

SYNCHRONICITY



GA no:	732240
Action full title:	SynchroniCity: Delivering an IoT enabled Digital Single Market for Europe and Beyond
Call/Topic:	Large Scale Pilots
Type of action:	Innovation Action (IA)
Starting date of action:	01.01.2017
Project duration:	36 months
Project end date:	31.12.2019
Deliverable number:	D2.5
Deliverable title:	Advanced data market place enablers
Document version:	Ver1.0
WP number:	WP2
Lead beneficiary:	14-DigiCat
Main author(s):	Andrea Gaglione, Daniel Puschmann, Alex Gluhak (DigiCat), Martino Maggio (ENG)
Internal reviewers:	Ignacio Elicegui Maestro (UC), Francisco Monsanto (UBIWHERE), Eunah Kim (UDG Alliance), Martin Brynskov (AU)
Type of deliverable:	Other
Dissemination level:	Public
Delivery date from Annex 1:	M23
Actual delivery date:	15.01.2020 (M37)



Co-funded by the
European Union

This deliverable is part of a project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 732240.

Co-funded by



Switzerland



South Korea

Executive Summary

IoT data marketplaces provide a means of unlocking existing IoT data silos and offer opportunities for generating new revenue streams and new value for all parties. The SynchroniCity vision is to scale these opportunities by envisioning a vibrant IoT data marketplace as a key enabler for the creation of a global market for IoT-enabled urban services with standardized interfaces and common information models. The IoT data marketplace will be a key element of the ecosystem transaction management set out by SynchroniCity, sharing this vision with the Open & Agile Smart Cities (OASC) community, which already encompasses more than 100 cities across the world.

This deliverable presents the advanced data marketplace enablers and complements the basic enablers introduced in deliverable D2.4 in order to make this vision a reality. The advanced enablers set the final version of the marketplace apart from its basic counterpart and from other similar solutions available in the market. In fact, the main objective of the platform is to build trust among users and stimulate the establishment of an ecosystem in which different stakeholders are incentivized to exchange data.

In the following, we provide a high-level summary of the key points addressed in the deliverable. For more details, please consult the respective sections of the deliverable.

SLA and license management

Our initial work for the development of this component led to the definition of the license agreements – presented in deliverable D2.4 – which define the terms of use for data. We have finalized the component and related API, by adding the support for the definition of service level agreements (SLA) between a data provider and a data consumer. Our goal is to let data providers and data consumers agree on a number of end-to-end parameters (metrics) that define the system performance and guarantee a certain level of QoS during the provisioning of data streams.

Feedback and reputation

To improve the confidence of data consumers that data available in the marketplace will satisfy their needs, we developed a five-star rating system according to which data consumers can rate data offerings on the basis of a scale from 1 to 5. Data consumers are allowed to add a comment along with the rating score and they are also able to change their rating at a later stage.

Transparency and accountability service

The underlying leading principle of the IoT data marketplace is to foster the development of a trusted and transparent ecosystem that should eventually attract stakeholders and grow the business around the exchange of IoT data. The transparency and accountability service is the enabler to realize this. This enabler leverages blockchain technology to store license agreements and SLA, as well as track the SLA during data provisioning, thus creating a visible and immutable audit trail aiming to promote a fair behavior among the platform users.

Federation management

A proper federation among closed data deployed in multiple RZs is still not possible within the current SynchroniCity framework because user identities are managed locally by the RZs. Open data can be federated using the context broker federation mechanism, where one instance of the context broker

can subscribe to another context broker, replicating the data. Furthermore, we have integrated the IoT data marketplace with an open data federation platform that allows it to expose open datasets in the catalog and create related data offerings regardless of the technology of the original open data portal/repository.

Eventually, we have extended the PoC prototype presented in deliverable D2.4 and integrated it with Manchester RZ IoT infrastructure other than Santander RZ's. We validated the prototype by showing how the transparency and accountability service allows to store the agreements associated with a data offering on a blockchain after a data consumer purchases the offerings.

We are currently finalizing the implementation and testing of the transparency and accountability service and will deliver a final version of the IoT data marketplace by the end of M24, well timed for the launch of the open call selected pilots.

Abbreviations

API	Application Programming Interface
CKAN	Comprehensive Knowledge Archive Network
D	Deliverable
DCAT	Data Catalogue Vocabulary
EC	European Commission
GA	Grant Agreement
GE	Generic Enabler
HTTP	Hypertext Transfer Protocol
IoT	Internet of Things
JSON	JavaScript Object Notation
NGSI	Next Generation Service Interfaces
PBFT	Practical Byzantine Fault Tolerance
RDF	Resource Description Framework
REST	Representational state transfer
SLA	Service Level Agreement
W3C	World Wide Web Consortium
WP	Work Package
WT	Work Task
XML	eXtensible Markup Language

Contents

1	Introduction.....	7
2	SynchroniCity IoT data marketplace – Advanced enablers	8
2.1	SLA and license management.....	9
2.2	Feedback and reputation.....	10
2.3	Transparency and accountability service	12
2.3.1	Storage of agreements.....	13
2.3.2	SLA tracking.....	14
2.4	Implementation details.....	16
2.4.1	Hyperledger Fabric.....	16
2.4.2	Blockchain application and integration details	17
2.5	Federation management	18
3	Validation.....	22
3.1	Data sets and data models.....	22
3.2	Marketplace in action.....	23
4	Conclusion.....	26
5	References.....	27
	Appendix	28

List of Figures

Figure 1. SynchroniCity IoT data marketplace – API and portal development.	8
Figure 2. Revised conceptual model of the SynchroniCity IoT data marketplace.	12
Figure 3. Overview of the interactions enabled by the transparency and accountability service. ...	13
Figure 4. Use case diagram: Create and sign agreement (data provider).	14
Figure 5. Use case diagram: Sign agreement (data consumer).....	14
Figure 6. Interactions between SynchroniCity components and the blockchain enabled by the transparency and accountability service.	15
Figure 7. Interactions of the blockchain application with the marketplace portal and the PEP proxy through RESTful interface.	18
Figure 8. Overview of the integration between Idra and the SynchroniCity IoT data marketplace..	20
Figure 9. Sequence diagram: Import datasets into the marketplace.	21
Figure 10. Marketplace portal – Catalog Santander.	23
Figure 11. Marketplace portal – Catalog Manchester.	24
Figure 12. Details page of a data offering.	24
Figure 13. REST interface of Hyperledger Fabric.....	25
Figure 14. Output logs of the blockchain nodes.	25

