International Telecommunication Union

ITU-T

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

Y.3176

(09/2020)

SERIES Y: GLOBAL INFORMATION INFRASTRUCTURE, INTERNET PROTOCOL ASPECTS, NEXT-GENERATION NETWORKS, INTERNET OF THINGS AND SMART CITIES

Future networks

Machine learning marketplace integration in future networks including IMT-2020

Recommendation ITU-T Y.3176

ITU-T Y-SERIES RECOMMENDATIONS

GLOBAL INFORMATION INFRASTRUCTURE, INTERNET PROTOCOL ASPECTS, NEXT-GENERATION NETWORKS, INTERNET OF THINGS AND SMART CITIES

GLOBAL INFORMATION INFRASTRUCTURE	
General	Y.100–Y.199
Services, applications and middleware	Y.200–Y.299
Network aspects	Y.300–Y.399
Interfaces and protocols	Y.400–Y.499
Numbering, addressing and naming	Y.500–Y.599
Operation, administration and maintenance	Y.600–Y.699
Security	Y.700–Y.799
Performances	Y.800–Y.899
INTERNET PROTOCOL ASPECTS	
General	Y.1000–Y.1099
Services and applications	Y.1100–Y.1199
Architecture, access, network capabilities and resource management	Y.1200–Y.1299
Transport	Y.1300–Y.1399
Interworking	Y.1400–Y.1499
Quality of service and network performance	Y.1500–Y.1599
Signalling	Y.1600–Y.1699
Operation, administration and maintenance	Y.1700–Y.1799
Charging	Y.1800–Y.1899
IPTV over NGN	Y.1900–Y.1999
NEXT GENERATION NETWORKS	
Frameworks and functional architecture models	Y.2000–Y.2099
Quality of Service and performance	Y.2100–Y.2199
Service aspects: Service capabilities and service architecture	Y.2200–Y.2249
Service aspects: Interoperability of services and networks in NGN	Y.2250–Y.2299
Enhancements to NGN	Y.2300–Y.2399
Network management	Y.2400–Y.2499
Network control architectures and protocols	Y.2500–Y.2599
Packet-based Networks	Y.2600–Y.2699
Security	Y.2700–Y.2799
Generalized mobility	Y.2800–Y.2899
Carrier grade open environment	Y.2900–Y.2999
FUTURE NETWORKS	**Y.3000–Y.3499**
CLOUD COMPUTING	Y.3500–Y.3599
BIG DATA	Y.3600–Y.3799
QUANTUM KEY DISTRIBUTION NETWORKS	Y.3800–Y.3999
INTERNET OF THINGS AND SMART CITIES AND COMMUNITIES	
General	Y.4000–Y.4049
Definitions and terminologies	Y.4050–Y.4099
Requirements and use cases	Y.4100–Y.4249
Infrastructure, connectivity and networks	Y.4250–Y.4399
Frameworks, architectures and protocols	Y.4400–Y.4549
Services, applications, computation and data processing	Y.4550–Y.4699
Management, control and performance	Y.4700–Y.4799
Identification and security	Y.4800–Y.4899
Evaluation and assessment	Y.4900–Y.4999

For further details, please refer to the list of ITU-T Recommendations.

Recommendation ITU-T Y.3176

Machine learning marketplace integration in future networks including IMT-2020

Summary

Recommendation ITU-T Y.3176 provides high-level requirements and the architecture for integrating machine learning (ML) marketplaces in future networks including IMT-2020. Based on these requirements, the Recommendation describes the architecture for the integration of ML marketplaces is taking into account the architectural framework in Recommendation ITU-T Y.3172 as a basis.

History

Edition	Recommendation	Approval	Study Group	Unique ID*
1.0	ITU-T Y.3176	2020-09-29	13	11.1002/1000/14402

Keywords

Architecture, federation, future networks, IMT-2020, life-cycle, machine learning, management, marketplace, model, network functions, requirements.

* To access the Recommendation, type the URL http://handle.itu.int/ in the address field of your web browser, followed by the Recommendation's unique ID. For example, http://handle.itu.int/11.1002/1000/11830-en.

FOREWORD

The International Telecommunication Union (ITU) is the United Nations specialized agency in the field of telecommunications, information and communication technologies (ICTs). The ITU Telecommunication Standardization Sector (ITU-T) is a permanent organ of ITU. ITU-T is responsible for studying technical, operating and tariff questions and issuing Recommendations on them with a view to standardizing telecommunications on a worldwide basis.

The World Telecommunication Standardization Assembly (WTSA), which meets every four years, establishes the topics for study by the ITU-T study groups which, in turn, produce Recommendations on these topics.

The approval of ITU-T Recommendations is covered by the procedure laid down in WTSA Resolution 1.

In some areas of information technology which fall within ITU-T's purview, the necessary standards are prepared on a collaborative basis with ISO and IEC.

NOTE

In this Recommendation, the expression "Administration" is used for conciseness to indicate both a telecommunication administration and a recognized operating agency.

Compliance with this Recommendation is voluntary. However, the Recommendation may contain certain mandatory provisions (to ensure, e.g., interoperability or applicability) and compliance with the Recommendation is achieved when all of these mandatory provisions are met. The words "shall" or some other obligatory language such as "must" and the negative equivalents are used to express requirements. The use of such words does not suggest that compliance with the Recommendation is required of any party.

INTELLECTUAL PROPERTY RIGHTS

ITU draws attention to the possibility that the practice or implementation of this Recommendation may involve the use of a claimed Intellectual Property Right. ITU takes no position concerning the evidence, validity or applicability of claimed Intellectual Property Rights, whether asserted by ITU members or others outside of the Recommendation development process.

As of the date of approval of this Recommendation, ITU had not received notice of intellectual property, protected by patents, which may be required to implement this Recommendation. However, implementers are cautioned that this may not represent the latest information and are therefore strongly urged to consult the TSB patent database at http://www.itu.int/ITU-T/ipr/.

Table of Contents

Recommendation ITU-T Y.3176

Machine learning marketplace integration in future networks including IMT-2020

1 Scope

This Recommendation provides high-level requirements and the architecture for integration of ML marketplaces in future networks including IMT-2020.

The scope of this Recommendation includes:

– The main motivations and challenges related to ML marketplace integration in networks,

– High level requirements of ML marketplace integration,

– Architecture for the integration of ML marketplaces in networks.

2 References

The following ITU-T Recommendations and other references contain provisions which, through reference in this text, constitute provisions of this Recommendation. At the time of publication, the editions indicated were valid. All Recommendations and other references are subject to revision; users of this Recommendation are therefore encouraged to investigate the possibility of applying the most recent edition of the Recommendations and other references listed below. A list of the currently valid ITU-T Recommendations is regularly published. The reference to a document within this Recommendation does not give it, as a stand-alone document, the status of a Recommendation.

[ITU-T Y.2701] Recommendation ITU-T Y.2701 (2007), *Security requirements for NGN release 1*.

[ITU-T Y.3101] Recommendation ITU-T Y.3101 (2018), *Requirements of the IMT-2020 network.*

[ITU-T Y.3172] Recommendation ITU-T Y.3172 (2019), *Architectural framework for machine learning in future networks including IMT-2020.*

[ITU-T Y.3174] Recommendation ITU-T Y.3174 (2020), *Framework for data handling to enable machine learning in future networks including IMT-2020.*

3 Definitions

3.1 Terms defined elsewhere

This Recommendation uses the following terms defined elsewhere:

3.1.1 machine learning (ML) [ITU-T Y.3172]: Processes that enable computational systems to understand data and gain knowledge from it without necessarily being explicitly programmed.

NOTE 1 – Definition adapted from [b-ETSI GR ENI 004].

NOTE 2 – Supervised machine learning and unsupervised machine learning are two examples of machine learning types.

