International Telecommunication Union

ITU-T

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

Y.4472

(08/2020)

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

Internet of things and smart cities and communities – Frameworks, architectures and protocols

Open data application programming interfaces (APIs) for IoT data in smart cities and communities

Recommendation ITU-T Y.4472

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.4472

Open data application programming interfaces (APIs) for IoT data in smart cities and communities

Summary

A growing number of smart cities and administrations are inclined to collaborate and mutualize their efforts and resources for IoT deployments and open data sharing. This Recommendation studies the concept and potential of developing a secure open and interoperable API in the context of IoT deployment and open data management in smart cities. It analyses current solutions implemented by administrations around the world, where applicable, including those adopted by smart cities, to share their data through open and interoperable interfaces. It subsequently specifies an open and interoperable API for secure open data architecture, as well as for supporting IoT data interoperability for smart cities.

This Recommendation presents a complete set of Open APIs dedicated to smart cities offering different features covering the needs of interoperable smart city framework development. In order to achieve interoperability between heterogeneous platforms and the development of smart cities, the Recommendation has proposed "interoperability points" in southbound and northbound interfaces in a smart city framework.

It provides a list of core API sets focusing on data interoperability, including context data management APIs, data transactions APIs, data storage APIs and security APIs. Through the mechanism of subscriptions, it is possible to get a performant and scalable context data managment. The data storage APIs allow a granular management of the saved data for all cases, in particular both for open data and private data. The data transaction APIs facilitate exposure and access to the data through a data marketplace. In addition, security and privacy APIs are seriously taken into account to provide secure data exchange.

It should be noted that data interoperability with open APIs can be completed by using common data models, which is briefly discussed. Common data models built upon the collaboration with several standard fora and European projects are open for public use.

The development of an interoperable framework makes smart city platforms cost efficient, flexible and extendable. Interoperability is not a choice but a must in smart city systems that embed multiple verticals.

History

Edition	Recommendation	Approval	Study Group	Unique ID*
1.0	ITU-T Y.4472	2020-08-29	20	11.1002/1000/14374

Keywords

API, interoperable, IoT, open data.

---

* 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.4472

Open data application programming interfaces (APIs) for IoT data in smart cities and communities

1 Scope

This Recommendation proposes open and interoperable APIs for secure open data exchange, as appropriate, as well as for IoT data interoperability for smart cities.

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.

None.

3 Definitions

3.1 Terms defined elsewhere

This Recommendation uses the following terms defined elsewhere:

3.1.1 application programming interface (API) [b-ITU-T T.170]: A boundary across which a software application uses facilities of programming languages to invoke software services. These facilities may include procedures or operations, shared data objects and resolution of identifiers.

3.1.2 data marketplace [b-FG-DPM TS D0.1]: An electronic marketplace whose main product is provisioning of data and/or related services around data.

3.1.3 data model [b-ITU-T J.380.8]: A data model is a formal view of the data items contained in an information store to which an information service implementing this standard will provide access and is specified for purposes of formulating and executing queries against the information store's data.

3.1.4 interoperability [b-ITU-T Y.101]: The ability of two or more systems or applications to exchange information and to mutually use the information that has been exchanged.

3.1.5 Internet of things (IoT) [b-ITU-T Y.4000]: A global infrastructure for the information society, enabling advanced services by interconnecting (physical and virtual) things based on existing and evolving interoperable information and communication technologies.

NOTE 1 – Through the exploitation of identification, data capture, processing and communication capabilities, the IoT makes full use of things to offer services to all kinds of applications, whilst ensuring that security and privacy requirements are fulfilled.

NOTE 2 – From a broader perspective, the IoT can be perceived as a vision with technological and societal implications.

3.2 Terms defined in this Recommendation

This Recommendation defines the following terms:

3.2.1 common data models: Common data models are schemas representing the format of the data sets. They are interoperability points between the entities exchanging data through the common data models.

3.2.2 context data (entity): Structured data that contains status information of context entities and their related attributes and metadata. A context entity can represent everything in the real world such as users, places and devices that can be abstracted and represented using a predefined data model.

3.2.3 data monetization: Generation of economic benefits from different data sources.

3.2.4 northbound interfaces: Interfaces permitting a given component to communicate with other components located at a higher level.

3.2.5 open data: Data that can be publicly accessible to all through open standards and protocols or through other means. The use and redistribution of open data can be subject to rules.

3.2.6 smart city platform: A city platform that offers direct integration of city platforms and systems, or through open interfaces between city platforms and third parties, in order to offer urban operation and services supporting the functioning of city services, as well as efficiency, performance, security and scalability.

3.2.7 southbound interfaces: Interfaces permitting a given component to communicate with other components located at a lower level.

4 Abbreviations and acronyms

This Recommendation uses the following abbreviations and acronyms:

API	Application Programming Interface
ENISA	European Union Agency for Network and Information Security
GSMA	GSM Association
IoT	Internet of Things
JSON	JavaScript Object Notation
JSON-LD	JavaScript Object Notation for Linked Data
M2M	Machine to Machine
OASC	Open & Agile Smart Cities
RDF	Resource Description Framework
TMF	TM Forum
XACML	extensible Access Control Markup Language

5 Conventions

None.

6 General considerations about open APIs for IoT data in smart cities and communities

Smart cities are a key driver of large-scale IoT markets where a complex set of diverse technologies, consisting of heterogeneous IoT devices, network infrastructure, cloud systems and multiple IoT platforms, go together to provide a diverse smart city and application. The scale of the smart city use cases varies from a single local network-based system to a large-scaled cross-platform deployment. Various communication technologies are used to support the use case requirements and may include diverse IoT protocols, security and privacy, data analytics and management, service orchestration, business management, etc.

Considering the heterogeneity that a smart city embeds, interoperability is a pivotal index to consider from the design phase. Yet, there are large sets of interoperability solutions that can be applied in different layers of the components, systems, platforms, networks and services. The challenges become greater when it comes to a federation of the smart city systems, while the needs grow.

