International Telecommunication Union

ITU-T

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

Y.3606

(12/2021)

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

Big Data

Big data – Deep packet inspection mechanism for big data in network

Recommendation ITU-T Y.3606

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
Computing power networks	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.3606

Big data – Deep packet inspection mechanism for big data in network

Summary

Recommendation ITU-T Y.3606 specifies a mechanism for deep packet inspection (DPI) applied to big data in the network context. The scope of Recommendation ITU-T Y.3606 includes: an introduction to the differences between generic DPI and big data DPI; an overview of big data processing procedures; the relationship between DPI and big data-related technologies; a data classification mechanism using DPI for big data in networks; a data pre-processing mechanism using DPI for big data in networks; a coordination processing mechanism for DPI in the context of big data in networks; and the interfaces between DPI and upper-layer big data-related methods.

History

Edition	Recommendation	Approval	Study Group	Unique ID*
1.0	ITU-T Y.3606	2021-12-06	13	11.1002/1000/14776

Keywords

Big data, deep packet inspection, mechanism.

* 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/software copyrights, 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 appropriate ITU-T databases available via the ITU-T website at http://www.itu.int/ITU-T/ipr/.

Table of Contents

Recommendation ITU-T Y.3606

Big data – Deep packet inspection mechanism for big data in network

1 Scope

The scope of this Recommendation includes:

a) an introduction to the differences between generic deep packet inspection (DPI) and big data DPI;

b) an overview of big data processing procedure;

c) the relationship between deep packet inspection and big data related technologies;

d) a data classification mechanism using deep packet inspection for big data in networks;

e) a data pre-processing mechanism using deep packet inspection for big data in networks;

f) a coordination processing mechanism for deep packet inspection for big data in networks;

g) the interfaces between deep packet inspection and upper-layer big data related methods;

h) other aspects of DPI mechanism for big data in network.

NOTE – Big data itself and big data technologies lie outside the scope of this Recommendation.

Implementers and users of the described techniques shall comply with all applicable national and regional laws, regulations and policies. The mechanisms described in this Recommendation may not be applicable to the international correspondence in order to ensure the secrecy and sovereign national legal requirements placed upon telecommunications, but they shall comply with the ITU Constitution and Convention.

The afore-mentioned national and regional laws, regulations and policies include laws, regulations and policies related to personal data.

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 X.200] Recommendation ITU-T X.200 (1994) | ISO/IEC 7498-1:1994, *Information technology – Open Systems Interconnection – Basic reference model: The basic model*.

[ITU-T Y.2770] Recommendation ITU-T Y.2770 (2012), *Requirements for deep packet inspection in next generation networks.*

[ITU-T Y.2771] Recommendation ITU-T Y.2771 (2014), *Framework for deep packet inspection.*

[ITU-T Y.2773] Recommendation ITU-T Y.2773 (2017), *Performance models and metrics for deep packet inspection.*

[ITU-T Y.3021] Recommendation ITU-T Y.3021 (2012), *Framework of energy saving for future networks.*

[ITU-T Y.3600] Recommendation ITU-T Y.3600 (2015), *Big data – Cloud computing based requirements and capabilities.*

[ITU-T Y.3601] Recommendation ITU-T Y.3601 (2018), *Big data – Framework and requirements for data exchange.*

3 Definitions

3.1 Terms defined elsewhere

This Recommendation uses the following terms defined elsewhere:

3.1.1 big data [ITU-T Y.3600]: A paradigm for enabling the collection, storage, management, analysis and visualization, potentially under real-time constraints, of extensive datasets with heterogeneous characteristics.

NOTE – Examples of datasets characteristics include high-volume, high-velocity, high-variety, etc.

3.1.2 deep packet inspection (DPI) [ITU-T Y.2770]: Analysis, according to the layered protocol architecture OSI-BRM [ITU-T X.200], of

- payload and/or packet properties (see list of potential properties in clause 3.2.11 of [ITU-T Y.2770] deeper than protocol layer 2, 3 or 4 (L2/L3/L4) header information, and
- other packet properties

in order to identify the application unambiguously.

NOTE – The output of the DPI function, along with some extra information such as the flow information is typically used in subsequent functions such as reporting or actions on the packet.

3.1.3 DPI analyser [ITU-T Y.2771]: A subsequent entity in the DPI processing path (within a DPI policy enforcement function) with focus on comparison functions between the particular packet headers and payloads of preselected packet flows. The primary scope of the DPI analyser is related to the evaluation of DPI policy *conditions* against *preselected* incoming packets.

NOTE – The DPI analyser may be located after a DPI scanner (see clause 3.2.5 of [ITU-T Y.2771]). The DPI analyser may provide the functionality of an intrusion detection system (IDS) analyser.

3.1.4 DPI engine [ITU-T Y.2770]: A subcomponent and central part of the DPI functional entity which performs all packet path processing functions (e.g., packet identification and other packet processing functions in Figure 6-1 of [ITU-T Y.2770]).

3.1.5 DPI node [ITU-T Y.2771]: A network element device that realizes the DPI related functions. It is thus a generic term used to designate the realization of a DPI physical entity.

NOTE – Functional perspective: the DPI node function (DPI-NF) comprises the DPI policy enforcement function (DPI-PEF) and the (optional) local policy decision function (L-PDF), hence, the DPI-NF is functionally equal to the DPI functional entity.

3.2 Terms defined in this Recommendation

This Recommendation defines the following term:

3.2.1 DPI meta-engine: An independent basic component of every processing stage in the inner path of a DPI engine.

NOTE – For example, if there are several independent components in the DPI scan function (DPI-ScF) stage, then every component is called a DPI-ScF meta-engine.

4 Abbreviations and acronyms

This Recommendation uses the following abbreviations and acronyms:

BDAP Big Data Application Provider

BDIP Big Data Infrastructure Provider

BDSU	Big Data Service User
DFA	Deterministic Finite Automation
DNNF	Determining Next Node Function
DP:	Data Provider
DPI	Deep Packet Inspection
DPI-AcEF	DPI Action Execution Function
DPI-AnF	DPI Analyser Function
DPI-ExF	DPI pre-extraction Function
DPI-NF	DPI Node Function
DPI-PEF	DPI Policy Enforcement Function
DPI-PIB	DPI Policy Information Base
DPI-ScF	DPI Scan Function
DPI-TrF	DPI pre-transformation Function
EMS	Element Management System
FIB	Forwarding Information Base
HTTP	Hypertext Transfer Protocol
ICT	Information and Communication Technology
IDS	Intrusion Detection System
L-PDF	Local Policy Decision Function
NFA	Non-deterministic Finite Automation
NMS	Network Management System
OLAP	Online Analysis and Process
OSI-BRM	Open System Interconnection-Basic Reference Model
PFF	Packet Forwarding Function
PIB	Policy Information Base
R-PDF	Remote Policy Decision Function
SNMP	Simple Network Management Protocol
TCAM	Ternary Content Addressable Memory
UDP	User Datagram Protocol
XML	extensible Markup Language
YANG	Yet Another Next Generation

5 Conventions

In this Recommendation, the words "shall", "should" and "may" sometimes appear, in which case they are to be interpreted, respectively, as "is required to", "is recommended" and "can optionally". The appearance of such phrases or keywords in material explicitly marked as informative are to be interpreted as having no normative intent.

6 Introduction of differences between generic DPI and big data DPI

Big data has some features that make it different from common data. Such features are related, for example, to volume, velocity and variety (see [ITU-T Y.3600]). The mechanism of traditional DPI has solved some of the problems for network application or service awareness; however, these three features present problems and challenges to DPI-related technologies. These features differentiate DPI for big data from traditional DPI that cannot work efficiently in the big data context.

In order to facilitate illustration, in this Recommendation traditional DPI is called "generic DPI" and DPI applied in the context of big data in networks is called "big data DPI".

The technology covered by this Recommendation is big data in telecommunication network settings. General purpose big data, and big data technologies and services, lie outside the scope of this Recommendation. The context of big data in networks is large-scale data sets stored, transported and used in telecommunication applications. This Recommendation does not address general-purpose requirements or frameworks for big data storage, transport or applications.

6.1 Difference between generic DPI and big data DPI for the velocity feature

Figure 6-1 depicts the difference between generic DPI and big data DPI in the case of the velocity feature; while not all differences are depicted, a typical aspect related to a DPI engine is illustrated. Figure 6-1 shows that generic DPI technologies do not take account of problems caused by a huge change in velocity for big data.

Figure 6-1-b shows that the core functional component of a generic DPI, i.e., a DPI engine, may be re-structured for parallel and distributed processing of big data with an increase in data velocity. A scheduler should be designed within the DPI engine.

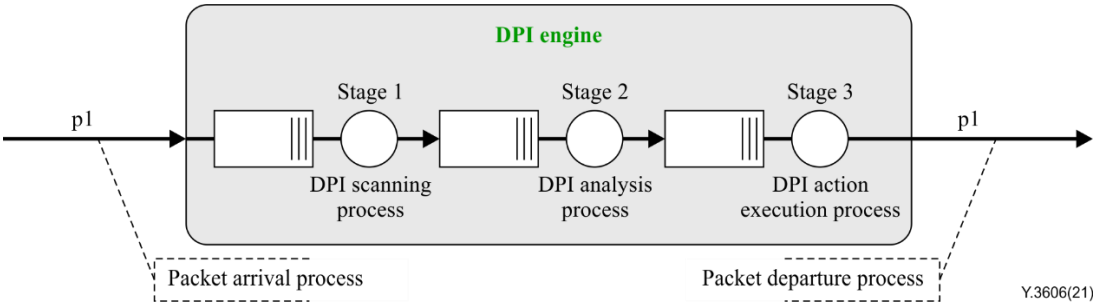
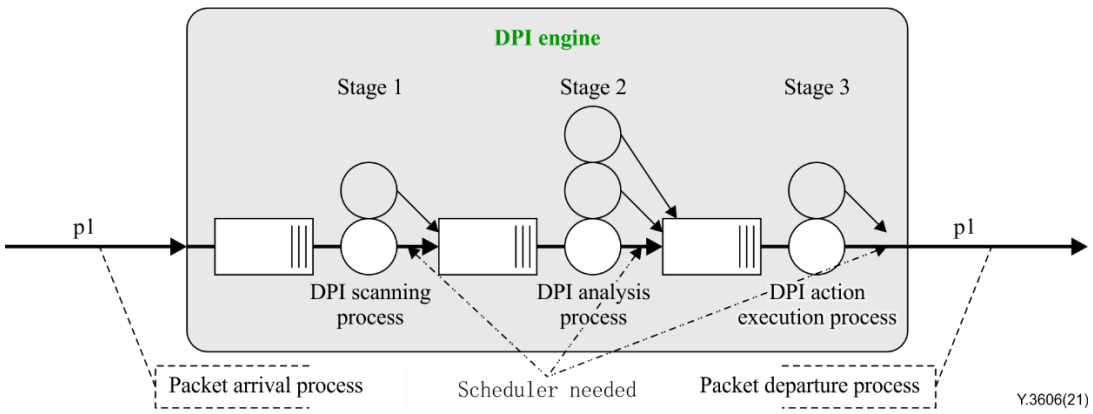


Figure 6-1-a – Generic DPI processing method related to a DPI engine



b

Figure 6-1-b – Possible big data DPI processing method related to a DPI engine

6.2 Difference between generic DPI and big data DPI for the volume feature

Figure 6-2 depicts the difference between generic DPI and big data DPI for the volume feature; while not all differences are depicted, an aspect related to DPI entities is illustrated. Figure 6-2-b gives a modified model for the DPI processing method. The volume of big data is possibly so huge that a single DPI-PEF does not have the ability to process the data in time. Therefore, a multi-entity coordination, scheduler and buffer mechanism should be introduced to treat the problems caused by big data.