3.1.2 machine learning database [ITU-T Y.3174]: A component which stores data related to machine learning in the machine learning overlay.

3.1.3 machine learning function orchestrator (MLFO) [ITU-T Y.3172]: A logical node with functionalities that manage and orchestrate the nodes in a machine learning pipeline.

3.1.4 machine learning model [ITU-T Y.3172]: Model created by applying machine learning techniques to data to learn from.

NOTE 1 – A machine learning model is used to generate predictions (e.g., regression, classification, clustering) on new (untrained) data.

NOTE 2 – A machine learning model may be encapsulated in a deployable fashion in the form of a software (e.g., virtual machine, container) or hardware component (e.g., IoT device).

NOTE 3 – Machine learning techniques include learning algorithms (e.g., learning the function that maps input data attributes to output data).

3.1.5 machine learning pipeline [ITU-T Y.3172]: A set of logical nodes, each with specific functionalities, that can be combined to form a machine learning application in a telecommunication network.

NOTE – The nodes of a machine learning pipeline are entities that are managed in a standard manner and can be hosted in a variety of network functions [b-ITU-T Y.3100].

3.1.6 machine learning sandbox [ITU-T Y.3172]: An environment in which machine learning models can be trained, tested and their effects on the network evaluated.

NOTE – A machine learning sandbox is designed to prevent a machine learning application from affecting the network, or to restrict the usage of certain machine learning functionalities.

3.2 Terms defined in this Recommendation

This Recommendation defines the following terms:

3.2.1 machine learning model metadata: Information which describes the characteristics of a machine learning model.

NOTE – Machine learning (ML) model metadata include, but is not limited to, the name of the ML model, ML model's author, version of the ML model, license information of the ML model, description of the data inputs and outputs of the ML model and the runtime environment of the ML model.

3.2.2 machine learning marketplace: A component which provides capabilities facilitating the exchange and delivery of machine learning models among multiple parties.

NOTE 1 – Examples of parties include suppliers and users of ML models. Capabilities provided to users of ML models include functionalities to find, learn about, deploy (or download) and use ML models. Capabilities provided to suppliers of ML models (e.g., data scientist) include functionalities to share (on-board, upload), describe (learn about) and market their ML models.

NOTE 2 – A network operator may use a machine learning marketplace deployed internally and/or externally to the network operator's administrative domains. Internal and external marketplaces differ only in the deployment perspective. A marketplace which is internal to a network operator may act as an external marketplace to another network operator and vice versa.

4 Abbreviations and acronyms

This Recommendation uses the following abbreviations and acronyms:

API Application Programming Interface

Enum Enumeration

ID Identifier

KPI Key Performance Indicator

ML Machine Learning

MLFO Machine Learning Function Orchestrator

MLDB Machine Learning Database

MPP Mobility Pattern Prediction

SRC Source

URI Uniform Resource Identifier

5 Conventions

In this Recommendation, 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 Recommendation 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.

6 Introduction

Use cases for the use of machine learning (ML) in future networks including IMT-2020 are described in [b-ITU-T Y-Sup.55] while an architectural framework for ML in future networks including IMT-2020 is presented in [ITU-T Y.3172].

The main motivations for ML marketplace integration in future networks including IMT-2020 are highlighted below:

1) Evolution of ML models may happen independently of the technology evolution in future ML underlay networks. Integration of ML marketplaces will provide a mechanism for network operators to follow the innovation curve in the domain of machine learning.

   NOTE – For example, as and when new, innovative and proven ML mechanisms and techniques are published by ML researchers and corresponding ML models are made available in the marketplace, network operators are able to use those models using the ML marketplace integration described in this Recommendation.

2) Efficient exchange and deployment of ML models using standard interfaces will help not only in solving network problems using ML techniques but also in sharing and monetizing ML techniques in the network domain.

However, the following main challenges need to be addressed for the integration of ML marketplaces in ML-enabled networks:

– ML models may be hosted in varied types of ML marketplaces (e.g., Acumos [b-LF Acumos]). While designing the ML application, network operators need interoperable mechanisms for identification of the ML marketplace which may be used as a source for ML models.

  NOTE 1 – This points to the need for network operators to use standard ML intent mechanisms to specify the ML marketplace to be used for the ML application.

– Network operators need interoperable, automated mechanisms to select the appropriate ML models from the ML marketplaces.

  NOTE 2 – For example, the latest advances in analytics or algorithms [b-DL Survey] have no dependency on the network architecture evolution of ML underlay networks. Cloud computing-hosted ML marketplaces may attract developers of innovative ML mechanisms and algorithms to host their ML models.

NOTE 3 – Metadata used for describing ML models stored in ML marketplaces may be exploited for the selection process. In such a context, availability of standardized ML model metadata is of prime importance.

– Network operators should be able to follow the innovation curve in the ML marketplace without sacrificing their network key performance indicators (KPI) [b-3GPP TS 28.554].

NOTE 4 – Network operators may use a ML sandbox to isolate the effect of training and testing of ML models from the live network.

– Lack of standard mechanisms to exchange ML models and related metadata between ML marketplaces and the network operator's ML deployment environments limits the interoperability. There is a need to study and develop standard mechanisms for the interfaces between ML marketplaces and network operator's ML deployment environments in order to allow for efficient exchange and deployment of ML models.

NOTE 5 – For example, exchange and deployment of ML models from a ML marketplace may be done using export of ML models to standard file format [b-ONNX], ML model compression using standard methods [b-ISO-IEC 15938-17] and secure file transfer to the network operator's ML deployment environments.

NOTE 6 – Deployment of internal ML marketplaces enables collaborative development and use of ML models within different ML deployment environments of the network operator, e.g., operations and maintenance, research and development. It also enables the development of regional or enterprise specific ML models which conform to local regulations and organizational policies on data handling.

– As mentioned in [ITU-T Y.3172], building complex ML models by chaining ML models is an important need for network operators. However, selecting the appropriate ML models from disparate ML marketplaces and chaining them requires support for such mechanisms from ML marketplaces.

NOTE 7 – This points to the need for the availability of standardized metadata for the ML models and the need to interoperate with ML sandbox.

– Efficient use of testing methods requires sharing of information between ML marketplaces and network operators, e.g., data which was used for testing, test results, current KPIs for the ML model.

– Deployment of ML models depends on the targeted deployment environment of the network operator, which differ from a virtualisation technology standpoint, e.g., container or virtual machine based [b-ETSI-GS-NFV-EVE]. ML models hosted in ML marketplaces need to be amenable to such different types of deployment environment.

NOTE 8 – For example, a containerization of ML model may be needed to deploy the model in a container-based virtualization environment of the network operator.

This Recommendation provides high-level requirements and architectural components for integrating ML marketplaces in future networks including IMT-2020.

7 High-level requirements for ML marketplace integration

This clause provides high-level requirements for ML marketplace integration, i.e., ML model metadata related requirements and ML marketplace requirements.

7.1 ML model metadata related requirements

REQ-ML-MKT-METADATA-01: The ML model metadata is required to indicate the training status and chaining status of the ML model.

NOTE 1 – Examples of indicated training status information include whether the ML model is trained or not, and in case of trained models, the training datasets used.

NOTE 2 – Examples of indicated chaining status information include whether the ML model is compatible with any other model to be chained.

REQ-ML-MKT-METADATA-02: The ML model metadata is required to indicate the testing maturity of the model.

NOTE 3 – Examples of testing maturity indications of a ML model include the following:

1) Performance benchmarks with respect to published or standard test cases, e.g., latency benchmarks of the ML model, memory requirements of the ML model, accuracy of the ML model.
2) Reference to standard test cases and corresponding compliance of the ML model to those test cases.
3) Applicability of ML output from the ML model to various types of ML underlay networks, e.g., ML output from mobility pattern prediction (MPP) model may be applicable to urban and rural areas.
4) A rank or score based on testing or user experience from network operators.

