pre-processed (pre-processor) before feeding this data to the ML algorithm (model). Output of ML-algorithm is then used to apply policies (policy) that will be implemented (sink). Note that although the location of source(s) and sink(s) depends on the ML-use case, but location of other functionalities, such as collector, pre-processor, ML-algorithm etc., are not specific to any use case. Instead these functionalities can be seen as logical entities whose placement (vitualized) depends on the capabilities of the network and requirements of the use case. This logical representation of a ML based network application is called ML-pipeline, and described in detail in [8].

The high-level logical architecture (see Fig 1) has three main components: the management subsystem, the multi-level ML pipeline, and the closed loop subsystem. Together, these subsystems facilitate the functionality and interfaces needed for ML in future networks. These functions defined in the logical architecture are logical functions, these blocks could be hosted in various technology specific network functions and the corresponding interfaces could be realized using extensions of existing technology-specific interfaces. These technology specific avatars of the logical architecture are studied in figures 2, 3, etc.

Several **key-issues** related to ML in 5G and future networks, listed in Section 6, form one of the most important contribution of this study. These key-issues may give rise to new areas of research and extensions of current standards to support future networks, which are beyond the scope of the present document. In essence, the following topics are important:

- Need for an ML-ML (Machine learning Meta Language). This will provide interoperable, declarative mechanism to specify the “intent” of the use case which uses ML in 5G.
- The level of capability exposure needed to enable dynamic ML based use cases in 5G and future networks. The characteristics of capabilities exposure mechanisms for future networks need to be studied, especially in the context of migrating existing networks to future ML based ones.
- MLFO: ML function orchestrator: A logical orchestrator which may be used for monitoring and managing the ML pipeline nodes in the system. Operations on ML in future networks would be controlled via MLFO (eg. Compression, scaling, chaining, updates, optimizations etc).

Future Steps

- Develop a future standardization direction, taking into consideration the activities currently undertaken by the various standards developing organizations (SDOs), industry and open-source forums.
- Liaise with these SDOs and forums to arrive at common understanding and channel contributions into standardization activities on ML in 5G and future networks.

The sections in this document are structured as below. Section 4 describes the high level requirements. These are derived from various use cases and architectural contributions. Section 5 describes both a logical architecture and its technology-specific realizations. Section 6 lists the key issues encountered by the FG while analysing these concepts.

4. High level requirements

In this document, requirements are classified as follows:

- The keywords "**is required to**" indicate a requirement which must be strictly followed and from which no deviation is permitted if conformance to this document is to be claimed.
- The keywords "**is recommended**" indicate a requirement which is recommended but which is not absolutely required. Thus such requirements need not be present to claim conformance.
- The keywords "can optionally" and "**may**" indicate an optional requirement which is permissible, without implying any sense of being recommended. These terms are not intended to imply that the vendor's implementation must provide the option and the feature can be optionally enabled by the network operator/service provider. Rather, it means the vendor may optionally provide the feature and still claim conformance with the specification.

ML-unify-001	Multiple sources of data are **recommended to** be used to take advantage of correlations in data.
Req applicant	Core/general
Significance/Description	In future networks, sources of data may be heterogenous, integrated with different Network Functions, and may report different formats of data. These varied “perspectives” can provide rich insights upon correlated analysis. Example: Analysis of data from UE, RAN, CN and AF is needed to predict potential QoS related issues in end-to-end user flows. Thus, the architecture construct for ML pipeline to be able to collect and correlate the data from these varied sources, is needed.
Traceability/Examples	ML5G-I-069 [9] describes SON modules that monitor and listen to all network alarms, KPI and then takes proper action to clear alarms or enhance network KPIs, or give network design recommendations without human intervention.

ML-unify-002	Multiple technologies and network layers (RAN, Core, Transport, 2G, 3G, future networks) is **required** to be supported, even non-3GPP external elements is **recommended** to be interfaced with, to achieve end-to-end user experience.
Req applicant	Core/general
Significance/Description	Future networks would have multiple technologies coexisting side by side eg. licensed and unlicensed wireless technologies, fixed and wireless converged (FMC) technologies, legacy and future technologies. Emergence of Network slicing [1][19] is one example where vertical technologies (eg. V2X) and their integration into future networks are important. Thus, it is important for that architecture to be capable of overlay with multiple underlying technologies (eg. 3G, 4G, 5G) and even support application functions like in-car entertainment or streaming data from drones or AR/VR headsets. The 5G End to end Key Performance Indicators (KPI) are defined in technology-specific standards (eg. See [26]).
Traceability/Example	ML5G-I-069 [9] describes the cognitive AI concept, in which data from network elements beside some other external elements, such as (sensors, power circuits and different IOT modules) are utilized to control and monitor these elements. This data is then used in various types of network parameter optimizations to achieve gains in coverage, capacity and quality.

ML-unify-003	The network architecture is **required** to support multi-level and distributed instantiation of the ML Pipeline. Data from different levels **may** be able to enrich multiple ML Pipelines (Algorithms) as needed. ML pipelines are **required** to be multi-level and/or multi-domain, connected together via logical interface 8. An ML pipeline **may** be instantiated in multiple levels (eg. core, edge, MEC).
Req applicant	Core/general
Significance/Description	Cloudification, SBA, flexible HAS imply that various network functionalities can be placed dynamically in multiple levels, domains and clouds. Thus, the ML pipeline should also be able to span these levels and interface accordingly. The functionality of the ML pipeline too would span these levels, domains and clouds based on the homing allocation criteria.
Traceability/Example	ML5G-I-069, ML5G-I-79R4, ML5G-I-72. .

	ML5G-I-105 [28] describes a use case where feature extraction (src) can be placed in P-GW and feature data is sent (over interface 8) to MEC where deep learning model is hosted. ML5G-I-81 [6] describes the multiple levels where analytics (ML pipeline) may be hosted in the network (e.g. RAN, AMF) utilizing locally available data as input while performing mobility pattern predictions or slice configurations.

ML-unify-004	When realizing an ML application using the ML pipeline, it is **recommended** that the number of logical entities impacted is kept minimum . In the ideal case, it is **required** to impact only src and sink of the ML pipeline.
Req applicant	Core/general
Significance/Description	Use case definition for ML in future networks will be done based on existing (or new) services or functions in the network. Use case definition for ML has to be loosely coupled with the network service definition in future networks,. Note: Based on the use case definition (refer to Intent), homing and other characteristics of the ML pipeline (eg. chaining) are decided. The Src and the Sink are points of tight integration (eg. APIs) with the technology specific network functions (eg. 3GPP RAN). The other nodes in the ML pipeline may be generic and does not have tight integration with technology specific network functions. Clear interface points between the ML pipeline and the underlying technology are proposed (at the source or sink or MLFO).
Traceability/Example	ML5G-I-79R4. ML5G-I-105 [28] describes feature extraction (src) and traffic classification (sink) placed in the user plane of P-GW. These are points where user plane data is handled by the ML pipeline nodes. The other nodes in the ML pipeline (e.g. the DNN model) does not have such interface dependencies with the 3GPP NFs.

ML-unify-005	Logical entities of the ML pipeline are **required** to be capable of splitting their functionalities and/or be hosted on separate technology-specific nodes. Similarly, multiple logical entities are **required** to be capable of being implemented on single node.
Req applicant	Core/general
Significance/Description	In future networks, homing allocation service for network functions will optimize the location and the performance accordingly. NFVO plays an important role in this. To carry forward such benefits to the ML use case, similar optimizations should be applied to ML pipeline nodes as well. Moreover, the constraints applicable to an ML pipeline (e.g. Training may need GPU and may need to be done in a sandbox domain) may be unique.
Traceability/Example	ML5G-I-49R2 [3] gives scenarios where depending on the latency budget, data availability and other considerations for MPP, ML pipeline could have the src and model hosted in core, edge or MEC.

ML-unify-006	An interface between ML pipelines of multiple levels **may** transfer trained models. (This **may** be one functionality of interface 8)
Req applicant	Interface-8
Significance/Description	Training the models has certain specific needs eg. availability of certain kinds of processors, availability of data. Once the training is done, it has to be sent to the NF which is hosting the model. This could be UE, RAN or CN as examples from 3GPP. Training may be done separately from the live network. Thus, sending trained models across multiple levels (eg. core, edge) is an important requirement.