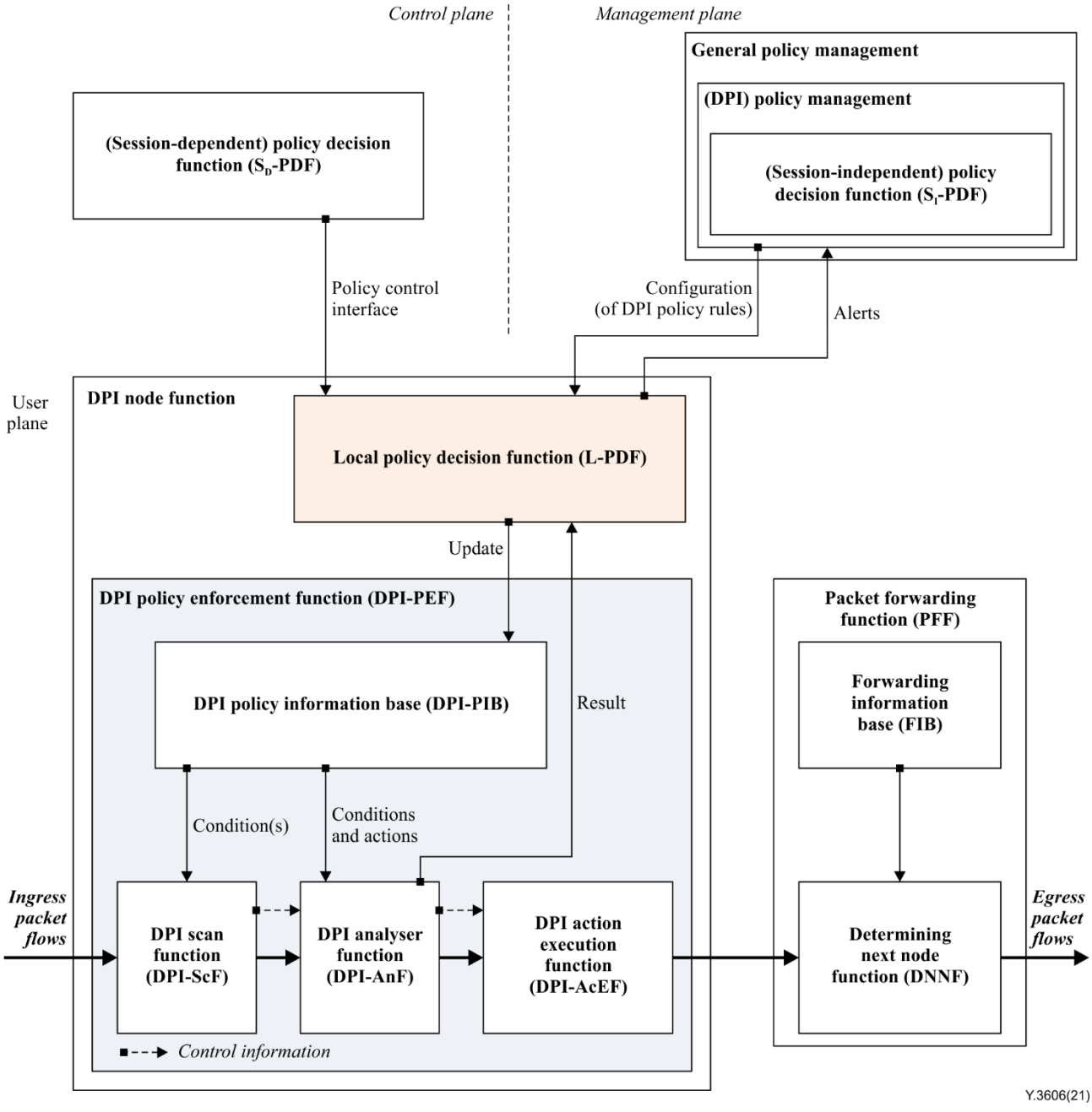
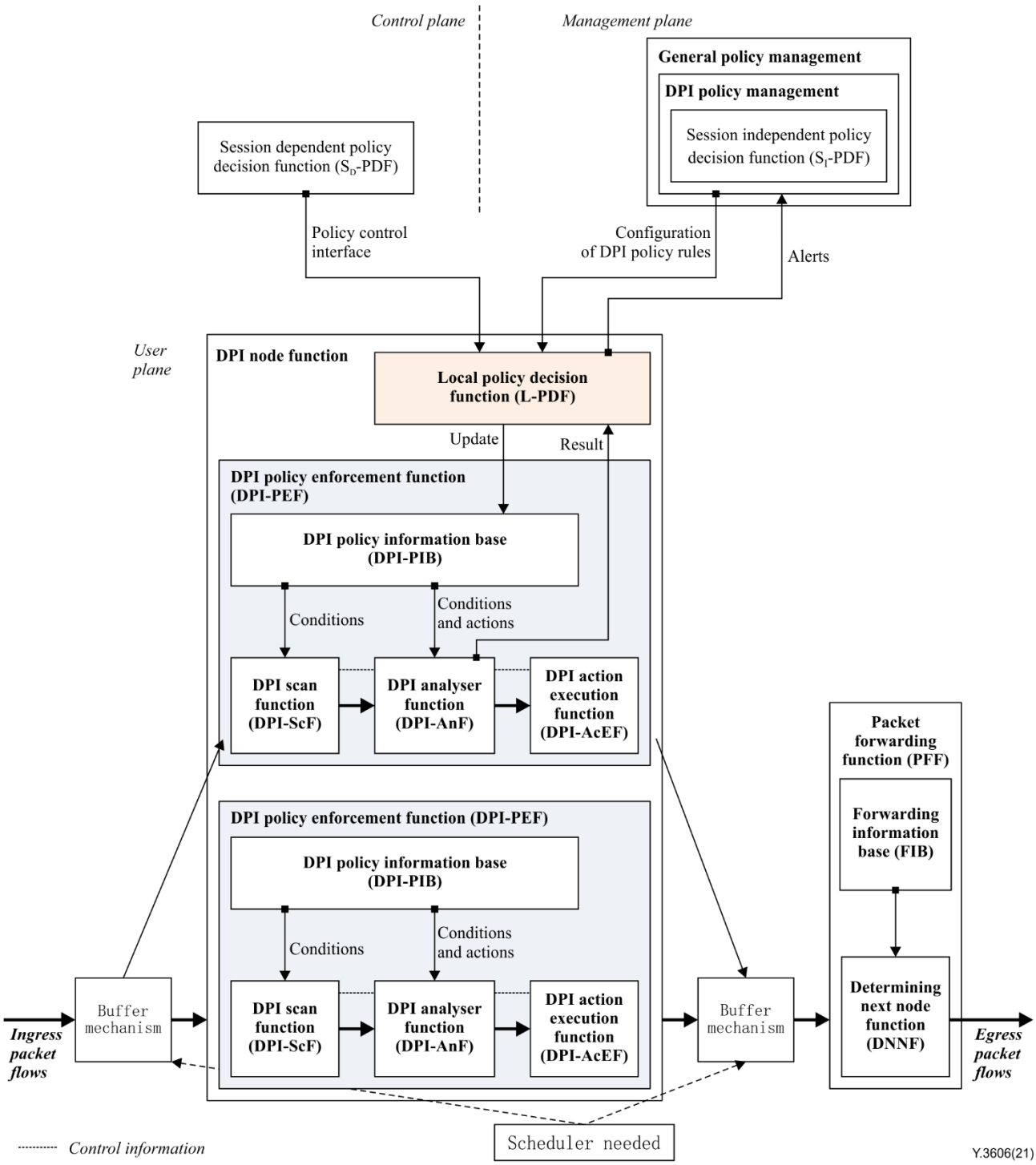


Figure 6-2-a – Generic DPI processing method within a DPI entity



NOTE – The packet forwarding function (PFF) lies outside the scope of this Recommendation.

Figure 6-2-b – Big data DPI processing method in big data in network context

6.3 Difference between general DPI and big data DPI for the variety feature

Figure 6-3 depicts the difference between generic DPI and big data DPI for the variety feature. As for big data in networks, besides structured data, there are also unstructured data and semi-structured data; the former linear rule representation (depicted in Figure 6-3-a) cannot describe the DPI rules very well because of data variety. Furthermore, as for consideration of aspects of performance and accurate data value extraction, etc., there are many new requirements for the new DPI process mechanism.

Figure 6-3-b is an example of information structure for the DPI rules applied to big data in the network context. Compared to the generic DPI rule information structure depicted in Figure 6-3-a,

the DPI rule information structure for the big data context should be designed with full consideration of the variety of big data. Then, the structure of DPI rule information is appropriate to be designed as multi-table style and one DPI rule in a rule table may refer to one or more rules in other rule tables.

This Recommendation studies how to use DPI-related technologies in context of big data in networks.

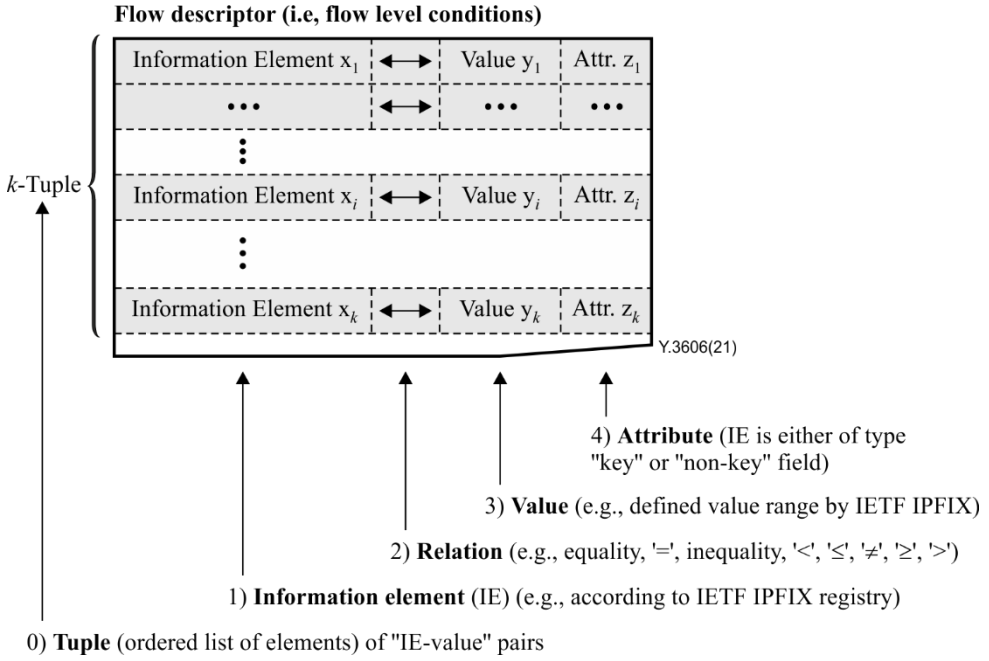
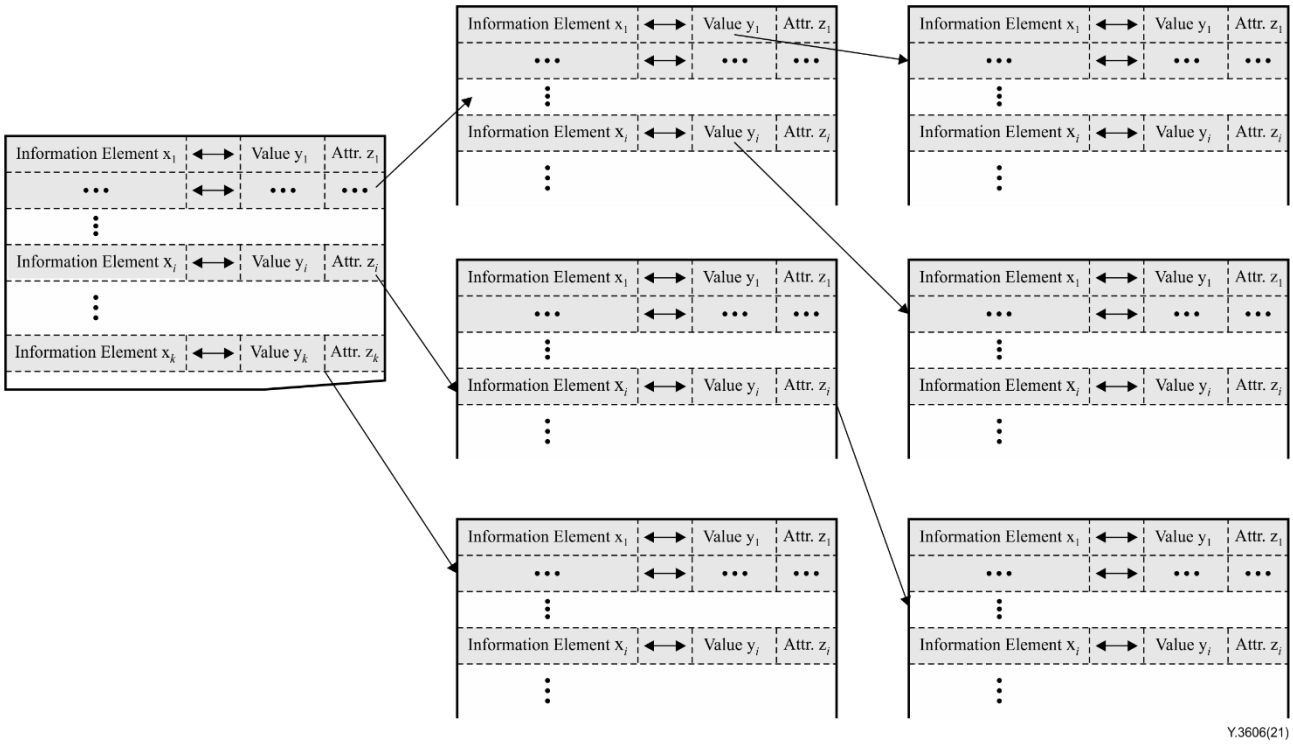


Figure 6-3-a – Generic DPI rules



NOTE – For IETF TPFIX, see [b-IETF RFC 5101].

Figure 6-3-b – Big data DPI rules for DPI applied in context of big data in network

7 Overview of big data processing procedure

7.1 An example model for big data processing

Figure 7-1 depicts an example of a model for big data processing. As to the example model, data collection is the starting point of the whole process, while visualization is its end point. The whole process of the example model mainly includes the following steps.

- Data collection: the step in which big data are collected from the network.
- Data pre-processing: the step in which big data are pre-processed before storage in the relevant database. This step includes data extraction, data transformation and data loading.
- Data storage: the step in which big data is stored in the relevant database.
- Data analysis: the step in which big data are analysed (including data mining, on-line analysis and processing (OLAP)).
- Prediction: the step in which future information is predicted based on data analysis. This step is optional.
- Visualization: the step in which results of data analysis and prediction are represented and displayed in some format (e.g., graph or table) that is easy to understand.