Instead of adding another complexity to thousands of existing solutions on interoperability of smart cities and further federation of two or more smart city platforms, this Recommendation is focusing on "data". The communication network protocols or protocols for platforms depend on the city infrastructure, budget, engineering, environment, etc. The common goal of the different smart cities is eventually the interoperability of the generated and obtained data for creating services. The need of open application programming interfaces (Open APIs) for IoT data lies on this point. The IoT data includes open data and also private data.

In order to ensure data interoperability between smart city platforms with minimal costs, "interoperability points" are introduced below. The core open APIs are defined to connect the interoperability points.

Interoperability points represent the main technical interfaces between smart cities' platform and external systems. Interoperability points are also a way to access basic smart city IoT functionalities and in particular to consume and provide data. They assure the replicability of solutions (i.e., services, applications) on different smart cities that are compliant to them, and they are completely decoupled from the specific technological implementation and deployment of the architectural components. Interoperability points are the logical and conceptual representation of a set of open APIs that can be concretely instantiated to provide a technologically-specific implementation.

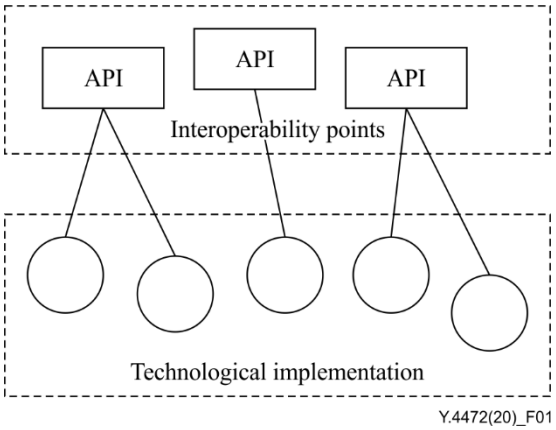

Figure 1 – Interoperability points

In general, there are two interoperability points in a generic smart city platform:

- **Southbound interfaces**: represent the main way for interacting with IoT devices/middleware and managing relevant IoT data. They include a set of interfaces used to connect a smart city platform to heterogeneous IoT devices and middleware. The southbound interfaces are intended to be exchanging IoT data with a smart city platform, hiding the complexity of the IoT protocols and communication issues, which are not covered by this Recommendation.
- **Northbound interfaces**: include a set of interfaces that provide IoT data and its elaboration to the external system and application interacting with the smart city platform. The northbound interfaces described in this Recommendation are not all those possible to be provided by a generic smart city platform, but only the basic ones which constitute an interoperability layer for IoT data provisioning.

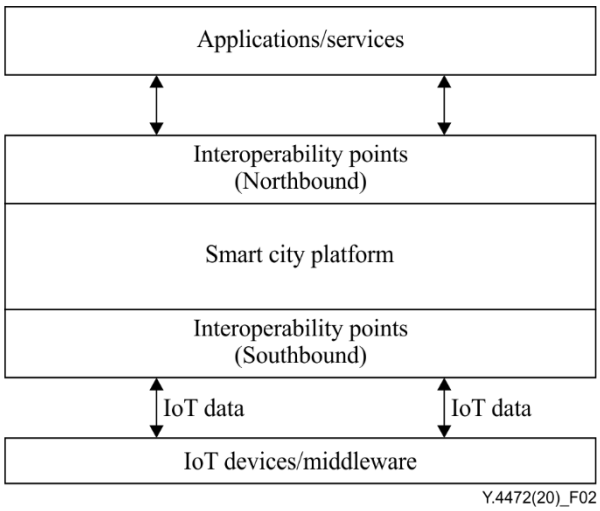

Figure 2 – Northbound and southbound interfaces

7 Requirements on open APIs for IoT data

The adoption of open APIs in smart cities and communities is a key enabler for the digital transformation of such ecosystems. To address this, stakeholders agree on a set of common expectations that allows the shaping of the following high-level requirements to be met by any open API in the IoT and smart city domain. The stakeholders are firstly the citizens living in smart cities, the authorities, the third-party service providers and any organization involved in the management of smart cities.

Context awareness and management: Context information is intended as structured data that contains status information about entities in the real world (e.g., sensor information in a smart city). The APIs have to provide support for (real-time) context management, allowing for instance to publish, consume and subscribe to context information.

Enable semantic interoperability: Semantic interoperability is enabled through the support, availability and future extension of data models, ontologies and controlled vocabularies to cover different needs of heterogeneous applicative domains.

Enabling the deployment of the corresponding data marketplace: The deployment fosters data transactions and monetization.

Extensibility: The API should be extensible to fulfil new functionalities not yet defined.

Interoperability between heterogeneous technologies feeding information to the data repository: The API has to be designed to simplify the interoperability between existing smart city and IoT platforms.

Openness: The APIs have to be accessible without (or with very limited) restriction by external users (e.g., developers) in order to foster their adoption and the reuse of data.

Open standards: The APIs have to be based on open standards that are publicly available and, preferably, promoted by international organizations. This allows to avoid the technological fragmentation of the API specification while supporting the no-vendor lock-in principle.

Privacy by design: The APIs should comply with privacy by design and personal data minimization principles. It should ensure compliance with applicable data protection regulations and preserve data subject rights.

Reliable and resilient accessibility: The APIs need to consume the exposed contextual data whilst guaranteeing that data is consistent in all its dimensions (time and content).

Security by design and easy integration with encryption, authentication and authorization frameworks: The APIs have to comply with the security by design principle and the most relevant standard protocols and techniques to address critical security, integrity and privacy issues. Access to the different APIs mentioned in this Recommendation is ensured by HTTPS transmissions with valid certificates.

Support for machine-readable data: The APIs are, by definition, based on a machine-readable format, but it is also necessary that it supports the exchange of machine-readable data and metadata formats (e.g., JSON, XML). Therefore, it will enable data processing by third-party applications and systems.

Semantic web support: The APIs have to be designed to satisfy the need for semantic data, including support for semantic languages (e.g., RDF), linked-data data formats (e.g., JSON-LD), and specific query languages.

Support for data-intensive and real time: The IoT and smart city domain can be characterized by the intensive (near) real-time data production that requires the use of specific technologies to access data under strict real-time and performance constraints while preserving its accuracy. The APIs have to support these types of technologies.