NOTE 4 – For example, a network operator may score the ML model (based on the use case under consideration) as "10" on a scale from "0" (lowest score) to "10" (top score). This score may be used by another network operator to select the ML model (to be used for a similar use case) from the ML marketplace.

REQ-ML-MKT-METADATA-03: The ML model metadata is required to indicate the input and output characteristics of the ML model.

NOTE 5 – Examples of indicated characteristics include the dimensions of input data and ML output, and data models used in exchange of data with ML underlay [ITU-T Y.3174].

REQ-ML-MKT-METADATA-04: The ML model metadata is required to indicate the compression status of the ML model.

NOTE 6 – Examples of indicated compression status information include whether the ML model is compressed or not, and in case of compressed models, the compression method used.

REQ-ML-MKT-METADATA-05: The ML model metadata is required to indicate the type of learning that the ML model supports.

NOTE 7 – Examples of learning type indication include supervised or unsupervised learning.

NOTE 8 – The indication about the learning type is used for the training or retraining of the ML model. Depending on the type of learning indicated, the MLFO is able to select an appropriate type of ML pipeline to be instantiated in order to train such an ML model.

REQ-ML-MKT-METADATA-6: The ML model metadata is required to indicate the resource requirements and constraints for ML models.

NOTE 9 – Specified resource requirements and constraints include compute, storage and network requirements for the ML model to function and to meet the KPIs of ML use cases. Examples are deployment environment of the ML model, prediction time, budget for communication latency, and validity time for ML output. Example of temporal validity for ML output is the case where prediction from the ML model may be valid for use only for a limited time, after that the prediction needs to be refreshed based on new data.

7.2 ML marketplace related requirements

REQ-ML-MKTPLACE-01: The ML marketplace is recommended to support functionalities for the searching of ML models based on the metadata describing these ML models.

NOTE 1 – For example, a ML marketplace may support an interface that takes metadata (or its components) as input for selection and returns a list of ML models satisfying the best match for those metadata.

REQ-ML-MKTPLACE-02: The ML marketplace is recommended to support functionalities for exporting ML models according to standardized formats.

NOTE 2 – Example of standardized model format for exporting includes ONNX [b-ONNX].

REQ-ML-MKTPLACE-03: The ML marketplace is recommended to support asynchronous notifications to the MLFO about additions, modifications and deletions of new ML models.

NOTE 3 – Asynchronous notifications are used for the discovery of ML models by the MLFO.

REQ-ML-MKTPLACE-04: The ML marketplace is required to support query of functionalities offered, on the interface between the ML marketplace and MLFO.

NOTE 4 – Discovery of ML models is an example of functionality supported by ML marketplace, which can be queried by MLFO.

REQ-ML-MKTPLACE-05: The ML marketplace is required to support functionalities for pushing ML models to the ML sandbox and/or ML pipeline subsystem.

NOTE 5 – Pushing ML models may be performed using secure file transfer protocols.

NOTE 6 – Pushing may be performed using a standardized format for ML models, including metadata.

REQ-ML-MKTPLACE-06: The ML marketplace is recommended to support usage of trained and untrained ML models.

NOTE 7 – In case of untrained ML models, the offline training may be done using stored data. This training may be done in the ML sandbox.

NOTE 8 – The ML marketplace may provide capabilities for publishing, storing, discovering and querying of trained as well as untrained ML models.

8 Architecture for ML marketplace integration

The architecture for ML marketplace integration is shown in Figure 1 and is intended to fulfil the requirements provided in clause 7.

NOTE 1 – Figure 1 is built on the basis or using elements of Figure 4 of [ITU-T Y.3172].

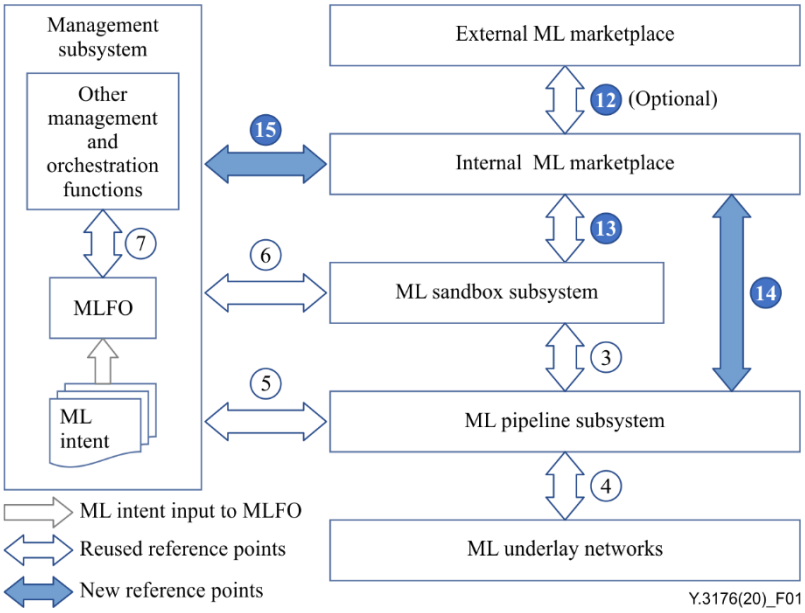

Figure 1 – Architecture for ML marketplace integration in network

Figure 1 includes a ML marketplace hosted internally to the network operator's administrative domain(s) (shown as "Internal ML marketplace" architectural component) which optionally interfaces with another ML marketplace hosted externally to the network operator's administrative domain(s) (shown as "External ML marketplace" architectural component).

NOTE 2 – The external ML marketplace component may represent a federation of different ML marketplaces hosted externally to the network operator's administrative domains.

In Figure 1, reference points 3, 4, 5, 6 and 7 are reused from [ITU-T Y.3172]. New reference points to [ITU-T Y.3172] are as follows:

– Reference point 12 is an optional reference point between the internal ML marketplace and the external ML marketplace components. This reference point is used in case of ML marketplaces' federation between network operators,
– Reference point 13 is the reference point between the internal ML marketplace and the ML sandbox subsystem,
– Reference point 14 is the reference point between the internal ML marketplace and the ML pipeline subsystem,
– Reference point 15 between the internal ML marketplace and the management subsystem allows the network operators to manage the internal ML marketplace.

Further details of these new reference points, including application programming interfaces (APIs) and corresponding information elements, are provided in clause 8.2 and the reused reference points from [ITU-T Y.3172] are called out. In addition, clause 8.1 describes new architectural components (internal and external ML marketplaces) as compared to [ITU-T Y.3172]. The use of these APIs is further explained in clause 8.3 using the sequence diagrams corresponding to the main scenarios for integration of ML marketplace with operator networks.

8.1 Architecture framework components

This clause describes new architectural components as compared to [ITU-T Y.3172].

8.1.1 Internal ML marketplace

A machine learning marketplace deployed internally to the network operator's administrative domain is an internal marketplace. The internal marketplace enables exchange of ML models within the network operator's administrative domain, e.g., query, select, push, discovery and deploy ML models by MLFO in the ML sandbox or ML pipeline subsystem.

NOTE – The internal marketplace can interface with external marketplaces, e.g., to a federation of ML marketplaces, and query, pull and discovery ML models from external marketplaces.

8.1.2 External ML marketplace

A machine learning marketplace deployed outside the network operator's administrative domain is an external marketplace. While the external marketplace cannot directly interface with the network operator's administrative domain, it enables exchange of ML models, e.g., query, pull, and discovery of ML models, with the internal marketplace via an optional interface, e.g., to a federation of ML marketplaces.