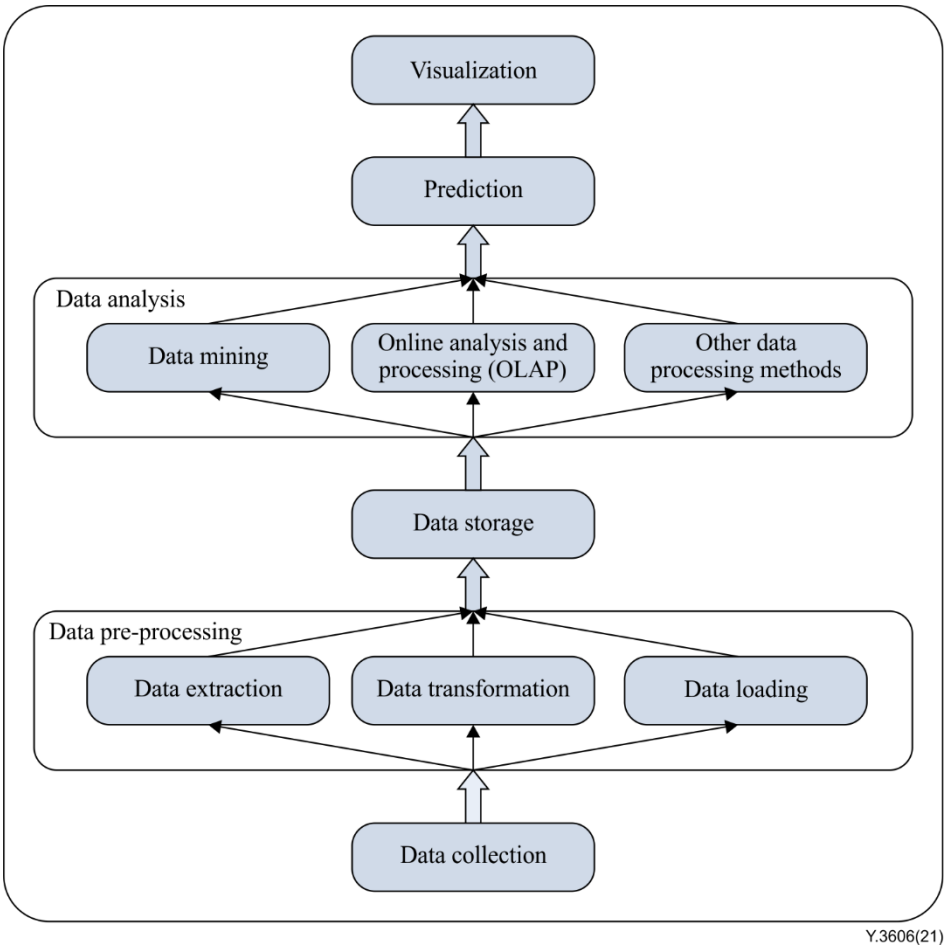


Figure 7-1 – Example model of big data processing

7.2 Big data processing model in ITU context

Figure 7-2 illustrates the cloud computing-based big data system context within ITU (see also Figure 7-1 of [ITU-T Y.3600]). [ITU-T Y.3601] also specifies a framework and requirements for big data exchange.

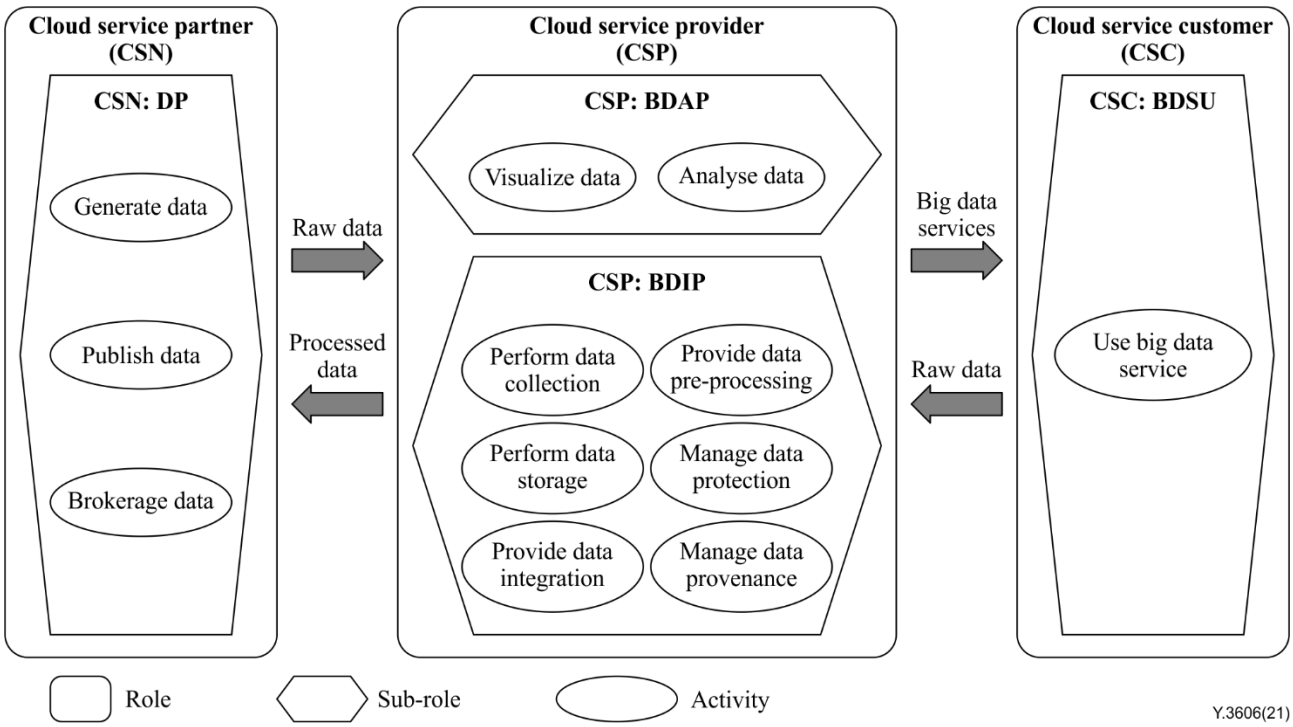


Figure 7-2 – Cloud computing based big data system context

8 Connection between deep packet inspection and big data process

8.1 Summary of relationship between DPI and big data processing model

Figure 8-1 depicts the relationship between DPI and the big data processing model described in clause 7. On the one hand, DPI functions can efficiently collect data, which is the basic process step of the big data process model. On the other hand, the output of big data process procedures can be beneficial to maintain and manage the DPI policy information base (DPI-PIB) and provide better service to the DPI engine (see [ITU-T Y.2771]).

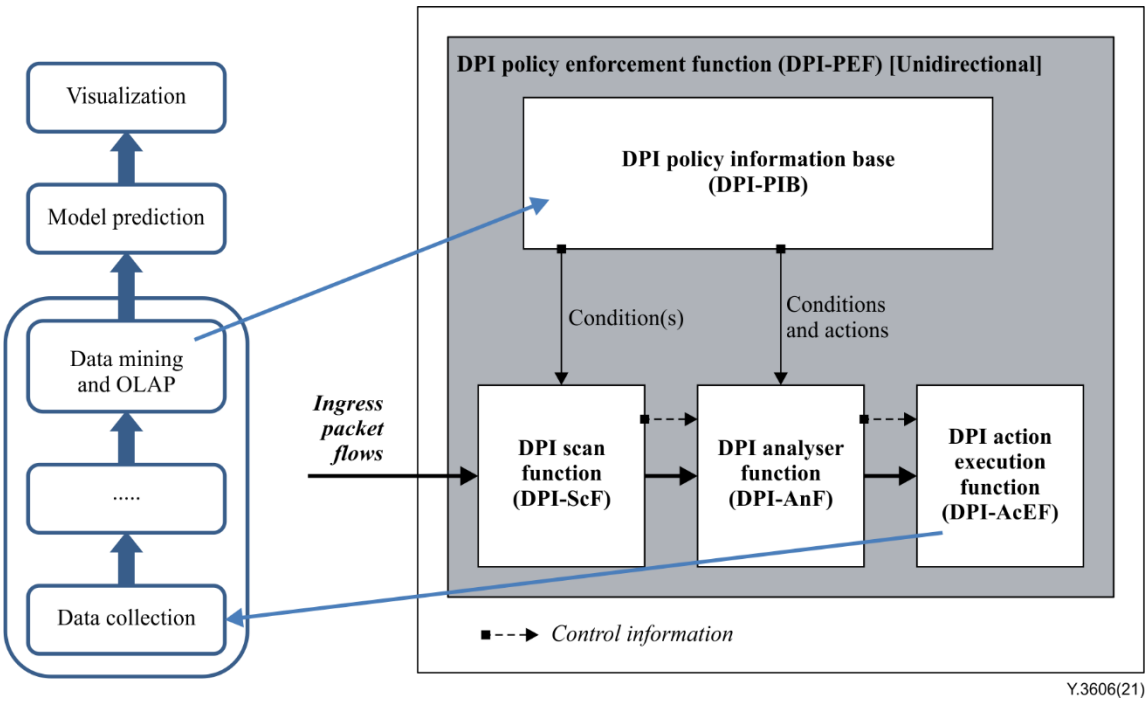


Figure 8-1 – Relation between DPI and big data processing model

8.2 DPI carries out the data collecting functions for big data

DPI can usefully be used for big data collection and pre-processing. DPI can first identify different network traffic flows and send them to different big data processing entities based on preset requirements. In addition, DPI can realize some pre-processing functions before it sends the data to big data processing entities.

8.3 Big data process carries out partial functions of R-PDF

The data carried on the network change over time and as a result policy information base (PIB) maintenance becomes more difficult. On the one hand, PIB scale increases. On the other hand, the PIB becomes less efficient because of continual changes in the data. Big data process procedures can help to find the pattern of the data and help to improve the real-time PIB; the PIB can then be appropriate for the network data over time.

9 Data classification mechanism for big data DPI

9.1 Basic mechanism of information representation

9.1.1 General information representation method

When designing a DPI product or system, the regular expression-based method is one most commonly used. In that case, representation and storage of the following three classes of information are necessary:

- regular expression to be used to match the data in a packet;
- data string from the packet to be matched;
- run-time status data to be used by the regular expression-based method.

9.1.2 Deterministic finite automation and non-deterministic finite automation

Deterministic finite automation (DFA) and non-deterministic finite automation (NFA) are two main algorithms of the regular expression-based method. NFA is a regular-express-central algorithm meanwhile DFA is a matched-data-string-central method. Both DFA and NFA have a large scale of

state graph for representation and storage; the state graph is possibly too large for a storage component.

9.1.3 Hybrid and distributed storage method

To solve the problem of large-scale state graph of DFA and NFA, based on big data related architecture, it is recommended that multiple storing components be used for its storage.

Each single storage component only stores one segment or one branch of the state graph.

When the processing component executes DFA or NFA functions, it fetches the state graph data from the corresponding storage components only if it is necessary.

9.2 Basic data classification mechanism used deep packet inspection

When DPI is used in the big data context, generally, a single DPI entity can hardly finish real-time processing when the velocity or volume of big data in a network exceeds a limit. There are two effective processing methods that should be used in such circumstances:

- distributed processing mode;
- cascaded processing mode.

9.2.1 Distributed processing mode

In distributed processing mode, the big data in a network are handled by two or more processing entities. Each processing entity only handles part of the data that the entity can process in a given time. A typical model of distributed DPI processing for big data in a network is depicted in Figure 9-1, in which big data in a network is split into smaller data parts that can each be processed by a single entity.

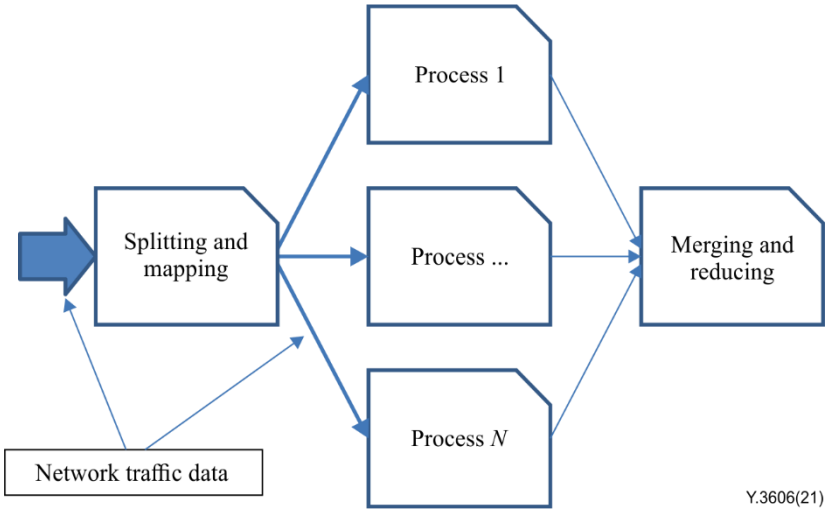


Figure 9-1 – A typical model of distributed processing of DPI for big data

9.2.2 Cascaded processing mode

In cascaded processing mode, a series of processing entities is coordinated to realize DPI functions. Each processing entity only realizes some DPI functions. A typical model of cascaded DPI processing for big data in a network is depicted in Figure 9-2, in which the procedure for DPI to process big data in network is composed of a series of steps, each of which can be carried out by a single DPI entity. Furthermore, the DPI entity handling larger traffic realizes simpler functions. For example, process 1 can only realize a Bloom filter meanwhile process *N* can realize a DFA.

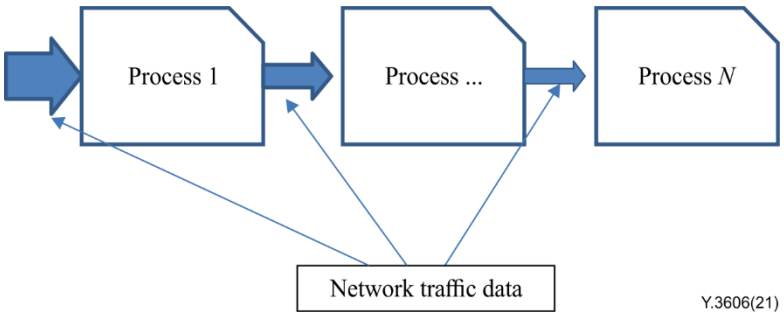


Figure 9-2 – A typical model of cascaded processing of DPI for big data

9.3 Data classification mechanism for structured data used DPI

Generally, the data classification function for structured data is easier to realize than that for non-structured data. There are two different application scenarios for DPI for structured data:

- protocol-known structured data;
- protocol-unknown structured data.