1 Introduction

The main ambition of the SynchroniCity project is to open up a global IoT market where cities and businesses can develop digital services to improve the lives of citizens and grow local economies. In fact, SynchroniCity aims to harmonize smart city solutions. Part of this harmonization effort is creating a marketplace for IoT data which allows IoT businesses to exchange and even trade IoT sensor data in order to generate secondary revenue streams out of that. The IoT data marketplace is a key element of the SynchroniCity *ecosystem transaction management* and is complemented by the marketplaces for hardware and services. The design of the data marketplace platform has been presented in D2.4 along with the enablers that provide basic functionalities – e.g., publish and purchase data offerings – by adapting/extending the FIWARE/TMForum Business API Ecosystem (BAE) Generic Enabler.

As a result of the continuous effort, this document introduces the *advanced data marketplace enablers*. They have been partially introduced in D2.4 with reference to this document, and this Deliverable presents them with a much more detailed description. The advanced enablers are those IoT data marketplace components that provide more complex tools and functionalities and contribute to build trust among the end users. Therefore, they set the data marketplace platform apart from its basic version and from other similar solutions available in the market. In fact, the goal of the data marketplace is not only to unlock IoT data sources and make them purchasable. A key design principle of the platform and the overall SynchroniCity framework is to foster the development of a trusted and transparent ecosystem which should eventually attract stakeholders and grow the business around the exchange of IoT data and smart city data.

By presenting the advanced data marketplace enablers, this document outlines the final version of the IoT data marketplace, which is part of the SynchroniCity advanced framework and paves the way to the experimentation through the pilots selected in the open call selected.

2 SynchroniCity IoT data marketplace – Advanced enablers

The IoT data marketplace platform plays an important role in realizing the SynchroniCity vision of a Digital Single Market. It provides visibility and accessibility of data sets and streams that are gathered in urban environments for easy sharing and trading. The guarantee that the data available on the marketplace adhere to the standardized data formats makes it easy for application developers to move their services across different cities. On top of that, data providers can specify service level agreements (SLA) when offering their data, which gives the data consumers a guarantee of what they can expect from the data delivery, improving their trust and confidence to commit to subscribe to data streams.

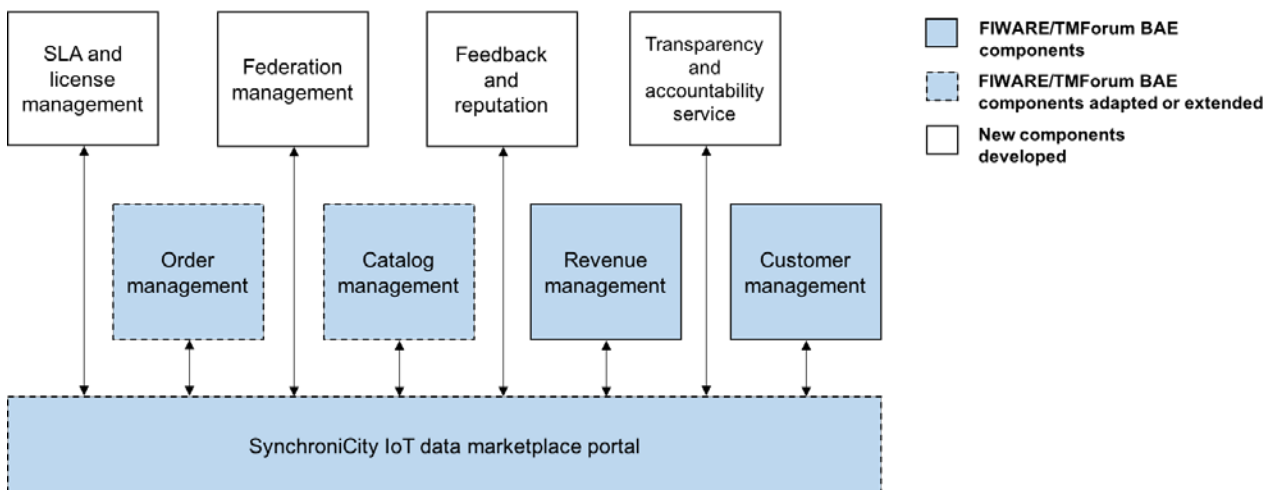


Figure 1. SynchroniCity IoT data marketplace – API and portal development.

Figure 1 shows the main components of the IoT data marketplace API and the portal and also highlights:

- the components of the FIWARE/TMForum BAE reused straight away;
- the components of the FIWARE/TMForum BAE adapted or extended to suit the requirements of a platform addressing IoT data;
- new components developed which provide more advanced features.

The basic version of the SynchroniCity IoT data marketplace - released in M15 - included all the components/enablers reused or adapted/extended from the FIWARE/TMForum BAE along with the initial version of the *SLA and license management* component. The basic enablers allow data providers to publish data offerings into the marketplace, and data consumers to purchase the offerings after agreeing on the license terms and conditions related to the use and distribution of data.

The ambition that led us towards the development of advanced enablers was to build trust among marketplace users which would incentivize them to join the ecosystem and exchange data. The idea is to foster a fair behavior among the users in adherence to SLA and license agreements as this way they will build a positive reputation and a better relationship with existing customers, thus increasing the influx of new customers. The advanced enablers set the final version of the marketplace apart from its basic counterpart and from other similar

solutions available in the market. The main components developed are summarized in the following text, whilst their detailed description is provided in the next sections.