8.2 Reference points

The functionalities enabled by reference points from [ITU-T Y.3172] shown in Figure 1 are described in clause 8.2.1 while new reference points 12, 13, 14 and 15 which are directly related to ML marketplaces are described in clause 8.2.2.

8.2.1 Reused reference points

8.2.1.1 Reference point 3

Reference point 3 [ITU-T Y.3172] is used for the deployment or update of ML models in the ML pipeline subsystem.

8.2.1.2 Reference point 4

Reference point 4 [ITU-T Y.3172] is used for handling data from ML underlay networks and configuring ML underlay networks.

8.2.1.3 Reference point 5

Reference point 5 [ITU-T Y.3172] enables ML model monitoring functionality. After a ML model is pushed to a ML pipeline subsystem, the MLFO triggers the monitoring operation in the ML pipeline subsystem and receives the monitoring results reported from the ML pipeline subsystem.

8.2.1.4 Reference point 6

Reference point 6 [ITU-T Y.3172] is used by the management subsystem to manage the ML models pushed from the internal marketplace to the ML sandbox. For example, the MLFO may trigger and monitor the training, optimization, chaining and deployment process of an ML model in the ML sandbox.

8.2.1.5 Reference point 7

Reference point 7 [ITU-T Y.3172] allows the MLFO to monitor the capabilities of ML underlay networks with respect to network intelligence.

8.2.2 New reference points

This clause describes new reference points to [ITU-T Y.3172].

8.2.2.1 Reference point 12

Reference point 12 between internal ML marketplaces and external ML marketplaces is optional and used for sharing ML models between internal and external ML marketplaces. Using reference point 12, the internal ML marketplace is capable of getting access to the ML models which are authorized to be shared by the external ML marketplace via this reference point.

The APIs supported through reference point 12 are described in the following clauses.

8.2.2.1.1 Model_Query API

The Model_Query API enables the internal ML marketplace to query from the external ML marketplace for a particular set of ML models that meet the requirements indicated in ML intent.

Model_Query-request:

Direction: Internal ML marketplace → External ML marketplace

Table 8-1 lists information elements in the Model_Query-request.

Table 8-1 – Model_Query-request information elements

Information element	Type	Mandatory/Optional/ Conditional	Description
Model identifier	String	Optional	Identifies of the ML model that is queried
Model metadata	<attribute, value> array See Note 1	Conditional See Note 2	Provides metadata about the queried ML model

NOTE 1 (applicable to all equivalent table cases in clause 8.2) – Each <attribute, value> pair corresponds to a metadata about the model. An array of such pairs may be used for this query. Examples of <attribute, value> pair are *<type of data input, real-time>, <source of data, UE>, <type of learning, supervised> <description, text description of model>*.

NOTE 2 – This information element is present only if the model identifier information element is not present in the Model_Query-request.

Model_Query-response:

Direction: External ML marketplace → Internal ML marketplace

Table 8-2 lists the information elements of the Model_Query-response.

Table 8-2 – Model_Query-response information elements

Information element	Type	Mandatory/Optional/ Conditional	Description
Result	Enum	Mandatory	Indicates the handling result ("success" or "failure") of a Model_Query-request.
Model Identifier	String	Conditional See Note 3	Identifier of a ML model that meets the criteria provided in the Model_Query - request
Model metadata	<Attribute, value> array	Conditional See Note 3	Contains the metadata of a ML model whose identifier is present in the Model_Query-response

NOTE 3 – This information element is present only if the Result information element indicates "success".

8.2.2.1.2 Model_Selection API

The Model_Selection API enables an internal ML marketplace to select ML model(s) from an external ML marketplace.

Model_Selection-request:

Direction: Internal ML marketplace → External ML marketplace

Table 8-3 lists the information elements of the Model_Selection-request.

Table 8-3 – Model_Selection-request information elements

Information element	Type	Mandatory/Optional /Conditional	Description
Model identifier	String	Mandatory	Identifier of the ML model that is requested to be selected

Model_Selection-response:

Direction: External ML marketplace → Internal ML marketplace

Table 8-4 lists the information elements of the Model_Selection-response.

Table 8-4 – Model_Selection-response information elements

Information element	Type	Mandatory/Optional /Conditional	Description
Result	Enum	Mandatory	Indicates the handling result ("Success" or "Failure") of the Model_Selection-request
Model identifier	String	Conditional See Note	Identifier of the ML model(s) that are successfully selected

NOTE – This information element is present only if the Result information element indicates "success".

8.2.2.1.3 Model_Pull API

The Model_Pull API enables the internal ML marketplace to pull a ML model from the external ML marketplace. This API allows a ML model available in a federated marketplace to be pushed to the internal marketplace.

Model_Pull-request:

Direction: Internal ML marketplace → External ML marketplace

Table 8-5 lists the information elements of the Model_Pull-request.

Table 8-5 – Model_Pull-request information elements

Information element	Type	Mandatory/Optional/ Conditional	Description
Model identifier	String	Mandatory	Identifier of the ML model that is requested to be pulled

Model_Pull-response:

Direction: External ML marketplace → Internal ML marketplace

Table 8-6 lists the information elements of the Model_Pull-response.

Table 8-6 – Model_Pull-response information elements

Information element	Type	Mandatory/Optional/ Conditional	Description
Result	Enum	Mandatory	Indicates the handling result ("Success", "Failure") of the Model_Pull-request
Model identifier	String	Conditional See Note 1	Identifier of the ML model that is accepted to be pulled
Model metadata	<Attribute, value> array	Conditional See Note 1	Contains metadata of the ML model that is accepted to be pulled
Model object	Object See Note 2	Conditional See Note 1	Contains the model object

NOTE 1 – This information element is present only if the Result information element indicates "success".

NOTE 2 – The Object may be represented using a URL which points to a repository where the ML model can be found for download or ML model itself. In case the ML model itself is present in the API, it may use a binary format.

8.2.2.1.4 Model_Discovery_Poll API

The Model_Discovery_Poll API enables the internal ML marketplace to poll for updated versions of ML models from an external ML marketplace. This is of interest for ML models that are or being deployed in a ML pipeline subsystem by the internal ML marketplace.

Model_Discovery_Poll-request:

Direction: Internal ML marketplace → External ML marketplace

Table 8-7 lists the information elements of the Model_Discovery_Poll-request.

Table 8-7 – Model_Discovery_Poll-request information elements

Information element	Type	Mandatory/Optional /Conditional	Description
Model identifier	String	Mandatory	The identifier of the searched ML model

Model_Discovery_Poll-response:

Direction: External ML marketplace → Internal ML marketplace

Table 8-8 lists the information elements of the Model_Discovery_Poll-response.

Table 8-8 – Model_Discovery_Poll-response information elements

Information element	Type	Mandatory/Optional /Conditional	Description
Result	Enum	Mandatory	Indicates the handling result ("Success", "Failure") of the Model_Discovery_Poll-request
Model identifier	String	Conditional See Note	Identifier of the ML model that is polled
Model metadata	<Attribute, value> array	Conditional See Note	Indicates the metadata of the polled ML model present in the external marketplace

NOTE – This information element is present only if the Result information element indicates "success".

8.2.2.1.5 Model_Discovery_Asynchronous_Update API

The Model_Discovery_Asynchronous_Update API enables the external marketplace to asynchronously indicate that a ML model has been updated and to provide the associated updated metadata. This is of interest for ML models from the external marketplace that are or being deployed by the internal ML marketplace.

Direction: External ML marketplace → Internal ML marketplace

Table 8-9 lists the information elements of the Model_Discovery_Asynchronous_Update.