Different data classification mechanisms can be used for protocol-known or protocol-unknown structured data, respectively.

9.3.1 Data classification mechanism for protocol-known structured data

Because the protocol that is used to transport structured data is determined, the packets that include such data can then be effectively identified and classified. It is appropriate and sufficient to use data and mask representation to design DPI policy conditions.

9.3.2 Data classification mechanism for protocol-unknown structured data

In this application scenario, the protocol that is used to transport structured data cannot be directly used by DPI functions, so it is impossible to use only data and mask representation to identify and classify the packets that include it.

An appropriate data classification mechanism for protocol-unknown structured data is that DPI policy conditions are laid out based on regular expression.

An alternative feasible classification mechanism for protocol-unknown structured data uses a two-step method:

- identification of the protocol structure that is used to carry the such structured data;
- use of a data classification mechanism for protocol-known structured data.

9.4 Data classification mechanism for non-structured data used DPI

The data classification mechanism for non-structured data is more complicated than that for structured data. There are five aspects related to data classification mechanism for non-structured data to be specified, as follows:

- multi-stage mechanism;
- packet-aggregate mechanism;
- decompression mechanism;
- classification mechanism based on a Bloom filter;
- classification mechanism based on regular expression.

9.4.1 Multi-stage mechanism

Figure 9-2 shows a cascaded process model to solve the problem that a single DPI entity cannot carry complicated DPI functions.

In a data classification mechanism for non-structured data, such a problem certainly exists and worsens somewhat. A multi-stage mechanism similar to the model given in Figure 9-2 is therefore appropriate in this case. That is to say, the first stage aims to perform a preliminary classification function and succeeding stages are responsible for more accurate classification.

9.4.2 Packet-aggregate mechanism

Under many circumstances, a non-structured data unit (e.g., audio or video) may be carried by thousands of packets. It is usually not effective to inspect such packets one by one. A feasible mechanism uses a packet-aggregate mechanism to merge these packets and inspect the merged data (see Figure 9-3).

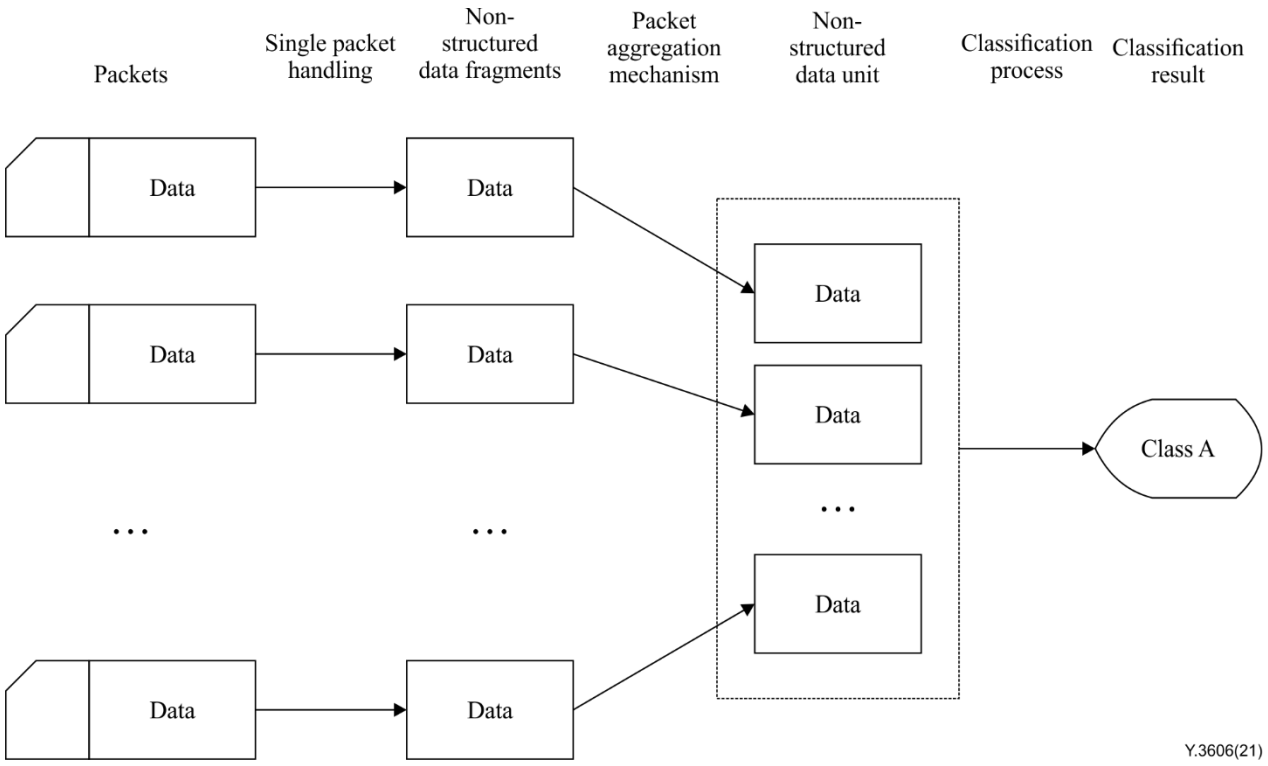


Figure 9-3 – Packet-aggregate mechanism

9.4.3 Decompression mechanism

Non-structured data are usually transported after being compressed, a decompression mechanism is then needed before classification processing.

Note that the packet-aggregate mechanism can also be applied to compressed non-structured data.

9.4.4 Classification mechanism based on a Bloom filter

A Bloom filter (see Appendix I of [ITU-T Y.2771]) is a good solution when applied in a high-efficiency and low-accuracy application scenario. A Bloom filter has a false-negative error rate of 0 while the false-positive error rate is greater than 0, therefore the succeeding process should tolerate the shortcomings described in clauses 9.4.1 to 9.4.3.

In the multi-stage mechanism described in clause 9.3.1, a classification mechanism based on a Bloom filter is appropriate for the first stage.

9.4.5 Classification mechanism based on regular expression

When applied in a high-accuracy application scenario, the regular expression-based mechanism is more appropriate than that which is Bloom filter related. Note that the regular expression-based mechanism has high time complexity and space complexity.

In the multi-stage mechanism described in clause 9.3.1, a classification mechanism based on regular expression is better applied to stages other than the first.

9.5 Data classification mechanism for semi-structured data used DPI

Semi-structured data differs from structured or non-structured types. It is more complicated than structured data, while it is simpler that non-structured data. It has some features that are similar to structured data and others to non-structured data.

NOTE –Compared with structured data, the structure of semi-structured data varies a lot. Nonetheless, semi-structured data is easier to transform to structured data than the non-structured type. For example, an object exchange model is typical semi-structured data.

An appropriate processing method is to transform semi-structured data to structured data as much as possible. Such data can then be handled as structured data. Data that cannot be transformed can be handled as non-structured data.

9.6 Energy efficient mechanisms for classification function

The energy efficient mechanisms for a classification function described in this clause are not mandatory but optional to DPI implementations.

Energy saving in the information and communication technology (ICT) field is an important issue, as has been identified in designing future networks. One basic objective of the development of future networks is the demonstration of environmental awareness, which may be realized via energy-saving technologies [ITU-T Y.3021]. The importance of these issues is increasing with the widespread implementation of network equipment and greater energy consumption. DPI is a network building block that needs more resources and power. Energy efficient mechanisms for classification functions of DPI need to be specified to save energy and for various other objectives. Energy efficient mechanisms are important for the scenario in which DPI is applied in the big data context, because the big data scenario needs more data process performance and energy dissipation is usually high.

Ternary content addressable memory (TCAM) with its high matching speed is widely used in the industry to store the DPI policy rule. The challenge is the high power dissipation of TCAM. A two-phase energy efficient mechanism for DPI can be used to lower power dissipation, especially in the scenario to identify certain classes of network traffic (e.g., attack traffic).

To take the identification of attack traffic as an example, the two-phase mechanism can be described as follows:

1) use of parallel Bloom filters to exclude normal packets not including attack signatures;
2) use of TCAM to inspect suspicious packets including real attack packets and false positive packets of the first phase.

The false negative probability of Bloom filters is zero and the false positive probability is very low. Since most network data do not include attack signatures, the method can get high DPI performance with low power dissipation.

The functional model of Bloom filter-based probabilistic DPI is illustrated in Figure 9-4. The policy rule for probabilistic DPI is as follows (note that the policy rule is just an example).

If packet 'P' contains signatures from a DPI signature set *'S'* (as policy condition, see [ITU-T Y.2770]), where the signature set is given by *'S'* = {*'S1'*, *'S2'*,..., *'Sm'*}, then discard the packet (as policy action, see [ITU-T Y.2770]).

The Bloom filter BF_S for signature set *S* is generated by the set of hash functions H_1, H_2, …, H_k (see Appendix I of [ITU-T Y.2771]). The policy rule is converted to:

If H_1('P'), H_2('P'),…, H_k('P') match BF_S, then discard the packet.

Before the DPI analyser function (DPI-AnF) compares the arriving packet against this DPI policy rule condition, the DPI-ScF needs to determine the offset and length in the arriving packet that are used to match against the DPI policy rule conditions.

There are two principal options:

1) For the protocol-stack aware DPI rule condition, the offset and signature length in the set '*S*' is known, and the scanner reports the offset and length information to the DPI analyser directly.

2) For the protocol-agnostic DPI rule condition, the DPI scanner needs to scan and determine the offset and length. The DPI scanner reports that information to the DPI analyser.

The DPI analyser generates the hash result of packet P using H_1, H_2, …, H_k, matches the results with the BF_S and reports the match result to the DPI action execution (DPI-AcEF) function. The DPI-AcEF reports the matching result ("true" or "false") and discards the packet when the match result is estimated to be true, or else passes the packet to the PFF. If the generated match result does not match the BF_S, the DPI scanner needs to scan from the 'offset+1' byte (offset = offset + 1), and the process continues until the end of the packet.

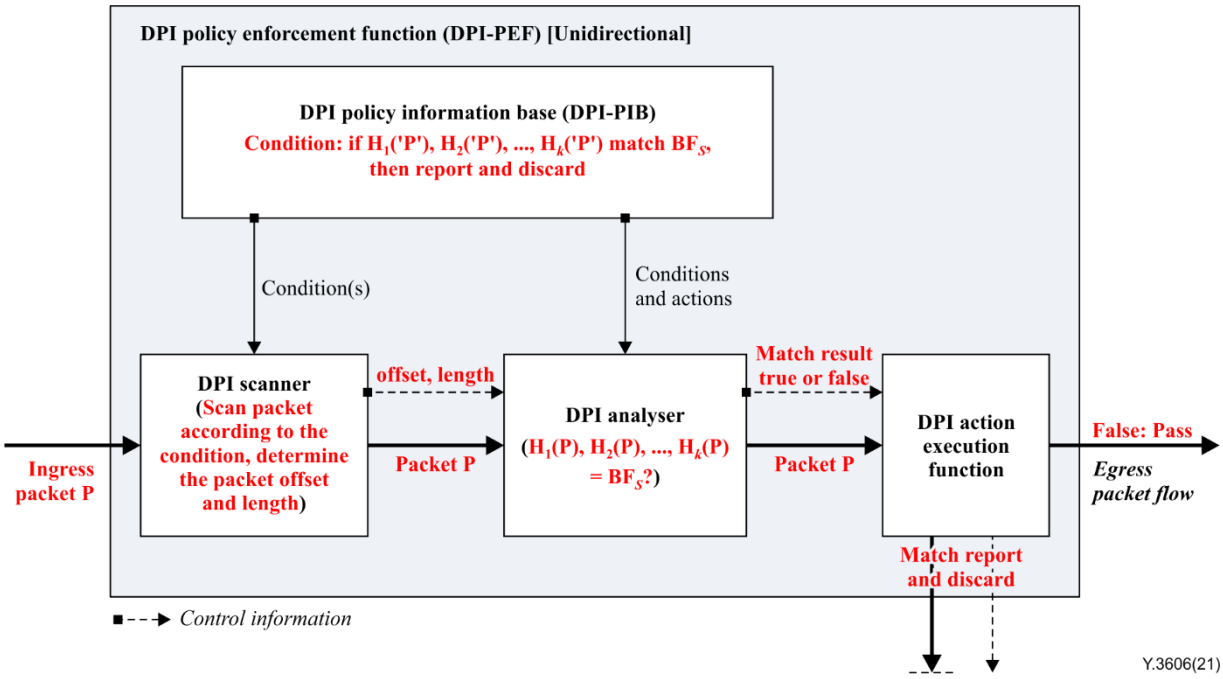


Figure 9-4 – Functional model of Bloom filter-based probabilistic DPI

Note that the match result of the DPI analyser is Boolean, i.e., true or false; hence there is no value indicating a positive match with probability *p* (with *p* between 0% and 100%). However, the entire DPI packet processing path stages as such lead to probabilistic DPI results due to the inherent false positive rate f-p, as part of the DPI policy condition.

10 Data pre-processing mechanism used big data DPI

10.1 Extension of DPI-engine

10.1.1 DPI pre-extraction function and DPI pre-transformation function

Generally, a DPI engine is composed of a series of functions that consist of the DPI-ScF, DPI-AnF and DPI-AcEF.