- *SLA and license management.* The feature to add data licenses to data offerings, has been developed as part of the basic enablers for the marketplace. The component has been completed to include features that allow to define the SLA associated with data offerings.
- *Transparency and accountability service.* A component that interacts with a blockchain network to: (i) store both the license agreements and SLA in a tamper-proof way; (ii) track the SLA during data provisioning. The component enables the creation of transparent and auditable trails of the agreements and also of the way data shared through the marketplace is provided and consumed.
- *Feedback and reputation.* A simple mechanism that allows data consumers to rate data offerings. A history of feedback and reputation will be tracked in order to give an ongoing picture of the trustworthiness of marketplace participants.
- *Federation management.* We have realized an integration with an open data portal that allows to easily import open data into the marketplace and vice versa. In the case of open data, we also exploit the subscription capabilities of the Orion Context Broker to enable partial federation between multiple instances of the SynchroniCity marketplace.

2.1 SLA and license management

The marketplace allows data providers to attach a license agreement to the data offering and specify SLA that the provider promises to adhere to when providing dynamic data streams.

As described in D2.4, the license agreements can be either chosen from a set of pre-defined licenses commonly used for open data or if they are not satisfactory for the needs and wants of the data provider, custom agreements can be specified based on a template. The latter allows to define: (i) business activity sectors for which the data may be used, (ii) purposes for purchasing and using the data, (iii) authorization to resell the data, (iv) geographical territories in which the data may be used and, (v) the date after which the authorization period to use the data ends.

On the other hand, SLA are contracts of quality of service (QoS) between a service provider and a service consumer and have been widely used for long time in telecommunication, IT services, and service oriented architectures [1]. As our marketplace allows to trade data sets - making them accessible to users (data consumers) - the services specified here concern the data streams between the sources of data providers and the end-user applications of data consumers. Our goal is to let data providers and data consumers agree on a number of end-to-end parameters (metrics) that define the system performance and guarantee a certain level of QoS during the provisioning of data streams. Currently, the marketplace allows to specify three metrics as part of the SLA:

- *Update rate:* expected number of updates of the data source in a given time frame (e.g., 100/day).
- *Response time:* average (or maximum) amount of time it takes for the Context Broker to respond to a data request (e.g., GET) sent by an end-user application (e.g., 400

ms). Note that this metric stands only if the data provider offers the data through its own brokering infrastructure.

- *Delay*: metric used for subscription-based data access, it notes the average (or maximum) amount of time to deliver a new update to the data consumer (e.g., 1000 ms).

Most IoT applications rely upon these parameters in order to work properly and satisfy the stakeholders' requirements, and other metrics can be added later on. When purchasing the access to data through the marketplace, a data consumer has to sign the license attached to the offering as well as the SLA. In order to create an immutable, transparent and tamper proof audit trail, this signed agreement is stored on a distributed ledger.

SLA and license management APIs are logically grouped into two sub-modules: license specification management and SLA specification management. They have already been introduced in D2.4. The main features are reported in this deliverable with more details being provided for the SLA specification management module. The final SLA and license management API specification is described in the Appendix (as well as online at: <https://synchronicityiotdm.docs.apiary.io/#>). The part related to the SLA management has been modified with respect to that presented in D2.4, in accordance with the metrics defined below.

- **SLA specification management.** A SLA specification is a detailed description of the SLA related to a particular data offering available in the marketplace. It exposes the following interfaces:
 - SLA specification collection: interfaces for retrieving a list of SLA specifications and for creating a new SLA specification.
 - SLA specification entry: interfaces for retrieving and updating a specific SLA specification entry.
- **License specification management.** A license specification is a detailed description of the terms and conditions by which the related data offering is made available through the marketplace. It exposes the following interfaces:
 - License specification collection: interfaces for retrieving a list of license specifications and for creating a new license specification.
 - License specification entry: interfaces for retrieving and updating a specific license specification entry.

Both the license agreements and the SLA are specified for the data offerings. Therefore, if a data provider wants to offer the same data with different SLA and different price, then multiple data offerings can be published each with their own SLA and pricing models, all referring to the same data source specification.

2.2 Feedback and reputation

In a marketplace where data providers are competing, it is necessary for a potential data consumer to be able to immediately gauge the quality of the data sets provided. The specification of the SLA by the data provider can already help to build confidence that the data provided will satisfy the needs of the consumer. However, by allowing data consumers who are already accessing the data to give a rating about the data sets they are consuming,

potential new end users can see beforehand how satisfied they might be when accessing the data.

Our feedback and reputation management component rely upon a five-star rating system according to which the consumer can score data offerings on the basis of a scale from 1 to 5. This is an already established system in the minds of people and they can see if it is good for them. Data consumers are allowed to add a comment along with the rating score and they are also able to change their rating at a later stage. This is necessary as some of the data provided on the marketplace will be streaming data and therefore might be subject to changes in the data quality as well as in the delivery by the data provider (in terms of update rate, response time and delay).

The average of the current rating is displayed in the marketplace. However, the history of the rating is also stored, as this can form one part of the reputation that the data providers are building over time. A data provider that consistently supplies the data in accordance to the service level agreements should be considered higher reputable than a data provider that had drops in quality or data delivery in the past.

The **feedback and reputation API** exposes two functionalities to the marketplace. The first allows the data consumer to set a rating to a data offering they acquired access to or update a previously rated offering. The second functionality allows the marketplace to retrieve the average of ratings so that it can be displayed together with the data offering in the portal. The feedback and reputation API specification is described in the Appendix (as well as available online at: <https://synchronicityiotdm.docs.apiary.io/#>).

The revised conceptual model of the SynchroniCity IoT data marketplace is shown in Figure 2, where shaded boxes have been used to indicate those elements that got changed or have been added since the initial model. In fact, besides adding the SLA and reputation elements, we have changed the license element which is now only separated into: (i) *standard license* that can be chosen from a predefined set (e.g., Open Data Commons, Creative Commons), and (ii) *personalized license*, according to which data providers can either set additional restrictions to the access rights of their data, such as exclusivity, geography, duration, and purpose, along with the permissions regarding redistribution, adaptation, and resale, or write themselves the terms and conditions associated to their data offerings.