Table 8-9 – Model_Discovery_Asynchronous_Update information elements

Information element	Type	Mandatory/Optional /Conditional	Description
Model identifier	String	Mandatory	Identifier of the updated ML model.
Model metadata	<Attribute, value> array	Optional	Contains the metadata of the updated ML model Indicates the updated metadata if the model is updated in the external marketplace

8.2.2.2 Reference point 13

Reference point 13 provides functionality to update the models from ML sandbox to the internal marketplace as well as push ML models from the internal marketplace to the ML sandbox.

8.2.2.2.1 Model_Update API

The Model_Update API enables the ML sandbox to send a newly trained ML model to the internal Ml marketplace.

Model_Update-request:

Direction: ML sandbox → ML marketplace

Table 8-10 lists information elements of the Model_Update-request.

Table 8-10 – Model_Update-request information elements

Information element	Type	Mandatory/Optional /Conditional	Description
Model identifier	String	Mandatory	Identifier of the newly trained ML model
Model metadata	<Attribute, value> array	Mandatory	Contains the metadata of the newly trained ML model
Model object	Object	Mandatory	Contains the ML model object

Model_Update-response:

Direction: ML marketplace → ML sandbox

Table 8-11 lists the information elements of the Model_Update-response.

Table 8-11 – Model_Update-response information elements

Information element	Type	Mandatory/Optional /Conditional	Description
Result	Enum	Mandatory	Indicates the handling result ("Success", "Failure") of the Model_Update-request
Model identifier	String	Conditional See Note	Identifier of the updated ML model

NOTE – This information element is present only if the Result information element indicates "success".

8.2.2.2.2 Model_Push API

The Model_Push API enables the internal ML marketplace to push a ML model to the ML sandbox upon trigger by the MLFO.

Model_Push-request:

Direction: ML marketplace → ML sandbox

Table 8-12 lists the information elements of the Model_Push-request.

Table 8-12 – Model_Push-request information elements

Information element	Type	Mandatory/Optional /Conditional	Description
Model identifier	String	Mandatory	Identifier of the pushed ML model
Model metadata	<Attribute, value> array	Mandatory	Contains the metadata of the pushed ML model
Model object	Object	Mandatory	Contains the ML model object

Model_Push-response:

Direction: ML sandbox → ML marketplace

Table 8-13 lists the information elements of the Model_Push-response.

Table 8-13 – Model_Push-response information elements

Information element	Type	Mandatory/Optional /Conditional	Description
Result	Enum	Mandatory	Indicates the handling result ("Success", "Failure") of the Model_Push-request
Model identifier	String	Conditional See Note	Identifier of the pushed ML model

NOTE – This information element is present only if the Result information element indicates "success".

8.2.2.3 Reference point 14

8.2.2.3.1 Model_Push API

The Model_Push API enables the internal ML marketplace to push a ML model to the ML pipeline subsystem upon trigger by the MLFO.

Model_Push-request:

Direction: ML marketplace → ML pipeline

Table 8-14 lists the information elements of the Model_Push-request.

Table 8-14 – Model_Push-request information elements

Information element	Type	Mandatory/Optional /Conditional	Description
Model identifier	String	Mandatory	Identifier of the pushed ML model
Model metadata	<Attribute, value> array	Mandatory	Contains the metadata of the pushed ML model
Model object	Object	Mandatory	Contains the ML model object

Model_Push-response:

Direction: ML pipeline → ML marketplace

Table 8-15 lists the information elements of the Model_Push-response.

Table 8-15 – Model_Push-response information elements

Information element	Type	Mandatory/Optional /Conditional	Description
Result	Enum	Mandatory	Indicates the handling result ("Success", "Failure") of the Model_Push-request.
Model identifier	String	Conditional See Note	Identifier of the pushed ML model

NOTE – This information element is present only if the Result information element indicates "success".

8.2.2.4 Reference point 15

Reference point 15 provides MLFO with functionalities of query, selection, discovery and deployment of ML models from the internal marketplace.

8.2.2.4.1 Model_Query API

The Model_Query API enables the MLFO to query from the ML marketplace for a particular set of ML models that meet the requirements indicated in ML intent.

Model_Query-request:

Direction: MLFO → ML marketplace

Table 8-16 lists the information elements of the Model_Query-request.

Table 8-16 – Model_Query-request information elements

Information element	Type	Mandatory/Optional /Conditional	Description
Model identifier	String	Optional	Identifier of the ML model that is queried. This is used if known by the MLFO.
Model metadata	<Attribute, value> array	Mandatory	Metadata criteria to be met in the query of ML models

Model_Query-response:

Direction: ML marketplace → MLFO

Table 8-17 lists information elements of Model_Query-response.

Table 8-17 – Model_Query-response information elements

Information element	Type	Mandatory/Optional /Conditional	Description
Result	Enum	Mandatory	Indicates the handling result ("Success", "Failure") of the Model_Query-request
Model Identifiers	Array of Strings	Conditional See Note	Identifiers of the ML models returned by the ML marketplace and successfully meeting the query criteria
Model metadata	<Attribute, value> array	Conditional See Note	Contains the metadata of the ML models meeting the query criteria

NOTE – This information element is present only if the Result information element indicates "success".

8.2.2.4.2 Model_Selection API

The Model_Selection API enables the MLFO to select a ML model from the ML marketplace for deployment in the ML pipeline subsystem.

Model_Selection-request:

Direction: MLFO → ML marketplace

Table 8-18 lists the information elements of the Model_Selection-request.

Table 8-18 – Model_Selection-request information elements

Information element	Type	Mandatory/Optional /Conditional	Description
Model identifier	String	Mandatory	Identifier of the ML model requested to be selected (already known from a previous search)

Model_Selection-response:

Direction: ML marketplace → MLFO

Table 8-19 lists the information elements of the Model_Selection-response.

Table 8-19 – Model_Selection-response information elements

Information element	Type	Mandatory/Optional /Conditional	Description
Result	Enum	Mandatory	Indicates the handling result ("Success", "Failure") of the Model_Selection-request
Model identifier	String	Conditional See Note	Identifier of the selected ML model

NOTE – This information element is present only if the Result information element indicates "success".

8.2.2.4.3 Model_Discovery_Poll API

The Model_Discovery_Poll API enables the MLFO to poll the ML marketplace for updated versions of ML models that are already deployed in ML pipeline subsystem.

Model_Discovery_Poll-request:

Direction: MLFO → ML marketplace

Table 8-20 lists information elements for Model_Discovery_Poll-request.

Table 8-20 – Model_Discovery_Poll-request information elements

Information element	Type	Mandatory/Optional /Conditional	Description
Model identifier	String	Mandatory	Identifier of the ML model that is polled MLFO already knows the ID from the search

Model_Discovery_Poll-response:

Direction: ML marketplace → MLFO

Table 8-21 lists the information elements of the Model_Discovery_Poll-response.

Table 8-21 – Model_Discovery_Poll-response information elements

Information element	Type	Mandatory/Optional /Conditional	Description
Result	Enum	Mandatory	Indicates the handling result ("Success", "Failure") of the Model_Discovery_Poll-request
Model identifier	String	Conditional See Note	Identifier of the polled ML model
Model metadata	<Attribute, value> array	Conditional See Note	Contains the metadata of the polled ML model Indicates the updated metadata if the model is updated in Marketplaces

NOTE – This information element is present only if the Result information element indicates "success".

8.2.2.4.4 Model_Discovery_Asynchronous_Update API

The Model_Discovery_Asynchronous_Update API enables the ML marketplace to asynchronously update the MLFO for models that are already deployed in ML pipeline subsystem.

Direction: ML marketplace → MLFO

Table 8-22 lists the information elements of the Model_Discovery_Asynchronous_Update.