In the big data context, DPI is recommended to have partial pre-processing functions. The DPI engine is then the appropriate component to realize such pre-processing functions.

However, none of DPI-ScF, DPI-AnF and DPI-AcEF are appropriate for realization of the pre-processing functions. So, two new functions are introduced here to meet the pre-processing requirements:

- DPI pre-extraction function (DPI-ExF): responsible for the partial or complete extraction function for big data;
- DPI pre-transformation function (DPI-TrF): responsible for the partial or complete transformation function for big data.

10.1.2 Extension of the inner path of a DPI engine

Based on specification in clause 10.1.1, Figure 10-1 shows the inner path of the DPI engine with pre-processing functions. The DPI-ExF and the DPI-TrF should be designed between the DPI-AnF and the DPI-AcEF. Note that the DPI-ExF and the DPI-TrF can be set to bypass mode according to application requirements.

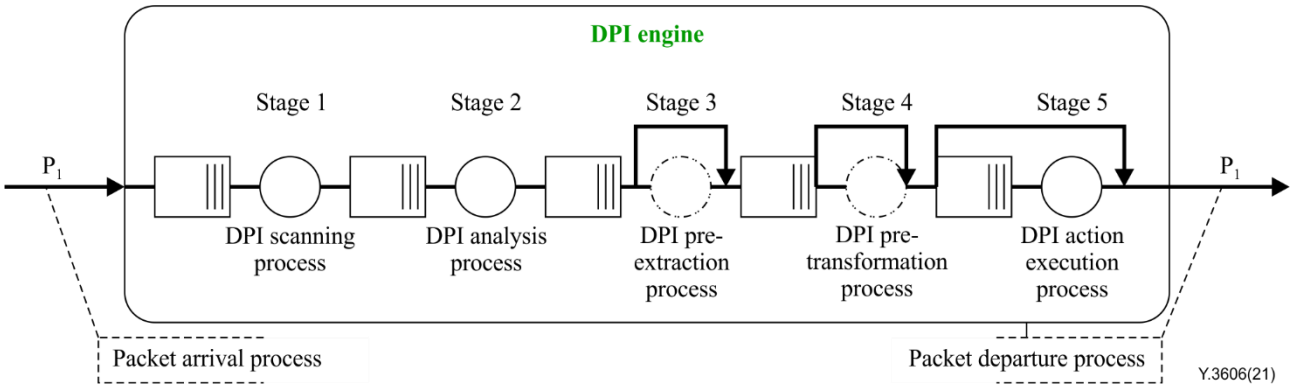


Figure 10-1 – Inner path of DPI engine supporting pre-processing

10.1.3 DPI meta-engine

Because the general architecture of big data processing model is based on distributed computing, the DPI architecture for big data is also distributed-computing based. In addition, if the DPI engine is the basic processing unit, then the efficiency would not be optimal because the processing stages within a DPI engine do not match each other in performance.

Therefore, a new concept or functional component, the DPI meta-engine, is introduced as a basic processing unit. The DPI meta-engine can be thought as minimal independent processing unit within a DPI engine. Furthermore, a DPI meta-engine can be thought as minimal independent processing unit within DPI-ScF, DPI-AnF, DPI-ExF, DPI-TrF or DPI-AcEF. In other words, all such DPI functions can be implemented based on a DPI meta-engine.

10.1.4 DPI multiple meta-engine

As is specified in clause 10.1.3, DPI-ScF, DPI-AnF, DPI-ExF, DPI-TrF and DPI-AcEF can be implemented based on a DPI meta-engine. There are then different kinds of DPI meta-engine corresponding to DPI-ScF, DPI-AnF, DPI-ExF, DPI-TrF and DPI-AcEF, respectively.

Within a DPI engine, it is possible to have more than one of each type of DPI meta-engine and the number of DPI meta-engines corresponding to DPI-ScF, DPI-AnF, DPI-ExF, DPI-TrF and DPI-AcEF can differ. e.g., there are two of the DPI-ScF and three of the DPI-AnF types (see Figure 10-2).

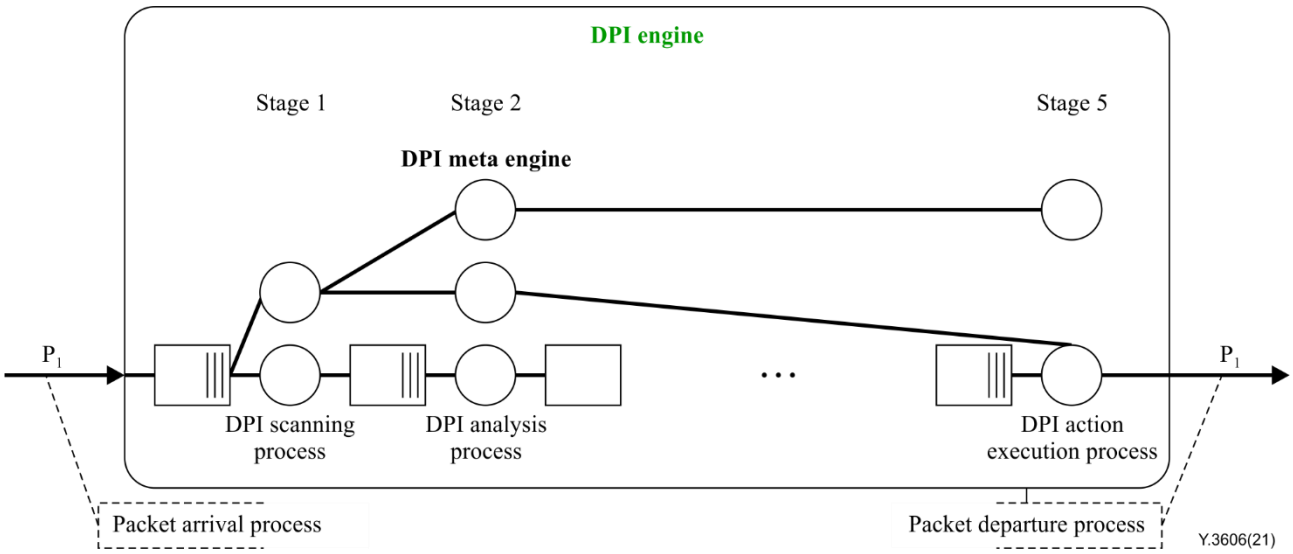


Figure 10-2 – DPI meta-engines in the data path

10.1.5 General model for an extended DPI engine

An example model of fully extended DPI engine can be seen in Figure 10-3, in which are depicted not only two new stages, but also multiple DPI meta-engines.

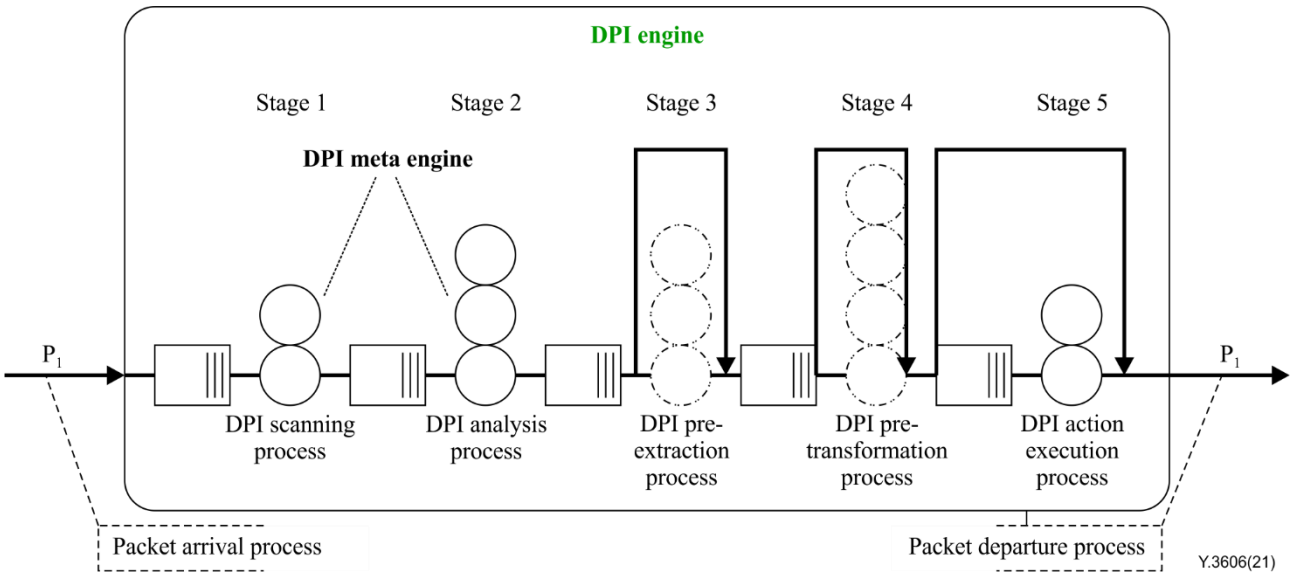


Figure 10-3 – Fully extended DPI engine model

10.1.6 DPI entity based on a DPI meta-engine

When a DPI function is realized based on a DPI meta-engine, the DPI entity will have better flexibility and high extendibility. In order to achieve expectable flexibility and extendibility, a control entity is needed to cooperate with the DPI entity.

Generally, a DPI entity based on DPI meta-engine and a DPI system based on DPI entities has the following features.

- The DPI system is composed of one or more DPI functional and one or more DPI control entities.
- Each DPI functional entity is composed of zero or more DPI-ScF, zero or more DPI-AnF, zero or more DPI-AcEF meta-engines and a policy rule base. The sum of the numbers of meta-engines of the DPI-ScF, DPI-AnF and DPI-AcEF types must be equal to or greater than 1.

- The DPI control entity is responsible for controlling the DPI functional entity and managing and scheduling DPI meta-engines through the DPI functional entity. Multiple DPI meta-engines execute their functions in order.
- The DPI control entity can collect information related to the DPI functional entities and all DPI meta-engines, and the DPI control entity can schedule the DPI meta-engines to realize corresponding functions according to input DPI requirements.
- Each DPI functional entity also has zero or more DPI-ExF and zero or more DPI-TrF meta-engines; the sum of the number of meta-engines of the DPI-ExF and DPI-TrF types can be zero or any value.
- In order to control the corresponding DPI functional entities, the control entity needs to collect basic information about them and all meta-engines in them. The basic information about the DPI functional entities includes identity and number of each type of meta-engine, while the information about meta-engines includes name, type, functional feature, whether the meta-engine is available and names of other meta-engines relevant to the meta-engine.
- When the control entity receives input from outside the DPI system, consisting of DPI functional requirements and information about all meta-engines, the control entity searches and identifies a group of meta-engines that can match the DPI functional requirements. The control entity then informs the corresponding DPI entities of which meta-engines need to be used to implement the DPI function. After getting the information from the control entity, the DPI entities configure those meta-engines to realize the DPI function.

10.1.7 Mechanism to implement DPI functions based on DPI meta-engine

The general mechanism to implement a DPI function based on a DPI meta-engine can be described as the following process.

1) The control entity should get the information related to the DPI functional entities. The information about the DPI functional entities includes identity and number of every type of meta-engine.
2) The control entity should get the information related to all DPI meta-engines. The information about meta-engines includes name, type, functional feature, whether the meta-engine is available and the names of other meta-engines relevant to them.
3) The control entity generates a directional graph that depicts the relations between the DPI meta-engines and a table that summarizes the information about them.
4) The control entity accepts the external DPI functional requirements.
5) The control entity analyses and concludes which DPI meta-engines are needed to implement the requirements.
6) The control entity configures DPI functional entities and DPI meta-engines to implement the requirements.

10.2 DPI application mode based on DPI meta-engine

10.2.1 Single DPI functional entity and multiple DPI meta-engines

The typical DPI mode for a single DPI functional entity realizes a certain DPI function. When the DPI functional entity is designed based on multiple DPI meta-engines, the DPI meta-engines in the DPI functional entity can be scheduled to work and cooperate in optimal status, the DPI function can then be implemented more efficiently.

Figure 10-4 depicts a scenario in which a single DPI functional entity with multiple DPI meta-engines implements the DPI function.

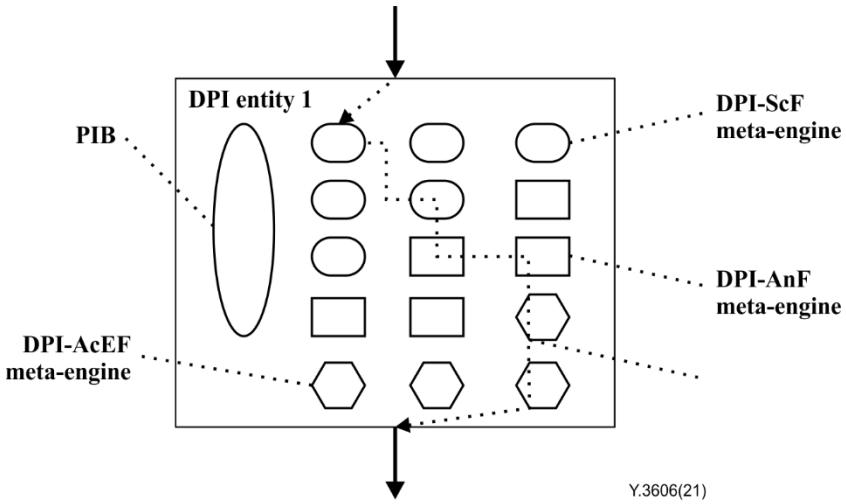


Figure 10-4 – Single DPI functional entity and multiple DPI meta-engine mode

10.2.2 Multiple DPI functional entity and multiple DPI meta-engine

Usually, a DPI function needs to rely on multiple DPI functional entities to cooperate. When DPI functional entities are designed based on multiple DPI meta-engines, both can be scheduled flexibly.