8 Interoperability points

The APIs listed in Table 1 are identified as interoperability points.

Table 1 – Interoperability points

Interoperability points	Description	Southbound/Northbound
Context Data Management API	Provide interfaces to manage IoT context data information	Southbound/Northbound
Data Storage API	Provide interfaces to access, in a uniform way, different private and open data	Northbound
IoT Data Transaction Management API	Provide interfaces to manage IoT data lifecycle and monetization process	Northbound
Security API	Provide interfaces to manage security primitives such as authorization and authentication	Southbound/Northbound
Common data models	A set of common data models that enable semantic interoperability in IoT data exchange	Southbound/Northbound

8.1 Context Data Management API

The Context Data Management API provides a standard way to manage and publish context information which comes from IoT devices and middleware. Context data represents information about real-world entities provided by several data sources such as IoT devices.

More specifically, the Context Data Management API consists of the set of APIs described in clauses 8.1.1 to 8.1.3:

- **Manage Context API**: provides methods for the creation, modification, and deletion of context entities. Input entities are issued with their attributes, in either a normalized way, by issuing structures, attributes, values and related data types, or in a compact way, by issuing

attributes directly as key-value pairs. In addition, it will be possible either to update or remove one or more entities as well as only specific attributes from each entity.

- **Query Context API**: provides methods for the discovery and retrieval of entities in the context data management component. Filters can be applied to refine the queries and only access entities that match specific metadata (e.g., id, entity type, etc.) or attributes' values. A query could be composed of a list of statements, each of them expressing a matching condition, and return all the entities that match all the matching conditions.
- **Content Subscription API**: provides methods to manage the subscriptions to asynchronous notification events about entity updates. It allows subscription to specific context entities, so when any update occurs on them, the subscriber application/service receives an asynchronous notification.

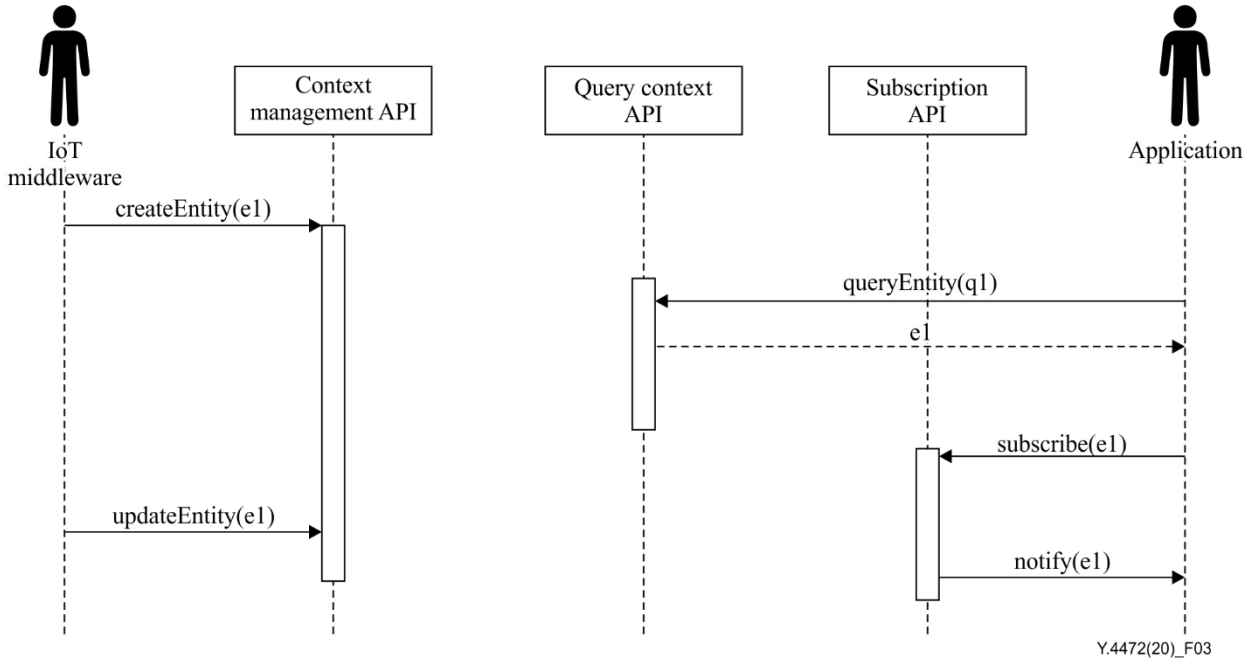

Figure 3 – Sequence diagram showing the interactions with the Context Data Management API

8.1.1 Manage Context API

See Table 2.

Table 2 – Manage Context API

Interface	Operation	Parameters
createEntity	*allow to create new context data entity*	*entity*
deleteEntity	*delete an existing entity*	*entityId*
updateEntity	*update/delete entity attributes*	*entityId, attributes*

8.1.2 Query Context API

See Table 3.

Table 3 – Query Context API

Interface	Operation	Parameters
getEntity	*retrieve a specific entity*	*entityId*
queryEntity	*retrieve entities which matches a specific query*	*entityId, filter*

8.1.3 Subscription API

See Table 4.

Table 4 – Subscription API

Interface	Operation	Parameters
subscribeEntity	*create a subscription to receive context updates*	*subscription, entityId*
notify	*send a notification*	*notification*

8.2 Data Storage API

The Data Storage API provides a standard way to access the data storage management component which allows for storing historical raw and aggregated time series information about the evolution of context data.

The data storage management component provides the functionalities that allow access to different storage technologies (e.g., different databases or cloud repositories) and exposes a uniform data storage interface. Data managed by this component can be classified into two types:

- Public or open data: provided without any restriction or with an open license that explicitly defines the rights to access, use and sharing the data.
- Private data: provided with strict restrictions (e.g., allow only access and use for specific purposes, sharing is typically not allowed); this category may include personal data, telecommunications data and energy supplier data.

Moreover, data can be distinguished between:

- actual data: with current values;
- historical data: including historical data over a period of time.