Traceability/Example	ML5G-I-085 [5] describes the scenario where DUDA hosts the real-time RAN data collection and pre-processing, prediction, parameter optimization and training tasks with low computational complexity in DU. DUDA needs to offer data feature requested for training prediction/decision models to the CUDA (CU Data Analytics) after pre-processing, while CUDA can assist DUDA to conduct some computationally intensive model training tasks. The trained model can be sent to the DUDA for deployment. ML5G-I-105 [28] describes Deep learning done at the MEC and trained model (classifier) updated at the P-GW.

ML-unify-007	Interface between ML pipelines of multiple levels **may** transfer data for training or testing models. (This **may** be one functionality of interface 8)
Req applicant	Interface-8
Significance/Description	Certain domains where the data is available may not have the training capabilities (eg. Resource contrained edge networks). In such cases, there may be a need to send the data for training or testing across to domains where the capacity for such operations is available (eg. Central data center).
Traceability/Example	ML5G-I-085. ML5G-I-108. ML5G-I-105 [28] describes the case where feature extraction from user plane is done at P-GW and the feature data is then sent to the model at MEC for training.

ML-unify-008	It is **required** to be possible to place/host the ML pipeline in a variety of network functions (eg. CN, MEC, NMS, etc) in a flexible fashion. The location of the logical ML pipeline nodes is **recommended** to be decoupled from their functionality, except in case of performance constraints.
Req applicant	MLFO, placement
Significance/Description	An orchestrator function which understands the needs and constraints of ML functions is needed to place/host the ML pipeline nodes at appropriate network functions. Some of the constraints could be availability of data which is specific to the use case, data transformation capabilities, performance constraints, training capabilities and model characteristics (eg. if the model is a neural network, then a GPU based system is desirable). Capability exposure is needed for placement and MLFO exploits this to achieve placement.
Traceability/Example	ML5G-I-49R2 [3] describes the case where based on the requirements of the use case, short-term or long-term predictions, the ML pipeline nodes may be hosted closer to the edge or at the core network. The placement may also be influenced by considerations on data availability. ML5G-I-79R4

ML-unify-009	Certain interfaces (eg. Interface 8) **may** be realized or extended using existing protocols (eg. RRC, GS MEC 011, MnS) [16] Src and Sink **may** need specific interfaces or APIs to extract data or configure parameters. For example: A Src running in the UE may use specific APIs to extract data from an VoLTE client. The ML pipeline **may** use interfaces provided by underlying platform (eg. EdgeX services) as source of data or sink of configurations. In that sense, these platform specific APIs **may** act as realizations of an interface to src and sink. Traditional 3GPP interfaces like Ng and Xn **may** need extensions so that they can realize the needs of the ML pipeline. Inter domain interfaces between edge-core and edge-edge, **may** be abstracted using interface 8,

	but realized using p2p platform specific interfaces (eg. EdgeX interfaces).
Req applicant	Realization, extension (eg. to 3GPP, MEC, EdgeX)
Significance/Description	There may be cases where a tight coupling at integration stage between the ML pipeline src and sink and the NF may not be avoidable. Eg. consider the case where the src runs in the RAN but needs measurements from the UE. In this case, the RAN needs to configure the UE for this measurement using RRC. In certain cases, an extension of such interfaces may be needed to achieve the ML function in the use case.
Traceability/Example	ML5G-I-49R2 [3] describes that Ng interface needs to be extended to support the UE level or flow level related information interaction between RAN (RDA/CU Data Analytics) and core network. ML5G-I-56R1, ML5G-I-085. ML5G-I-072 [7] describes an edgeX based use case where edge-core interface is needed for deep-learning done at cloud and ML based prediction at the edge cloud.

ML-unify-010	It is **required** that a standard method exist to translate the use-case specifications (eg. Intention) into an analytic ML pipeline (defined as **Intent** in rest of the document) It is **recommended** that a machine readable format is used to instruct use-case specifications to MLFO in order to instantiate the ML Pipeline.
Req applicant	Design-time: Intent, ML-ML
Significance/Description	Automation using intent specification and corresponding translation into configurations is a characteristic of future networks. Extending this technique to ML, intent specification of ML use case and correspondingly translate that into pipeline configurations should be supported. Note: Intent specification of ML use case allows overlaying of ML on top of existing declarative specification of network services, eg. those defined in [14].
Traceability/Example	ML5G-I-56R1 [4] describes Intention interpretation is a kind of solution for intelligent configuration. Intention interpretation function can translate it into the configuration that can be implemented in network devices. ML5G-I-79R4

ML-unify-011	Intention is **required** to specify the sources of data, repositories of models, targets/sinks for policy output from models, constraints on resources and use case.
Req applicant	Design-time: Intent. ML-ML
Significance/Description	The separation between technology agnostic part of the use case and technology specific deployment (eg: 3GPP) is captured in the design time of future network services. Intent specification for the ML use cases achieves this separation for the ML overlay. See Sec 1.5 and Sec 1.6 for definitions.
Traceability/Example	ML5G-I-56R1 [4] defines Intent as a template which captures the requirement of the operator. Intention interpretation function can translate it into the configuration that can be implemented in network devices, it goes further to say that the template might even be in a natural language.

ML-unify-012	<u>Any split of the ML pipeline is **required** to be flexible</u> based on the use cases and contraints defined in the intent.
Req applicant	Design-time: Intent, ML-ML
Significance/Description	Platform capabilities may change (hardware may be added or removed), network capabilities

	may change (capacity may increase or decrease), NF's may be scheduled or (re)configured dynamically by the NFVO. These dynamic changes may necessitate a change in the split and placement of the ML pipeline (eg. A decision may be taken to colocate the source and collector based on changes in the link capacity, or a decision may be taken to instantiate a new source based on scale out of a VNF, etc).
Traceability/Example	ML5G-I-56R1

ML-unify-013	(NOP external) 3rd party service providers are **required** to be able to describe the requirements and capabilities for an ML pipeline using Intent.
Req applicant	Design-time: Intent, ML-ML
Significance/Description	3rd party service providers may offer innovative services on top of future networks. For ML, it means new algorithms. A collaboration between 3rd party providers and operators may bring new sources of data or aggregation mechanisms (eg. a new smartphone application which interfaces with sensors on the UE). Intention as a declarative mechanism should extend the capabilities to include such 3rd parties, and they should be able to include these nodes into the specification so that end users can enjoy such innovative services offered by 3rd party service providers. Example of such a use case: 3rd Party (eg. Skype) want to optimize call quality over the network by running ML application that configures network parameters. 3Rd party can setup this ML use-case using interface to ML Pipeline.
Traceability/Example	ML5G-I-56R1 ML5G-I-105 [28] provides a use case where deep learning could be provided as an MEC application.

ML-unify-014	Time constraints of use cases are **required** to be captured in the Intent.
Req applicant	Design-time: Intent
Significance/Description	Different ML use cases have varied time constraints. At the tightest scale, RAN use cases like beamforming, scheduling, link adaptation would have 50us – 100us latency criteria. Next comes transport and 5GC use cases which need 10ms-s latency criteria. The least demanding in terms of latency are management level use cases like anomaly detection, coverage hole detection, etc which can afford minutes, hours or days of latency. These criteria form an important input to the MLFO while determining the placement, chaining and monitoring of an ML Pipeline.
Traceability/Example	ML5G-I-085

ML-unify-015	Placing and split of ML pipeline nodes is **required** to consider various constraints (eg. resource constraints of the NF, latency constraints specific to the use case) The model is **required** to be able to be placed in a flexible manner in an NF which is most optimal for the performance of the use case (eg. As per [26]) and constraints defined in the intention. The split of the ML pipeline is **required** to be flexible based on the use cases and constraints defined in the intention. Placement and hosting of ML pipeline are **required** to be flexible based on constraints mentioned in the intent. eg. placement could be in the core, RAN, or MANO. It is recommended that the constraints for online training and prediction for real-time applications (e.g., 1ms~10ms) are captured in the intent. This **may** be input to placing these nodes in Network Functions which can provide optimal performance for the use case (eg. As per [26]).
Req applicant	MLFO, intent, constraints
Significance/Description	The positioning and placement of ML pipeline nodes onto VNFs forms a major part of the

	realization of the ML use case with a specific technology (eg. 3GPP). Thus, it forms the link between 2 domains: tech-agnostic ML pipeline (overlay) and tech specific network underlay (eg. 3GPP). The needs, constraints and status of each domain needs to be taken into consideration while making this mapping or linkage. Thus, these requirements form an important part of MLFO which achieves this mapping of overlay to underlay and provides a smooth migration path to the operator.
Traceability/Example	ML5G-I-49R2, ML5G-I-56R1, ML5G-I-085. ML5G-I-105 [28] describes user plane data classification using DNN. Since this is a latency-senstive application, the model is hosted at the P-GW, whereas the training could be done at MEC.