Figure 10-5 depicts a scenario in which multiple DPI functional entities with multiple DPI meta-engines implement the DPI function.

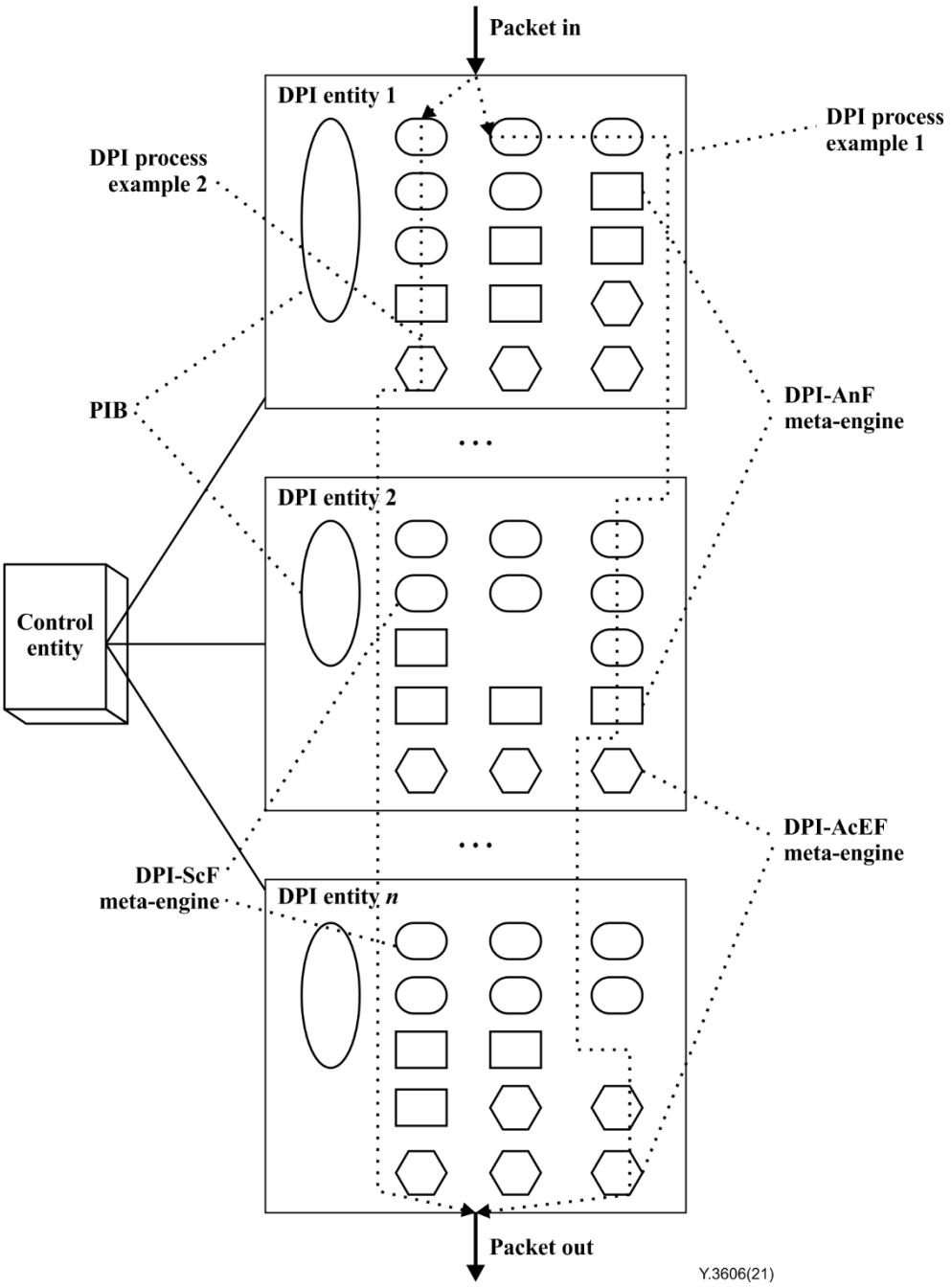


Figure 10-5 – Multiple DPI functional entity and multiple DPI meta-engine mode

10.3 Data pre-processing mechanism for structured data using DPI

Clause 10.1 specifies that the extended DPI engine includes the DPI-ExF and DPI-TrF, as well as generic DPI functions such as DPI-ScF, DPI-AnF and DPI-AcEF.

As to structured data, the functions of extraction and transformation can be carried out by coordination between DPI entities and big data processing entities (the entities that implement big data processing functions). Figure 10-6 depicts such a scenario.

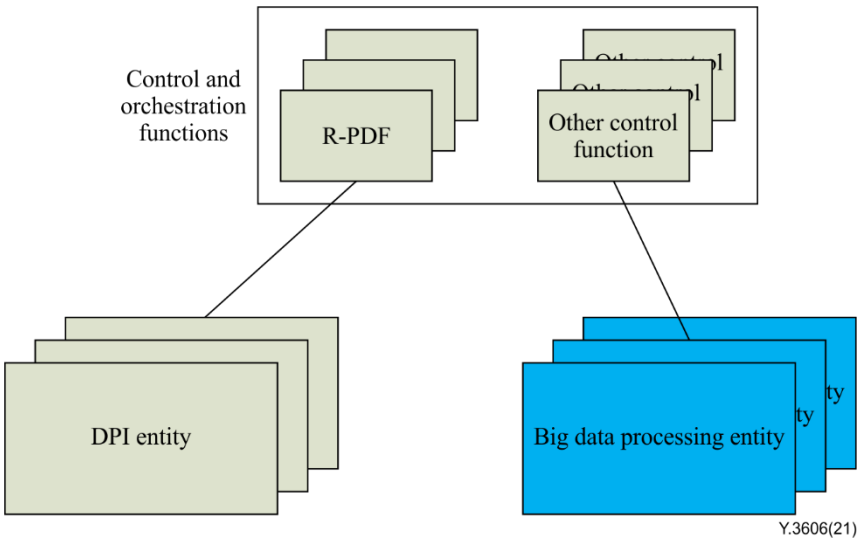


Figure 10-6 – Coordination for pre-processing function

Generally, DPI entities can realize the data DPI-ExF and data DPI-TrF, while DPI entities realize those functions based on configuration information from the remote policy decision function (R-PDF) (controller or management system, see [ITU-T Y.2770] and [ITU-T Y.2771]). Besides, DPI entities can realize functions such as checking and processing incorrect, incomplete and duplicated data.

It is a good solution for a DPI entity to realize the data DPI-ExF and the data DPI-TrF based on a DPI meta-engine. In addition, it would be more flexible if each DPI meta-engine is designed to implement a single simple data DPI-ExF or the data DPI-TrF.

10.4 Data pre-processing mechanism for non-structured data by DPI

Non-structured data is data of which the structure is not regular or integrated, where there is no pre-defined data model and which is difficult to represent in a two-dimensional table. Non-structured data includes line drawings, images, graphs or videos.

For non-structured data, the extraction and transformation functions can also be carried out by coordination between entities related to DPI and big data processing. However, because of the difference between structured and non-structured data, it is difficult for DPI entities to detect incorrect and incomplete data in the latter. Consequently, pre-processing for incorrect and incomplete data is recommended for implementation by big data processing entities.

11 Coordination processing mechanism of big -data DPI

11.1 General coordination processing mechanism

In the context of big data in networks, a DPI function is usually realized by a group of DPI components (e.g., DPI engine and DPI entity). Effective cooperation among DPI components is then important.

Such DPI components can be divided into those that are homogeneous and heterogeneous. If two independent DPI components can accept inputs of the same kind, while their outputs are identical when they have the same inputs, then they are termed homogeneous. Otherwise, they are termed heterogeneous. Figure 11-1 depicts an example involving homogeneous and heterogeneous DPI components: a) DPI components A and D, as well as B and C are homogeneous; b) DPI components A and B or C, as well as B and A or D are heterogeneous.

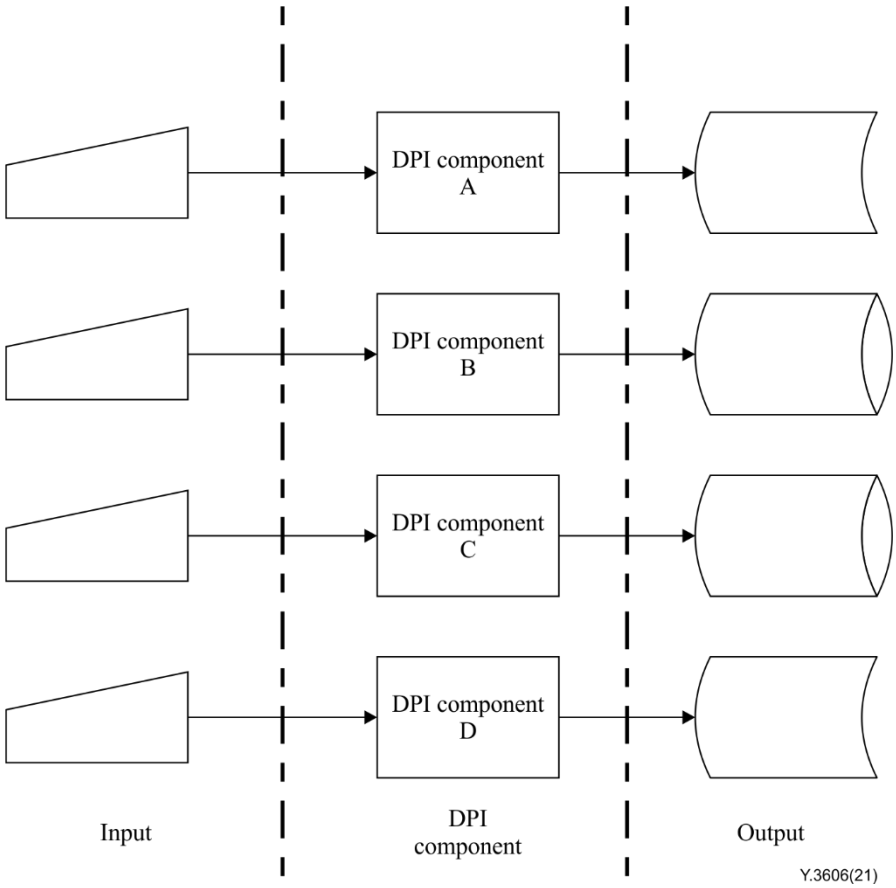


Figure 11-1 – An example about homogeneous and heterogeneous DPI components

Generally, if a group of DPI components cooperate to carry out a certain DPI function, there are three kinds of coordination processing mode: parallel; serial; and hybrid.

If all DPI components operate:

a) simultaneously and independently, that is called parallel mode;

b) in sequence and the output of the preceding DPI component is the input of the succeeding DPI component, that is called serial mode;

c) in parallel mode while others operate in serial mode, that is called hybrid mode.

Figure 11-2 illustrates these three modes. Figure 11-2-a depicts the parallel mode for a group of DPI components, Figure 11-2-b the serial mode and Figure 11-2-c gives an example of hybrid mode. In Figure 11-2-c, DPI components B, C and D work in parallel mode (denoted the DPI parallel components group (BCD)), DPI components E and F work in parallel mode (denoted the DPI parallel components group (EF)). Meanwhile, DPI component A, the DPI parallel components group (BCD) and DPI parallel components group (EF) work in serial mode.

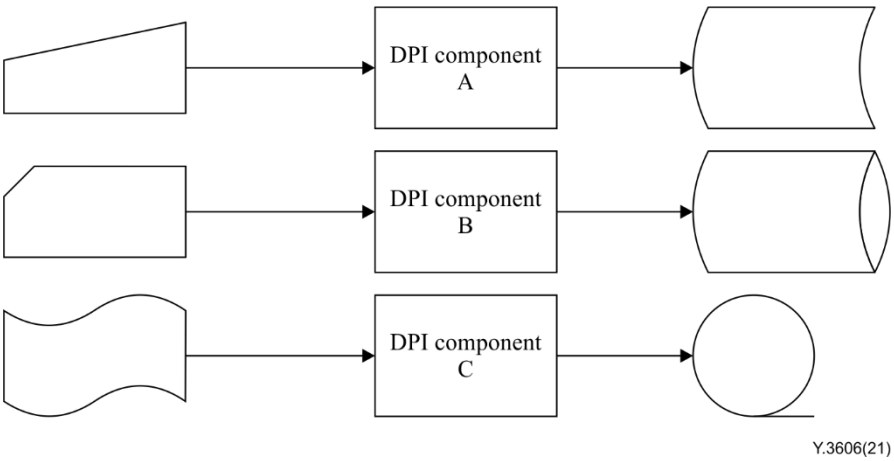


Figure 11-2-a – Depiction of parallel mode

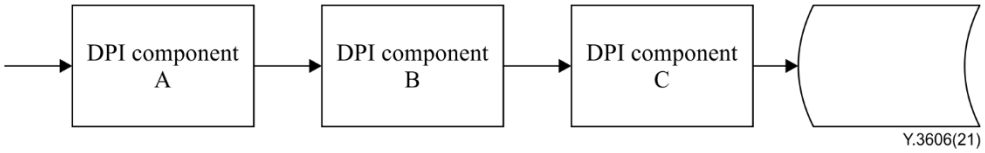


Figure 11-2-b – Depiction of serial mode

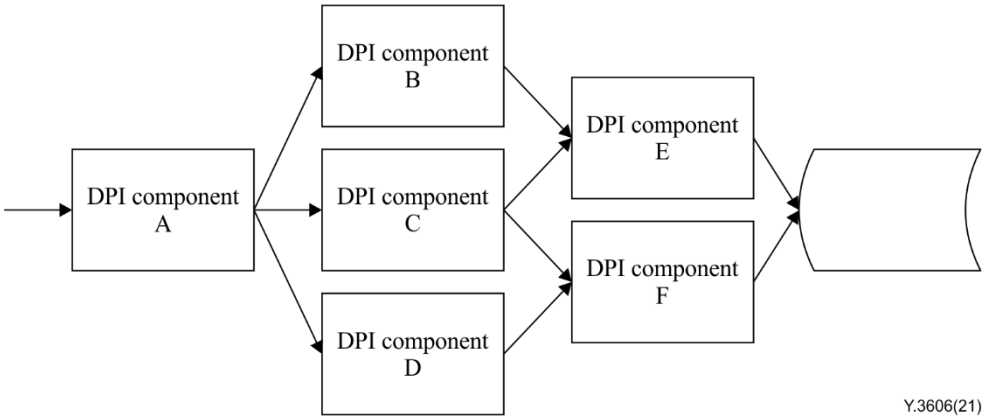


Figure 11-2-c – Depiction of hybrid mode

Figure 11-2 demonstrates that homogeneous DPI components are appropriate for parallel mode and two or more sequential DPI components are usually heterogeneous.