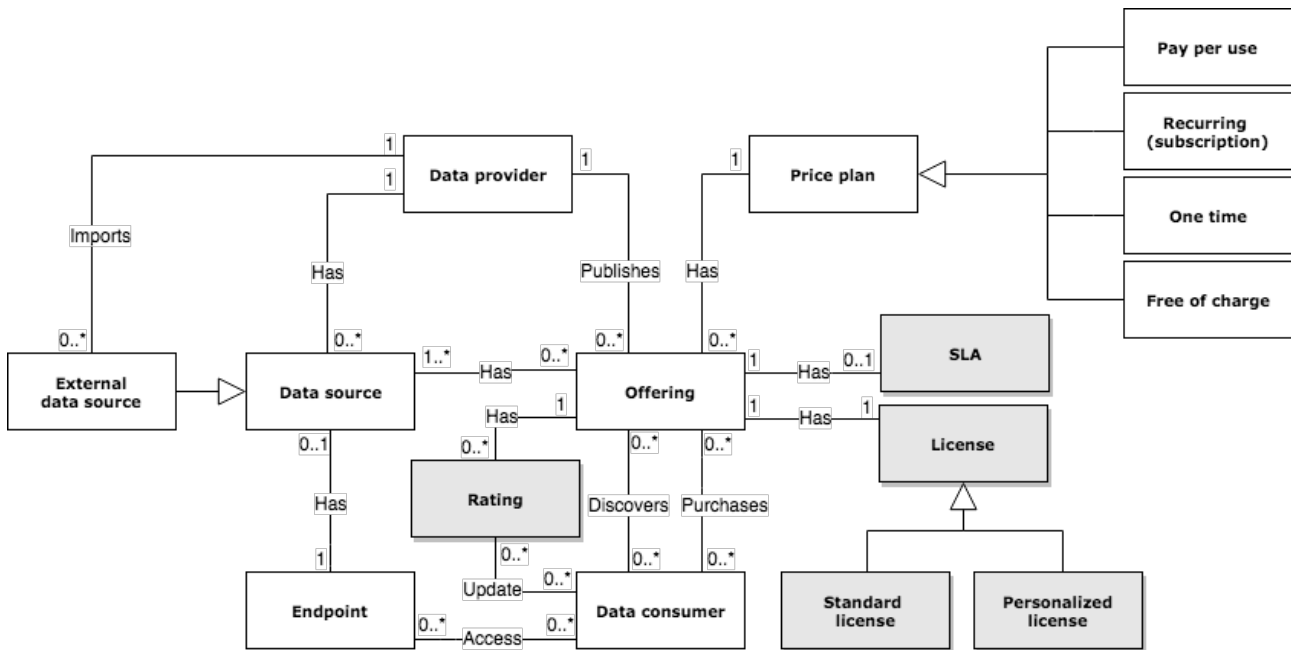


Figure 2. Revised conceptual model of the SynchroniCity IoT data marketplace.

2.3 Transparency and accountability service

As a rating system by default is a more subjective form of reputation and can be gamed by malicious users, we had to conceive a fair mechanism that allows to keep track of the actual behaviors of the marketplace users when it comes to accessing data and with respect to the agreements established between data providers and consumers at the time of purchase. In fact, a key design principle of the marketplace platform and the overall SynchroniCity framework is to foster the development of a trusted and transparent ecosystem that should eventually attract stakeholders and grow the business around the exchange of data sources. To this aim, we have developed the transparency and accountability service, a component that enables interactions between the marketplace / other SynchroniCity components and a distributed ledger technology (DLT) or a blockchain network to:

- store both the license agreements and SLA between data providers and consumers;
- track the SLA adherence during data provisioning.

Therefore, the service allows to create a visible and immutable audit trail which promotes fair behavior among the platform users as they are incentivized to build up a good reputation naturally.

Past behavior is a strong indicator for future behavior, therefore, knowing the history of behavior of a potential business partner beforehand will grow the trust that is necessary to enter into business relations with them.

Figure 3 shows the high-level view of the interactions enabled by the transparency and accountability service.

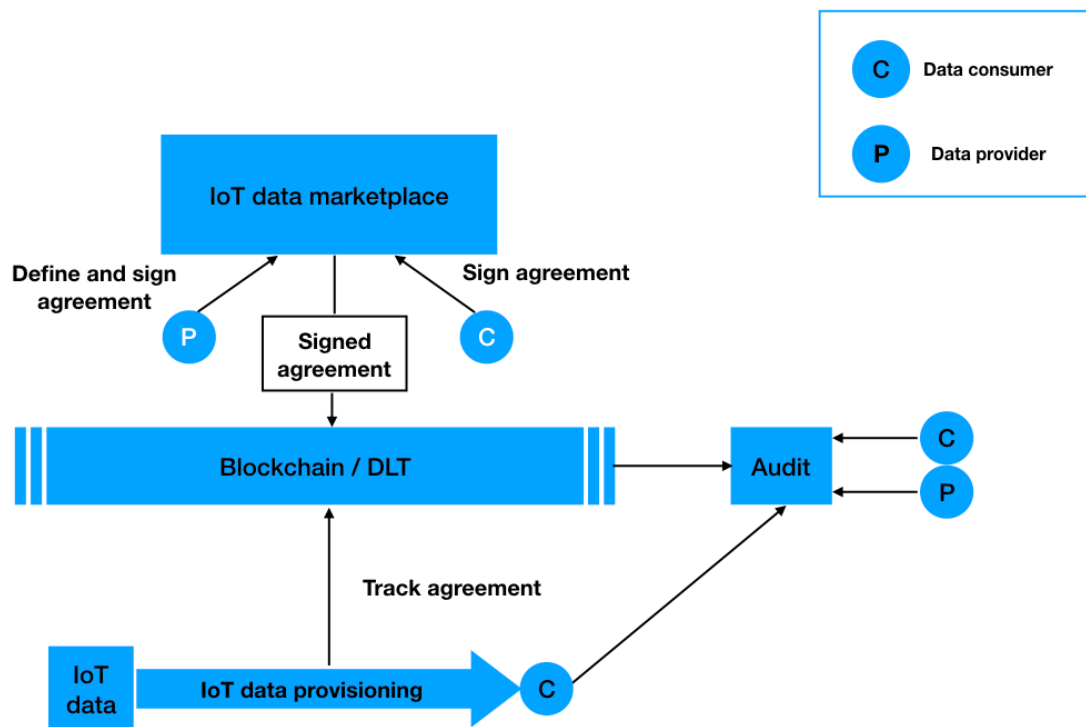


Figure 3. Overview of the interactions enabled by the transparency and accountability service.