ML-unify-016	Model selection is **required** to be done at the setup time of the ML pipeline, using data from the src.
Req applicant	MLFO: setup
Significance/Description	Advances in ML algorithmics suggest that in future networks there would be models with varied characteristics (eg. using variety of optimization techniques and weights) which suit different problem spaces and data characteristics. ZSM [22] requirement brings in discovery and onboarding of source of data dynamically. To extend the ML use case to such devices and sources of data, model selection has to be done dynamically, based on the data provided by the source.
Traceability/Example	ML5G-I-56R1 describes the requirement for model selection based on requirement of the operator specified in the intent.

ML-unify-017	Model training is **required** to be done in sandbox using training data. A sandbox domain **is recommended to** be used to optimize the ML pipeline. Simulator functions hosted in the sandbox domain **may** be used to derive data for optimizations.
Req applicant	Non-functional (sandbox)
Significance/Description	Model training is a complicated function, it has several considerations: use of specific hardware for speed, availability of data (eg. Data lakes), parameter optimizations, avoiding bias, distribution of training (e.g. Multi-agent reinforcement learning), the choice of loss function for training. training approach used exploration of hyper parameters, are some examples of them. Moreover, in future networks, operators would want to avoid the disruption of service while model training and update are performed. These considerations point to use of a simulator for producing the data for training the models, as well as its use in a sandbox domain.
Traceability	ML5G-I-56R1, ML5G-I-79R4

ML-unify-018	The capabilities to enable a closed loop monitoring and update, based on the effects of the ML policies on the network, are **required**.
Req applicant	Non-functional (closed loop)
Significance/Description	Closed loop is needed to monitor the effect of ML on network operations. Various KPIs are measured constantly and the ML algorithm's impact on them as well as on the ML pipeline itself (due to operations of the MLFO) are monitored and corrected constantly. These form inputs to the simulator which generate data that can cover new or modified scenarios accordingly in future (eg. a new type of anomaly is detected in the network, the simulator is modified to include such

	kind of data which can train the model to detect that too).
Traceability	ML5G-I-56R1. ML5G-I-109-R2 describes the case where continuous improvement of the automated fault recovery process workflows is important. Hence, not only the RCA is provided to the autonomous functions for configuring the NFVO, the effect produced by ML in autonomous functions are evaluated and used in a closed loop to optimize the autonomous function itself.

ML-unify-019	A logical orchestrator (MLFO: ML function orchestrator) is **required** to be used for <u>monitoring</u> and managing the ML pipeline nodes in the system. MLFO monitors the model grading (refer [27]), and Model <u>re-selection</u> is **recommended** to be done when the grading falls below a predefined threshold.
Req applicant	MLFO (monitoring)
Significance/Description	The varied levels and sources of data (core, edge), including the simulator and the sandbox domain implies that there could be various training techniques including distributed training. Complex models which are chained (or derived) may infact be trained using varied data. The performance of such models could be determined and compared in the sandbox domain using simulator. Based on such comparisons, operators could then select the model (based on internal policies) for specific use cases. This could be used in conjunction with MLFO to re-select the model. Note: grading may involve network performance grading also along with model grading.
Traceability	ML5G-I-79R4, ML5G-I-56R1

ML-unify-020	<u>Flexible chaining</u> of ML functions is **required** to be done based on the hosting and positioning on different NFs and domains. This is to realize the hybrid or distributed ML functions (refer the traceability below for details). ML pipeline deployment **may** be split and multi-level. <u>Chaining of ML pipeline nodes across these levels</u> **may** be needed to achieve the use case (eg. the split of ML functions based on gNB-CU/gNB- DU architecture defined in [5]). Chaining of logical functions **may** be used to build complex analytic ML pipeline.
Req applicant	MLFO (chaining)
Significance/Description	NFV architecture along with SBA and the emergence of service orchestration mechanisms like ONAP will enable operators, in near future, to rapidly design, develop and deploy network services. MLFO needs mechanisms including flexible chaining to keep up with the innovation in the underlay space. As the underlying network service evolve and deploy rapidly, so does the ML pipeline on top of them, using these MLFO techniques. This requirement aims to give the ML pipeline overlay, the ability to adapt to dynamic service creation and orchestration.
Traceability	ML5G-I-56R1, ML5G-I-085, ML5G-I-081, ML5G-I-79R4

ML-unify-021	<u>Support for plugging in and out new data sources or sinks</u> to a running ML pipeline is a **requirement**
Req applicant	MLFO (unstructured data)
Significance/Description	Certain advanced network services to be defined in future networks (eg. mIoT) require handling of unstructured data from a huge number of sources which may be under ZSM [22]. One such use case is the analysis of logged data for anomaly detection in networks. MLFO needs mechanisms to perform operations like selecting models based on metadata derived from unstructured data and scaling the ML pipeline nodes based on the incoming data.

Traceability	ML5G-I-072

ML-unify-022	A sharing mechanism for data and inputs between various nodes in the pipeline is **recommended** to be in the form of a distributed, shared, highly performant data storage.
Req applicant	General: core: DB
Significance/Description	Cross layer, cross domain sharing of data across various levels, domains and clouds is needed to take correlated ML decisions in future networks. Concepts like data lakes are emerging in future clouds and can be exploited in operator clouds too. Governance mechanisms for data are mandated by regulations in certain areas.
Traceability	ML5G-I-79R4, see also fig 5a and 5b in this document.

5. Unified Architecture

Unified architecture stands for a common high level logical architecture for ML in future networks. However, to understand the deployment options in various technology domains, this architecture has to be considered along with its technology-specific realizations. These are described in this section.

5.1 Unified Logical Architecture

The unified logical architecture is derived from the high level requirements described in the previous section. Reuse of existing standards wherever possible, is a guiding principle applied while arriving at this architecture. This exercise allows us to study the gaps of existing standards[17]. The level of abstraction used while deriving this logical architecture is such that, while all the basic requirements can be captured using these building blocks, extensions and technology specific customizations are possible in each standard domain (eg. 3GPP).

The three main building blocks of the unified logical architecture (Fig 1) are:

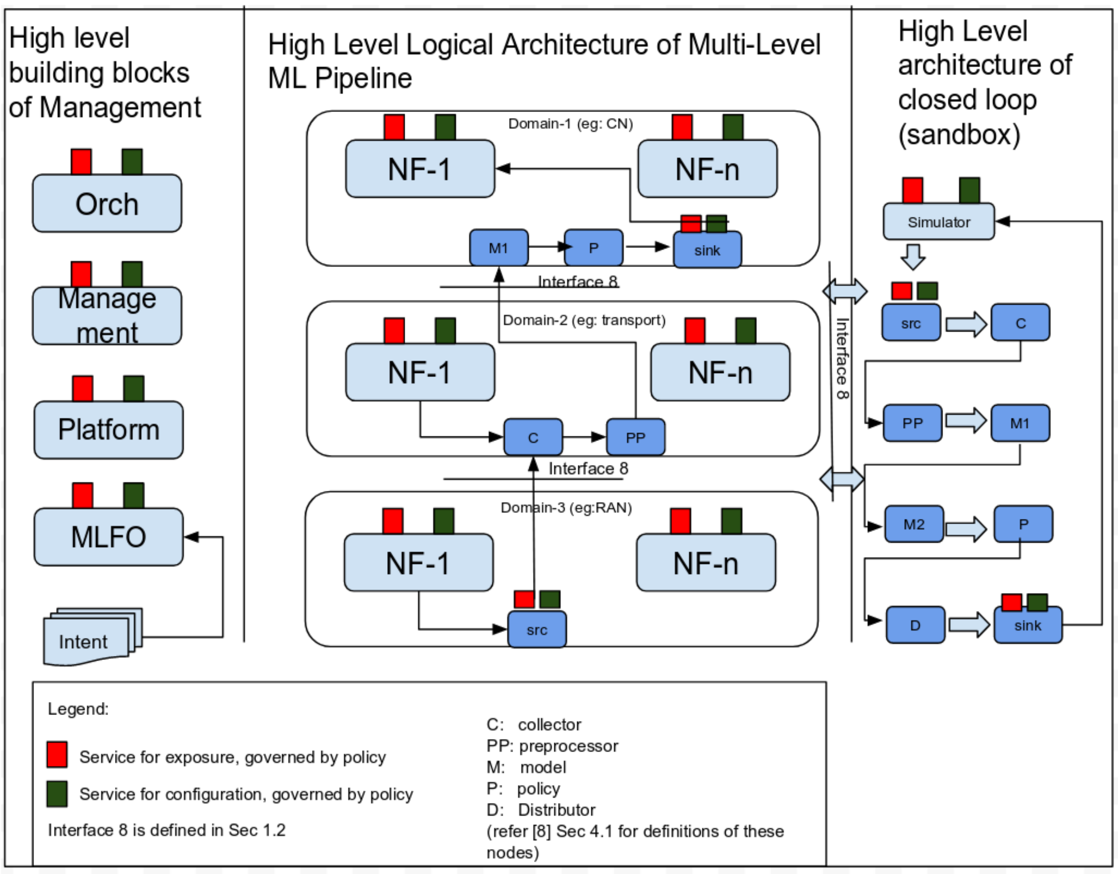
- Management subsystem: This includes orchestration, various existing management entities (eg. VNFM, EMS), management of platform (eg. VIM). In addition, we also define a new logical entity MLFO (refer to Sec 1.3 above). Monitoring and management of these functions is achieved using service based architecture defined in [1]. Intent (refer to Sec 1.6) allows the operator to specify and deploy ML services on top of existing ones without tight coupling with the underlying technology used for realization.