Table 8-22 – Model_Discovery_Asynchronous_Update information elements

Information element	Type	Mandatory/Optional /Conditional	Description
Model identifier	String	Mandatory	Identifier of the updated ML model
Model metadata	<Attribute, value> array	Mandatory	Metadata of the updated ML model

8.2.2.4.5 Model_Deployment API

The Model_Deployment API enables the MLFO to request the deployment of an ML model in the ML pipeline subsystem.

Model_Deployment-request:

Direction: MLFO → ML marketplace

Table 8-23 lists the information elements of the Model_Deployment-request.

Table 8-23 – Model_Deployment-request information elements

Information element	Type	Mandatory/Optional /Conditional	Description
Model identifier	String	Mandatory	Identifier of a ML model to be deployed MLFO already knows the ID
ML pipeline identifier	String	Mandatory	Identifier of the ML pipeline to be deployed. The ML pipeline may be deployed in a ML sandbox or in a ML underlay network. See Note 1.

NOTE 1 – This identifier may be in the form of a URI.

Model_Deployment-response:

Direction: ML marketplace → MLFO

Table 8-24 lists the information elements of Model_Deployment-response.

Table 8-24 – Model_Deployment-response information elements

Information element	Type	Mandatory/Optional /Conditional	Description
Result	Enum	Mandatory	Indicates the handling result ("Success", "Failure") of the Model_Deployment-request.
Model identifier	String	Conditional See Note 2	Identifier of deployed ML model
ML pipeline identifier	String	Conditional See Note 2	Identifier of the ML pipeline in which the ML model is deployed. This may be deployed in a ML sandbox or an ML underlay network. See Note 3.

NOTE 2 – This information element is present only if the Result information element indicates "success".

NOTE 3 – This identifier may be in the form of a URI.

8.3 Sequence diagrams

This clause provides sequence diagrams related to the integration of an ML marketplace in a network. These sequence diagrams cover the following aspects: model search, model selection and push, model discovery, model training and model deployment in a ML pipeline subsystem.

NOTE – In the sequence diagrams provided in this clause, the following points are important for integrating ML marketplaces in networks:

– Use of ML intent and MLFO to search and select the ML model from the ML marketplace, based on the needs of the ML application.

– Rich, standard metadata is used in the interface between the ML marketplace and the MLFO to search and select the ML model most suited for the requirements of the ML application.

– Interface between ML sandbox and/or ML pipeline subsystem and ML marketplace is used to push the ML model from the ML marketplace to the ML sandbox and/or ML pipeline subsystem.

8.3.1 Model search

The model search sequence diagram is shown in Figure 2.

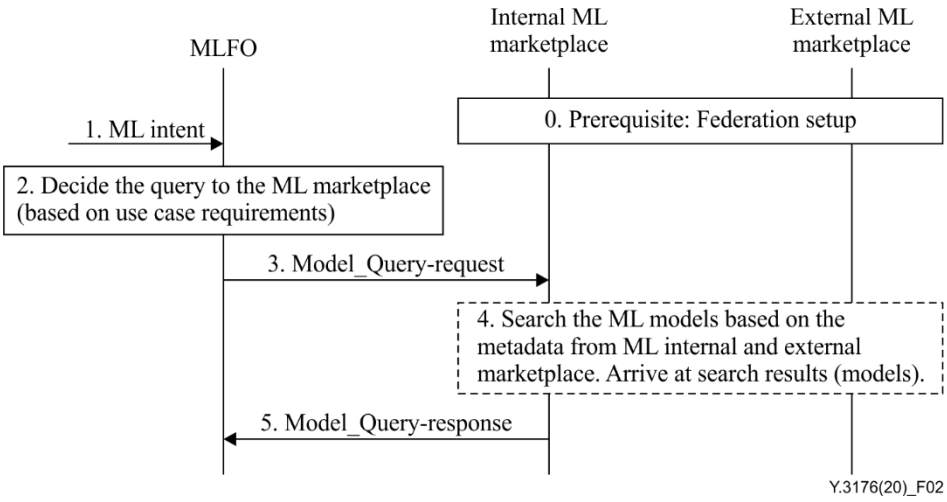

Figure 2 – Model search

Prerequisites: External ML marketplace and internal ML marketplace are federated.

The sequences shown in Figure 2 are as follows:

1) An ML intent which describes the ML use case requirements is received by the MLFO.

2) Based on the received ML intent, the MLFO generates ML pipeline requirements, which may include the functional and performance requirements of the ML model. The MLFO decides whether the model query can use the federation mode of ML marketplaces.

   NOTE – The federation mode specifies how the ML models are searched, i.e., from the internal ML marketplace only or throughout the ML marketplace federation, i.e., whether ML models from external ML marketplaces can be searched or not. The federation mode is part of the model metadata (see Table 8-1).

3) The MLFO sends a Model_Query-request message to the ML internal marketplace based on the use case requirements from Step 2.

4) If the query uses the federation mode of ML marketplaces, the internal ML marketplace searches for ML models that meet the requirements according to the received ML model metadata from internal and external ML marketplaces and arrives at best matching ML models.

5) The internal ML marketplace responds to the MLFO with a list of matching ML models and their associated information which may include ID, metadata and source of the model.

8.3.2 Model selection and push

The model selection and push sequence diagram is shown in Figure 3.

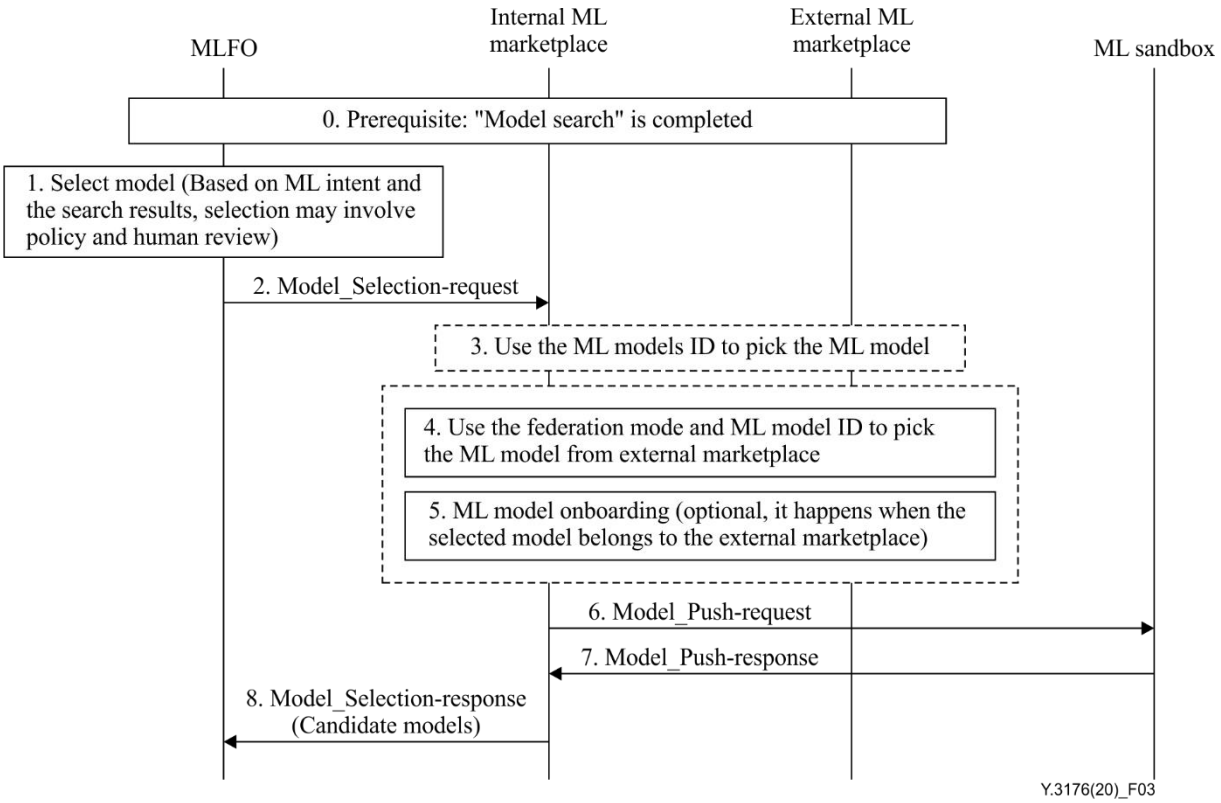

Figure 3 – Model selection and push

The sequences shown in Figure 3 are as follows:

Prerequisites: "Model search" (see clause 8.3.1)is completed.