2.3.1 Storage of agreements

Upon acquisition of a data offering on the marketplace, the data consumer has to sign the agreement laid out by the data provider. The agreement includes the attached data license, the SLA and the signatures of the data provider and data consumer. This signed agreement is stored on the distributed ledger.

Having the agreement stored on the blockchain builds the foundation for building an immutable audit trail. Once two parties finalized the deal of data provisioning, neither can alter the details of the agreement afterwards.

Figure 4 and Figure 5 show the use case diagrams of the system with respect to storing the agreements on the blockchain. First, when publishing a data offering, a data provider creates the agreement and signs it, before the signed agreement referring to that specific data offering gets stored on the blockchain (Figure 4). Then, when purchasing a data offering, a data consumer signs the related agreement, before the double-signed agreement gets stored on the blockchain (Figure 5).

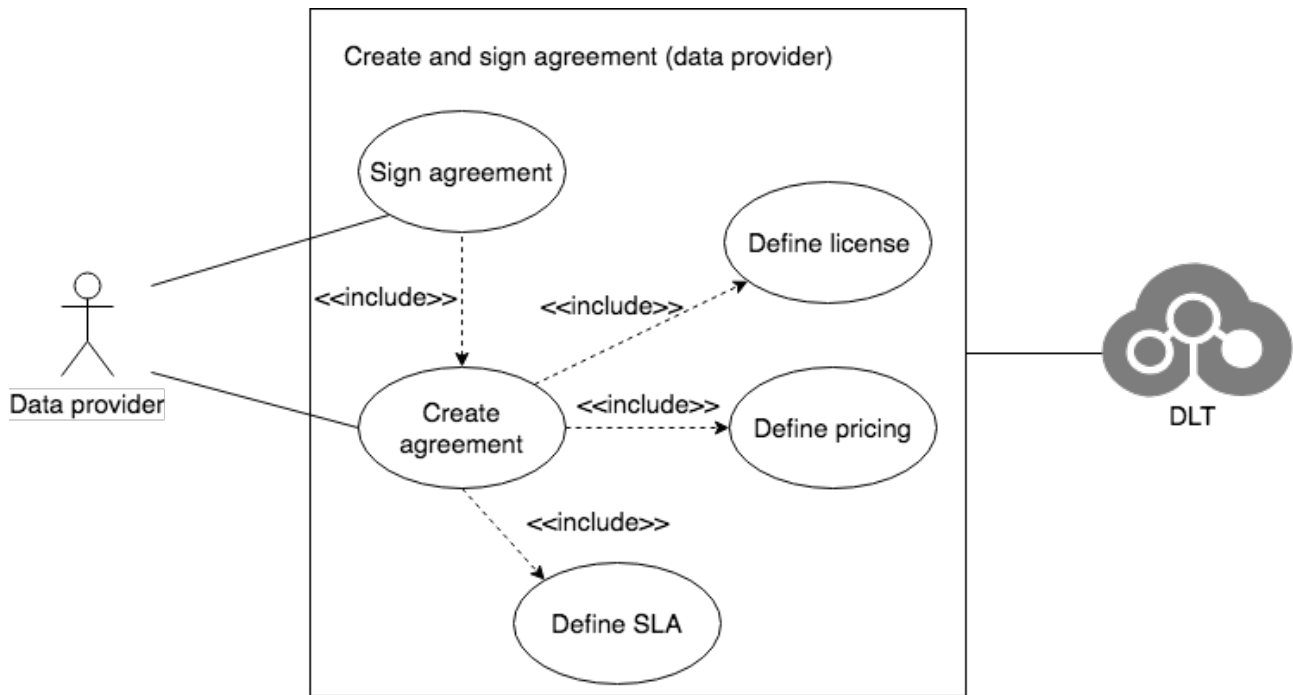


Figure 4. Use case diagram: Create and sign agreement (data provider).

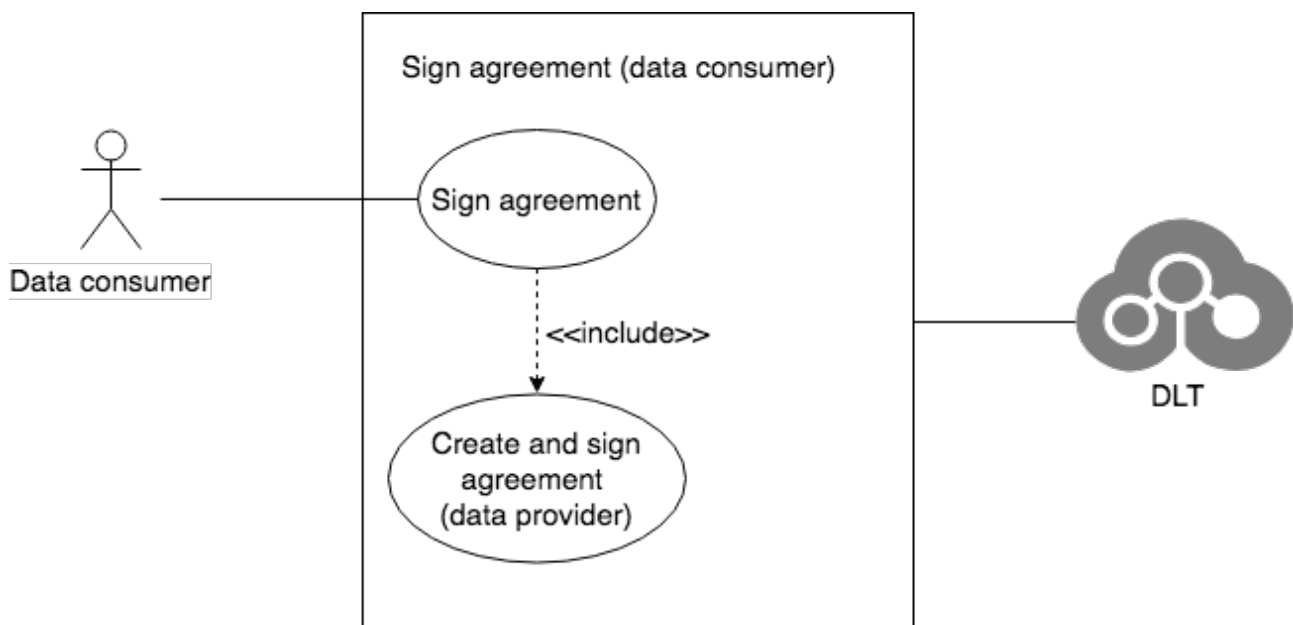


Figure 5. Use case diagram: Sign agreement (data consumer).