- Multi-level ML pipeline: The ML pipeline (refer to Sec 1.1) is a logical pipeline, which can be overlaid on existing network functions (eg. VNFs as defined by ETSI or CNF as defined by CNCF). MLFO provides instantiation and setup services for the ML pipeline, using in turn the services of the NFVO where needed. The deployment of an ML pipeline may span across different levels and domains or clouds. MLFO coordinates this deployment. In this context, interface-8 (refer to Sec 1.2) is important to achieve the chaining of such a multi-level deployment. Specific integration aspects of such an overlay of an ML pipeline on a specific technology (e.g. how to integrate ML pipeline across various NF's in 3GPP CN and RAN) may require extension of existing interfaces or definition of specific APIs. Some of these would be part of the standards gap analysis [17].

Note-1: in Fig 1, we show examples of three domains: CN, transport and RAN. These are treated as administrative domains. These may be owned, operated and administered by different entities (e.g.

in roaming cases).
Note-2: in Fig 1, we show an example of the MLP (ML Pipeline) overlay on these domains. However, based on specific use cases, other ways of distributing the MLP nodes are possible. These are shown in section 5.2.

- Closed-loop subsystem: future wireless networks present a dynamic environment. Various conditions may change in the network (eg. air interface conditions, UE's position, platform capabilities and platform capacity etc). A closed loop subsystem allows the ML pipeline to adapt to this dynamic environment. It is driven by a simulator and monitored by the MLFO using the parameters defined in the Intent. Such a “sandbox” environment allows operators to study the effect of ML optimizations before deploying them on a live system. As mentioned in Sec 1.6, updates from the network are fed back into the closed loop so that the ML pipeline can adapt to dynamically changing environment in the network.


High level building blocks of Management
Orch
Manage ment
Platform
MLFO
Intent
High Level Logical Architecture of Multi-Level ML Pipeline
Domain-1 (eg: CN)
NF-1
NF-n
M1
P
sink
Interface 8
Domain-2 (eg: transport)
NF-1
NF-n
C
PP
Interface 8
Domain-3 (eg:RAN)
NF-1
NF-n
src
Interface 8
High Level architecture of closed loop (sandbox)
Simulator
src
C
PP
M1
M2
P
D
sink
Legend:
🟥 Service for exposure, governed by policy
🟩 Service for configuration, governed by policy
Interface 8 is defined in Sec 1.2
C: collector
PP: preprocessor
M: model
P: policy
D: Distributor
(refer [8] Sec 4.1 for definitions of these nodes)

5.2 Realization(s) of the logical architecture

Fig 2 below (see also [3]) gives an example of realization of the logical architecture on a 3GPP system along with MEC and management systems. The realization is achieved in the following manner:

- Use the ML Pipeline (refer to Sec 1.1) to show the positions in this realization wherever the nodes in the ML pipeline could be hosted. e.g. CN, RAN, MEC, NMS, etc.
  - Consider arrows: 1->2->4->ML pipeline1: This pipeline uses inputs from the UE to make predictions at CN (e.g. MPP based use cases).
  - Consider arrows: 9->2->4->ML pipeline1: This pipeline uses inputs from the RAN and possibly a combination of UE and RAN, to make predictions at CN (e.g. MPP based use cases).

    - Consider arrows: 10->7->ML pipeline2->8: This pipeline uses inputs from the MEC platform to make predictions at the edge and apply these back to MEC. It could also use side info from UE and RAN (e.g. caching decisions made at the MEC, local routing decisions at the MEC).
    - Consider arrows: 3->4->ML pipeline1->5: This pipeline uses inputs from CN and possibly a combination of UE and RAN inputs to make predictions at CN, apply it to NMS parameters which may in turn affect configurations in different domains (e.g. SON decisions made at the CN).
    - Consider arrows: ML pipeline3->6: local predictions at the NMS which may in turn affect configurations in different domains (e.g. parameter optimizations based on data analytics).
- Call out extensions in 3GPP interfaces or MEC interfaces where applicable.
    - Consider arrow 1 below: this could be realized as an extension of RRC.
    - Consider arrow 2 below: this could be realized as an extension of N2 interface [1]
    - Consider arrow 5 below: could be realized via a reuse of ETSI SOL002 [21]
    - Consider arrow 10 below: this could be realized via a reuse or extension of GS MEC 011 [16]
- Give instances of constraints applicable for placement of the ML pipeline in 3GPP.
    - UE is a resource constrained device, hence only a source is instantiated in the UE.
    - As mentioned in ML-unify-014, RAN and MEC might have latency constraints on their use cases. Hence those models are hosted in the RAN itself as ML pipeline 2. Those data from the RAN and UE which are not used in such latency-bounded use cases, are sent to the CN via arrow 2 below.