1) Based on the ML intent (received in Step 1, Figure 2), the MLFO selects a ML model from the search results (received in the Model_Query-response message, see Step 5 in Figure 2).

   NOTE – Initial deployments of integrated ML marketplaces in networks may involve a human supervision of the ML models selected by the MLFO. This may be needed to verify whether the selected model is appropriate for the use case or whether it is as per network operator's defined policies. At a later stage, the selection of models is completely automated in the MLFO and this will remove the need for any human intervention.

2) The MLFO sends a Model_Selection-request message to the internal ML marketplace to indicate the selection, which may include ID of the selected model, the address of the ML pipeline subsystem in the ML sandbox, etc.

3) The ML internal marketplace or external marketplace picks the selected ML model using the ML model ID received in the Model_Selection-request message from the MLFO.

4) If the selected model belongs to the external ML marketplace, the internal ML marketplace uses the federation mode of ML marketplaces and the ML model ID to pick the ML model.

5) The selected model is on-boarded to the ML internal marketplace if the selected model belongs to an external ML marketplace.

6) The internal ML marketplace sends a Model_Push-request message to send the selected model object to the targeted ML sandbox.

7) The ML sandbox sends a Model_Push-response response message to the internal ML marketplace to indicate whether or not the model object was successfully received.

8) The internal ML marketplace sends a Model_Selection-response message to the MLFO to indicate whether or not the model selection was successfully completed.

8.3.3 Model discovery

The model discovery sequence diagram is shown in Figure 4.

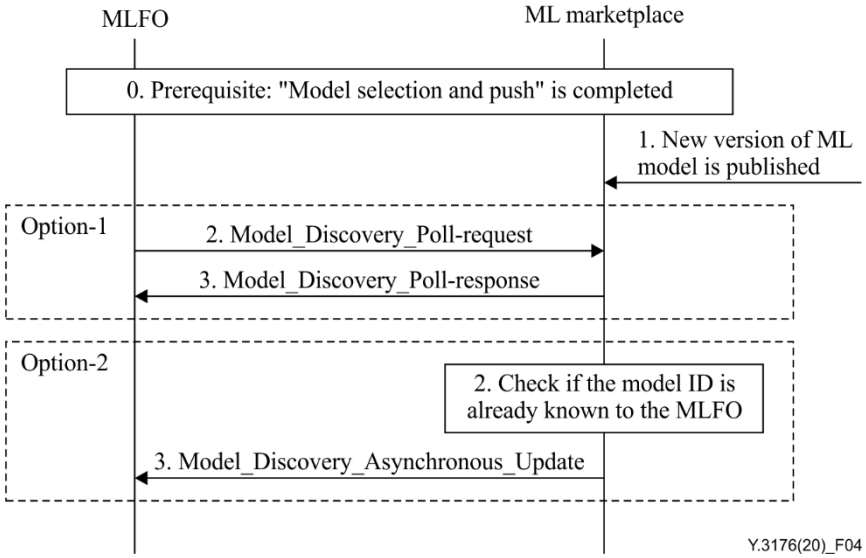

Figure 4 – Model discovery

The sequences shown in Figure 4 are as follows:

Prerequisites: "Model selection and push" (see clause 8.3.2) is completed.

1) A new version of a ML model is published in the ML marketplace.

   NOTE – This applies to both external and internal ML marketplaces.

Option-1:

2) A Model_Discovery_Poll-request message has been received from the MLFO.
3) The ML marketplace responds with a Model_Discovery_Poll-response message to the MLFO.

Option-2:

2) The ML marketplace checks if the ML model is known to the MLFO.
3) A Model_Discovery_Asynchronous_Update message containing ML model metadata or the ID of the newly published ML model is sent to the MLFO by the ML marketplace.

8.3.4 Model training

The model training sequence diagram is shown in Figure 5.

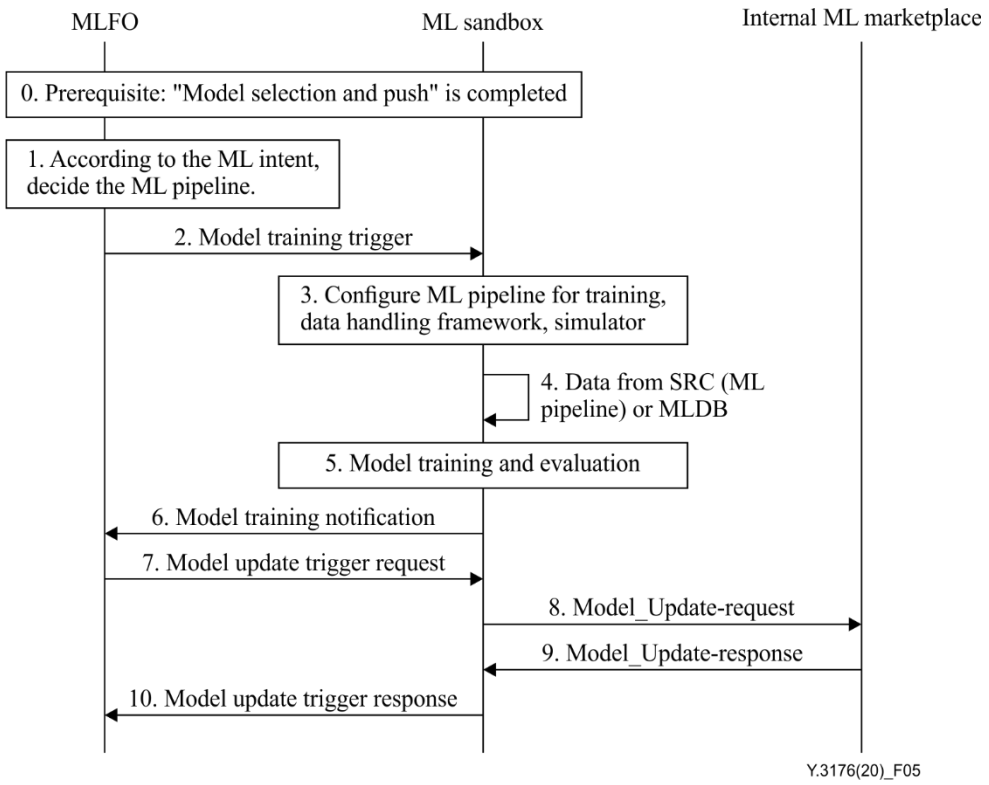

Figure 5 – Model training

The sequences shown in Figure 5 are as follows:

Prerequisites: "Model selection and push" (see clause 8.3.2) is completed.

1) According to the received ML intent, the MLFO provisions a ML pipeline for training purposes.
2) The MLFO triggers the training of models in the ML sandbox.
3) The ML pipeline for training and the data handling components [ITU-T Y.3174] are configured in the ML sandbox.
4) Data from a source (SRC) node [ITU-T Y.3172] or machine learning database (MLDB) [ITU-T Y.3174] is given as input for training the ML model.