To access these two types of data, the data storage interface often provides two different kinds of APIs:

- **Private Data set API**: enables authorized users to access data through provided methods to retrieve both historical raw and aggregated time series information. These series can be provided according to specific temporal aggregation methods, such as mean, max, min values, etc.
- **Open Dataset API**: provides a unique access point to search and discover open data sets, including historical data where applicable, which come from different open data management systems (ODMS) portals.

These two APIs are described in clauses 8.2.1 and 8.2.2.

8.2.1 Private Data set API

Data sets are *private.* Table 5 describes the Private Data Set API.

Table 5 – Private Data Set API

Interface	Operation	Parameters

search	Get the results of a search.
	
	For example, a search with the query "q=open+street+map" returns these results:
	
	{
	"count": 5,
	"results": ["open-street-map-sample-file", "transport-the-18th-century-cassini-roads-and-cities-dataset", "transport-exports-by-mode-of-transport-1966", "transport-traffic-commissioners-local-bus-service-registration", "food-hygiene-rating-scheme-camden"]
	}
	
	[b-CKAN]
		The query composed by words separated by the "+" character
		Respond "403 Unauthenticated" to inform that this is a private data set.

authenticate	Post the credentials of the user.	User's credentials.

In most cases, authentication is needed to access private data sets; however, data owners may give access to anyone without any technical barriers or identity control.

Authentication is handled typically by API Proxy using OAuth standards. Generally, OAuth in its actual version 2.0 provides clients with a "secure delegated access" to server resources on behalf of a resource owner.

In the standard case, the server exchanges the authorization code for an access token by making a POST request to the authorization server's token end point: grant_type=authorization_code.

8.2.2 Open data set API

Data sets are *public* [b-CKAN]. Table 6 describes the Open data set API.

Table 6 – Open data set API

Interface	Operation	Parameters
Search	Get the results of a search. For example, a search with the query "q=open+street+map" returns these results: { "count": 5, "results": ["open-street-map-sample-file", "transport-the-18th-century-cassini-roads-and-cities-dataset",	The query composed by words separated by the "+" character

Table 6 – Open data set API

Interface	Operation	Parameters
	"transport-exports-by-mode-of-transport-1966", "transport-traffic-commissioners-local-bus-service-registration", "food-hygiene-rating-scheme-camden"] } [b-CKAN]	

Get	Get the data from a specific data set.
	
	For example, the results of the operation are:
	
	{
	"count": 32,
	"display_name": "test-date",
	"name": "test-date"
	}
	
	[b-CKAN]
		The name of the data set

Update	Update the data of a specific data set. For example, if a modification is done through this operation, the results are:
	{
	"count": 32,
	"display_name": "test-date-updated",
	"name": "test-date-updated"
	}
	
	[b-CKAN]
		The name of the data set

View	View the content of a specific data set. For example, this operation displays these results:
	{
	"count": 32,
	"display_name": "test-date-updated",
	"name": "test-date-updated",
	"tags": "education"
	}
	
	[b-CKAN]
		The name of the data set

CKAN is in use by numerous governments, organizations and communities around the world; currently the majority of open data sets globally are available and can be exposed via CKAN standards (ckan.org), originally from OKF (okf.org).

The data management via CKAN is described in the Data Management API (REF); in using the API you will be performing HTTP requests to URLs like: /api/rest/dataset with this returning data in the JSON format.

8.3 IoT data transaction management API

The IoT data transaction management API provides visibility and accessibility to data sets and streams gathered in urban environments and published on a data marketplace, which acts as a catalogue for easy sharing and trading. Data available on data transaction systems such as marketplaces adheres to standardized data formats, thus making it easy for application developers, which are data consumers, to move their services across different cities. On top of that, data providers can specify data license agreements and service level agreements when offering their data, which gives data consumers a guarantee of what their rights to use and redistribute data are, as well as what they can expect from the quality of data delivery, improving their trust and confidence to commit to subscribe to data sources.

The conceptual model of a marketplace for IoT data is shown in Figure 4. It captures the core abstractions of its architecture which provide the key concepts enabling interaction with the platform. The figure also includes optional conceptual elements represented in dashed lines: even if not compulsory, their implementation would enrich the basic functionality of the marketplace with features that may be required by a subset of users/stakeholders. At the core of the model is the notion of data offering. A data offering is related to a single data source, a digital asset registered into the marketplace by a data provider. A data source has an end point used to access it. When a data offering is published on the marketplace by a data provider, it becomes discoverable by data consumers, who may acquire access to the end point of the data source that a data offering is related to. Optionally, external data sources (e.g., data sets already published in open data portals) can be imported into the marketplace by data providers which can then publish related offerings. Several data offerings may refer to the same data source and provide different data license agreements and price plans. Optionally, a service level agreement could be defined which gives data consumers a guarantee of what they can expect from data delivery (e.g., number of data messages over a given period, data completeness). The marketplace could optionally provide a reputation feature that allows data consumers to set a rating score to a data offering they acquired access to or update a previous rating score.