Fig 2: Multi-level ML pipeline in 3GPP and MEC

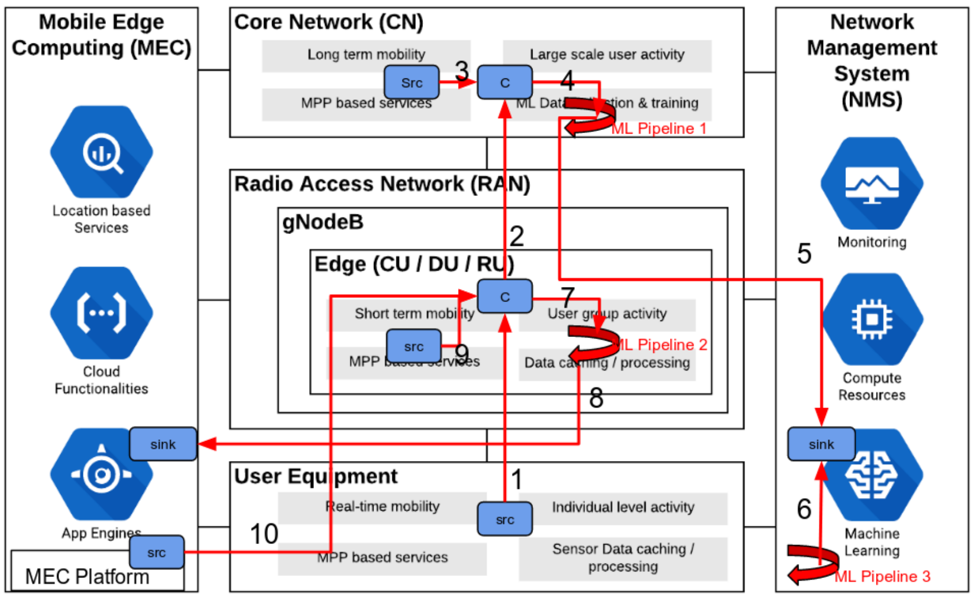


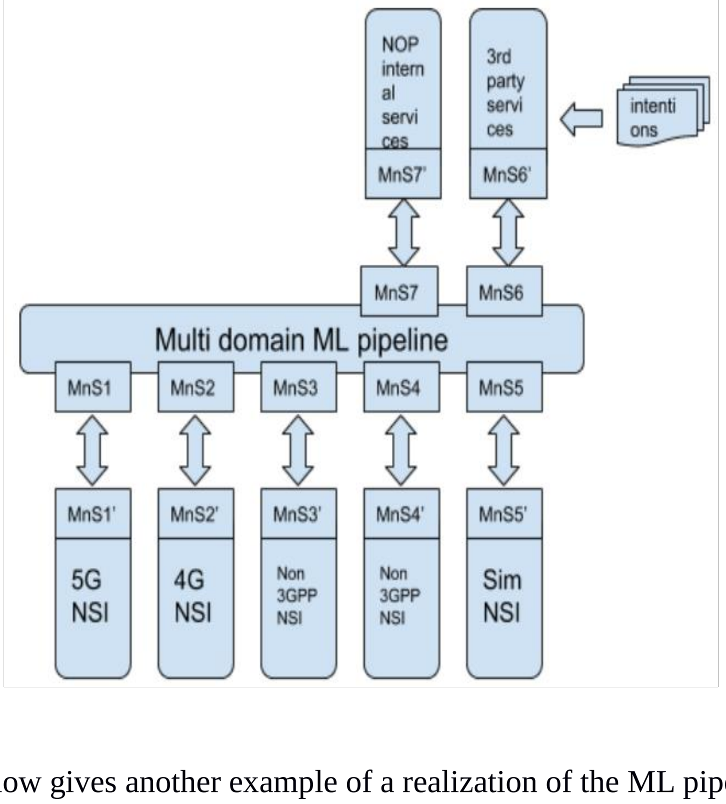
Fig 3 below (see also [1]) gives the interface points for the logical ML pipeline with various technology services. The interface points are achieved in the following manner:

- The ML pipeline is loosely coupled with 3GPP and other technologies. The ML pipeline has clear interface points where it interacts with 3GPP Network services. This allows the ML pipeline to evolve separately from underlying technologies, while allowing all forms of 3GPP and non-3GPP networks, even simulated ones, to benefit from ML services.
- In Fig 3, MnSx stands for producer of analytics services (and consumer of data) whereas MnSx' stands for consumer of analytics service (and producer of data).

Note: data may be shared as described in ML-unify-022.

- This also provides an interface point for 3rd party service providers who may provide innovative services on top of future networks. These may be ML-based, e.g. new ML algorithms or optimization mechanisms, or an OTT service (eg. VoLTE). A plugin mechanism for these 3rd party providers is needed to handle the ML needs of such services. Intention as a declarative mechanism should extend the capabilities to include such 3rd parties, and they should be able to include these nodes into the specification so that end users can enjoy such innovative services. Please see ML-unify-13 for an example of such a service.
- Network Operator may introduce services (eg. SON based on analysis of data from the network) which takes advantage of the ML pipeline. These too, interface with the ML pipeline instances via MnS interface points.

Fig 3: MnS based interface points for ML Pipeline

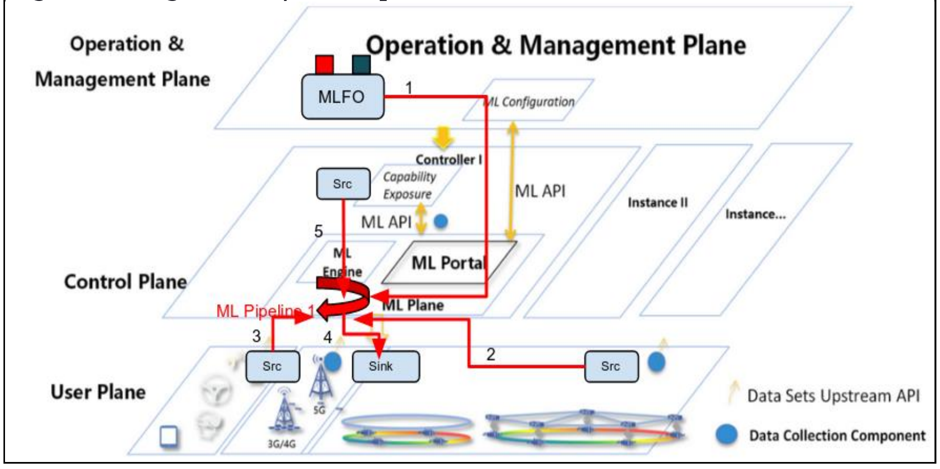


In Fig. 4 (see also [4]) below gives another example of a realization of the ML pipeline in a 3GPP network. Here the focus is on:

- Standard mechanisms for configuration of the ML pipeline using MLFO. MLFO inturn may use Intent as an input.
- The MLFO interface (see arrow 1 below) includes monitoring and management of the ML pipeline using MLFO (including model selection and reselection, see also ML-unify-019).

- Distributed and multi level placement of src, sink and ML pipeline in general, using MLFO.
- Slice creation based on user data (see [28]), integration with application logic (see [9]) are examples of use cases which can be achieved by placement of src and sink as shown below. Such placement of src and sink may require specific interfaces which integrate the data collection (see arrows 2,3,5) and network configuration (see arrow 4) respectively.

Fig 4: Management of ML Pipeline



In Fig. 5 below (see also [7]), we demonstrate the realization of the ML pipeline in the EdgeX [12] which is a vendor-neutral open source software platform. This enables interaction with devices, sensors, actuators, and other IoT objects. It provides a common framework for Industrial IoT edge computing. The supporting services (SS) layer of EdgeX layer provide edge analytics and intelligence.
To demonstrate the realization of the ML pipeline:

- ML Pipelines are instantiated in core cloud and edge clouds. For example, in Fig. 5, three ML pipeline instances are shown. They coordinate using interface 8 (refer Sec 1.2).
  - Arrow-4 shows a local analytics service (ML pipeline 2) based on platform inputs.
- In the core cloud, an ML pipeline will have the core NFs as src. This may be used in correlation with data from the edge (refer to [7]).
  - Arrow-5 shows a local analtics service (ML pipeline 3) based on local inputs at the core cloud and forwarded inputs from edge cloud 1 via arrow 2.
- In the edge cloud, an ML pipeline will have EdgeX platform service and its other services as src. These may be used in correlation with the data from another edge (e.g. in the 3GPP V2X use case or mobility case mentioned in [23]).
  - Arrow-1 shows a "store and forward" of data collected at edge cloud 2 to edge cloud 1. This data is then analysed by ML pipeline 1.
- Collaborative interfaces Fig 5 would be realized using interface 8 (refer to Sec 1.2). This helps in enriching the data available at any pipeline with side info from other instances.
  - Arrow-3 shows a local analytics service (ML pipeline 1) based on local inputs and forwarded inputs from edge cloud 2.

Fig 5: EdgeX-based instance of ML pipeline

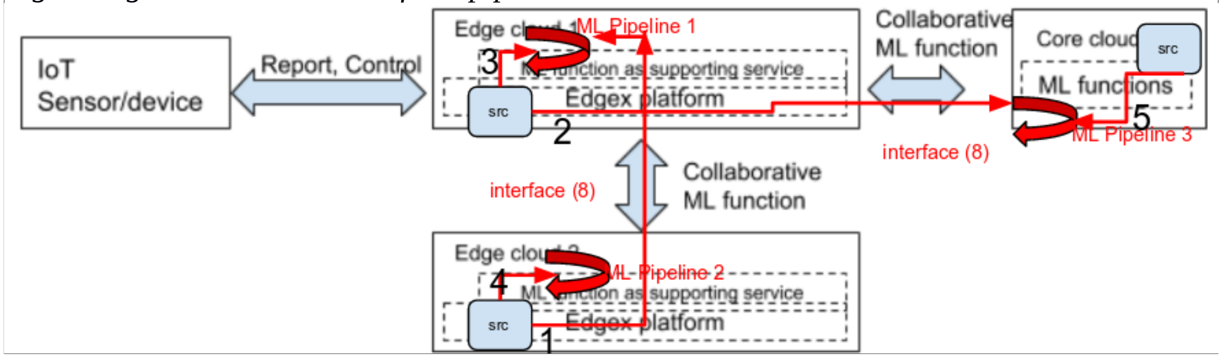


Fig 5a and 5b shows specific examples of this instance:

Fig 5a shows Edge-platform-based ML prediction. It may include:

- Intelligent network traffic control technology Edge computing technology based on EdgeX (Open source edge platform).
- Intelligent Traffic Analysis and ML-based prediction (eg. Deep learning, Reinforcement learning)
- Hybrid intelligent network control approach combined by optimizing prediction and control
    - Fig 5a shows: (arrow 1) Collecting data from sensor/device, (2) Stored data in EdgeX core service. (3) Data preprocessing by EdgeX expert service, (4) Transforming data-set to cloud for ML training in sandbox domain, (5) Model serving with the trained model from ML prediction in EdgeX, (6,7) ML decision in EdgeX platform.