In serial mode or the serial part of the hybrid mode, note that the output of the preceding DPI component(s) is/are the input of the succeeding DPI component(s). Therefore, a synchronization mechanism is necessary between the neighbouring DPI components in order to avoid congestion and data loss.

In parallel mode or the parallel part of the hybrid mode, parallel DPI components usually have the same input source. A traffic dispatch and balance mechanism is then usually needed between the input source DPI component and succeeding parallel ones.

In hybrid mode, some DPI components always work in serial mode, but there is great difference between the output of the preceding DPI component and the processing capability of the succeeding one. This circumstance causes severe waste of the capability of some DPI components, even if the synchronization mechanism is used. An appropriate solution is to ensure that the number of preceding DPI components and the number of the succeeding ones are different; these numbers can

be dynamically adjusted. A scheduling mechanism is appropriate in order to realize such dynamic adjustment. Figure 11-3 illustrates this problem and solution in detail.

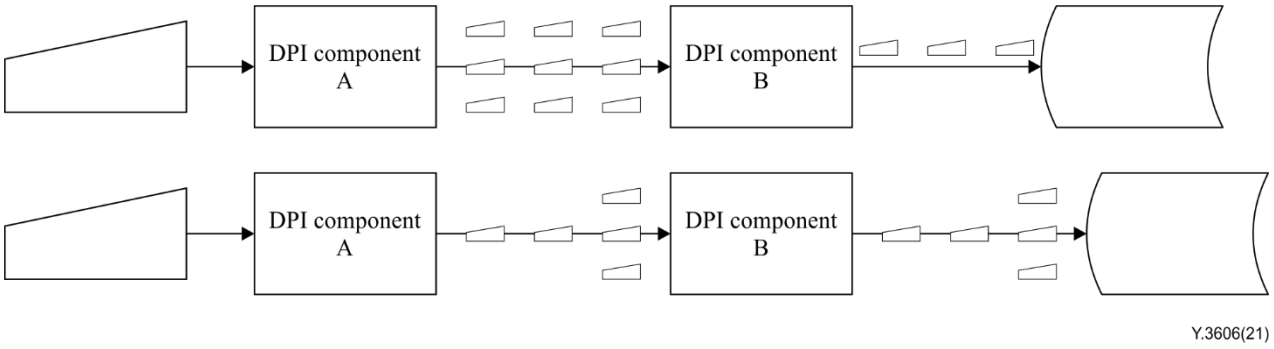
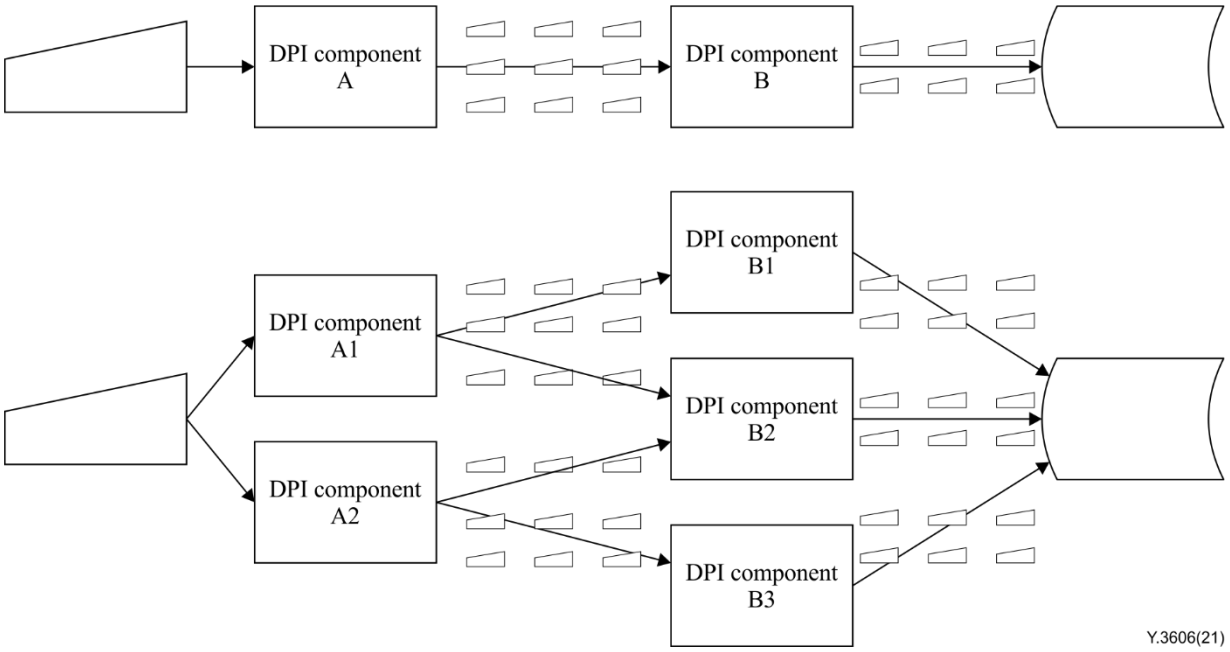


Figure 11-3-a – Problem that neighbouring DPI components have different capability



b

Figure 11-3-b – Solution for the problem that neighbouring DPI components have different capability

Figure 11-3-a shows the problem produced by a capability difference between DPI components in sequence. The upper half of Figure 11-3-a depicts the situation in which the preceding DPI component has twice the output of that which can be processed by the succeeding one. Meanwhile the lower half of Figure 11-3-a depicts a scene in which the preceding DPI component often has less output than that which can be processed by the succeeding one. In these scenarios, the capability of either the preceding or the succeeding DPI component is wasted.

Figure 11-3-b shows the solution corresponding to the problem caused by the capability difference between DPI components in sequence. The upper half of Figure 11-3-b depicts the situation in which the preceding DPI component has half the output of that which can be processed by the succeeding one. The lower half of Figure 11-3-b gives the solution by using two preceding DPI components and three succeeding ones.

11.2 Coordination processing mechanism for multiple DPI engines

11.2.1 General structure of multiple DPI engines within a DPI entity

The general structure of multiple DPI engines within a DPI entity is depicted in Figure 11-4. One typical feature of the above structure is that two or more DPI engines are designed within a DPI node and the above DPI engines share an identical DPI-PIB.

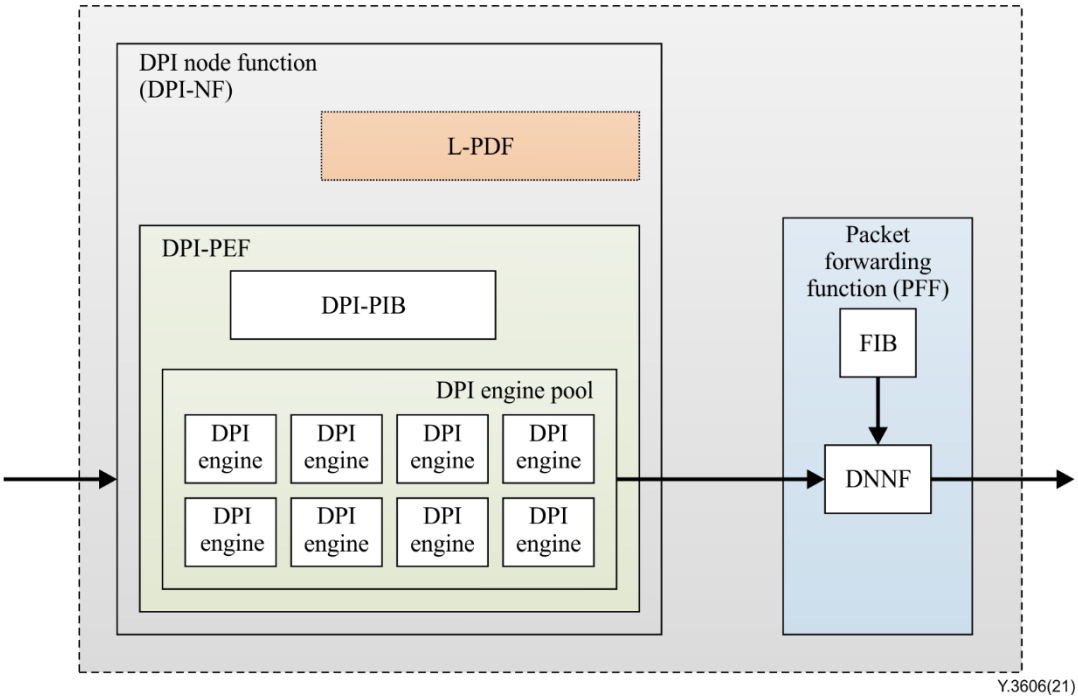


Figure 11-4 – General structure of multiple DPI engines within a DPI entity

11.2.2 Coordination process mechanism for multiple DPI engines within a DPI entity

The coordination process mechanism of multiple DPI engines within a DPI entity is depicted in Figure 11-5. Because all DPI engines within a DPI node use the same DPI-PIB, a PIB access controller is designed to modulate the access from DPI engines. In addition, a DPI engine scheduler is designed to configure the work mode (serial, parallel or hybrid) for the DPI engines. This work mode configuration needs support by physical resources (e.g., if DPI engine A and B are configured to work in parallel mode, they should be homogeneous DPI components).

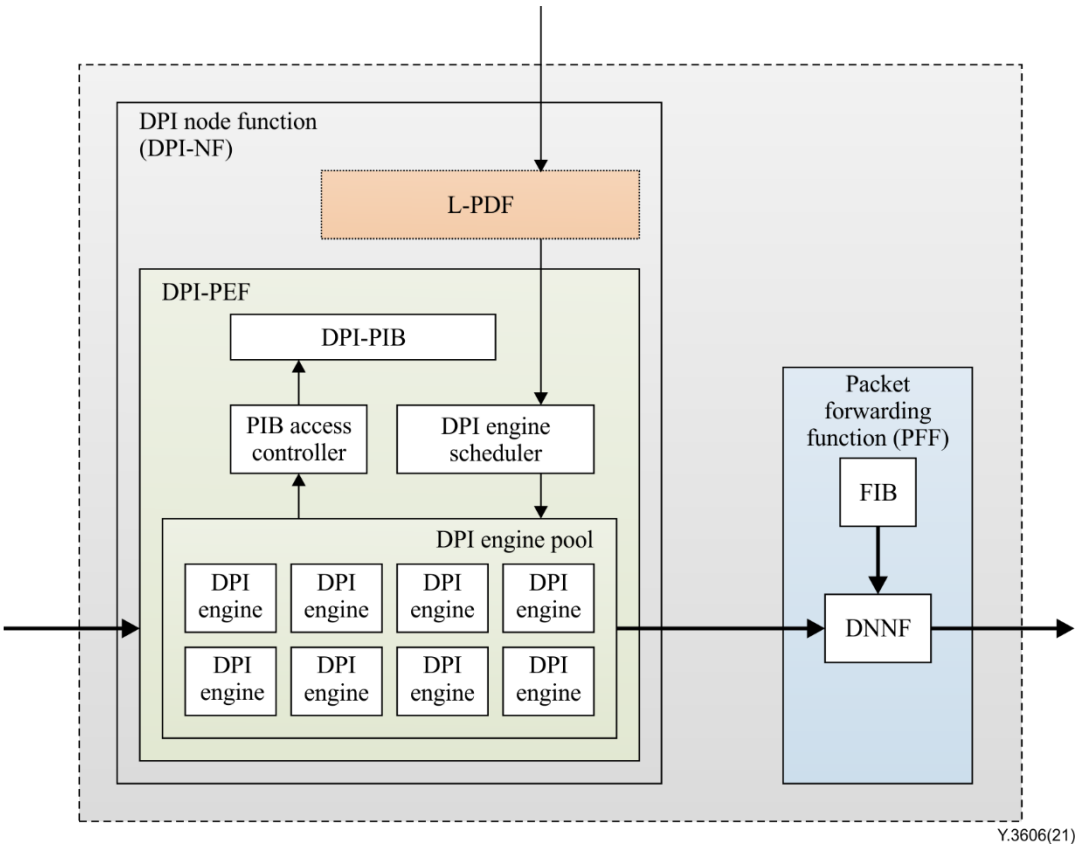


Figure 11-5 – Coordination mechanism of multiple DPI engines within a DPI entity

11.3 Coordination processing mechanism for multiple DPI nodes

The coordination process mechanism for multiple DPI nodes is depicted in Figure 11-6. Note that the DPI nodes are not deployed in the same physical location (Figure 11-6 just presents a logical view in which all available DPI nodes are organized to form a DPI node cluster).