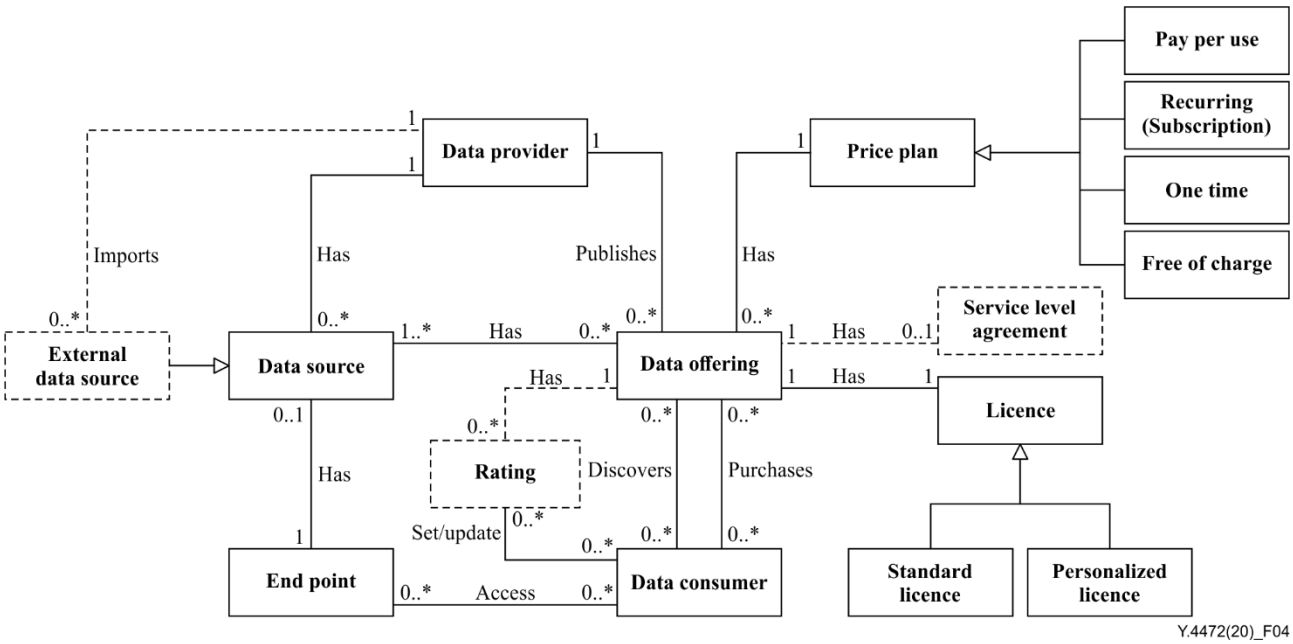

Figure 4 – Conceptual model of the marketplace for IoT data. Optional conceptual elements are represented in dashed lines. [b-SynchronicityMarketplace]

The marketplace may be accessible through the IoT Data Transaction Management API which can also expose a web portal where data providers and data consumers can interact with the platform (e.g., publish offerings, discover and acquire offerings). Identity management, authentication, authorization and accounting functionalities are ensured by a security component integrated with the marketplace and discussed in the next section. The main functionalities exposed by the IoT data transaction management API are listed in Table 7.

Table 7 – Main functionalities exposed by the IoT Data Transaction Management API

API	Functionality exposed	Description
IoT Data Transaction Management API	*publishDataOffering()*	Create a new data offering in the marketplace catalogue. License agreement and price plan is attached to the offering. Optionally, SLA can be set.
	searchDataOffering()	Look up data offerings in the catalogue. Filters (e.g., data type, location) can be applied.
	acquireDataOffering()	Acquire access to a specific data offering by receiving an access token.

Figure 5 shows two sequence diagrams that illustrate the main interactions of two different types of users, data providers and data consumers, with the IoT Data Transaction Management API.

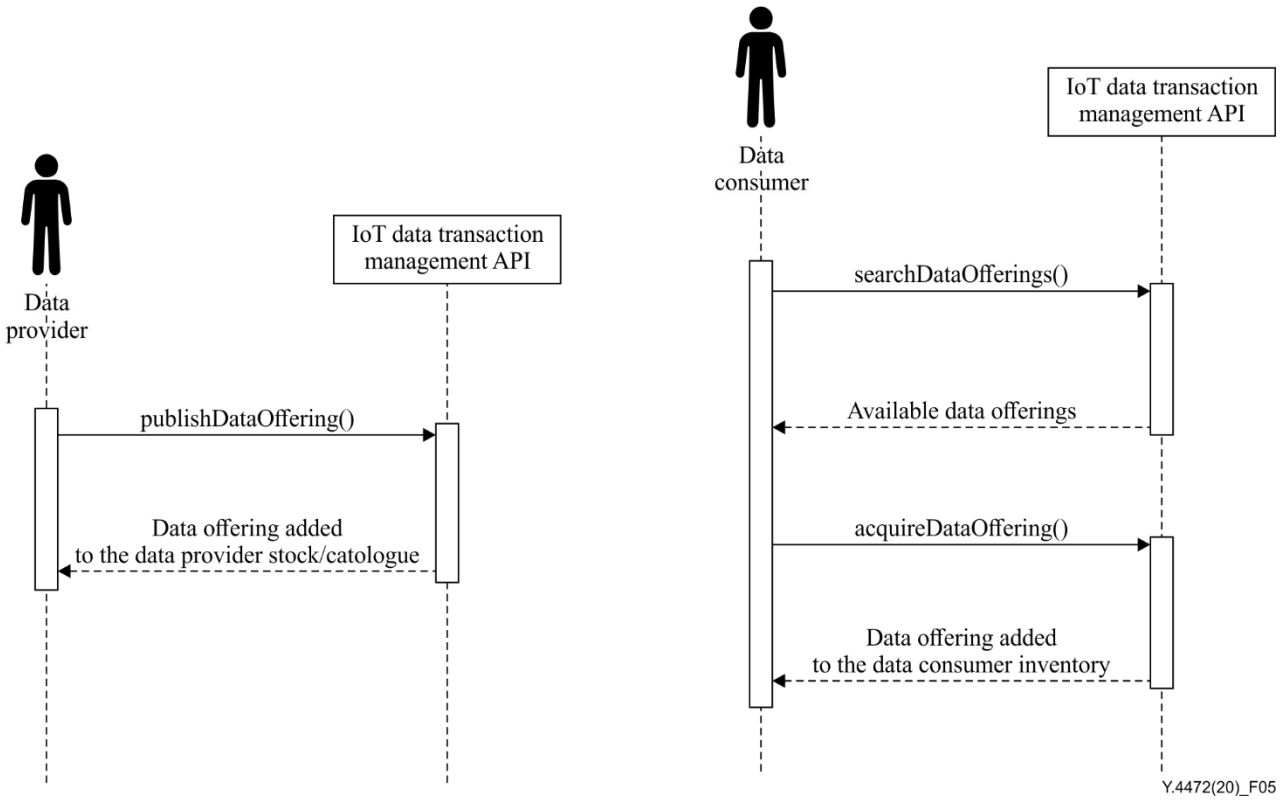

Figure 5 – Sequence diagrams showing the interactions of a data provider (left) and a data consumer (right) with the IoT Data Transaction Management API

8.4 Security API

The security interface provides a unified approach for the management of security policies as a viable and scalable means to define and enforce security rules consistently among the large variety of accessible resources (e.g., IoT devices, data and services).