Fig 5a: Edge-platform-based ML prediction

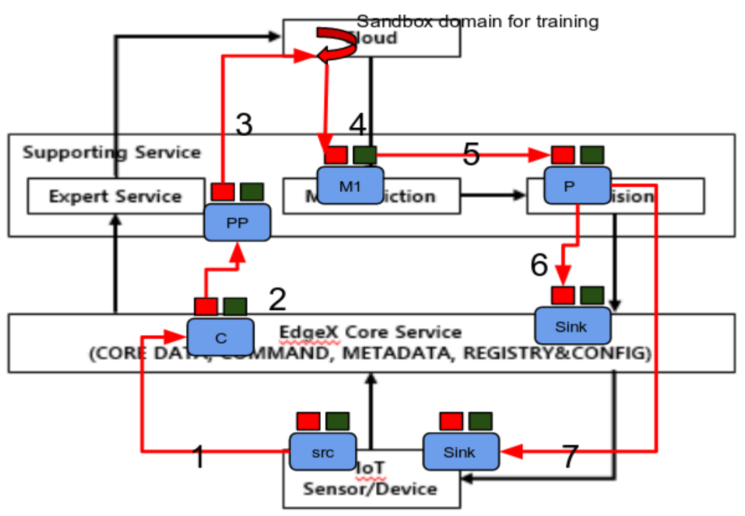
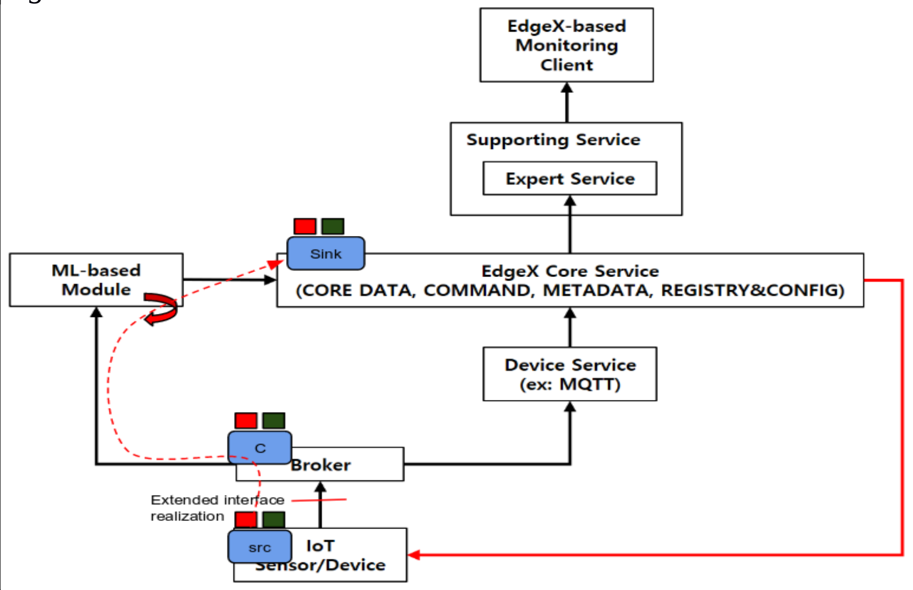


Fig 5b shows real-time monitoring and control service based on edge computing. It may include:

- Intelligent IoE edge-based monitoring and control system for analyzing and processing real-time sensor optimization
- ML-based intelligent IoE service and optimal control system using edge computing
- Intelligent real-time edge computing solution
- Fig 5b shows: (1) Real-time sensor/device data collecting by EdgeX device service via broker, and also collecting and processing data following by ML-based module, (2) Collecting and saving the data in DB (eg. MongoDB) via EdgeX core service, (3) Real-time monitoring the stored sensor/device data from EdgeX-based monitoring client via EdgeX

expert service, (4) Optimizing real-time control to IoT sensor/device via EdgeX core

Fig 5b:


EdgeX-based Monitoring Client
Supporting Service
Expert Service
Sink
ML-based Module
EdgeX Core Service (CORE DATA, COMMAND, METADATA, REGISTRY&CONFIG)
Device Service (ex: MQTT)
C
Broker
Extended interface realization
src
IoT Sensor/Device

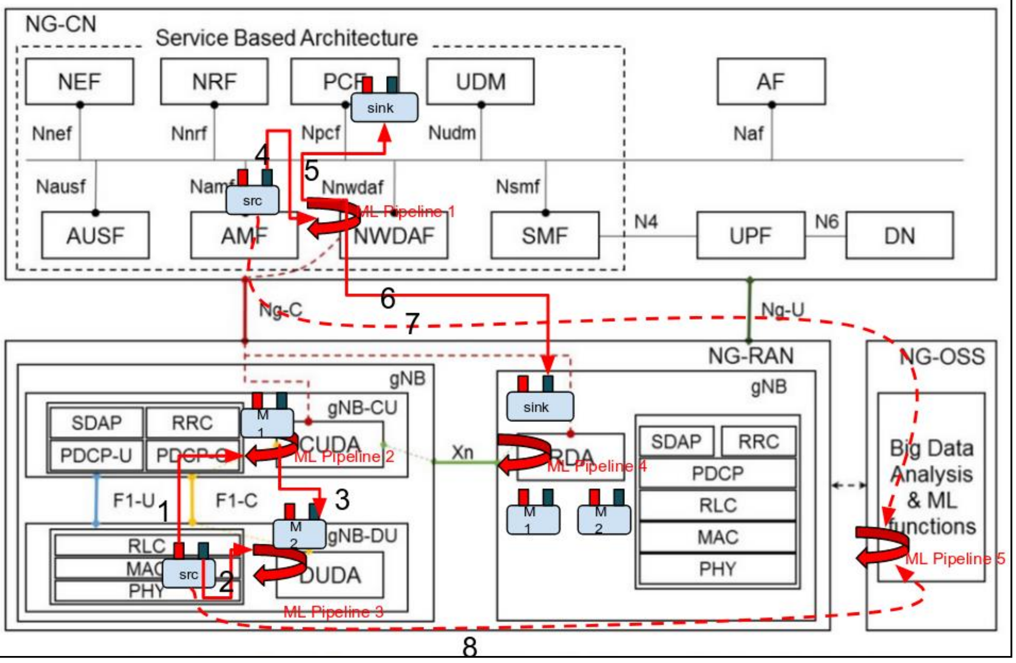
Real-time monitoring and control service based on edge computing

service.

Fig 6 below (see also [5]) shows a unique example where:

- In a Service based architecture, the ML pipeline is hosted in the NWDAF [1].
  - ML pipeline 1 may have AMF as src (arrow 4) and PCF as sink (arrow 5) to realize a particular use case (e.g. mobility based policy decisions).
- Resource constrained DU hosts part of the ML pipeline but not the training. The training is done at the CU and the trained model is distributed to the DU, where it is hosted.
  - DU hosts M2 which is updated from CU via arrow 3.
  - Data for training the model in CU is provided via arrow 1.
- Collapsing of the interface between ML pipelines is an option as shown in the RDA option . This brings out the need for flexibility in deployment. See ML-unify-012, ML-unify-015 and ML-unify-020.
  - M1 and M2 are hosted in CUDA (CU Data Analytics) and DUDA (in ML pipeline 2 and 3 respectively) in the 3GPP split deployment, whereas they are collapsed (merged) in the other 3GPP alternative deployment options [1].
- The extension of 3GPP interfaces for carrying information specific to ML pipeline execution and training is a requirement here.
  - Eg: (see [5]) RDA is primarily used to support optimization in the RAN. It also needs to provide data subscription services for NWDAF and business & operation Support System (OSS)/OSS/MANO, and
  - upload pre-processed subscription data to NWDAF and BOSS/OSS (arrow 8) for further big data operations and services.
  - RDA can also subscribe to the NWDAF data analysis results for the RAN-side service optimization (arrow 6).

Fig 6: Hierarchical, distributed instances



Note: CUDA stands for CU Data Analytics.