2.3.2 SLA tracking

After acquiring the data offering, the data access is handled through the PEP proxy Wilma Plus (see <https://gitlab.com/synchronicity-iot/fiware-pep-proxy.git>). We can exploit this fact for tracking the adherence to the agreed upon terms from both parties, data provider and data consumer alike.

- **Adherence to the SLA by the data provider:** the SLA management allows the data provider to specify guarantees for the update rate, response time and delay. If one or more metrics have been selected for the data offering, we monitor if the data delivery meets these guarantees.

Whenever the PEP proxy Wilma Plus registers a data access request, this is stored on the distributed ledger. This way, participants of the marketplace cannot later hide their actions in case of misbehavior. This history of past behavior can now be used to ensure better trust between parties. Before doing business with another marketplace participant, an audit trail of past data transactions can be inspected. A participant with a long history of good behavior is much more likely to adhere to the laid-out agreements in the future as well, whereas a participant with many instances of contract breaches in the past may do so in the future as well. This allows all participants to make informed decisions on whom they want to make business with and who they would rather avoid as a potential business partner. At the same time this mechanism strongly incentivises the marketplace participants to behave according to their agreements, as only a clean history of behavior will guarantee the willingness of collaboration from other participants of the marketplace.

The interactions between the SynchroniCity components, namely the Orion Context Broker and PEP proxy Wilma Plus, and the blockchain that enable the feature described above are captured in Figure 6. The arrows representing a *request data access* and the consequent *redirect access request* are shown in dashed lines because they are not needed in all the cases (e.g., subscriptions).

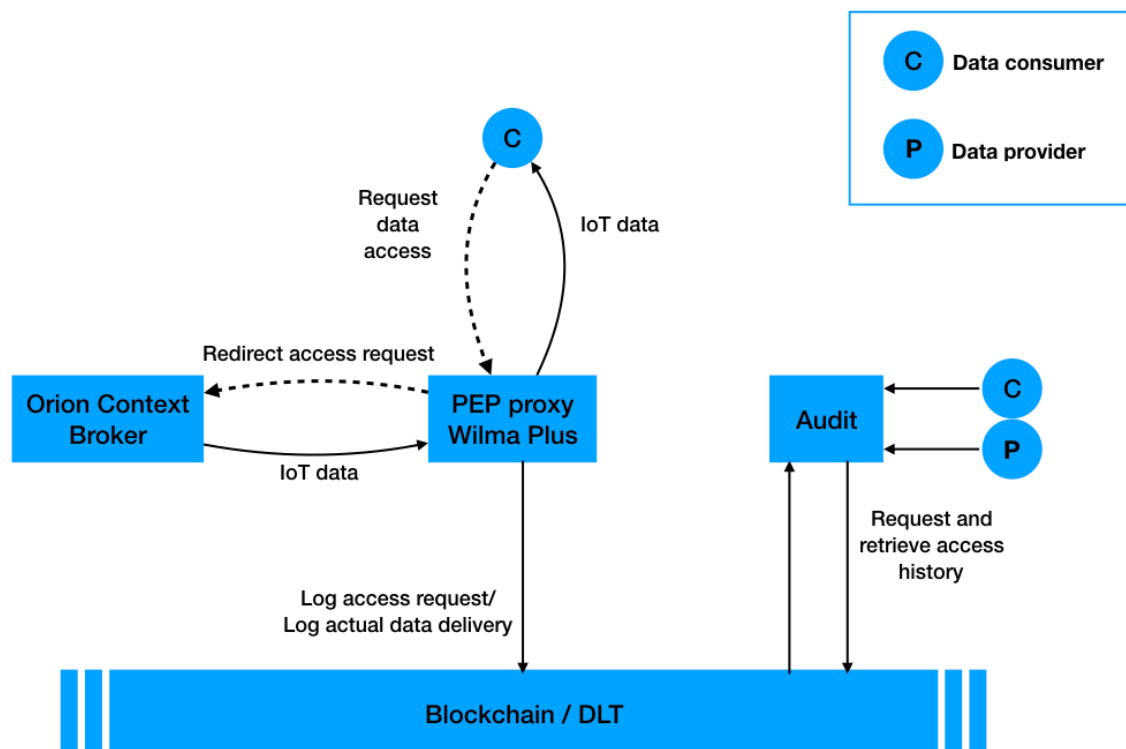


Figure 6. Interactions between SynchroniCity components and the blockchain enabled by the transparency and accountability service.

2.4 Implementation details

This section describes in more detail the blockchain application that implements the features of the transparency and accountability service. We first describe briefly the technology used to create the application and motivate our choice. Then we describe how we implemented the storage of agreements on the blockchain and how the blockchain is used to create an immutable audit trail, which enables to track adherence to the SLA.

2.4.1 Hyperledger Fabric

Hyperledger Fabric [2] is a platform for distributed ledger solutions underpinned by a modular architecture delivering high degrees of confidentiality, resiliency, flexibility, and scalability. It is designed to support pluggable implementations of different components and accommodate the complexity and intricacies that exist across the economic ecosystem.

At the heart of a blockchain network is a distributed ledger that records all the transactions that take place on the network. The information stored on a blockchain is append-only, using cryptographic techniques that guarantee that once a transaction has been added to the ledger it cannot be modified. This property of *immutability* makes it simple to determine the provenance of information because participants can be sure information has not been changed after the fact.

The information of the ledger is synchronized across network nodes using a consensus mechanism. We chose Hyperledger Fabric as the technology to implement our blockchain application because it offers a private and permissioned ledger, where read and write accesses can be restricted and the members of a Hyperledger Fabric network enroll through a trusted Membership Service Provider (MSP).