With respect to the standard identity management, authentication, authorization and accounting components which reflect the OASIS security standard (e.g., XACML, oneM2M, GSMA, and ENISA) policy management is decoupled from devices and services. Policy can be managed independently, thus focusing on providing business value and compliance to data protection regulations. Key management, encryption, digital signature and data anonymization functionalities are directly linked to resources and governed by the policy management so that implementation of changes and enforcement are simplified by deploying policies on the fly affecting each point of use immediately. The main APIs that constitute the security interface are described next.

Identity and Authentication Management API: allows the registration, import and management of users and roles and to perform authentication. Upon successful registration an identifier is assigned to the users. The latter can also be registered by importing information from an external source. The API allows authentication to be performed by receiving user credentials, interacting with the authorization component and issuing authentication tokens. Users with admin privileges can retrieve, update and delete users required to pass the user identifier to the API, specifying which operation has to be performed and the respective additional information. In this case the API will return the outcome of the operation (e.g., success, fail).

Authorization and Accounting Management API: grants or denies permission to access resources and to log access requests by communicating with the identity and authentication management and the policy management API.

Policy Management API: allows the creation, retrieval, update and deletion of authorization policies. Policies might be simply based on the roles that users have in the system or embed more complex rules specified in a standard mark-up language (e.g., XACML).

Data Protection and Privacy Management API provides functionality to ensure confidentiality, authentication, integrity, non-repudiation, data minimization, data retention and consent capabilities around the collection and dissemination of data. It provides methods to encrypt/decrypt data, digitally sign data (and verify signatures), and anonymize data using specific algorithms opportunely selected. Figure 6 illustrates the sequence diagram of the interactions between the APIs of the security interface in a data access request. A description of such interactions which highlights specific components of the APIs is detailed next.

The data consumer (e.g., a service developer that has acquired access to a specific data source via the marketplace) sends an authentication request (login) to the Identity and Authentication Management API, which then responds with an access token that the user will need to use to access the resource (data source). When the user wants to access a data source, he sends an access request to the authorization and accounting management API. If XACML is used, the request is intercepted by the authorization PEP proxy sub-component which gets authorization info from the token (i.e., the user role) and checks the validity and status of the token through the Identity and Authentication Management API. Eventually, the authorization PDP sub-component evaluates the access request after retrieving the authorization policy from the Policy Management API. Depending on this decision, the PEP proxy blocks or forwards access to the resource (through the context data management API). To track the amount of resources accessed, access information is logged into the accounting sub-component of the Authorization and Accounting API.

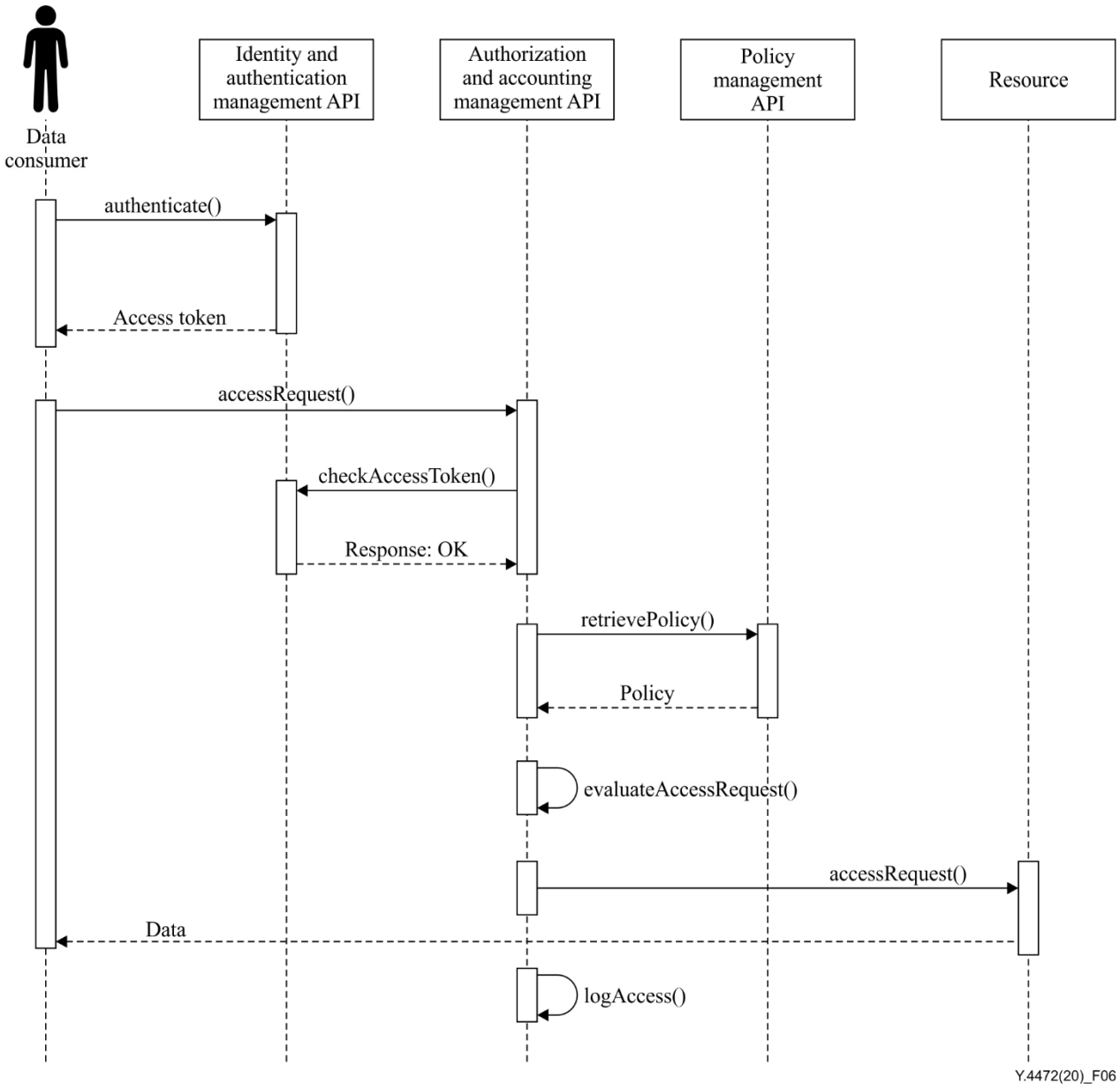

Figure 6 – Sequence diagram showing the interactions between the APIs of the security interface in a data access request

The sequence diagram in Figure 7 shows the interactions between the APIs of the security interface for creating and retrieving a security and privacy policy. The data protection and privacy management API enforces security and privacy measures previously defined, such as encryption and anonymization, defined by a policy admin through the Policy Management API, directly on the resource (e.g., data).