In Fig. 7 below (see also [5]), we describe the mechanism of incorporating timing constraints of various 3GPP use cases into their realization using ML pipeline. The timing constraints are captured in Intents, which are in turn processed by MLFO to determine instantiation choices, like positioning of various ML pipeline nodes. In Fig. 8, RAN use cases have strictest latency constraint (50us – 10ms). Therefore, the MLFO may choose to position the entire ML pipeline 2 in the RAN. In contrast, use cases related to 5GC have 10ms-s latency budgets. Hence, the MLFO may choose to enrich the data in ML pipeline 1 with side information from the RAN. The same is applicable to ML pipeline 3.

Fig 7: Timing Constraints and Intents

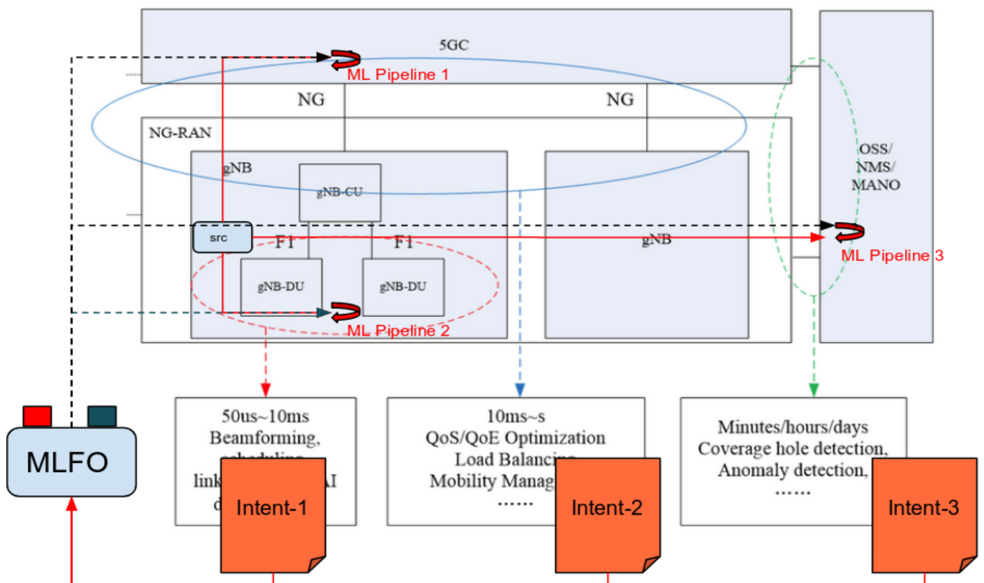

5GC
ML Pipeline 1
NG
NG
NG-RAN
gNB
gNB-CU
OSS/ NMS/ MANO
src
F1
F1
gNB
ML Pipeline 3
gNB-DU
gNB-DU
ML Pipeline 2
MLFO
50us~10ms Beamforming, scheduling, link ... AI ...
Intent-1
10ms~s QoS/QoE Optimization Load Balancing Mobility Manag... ......
Intent-2
Minutes/hours/days Coverage hole detection, Anomaly detection, ...
Intent-3

Fig. 8 (see also [6]) shows a unique realization where NWDAF functions are hierarchical across three domains: CN, AMF and RAN. This split allows certain specific data to be used for local decisions at these NFs. From an ML pipeline perspective, this would mean that the pipelines are chained, so that the output of one could feed into the input of another.

- Arrow 1, arrow 2, show the control of NS manager using the ML pipeline in NWDAF. This enables use cases like dynamic Slice configuration using ML.
- Arrows 3,4, and 5,6 are local NWDAF functions in AMF and RAN respectively (e.g. AMF can customize connection management, registration management, mobility restriction management for UEs based on the long term UE MPP).
- Arrow 7 shows chaining, so that the output of a remote NWDAF can feed into the local NWDAF as input (e.g. while performing short term MPP).

Fig 8: Hierarchical Instances

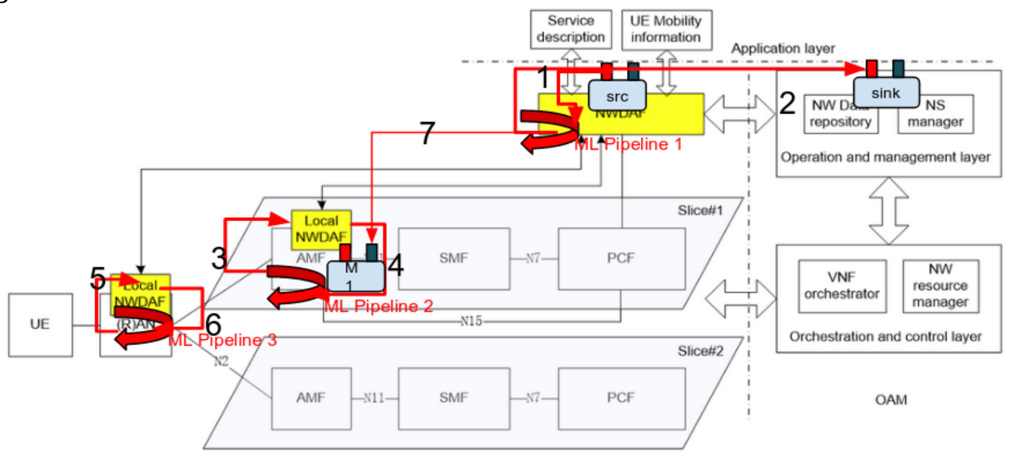


Fig. 9 shows the proposed architecture [24] to achieve closed loop automation in operation and management on 5G networks.

- The management system should be automated to the extent possible to promptly react to failures in the NFV. The operator wants to promptly discover such failures, which result in increasingly unstable behavior before the process escalates into critical failure. Root cause analysis is also important to properly convey relationship information between failure type and location to automation function.
  - Line 1 in Fig 9 shows the src – collector interface. Line 2 shows the Collector to ML pipeline interface. Line 3 conveys the result of the ML pipeline (RCA or predictive detection) to the policy function in the Automation function.
  - NFVO is configured based on policy or workflows (line 4).
- Continuous improvement of the automated fault recovery process workflows is important.
  - Line 6 provides the output corresponding to this improvement to the sink hosted in the automation function.
- As mentioned in ML-unify-019, the ML pipeline is configured and monitored by MLFO via Line 5.

Fig 9: ML in closed loop automation

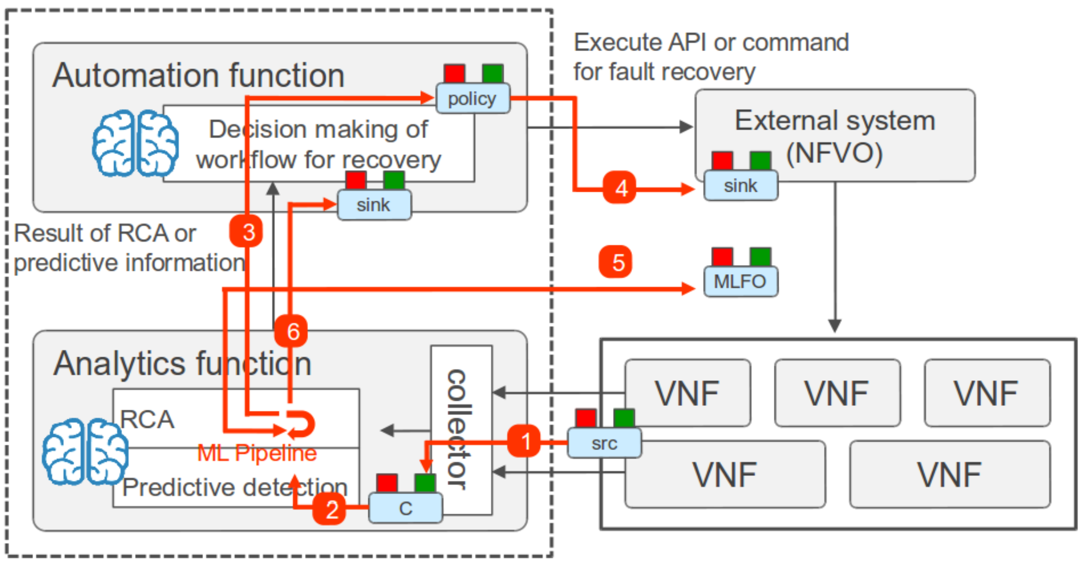
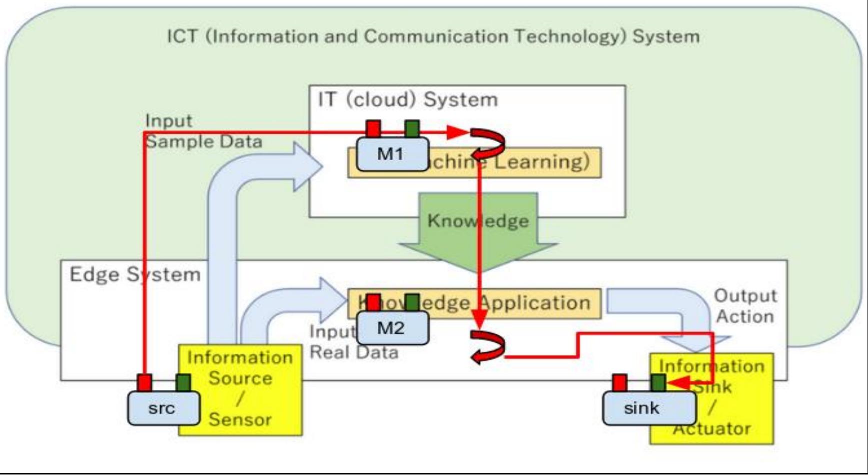