Coordination functions include topology analysis, orchestration, scheduling and remote PDF, and can be designed in the controller or element management system/network management system (EMS/NMS). It is also feasible to design some coordination functions in the controller and others in the EMS/NMS.

According to clause 11.1, there are three coordination modes among DPI components: parallel, serial and hybrid. The precondition for two or more DPI nodes to coordinate in parallel mode is that they have identical preceding network nodes. The precondition for two or more DPI nodes to coordinate in serial mode is that one of them precedes the other. Topology analysis is therefore necessary to decide which coordination mode is appropriate.

In addition, the capability of various DPI nodes, on which the suitability of the coordination mode also depends, is different. Capability analysis is therefore also a necessary function.

Resource allocation and sharing are also necessary among DPI nodes to coordinate to provide better functions. The orchestration function is then useful for DPI node coordination.

In order to carry out DPI functions, it is not necessary for all available DPI nodes to participate. The schedule function is then deployed to arrange appropriate coordination of DPI nodes to carry out a given mission.

Coordination functions also need the R-PDF when the DPI-PIB is to be shared among two or more independent DPI nodes.

The coordination processing mechanism for multiple DPI nodes described in this clause is not mandatory but optional to DPI implementations.

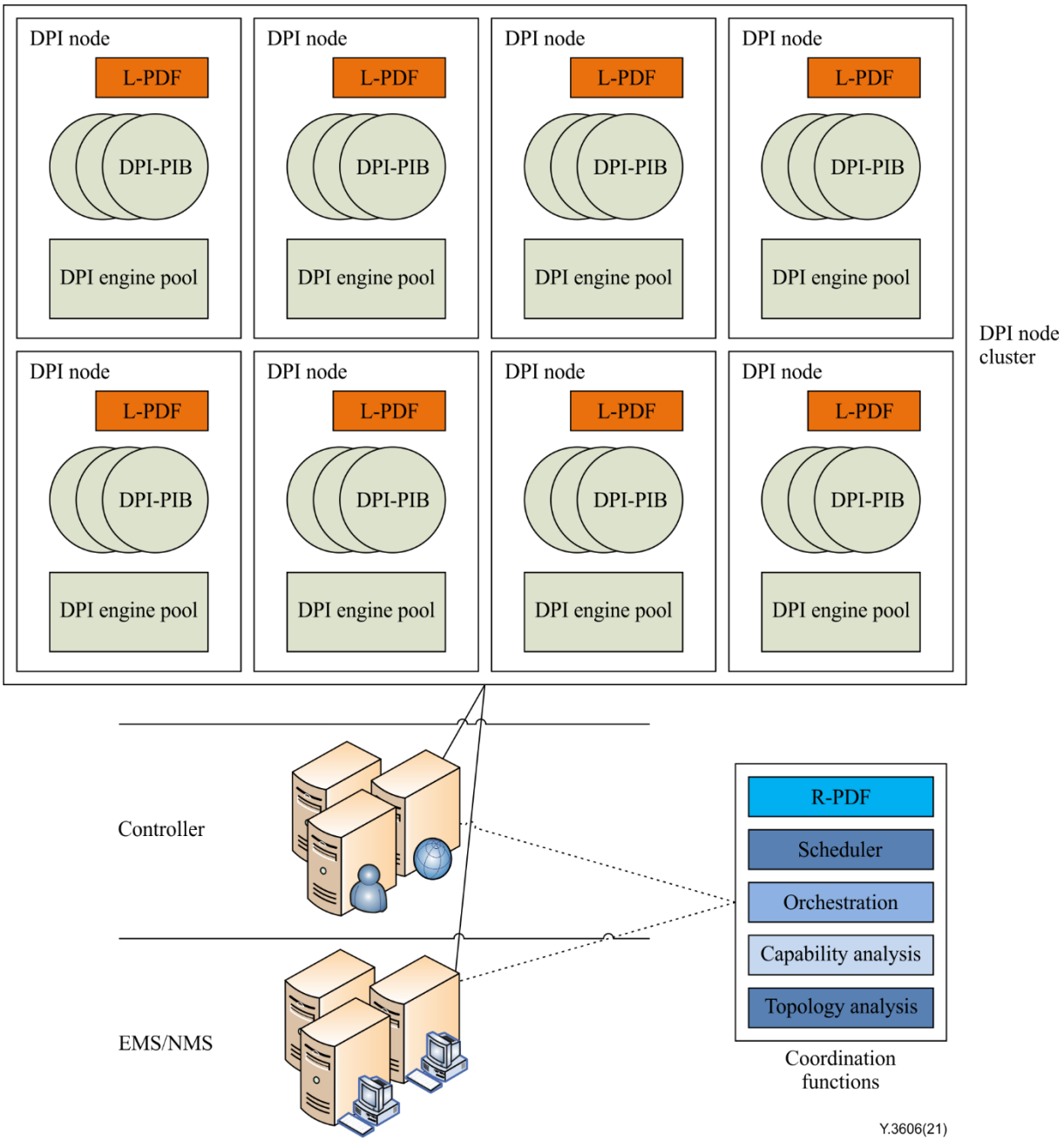


Figure 11-6 – Coordination mechanism of multiple DPI nodes

12 Interfaces between deep packet inspection and the upper-layer big data-related method

DPI is an infrastructure technology for big data-related methods. DPI is therefore a lower-layer technology for big data. Figure 12-1 depicts two interfaces between the DPI and big data layers as follows:

- U_u: the upstream interface is responsible for handing the data from the DPI to the big data layer;
- U_d: the downstream interface is responsible for handing the rule from the big data to the DPI layer.

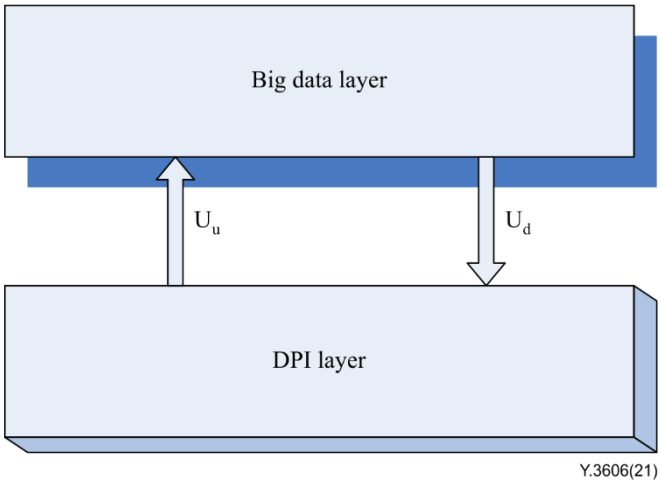


Figure 12-1 – Interfaces between the DPI and big data layers

12.1 Downstream interface

The U_d is used to send some rules to DPI entities in order that the DPI entities can take actions according to the rule and collect appropriate data for the big data layer. As an interface, data exchange is the core function of U_d. In order to implement the data exchange function, the protocol used by the interface and data carried at the interface should be clearly specified.

There are a lot of candidate protocols that can be adopted by U_d, such as the raw user datagram protocol (UDP), hypertext transfer protocol (HTTP) and simple network management protocol (SNMP). The criteria for the choice of a protocol for U_d are high efficiency, ease of realization and comprehensibility by the entities bordering the interface.

Data carriage via the U_d is an action that the big data layer requires the DPI entities to take. There are two kinds of action related to the rules, as follows:

- which kind of data the DPI entities should pick up;
- how the DPI entities submit the data to the big data layer.

In order to specify which data DPI entities should pick up, a rule similar to DPI conditions (see Figure A.1 of [ITU-T Y.2770]) can be used. Figure 12-2 also describes examples of such rules.

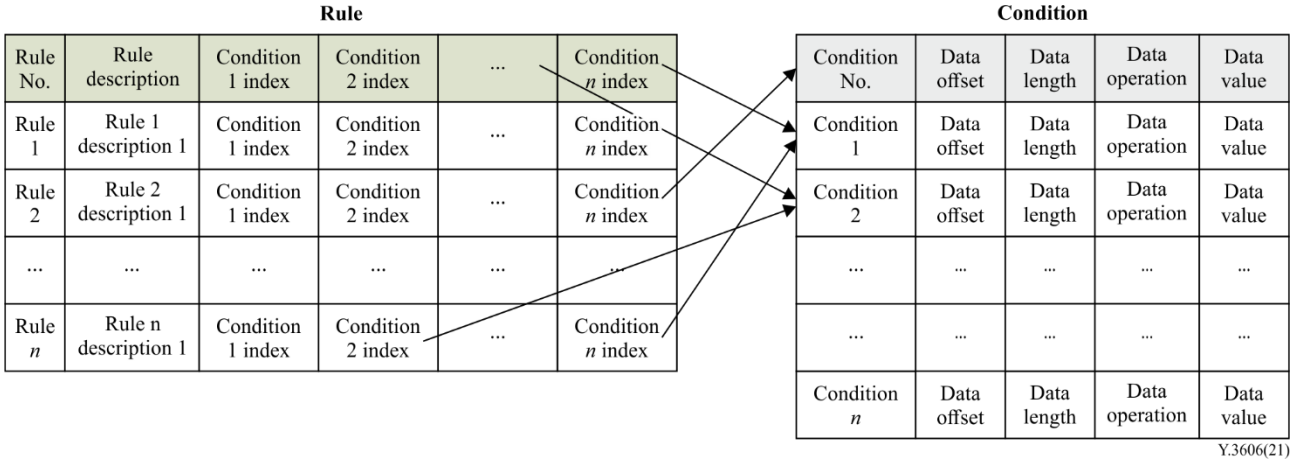


Figure 12-2 – Rules and conditions for DPI

When DPI entities submit data to the big data layer, it should at least communicate the data format, such as extensible markup language (XML) and yet another next generation (YANG), to the DPI

entities. This is a better solution than the big data and DPI layers negotiating to determine the data forma.

12.2 Upstream interface

The U_u is used to send data to the big data layer and to facilitate date use by it. Similarly, the U_u should clearly specify the protocol used by the interface and the data format.

Generally, U_u can use any protocols to exchange data. However, considering that this interface is mainly targeted at the transport of large-volume data, high efficiency and low overhead are two recommended features, so raw UDP is a good option.

Data format should be specified by the big data layer and configured through the U_d.

13 Other aspects of the DPI mechanism for big data in networks

13.1 Manageability

Note that the work mode of DPI components in the big data context is very different from the traditional one. So, new management aspects need taking into account.

First, new management objects, such as DPI-ExFs, DPI-TrFs and new interfaces, should be added to the management plane.

Second, because of the various coordination modes of a group of DPI components, the management plane needs to be aware of the current coordination mode of the DPI components and have the capability to configure the coordination mode.

Third, because of the requirements to configure the appropriate coordination mode, the management plane shall be aware of the resource status of all DPI components that need to work in that mode.

13.2 Applicability

When DPI functions are deployed in the big data context, because of the diversity of DPI components, some other factors should be taken into account:

If information about DPI entities is stored in distributed network nodes, then those nodes should have similar storage access capability in case a slow network node worsens the performance of the whole DPI system.

If DPI components work in coordination mode, the adaptability of process capability of the DPI components should be ensured. For example, if two or more components work in cascaded mode, then the succeeding DPI component should adapt the preceding DPI component in case some valuable information is lost.

When some functions are implemented by coordination between DPI entities and an external big data processing system, the coordination mode and adaptability should be taken into account.

13.3 Availability

In the big data context, a DPI function is usually implemented by a number of DPI components and relies on cooperation among different DPI components. In such circumstances, although reliability of the DPI components is very important, some further measures should be taken to improve the availability of entire DPI systems.

When DPI components work in cascaded mode, a protection mechanism is needed to prevent a failed DPI component from making the DPI system out of order.

When DPI components work in distributed mode, the functions allocated to a failed one should transfer to others that belong to the same distributed group. When DPI information is stored in

distributed network nodes, if one of the network nodes fails, the information stored in the network node should be recovered and made available by using other appropriate methods. For example, DPI components can provisionally retrieve information from the management plane until the failed network node is restored to normal status.

14 Performance consideration

When DPI function entities are used to implement data collection functions for big data, they should have the appropriate processing capability in order to ensure that data loss does not cross the threshold specified by network service providers.

When DPI function entities are used to implement data pre-processing functions for big data, the DPI function entities should have a processing capability that matches the succeeding data processing functions of big data.

When used in the big data context, DPI functional entities should have the capability to ensure that all performance indexes described in [ITU-T Y.2773] do not worsen.

15 Security considerations

This Recommendation has the same security requirements as [ITU-T Y.2770].

In addition, according to [ITU-T Y.2770], the DPI-FE and the information pertaining to DPI operations should be under protection against threats. It shall be guaranteed that information exchange between the R-PDF and L-PDF can occur safely.

If information is processed by several parallel processing entities, it is necessary to ensure that information is integrated.

If information is processed by a series of processing entities in order, it is necessary to ensure information security when information crosses those processing entities.

The regulation, privacy and security application aspects of DPI lie outside the scope of this Recommendation. Vendors, operators and service providers are required to take into account national regulatory and policy requirements when implementing this Recommendation.

Bibliography

[b-ITU-T Y.2772] Recommendation ITU-T Y.2772 (2016), *Mechanisms for the network elements with support of deep packet inspection.*

[b-IETF RFC 5101] IETF RFC 5101 (2008), *Specification of the IP flow information export (IPFIX) protocol for the exchange of IP traffic flow information*.

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, 2022