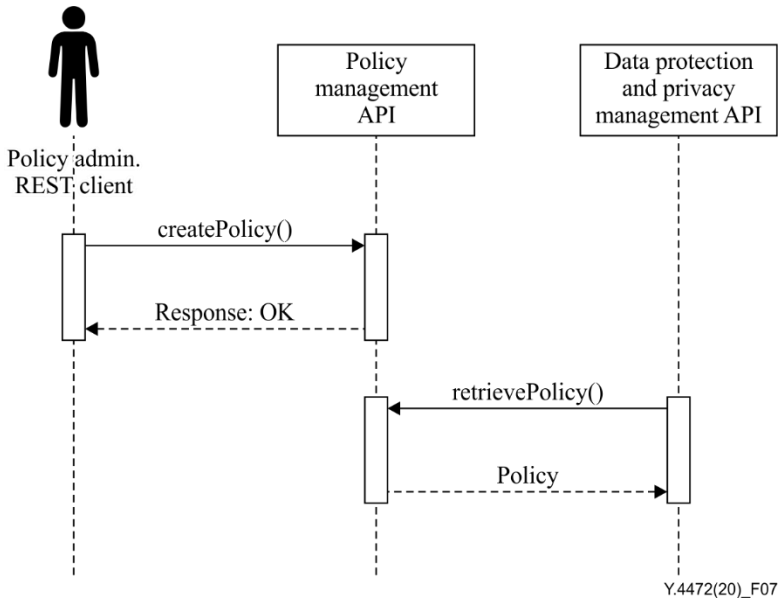

Figure 7 – Sequence diagram showing the interactions between the APIs of the security interface for creating and retrieving a security policy

The main functionalities exposed by the APIs of the security interface are listed in Table 8.

Table 8 – Main functionalities exposed by the APIs of the security interface

API	Functionality exposed	Description
Identity and Authentication Management API	*register()*	*Sign up new user*
	authenticate()	*Sign in user*
	checkAccessToken()	*Check the validity and status of the access token*
Authorization and Accounting Management API	*accessRequest()*	*Handle resource (data) access requests from users*
	logAccess()	*Log access information*
	evaluateAccessRequest()	*Evaluate the access request against a specified policy*
Policy Management API	*createPolicy()*	*Define authorization policies (e.g., role based) as well as security and privacy measures*
	updatePolicy()	*Update an authorization policy*
	retrievePolicy()	*Get a particular authorization policy*
	deletePolicy()	*Erase an authorization policy*
Data Protection and Privacy Management API	*encrypt() / decrypt()*	*Encrypt / decrypt data*
	sign()	*Digitally sign data*
	verify()	*Verify signature*
	anonymize()	*Anonymize data using a specified algorithm*

8.5 Common data models

One of the core elements in this Recommendation refers to the adoption of common data models for representing the information linked to the observations. When considering the interoperability as a means for breaking the vertical silos that have been characterizing the urban ecosystems, it is mandatory to adopt homogeneous data models agreed by the different stakeholders. In this way the synergies between services can be put into practice. Just as an example, a waste management company can share real-time information related to the positions of the trucks along the city streets. Sharing that information with the traffic management department allows car drivers to have access, in real time, to information related to those streets which should be avoided due to potential traffic congestion. Further to this, when data is provided with the corresponding ontologies, the closer we are to implementing true autonomic cities. This means that thanks to data interoperability between services, knowledge acquisition is faster and more reliable fostering decision making without human intervention.

In terms of design and implementation, data models have to be conceived to gather most of the usual needs linked to the urban services. Furthermore, they have to be easily scaled according to new requirements imposed by the evolution of the technology and the urban ecosystems' own characteristics.

Last but not least, in the same way that infrastructure federation is becoming a reality, data federation will become a key enabler in optimizing intercity interactions. Thus, providing an optimization opportunity which will go beyond the legacy local approaches.

8.6 IoT device management API suite

From a smart city point of view, one of the most fundamental issues is setting up the management of IoT devices. These APIs are managing IoT devices and IoT agents, which are collections or federations of IoT devices. What is important is that the suite provides a normalized interface to configure and manage the IoT device, get alarms, configure and monitor IoT devices, and provide secure access to them. The same set of open APIs can be deployed in the silicon of the IoT device or used by an agent.

It includes APIs for:

- IoT device inventory
- IoT device onboarding
- IoT device discovery
- IoT device type management
- IoT state management
- IoT alarm management
- Streaming of IoT data events
- Streaming of IoT management events
- IoT configuration management
- IoT data multi-protocol data access MQTT and NGSI-LD
- IoT API authorization

An example of such a suite is the TMF IoT Device Management Suite of APIs. [b-TMF908]

8.7 IoT service management API suite

The ability to create services around all the IoT device data that is collected within the smart city is essential.

The TMF IoT Service Management API allows the building of services based on the data provided by the IoT devices through the IoT device management APIs.

It includes APIs for:

- IoT service inventory
- IoT service onboarding
- IoT service discovery
- IoT service type management
- IoT service state management
- IoT service problem management
- Streaming of IoT service data events
- Streaming of IoT service management events
- IoT service configuration management
- IoT service data multi-protocol data access MQTT and NGSI-LD
- IoT service test management
- IoT service API authorization

An example of such a suite is the TMF IoT Service Management Suite of APIs [b-TMF914].