Fig 10 shows ML Model executed on cloud server, when there is no severe timing requirement. However use cases which require severe timing conditions are expected to be executed on edge part using the outcome of ML executed on cloud server. There may be transfer of learning [25] or output between the models in the two domains.

Fig 10: transfer of learning between domains

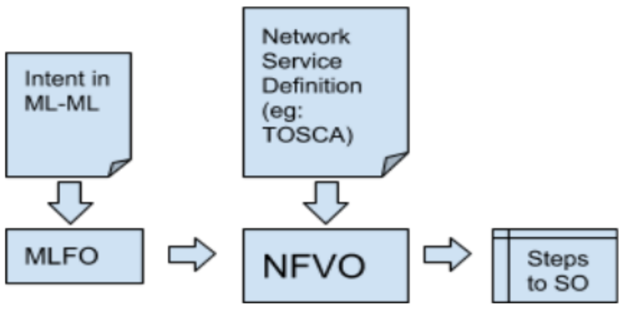


6 Key Architectural issues

a) ***Need to standardise a ML-ML (Machine learning Meta Language)? Can we get feedback from SG13 of ITU?***

Description/Significance:

- ML-ML should be specified as a technology-agnostic, declarative specification language, which can be used to specify how ML pipeline can be "composed" for a use case.
  E.g. NOP could specify using ML-ML that a new ML based use case needs to be introduced – analysing alarms, RCA and predicting problem areas in the network – by deploying a new src in the NMS subsystem, using a newly available model in a public repository, and direct the output to a fault recovery module via a policy function. Such a specification (Intent) could be agnostic of its underlying implementation in specific technology (e.g. 3GPP) by various vendors.
- ML-ML is used to make declarative specification, from which flows the interpretation and realisation of such a ML pipeline in, say 4G, 5G or any future networks, including simulated networks in a standard, predictable, interoperable manner, with no surprises.
- SO (Service orchestration) mechanisms, implemented by different vendors, need to understand the requirements specified in this format and control the steps in SO according to this input.
- It is important to differentiate the language (ML-ML) from the specifications (Intent) written in that language.
- This specification forms the mapping between ITU requirements and 3GPP, and other, realizations of the use case (because of clear demarcation of use case specific nodes and ML pipeline nodes in generic architecture).



Note: A possible realization of ML-ML may be using basic constructs in existing meta-languages and MLFO may be realized as a function as part of NFVO. Such implementation details are beyond the scope of the current study.

Current work and gaps:

- ETSI defines service orchestration. Network Service catalogs for E2E service description with VNF and PNF descriptors and service graphs are specified.
- Container based orchestration platforms (eg. Kubernetes) has their own mechanisms for service orchestration.
- Efforts are on to integrate these two, in ONAP.
- Current efforts are focussed on providing e2e service. Impact of introducing ML pipeline on such service orchestration mechanisms should be studied. Reuse of existing mechanisms should be maximised, but at the same time, gaps in integrating ML in a descriptive fashion while orchestrating VNF, PNF and CNF should be studied.

b) Study the level of capability exposure needed to enable dynamic ML based use cases in 5G and future networks?

Description/Significance:

- Capability exposure and creation of ML based services in future networks are tightly related to each other.
- ML use cases depends heavily on availability of data to analyse.
- Measurement data and context data has been identified in [2].
- Some of these data are standardised while others are not.
- Introduction of SBA (Service based Architecture) in 3GPP implies that a means of obtaining these data is via services from MnS producers which expose such data. It can be consumed by MnS consumers which use them for analytics.
- This may in turn be governed by EGMF (exposure governance management function) as defined in 3GPP.
- Dynamic and rapid service creation can be achieved in future networks only by mapping the capability exposure with service creation, discovery and chaining.

Current work and gaps:

- 3GPP specifies [1] southbound interfaces between NEF (network exposure function) and 5GC Network Functions e.g. N29 interface between NEF and SMF, N30 interface between NEF and PCF, etc.
- 3GPP NFs expose capabilities and events to other NFs via NEF.
- eg. The AMF provides the UE mobility related event reporting to NF that has been authorized to subscribe to the UE mobility event reporting service via NEF.
- But these are not dynamic, nor granular. Furthermore, Cloud native events need to be supported along with discovery and chaining.
- In combination with create and deploy MnS dynamically, (3rd party or NOP internal, dynamically), discovery, chaining and integration with NEF has to happen dynamically. Requirements and mechanisms for these needs studying further.

c) Study the requirements for "division of ML labour" between clouds? How is multi-level ML interface realized? What is the ML interface between clouds? is it a NFVO-NFVO interface?

Description/Significance:

- Multi-level interfaces are used in almost all use cases [3][4]
- Different orchestration abilities may be present in different clouds. eg. Edge clouds may not have all orchestrator functions.
- When designing, developing, testing, deploying and managing ML workloads across such multiple clouds, interface between them will be subjected to specific needs. eg. Design time specification and runtime deployment of ML pipeline nodes across domains, monitoring ML pipeline nodes, decision of where analytics is done (based on the capability of the cloud, eg, a resource constrained edge node may not host training function, but it may host the runtime model predictor). These are explained in great detail in [10].
- Requirements of ML functions on such cloud-cloud interface has to be studied to enable smooth deployment of ML functions across clouds.

Current work and gaps:

- Many opensource forums are studying the implementation of generic cloud. eg. ONAP, Akraino and openstack edge.
- In ETSI-NFV, IFA022 studies connectivity service instantiations between different NFVI-PoPs for the Network Service Life Cycle Management. A Network Service, is instantiated by the interactions among OSS/BSS, NFVO, WIM/VIM, and Network Controllers. The current IFA022 analyses the interactions among OSS/BSS, NFVO, WIM/VIM with reference to the current ETSI-NFV standards specifications.
- draft-bernardos-nfvrg-multidomain-05 analyzes the problem of multi-provider multi-domain orchestration, by first scoping the problem, then looking into potential architectural approaches, and finally describing the solutions being developed by the European 5GEx and 5G-TRANSFORMER projects.
- None of these addresses the needs of MLFO [8]

d) What is the relationship between ML pipeline nodes and 3GPP NFs? are the pipeline nodes managed by 3GPP? are the connections between them managed by 3GPP?

Description/Significance:

- ML pipeline nodes are an overlay on top of 3GPP. They could be hosted on any 3GPP NF by orchestration methods. They are managed by MLFO which in turn is managed by NFVO. MLFO may use interfaces and coordination with 3GPP and non-3GPP to manage the pipeline. (see sequence diagrams fig 9 and fig 10 of [8]).

- Further, ML pipeline acts as non-3GPP service, and interact with 3GPP NF using 3GPP defined MnS. Requirements for Nodes in ML pipeline will be defined by ITU (and not by 3GPP).

- ML pipeline exposes only Type A components towards 3GPP, because operation/notification has to be produced/consumed towards 3GPP NF MnS. Type B and C, even if exposed may be emulated (not real 3GPP NF).

- Interaction with legacy 3GPP Nfs needs to be studied using wrapper services. These wrapper services expose interfaces towards ML pipeline using standard interfaces but implement legacy or vendor-specific interfaces towards the NF. This aspect to provide a smooth migration path to operators needs further study.

____