Hyperledger Fabric has a ledger subsystem comprising two components: the world state and the transaction log. Each participant has a copy of the ledger to every Hyperledger Fabric network they belong to. The world state component describes the state of the ledger at a given point in time. It is the database of the ledger. The transaction log component records all transactions that have resulted in the current value of the world state; it is the update history for the world state. The ledger, then, is a combination of the world state database and the transaction log history.

In Hyperledger Fabric, *chaincode* is the *smart contract* that can be invoked by an application external to the blockchain when that application needs to interact with the ledger. In most cases, chaincode interacts only with the database component of the ledger, the world state (querying it, for example), and not the transaction log.

Depending on the needs of a network, participants in a Business-to-Business (B2B) network might be extremely sensitive about how much information they share. For other networks, privacy will not be a top concern. Hyperledger Fabric supports networks where privacy is a key operational requirement (using channels) as well as networks that are comparatively open.

Transactions must be written to the ledger in the order in which they occur, even though they might be between different sets of participants within the network. For this to happen, the order of transactions must be established and a method for rejecting bad transactions that have been inserted into the ledger in error (or maliciously) must be put into place. This is a thoroughly researched area of computer science, and there are many ways to achieve it, each with different trade-offs. For example, Practical Byzantine Fault Tolerance (PBFT) can provide a mechanism for file replicas to communicate with each other to keep each copy consistent, even in the event of corruption.

Hyperledger Fabric has been designed to allow network starters to choose a consensus mechanism that best represents the relationships that exist between participants. As with

privacy, there is a spectrum of needs; from networks that are highly structured in their relationships to those that are more peer-to-peer.

In our blockchain application, the peer network is formed by the cities that participate in the marketplace. Each transaction has to be verified by all peers in the network before it (and any assets created during the transaction) are stored on the ledger.

2.4.2 Blockchain application and integration details

The Hyperledger Fabric application is comprised of two main parts: the first is the model that defines the different participants taking part in the blockchain network, all the assets that can be stored on the blockchain using the application and the allowed transactions and the required data needed to carry out the transactions. The model is written in the *Hyperledger Composer* modeling language.

- **Participants:**
 - o *Data provider*: a marketplace participant that shares data through the marketplace. Only data providers can specify the agreements (including SLA and license) that have to be signed by data consumers.
 - o *Data consumer*: a marketplace participant that discovers, acquires access to and consumes data through the marketplace. Before acquiring access, the data consumer has to sign the agreement laid out by the data provider.
- **Assets:**
 - o *Signed agreement*: this asset contains the information about the data license and SLA attached to a data offering and the signature of both the data provider that owns the data and the data consumer that acquired access rights to the data.
 - o *Data access*: this asset logs the access requests a data consumer makes to the data and the relevant information associated with the data delivery in order to assess adherence to the SLA.
- **Transactions:**
 - o *Sign agreement*: this transaction takes the signatures of data provider and data consumer as well as the SLA and license and stores the signed agreement on the blockchain.
 - o *Register data access*: this transaction takes a reference to the data consumer and the metrics defined by the SLA (delay, response time, update rate) and stores it as a *data access* asset on the blockchain.

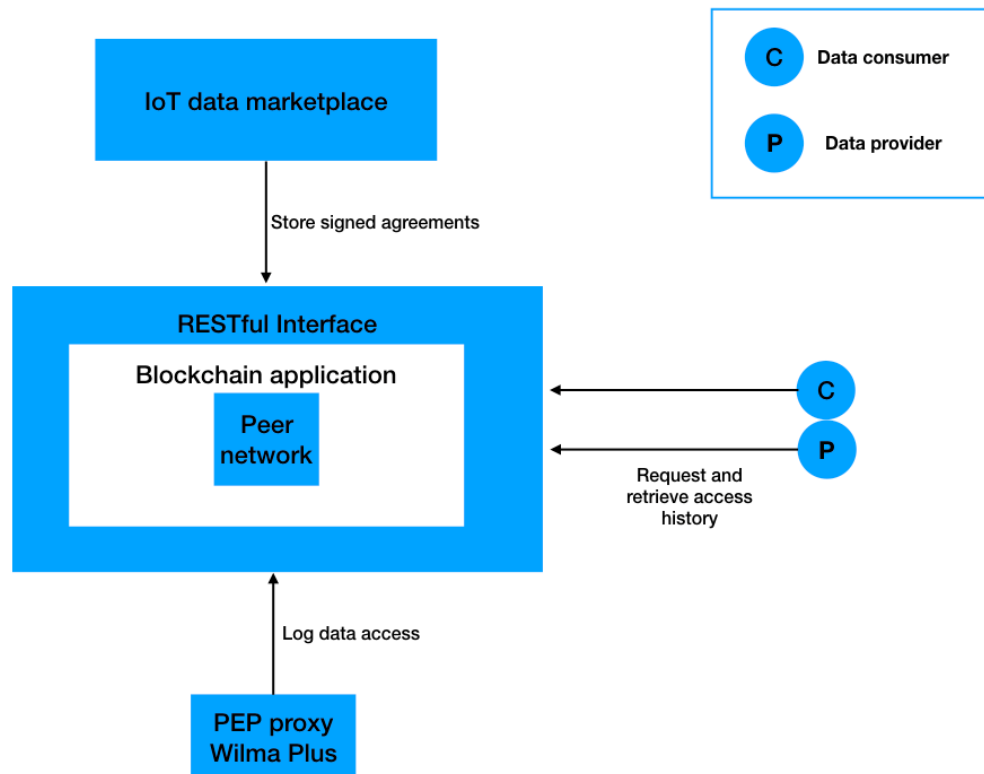


Figure 7. Interactions of the blockchain application with the marketplace portal and the PEP proxy Wilma Plus through RESTful interface.

The second part is the chaincode, which is written in NodeJS. Here the actual application logic is defined. Each function corresponds to a transaction, which was defined in the Hyperledger Composer modeling language.

Figure 7 shows the interactions of the blockchain application with the marketplace portal and the PEP proxy Wilma Plus. The integration is realized by exposing a RESTful interface. It exposes the sign agreement transaction to the in order to finalize agreements between data providers and data consumers.

The register data access transaction is exposed through the RESTful interface to the PEP proxy, every time a data consumer queries the data.