   NOTE – The SRC node in the ML pipeline [ITU-T Y.3172] provides data from simulated and/or live underlay network.
5) ML model training and evaluation are done in the ML sandbox.
6) After the ML model is trained and evaluated, the ML sandbox notifies the MLFO.
7) The MLFO sends a model update trigger request message to the ML sandbox.
8) The ML sandbox sends a Model_Update-request message to the internal ML marketplace. The newly trained model is then published in the internal ML marketplace.
9) The internal ML marketplace notifies the result by sending a Model_Update-response message to the ML sandbox.
10) The ML sandbox notifies the result by sending a model update trigger response to the MLFO.

8.3.5 Model deployment

The model deployment sequence diagram is shown in Figure 6.

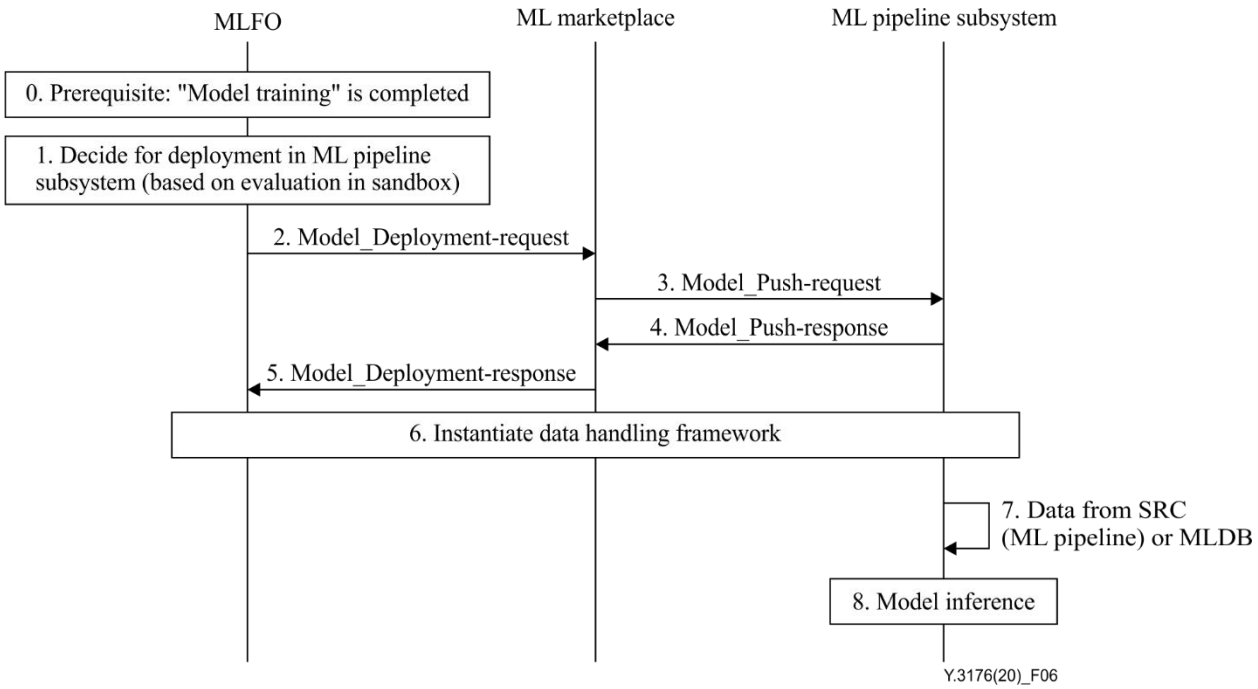

Figure 6 – Model deployment

The sequences shown in Figure 6 are as follows:

Prerequisites: "Model training" (see clause 8.3.4) is completed.

1) Based on the reached performance of the ML model in the ML sandbox, the MLFO decides that the ML model is ready for deployment in the ML pipeline subsystem.
2) The MLFO sends a Model_Deployment-request message to the ML marketplace.

   NOTE –This applies to both external and internal ML marketplaces.
3) A Model_Push-request message is sent to the ML pipeline subsystem.
4) After the model is pushed to the ML pipeline subsystem, a Model-Push-response message is sent back to the ML marketplace.
5) A ML_Deploy-response message is sent back to the MLFO acknowledging the successful deployment of the ML model in the ML pipeline subsystem.
6) According to clause 8.3.1 of [ITU-T Y.3174], MLFO instantiates the data handling components.
7) Data from SRC or MLDB is used as input for model inference.
8) Model inference is executed to make analyses, classifications or predictions, etc.

9 Security Considerations

General network security requirements and mechanisms in [ITU-T Y.2701] and [ITU-T Y.3101] are applicable for the integration of ML marketplaces in future networks including IMT-2020.

Bibliography

[b-ITU-T Y.3100] Recommendation ITU-T Y.3100 (2017), *Terms and definitions for IMT-2020 network*.

[b-ITU-T Y.3104] Recommendation ITU-T Y.3104 (2018), *Architecture of the IMT-2020 network*.

[ITU-T Y.3110] Recommendation ITU-T Y.3110 (2017), *IMT-2020 network management and orchestration requirements*.

[ITU-T Y.3111] Recommendation ITU-T Y.3111 (2017), *IMT-2020 network management and orchestration framework.*

[b-ITU-T Y-Sup.55] Supplement 55 to ITU-T Y.3170-series (2019), *Machine learning in future networks including IMT-2020: use cases*.

[b-ETSI-GS NFV-EVE] ETSI GS NFV-EVE 004 V1.1.1 (2016), *Network Functions Virtualisation (NFV)*; *Virtualisation Technologies; Report on the application of Different Virtualisation Technologies in the NFV Framework*.

[b-ETSI GR ENI 004] ETSI GR ENI 004 V2.1.1 (2019), *Experiential Networked Intelligence (ENI); Terminology for Main Concepts in ENI*.

[b-ISO-IEC 15938-17] ISO/IEC WD 15938-17 *Information technology – Multimedia content description interface – Part 17: Compression of neural networks for multimedia content description and analysis*.

[b-3GPP TS 28.554] 3GPP TS 28.554 Release 15, *Management and orchestration; 5G end to end Key Performance Indicators (KPI)*.

[b-DL Survey] Zhang, Chaoyun, Paul Patras, and Hamed Haddadi (2019), *Deep learning in mobile and wireless networking: A survey*. IEEE Communications Surveys & Tutorials.

[b-LF Acumos] Acumos AI Marketplace.
https://www.acumos.org

[b-ONNX] Open Neural Network Exchange (ONNX).
https://github.com/onnx/onnx

SERIES OF ITU-T RECOMMENDATIONS

Series A	Organization of the work of ITU-T
Series D	Tariff and accounting principles and international telecommunication/ICT economic and policy issues
Series E	Overall network operation, telephone service, service operation and human factors
Series F	Non-telephone telecommunication services
Series G	Transmission systems and media, digital systems and networks
Series H	Audiovisual and multimedia systems
Series I	Integrated services digital network
Series J	Cable networks and transmission of television, sound programme and other multimedia signals
Series K	Protection against interference
Series L	Environment and ICTs, climate change, e-waste, energy efficiency; construction, installation and protection of cables and other elements of outside plant
Series M	Telecommunication management, including TMN and network maintenance
Series N	Maintenance: international sound programme and television transmission circuits
Series O	Specifications of measuring equipment
Series P	Telephone transmission quality, telephone installations, local line networks
Series Q	Switching and signalling, and associated measurements and tests
Series R	Telegraph transmission
Series S	Telegraph services terminal equipment
Series T	Terminals for telematic services
Series U	Telegraph switching
Series V	Data communication over the telephone network
Series X	Data networks, open system communications and security
Series Y	**Global information infrastructure, Internet protocol aspects, next-generation networks, Internet of Things and smart cities**
Series Z	Languages and general software aspects for telecommunication systems

Printed in Switzerland
Geneva, 2020