Appendix I

Instruction for open API implementation

(This appendix does not form an integral part of this Recommendation.)

This appendix explains implementation guidelines. The technical activities of each smart city platform compliance can vary in terms of both complexity and duration but it is possible to define some milestones: Figure 1 describes this approach presenting four progressive steps with different levels of compliance with the open APIs that can be reached by cities indicating different degrees of interoperability with the platform:

Figure I.1 – Compliance levels of the open API framework

In the following section are described in detail the different phases and potential activities to perform in order to reach the specific compliance level.

Level 0 – Initial situation: proprietary/specific systems/API
The existing smart city platform is made and not compliant with open APIs. The IoT infrastructure and software platforms are managed/owned by the municipality of external providers. The IoT devices communicate with the infrastructure using specific/proprietary protocols. The data is accessible through specific/proprietary API. The data models are specific, not aligned with open standards. All the systems and applications are strictly tailored by the specific municipalities.

Level 1 – Entering interoperability process: asset identification
The cities want to achieve data interoperability with the open APIs and defined a technical interoperability architecture in order to make the existing IoT infrastructure compliant with the specifications of this Recommendation (interoperability points). The cities identified which data sets and services can be brought for the work.
Activities to be performed to achieve the level
The technical activities in this stage should be focused on two different tracks: the identification of the possible interactions between the IoT infrastructure of the city and the open API framework, estimating the minimum technical requirements needed to implement the "interoperability points". In this phase, it should be identified which components (hardware and software) of the existing city infrastructure have to be adapted using specific connectors or data injectors to convert native interfaces (e.g., REST API) to the open APIs. In parallel, it is necessary to investigate and identify which data sets should be exposed following the open API specifications: these data sets can include, for instance, real-time data coming from IoT devices, historical and open data, micro-services data.
Mandatory requirements
Knowledge of the open APIs framework and interoperability points specifications
Optional requirements
N/A

Level 2 – Implementation of the open APIs
A city implemented the open APIs: city data sets are accessible through open and standard interfaces that represent the main interoperability points with the open APIs framework.
Activities to be performed to achieve the level
The activities inside the city are devoted to the provisioning of the data using open and standard APIs: the city should work together with the technical partners to adapt the existing interfaces to the open APIs, in particular to the context management one. The city can use existing software components or can develop a specific software adapter to inject data into the context data management component. This can be considered the first step to share the city data with the open APIs and be part of the ecosystem.
Mandatory requirements
• Implementation context management APIs for real-time data provisioning (Interoperability points) • Implementation of security APIs to be compliant with the open APIs authentication and authorization (Interoperability points)
Optional requirements
• Implementation of data storage API to give access to the city's historical data (Interoperability points) and open data repositories

Level 3 – Adoption of common data models
The city adopts common data models to represent the city data. The common data models are domain specific and their usages enable the replicability of the applications and services based on them between different cities.
Activities to be performed to achieve the level
The city has to map its own legacy data models with common data models in order to select the most suitable ones for the specific smart city domains. The common data model can be extended or new ones can be created to represent new entities or attributes which are not covered by the current ones. The ones identified and mapped to the correct data model specific adapters should be implemented to convert the original data in the new format. *As a reference, EU H2020 SynchroniCity project provides tools and guidelines to speed up the mapping and conversion of the city data.*
Mandatory requirements
• Adoption of common data models (Interoperability points)
Optional requirements
N/A

Level 4 – Data transaction ecosystem participation
The city is fully part of the open API ecosystem by actively participating in the data transaction of the city data using the open APIs. The city data is included in a data marketplace catalogue that supports the open APIs and can be accessed by third-party applications and external stakeholders. The marketplace supports also monetization mechanisms allowing, for instance, to sell real-time or specific historical data sets. The active participation to the marketplace enables the concrete realization of a digital single market.
Activities to be performed to achieve the level
From a technical point of view, a city, being compliant with the previous levels, is ready to be part of a marketplace built upon the open APIs. The city can also decide to access an existing data marketplace or to install a specific version customized for the city's local ecosystem. If the city already has a marketplace supporting the open APIs, it could be federated to the other marketplaces using the same open APIs. In this case, limited technical activities will be necessary to implement the federation. In addition to that, the city will be involved in other business and strategic decisions to define the terms and approaches to be part of the other marketplace (e.g., proper handling of ownership, service access terms, data licenses etc.). It will be possible to have different levels of integration starting from the simple participation to the marketplace catalogue, to promote specific data assets, up to supporting all the other functionalities, such as revenue sharing, feedback collection, SLA management etc.
Mandatory requirements
• Compliance with data transaction APIs (Interoperability points)
Optional requirements
• Deployment of a local marketplace instance

Bibliography

[ITU-T J.380.8] Recommendation ITU-T J.380.8 (2011), *Digital program insertion – Advertising systems interfaces – General information service.*

[b-ITU-T T.170] Recommendation ITU-T T.170 (1998), *Framework of the T.170-Series of Recommendations.*

[b-ITU-T Y.101] Recommendation ITU-T Y.101 (2000), *Global Information Infrastructure terminology: Terms and definitions.*

[b-ITU-T Y.4000] Recommendation ITU-T Y.4000/Y.2060 (2012), *Overview of the Internet of things.*

[b-CKAN] CKAN (accessed 2020)
<https://docs.ckan.org/en/2.8/ >

[b-FG-DPM TS D0.1] ITU-T Technical Specification D5 (2019), *Data economy: commercialization, ecosystem and impact assessment*.
<http://handle.itu.int/11.1002/pub/813b086a-en>

[b-SynchronicityMarketplace] SynchroniCity Project (accessed 2020), *D.2.5 Advanced data market place enablers*.

[b-TMF908] TMF908 (2020), *IoT Agent and Device Component Suite API Specification*, TM Forum.
<https://www.tmforum.org/resources/specification/tmf908-iot-agent-and-device-component-suite-api-specification-v1-0/>

[b-TMF914] TMF914 (2020), IoT Service Management API Component Suite v4.0.0, TM Forum
<https://www.tmforum.org/resources/specification/tmf914-ai-management-api-component-suite-user-guide-v4-0/>

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, 2021