Furthermore, the RESTful interface allows everyone to see the history of the behavior of marketplace participants, by querying the data access assets that are stored on the blockchain, which creates a fully transparent reputation system.

2.5 Federation management

Federation among closed data deployed in multiple RZs is still not possible within the current SynchroniCity framework because user identities are managed locally by the RZs. However, we have integrated the IoT data marketplace with Idra – Open data federation platform [3] that allows it to expose open datasets in the catalog and create related data offerings regardless of the technology of the original open data portal/repository.

Idra is a web application able to federate existing Open Data Management Systems (ODMS) catalogs, based on different technologies (CKAN [4], DKAN [5], Socrata [6], Orion Context Broker), providing a unique access point to search and discover open datasets coming from

heterogeneous sources. Idra levels out the representation of collected open datasets by adopting international standards (DCAT-AP [7]) and provides a set of RESTful APIs to be used by third party applications. Idra's goal is to look for a wider solution to the main problems deriving from the growing interest in open data: from easier access to interoperability and support for publishing data, to an efficient and constructive exploitation for citizens and developers.

The Data Catalog Vocabulary (DCAT) [8] is a standard RDF vocabulary, issued by the W3C in 2014, designed to facilitate interoperability between catalogs of data published on the Web. It defines, through well-defined ontologies (e.g., FOAF [9], SKOS [10], vCard [11], DCMi [12]), the metadata representation related to datasets in a catalog and the methods for accessing data. DCAT allows publishers to increase the detection and enable applications to consume metadata from multiple catalogs, by performing a federated search between the various sites. A dataset in DCAT is defined as a *data collection, published or edited by a single agent and available for access or download in one or more formats*. A dataset should not be available as a downloadable file, which instead is represented by the distribution. DCAT-AP is a specification, based on the DCAT vocabulary, for the description of datasets related to the public sector in Europe. The main objective is to make public sector data better searchable, between different sectors and national borders.

Idra does not directly hold open data resources (data files). The imported metadata contain only direct links to access or download the URL of the resource itself, which remains on the ODMS node.

Idra functionalities can be also accessed by a generic external system, a client application such as the SynchroniCity IoT data marketplace, by using the available APIs [13]. Specifically, the End User API [14] allows to perform queries among federated datasets, retrieving related metadata both in JSON format and in several DCAT-AP serialization formats. These APIs are part of the Data Storage API of the SynchroniCity platform.

To summarize, the following functionalities are exposed:

- **Get Catalogs:** returns the paginated list of the active ODMS Catalogs.
- **Get Catalog:** returns the details of a specific ODMS Catalogue by its ID.
- **Metadata Search:** performs the federated search on all the dataset metadata belonging to the federated ODMS catalogs and returns the results as a JSON payload. It is possible to specify several options, such as:
 - o ODMS Catalog to search on (one or many).
 - o DCAT fields to search on, such as title, description, author, license.
 - o Whether the search is multilingual, by using translated filter values, retrieved from the EuroVoc [15] thesaurus.
 - o DCAT Categorization: datasets can be organized and retrieved by selecting the DCAT *Theme* metadata.
- **Metadata Search DCAT-AP:** returns the same results as the previous API but serialized according to DCAT-AP profiles and serialization formats:
 - o Profiles: metadata fields of resulting datasets can be returned both in DCAT-AP and DCAT-AP_IT profiles.
 - o Formats: datasets payload, according to the selected profile, can be represented in different RDF serialization syntaxes, such as RDF/XML [16], JSON-LD [17], Turtle [18], N3 [19].

The integration of Idra with the SynchroniCity IoT data marketplace is feasible by leveraging the RESTful API services exposed by both platforms. This way, the marketplace is able to import datasets metadata in its catalog and expose open datasets as data offering. Figure 8 shows how datasets metadata are retrieved from the federated ODMS catalogs and imported, through the Platform API, into the marketplace.

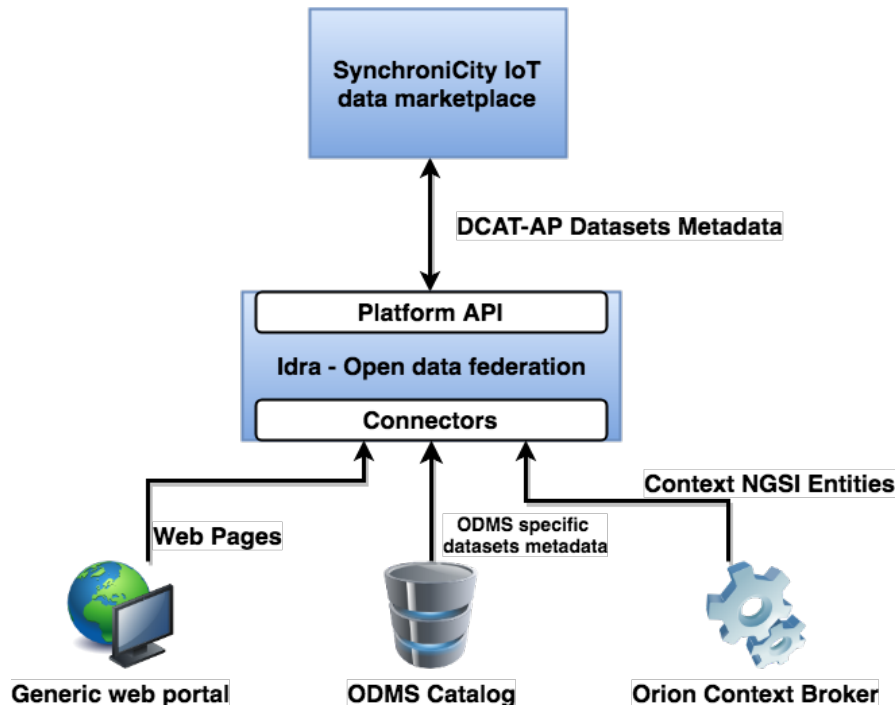


Figure 8. Overview of the integration between Idra and the SynchroniCity IoT data marketplace.

Figure 9 details the interaction flow between Idra and SynchroniCity IoT data marketplace when importing datasets into that.

In a preliminary federation phase, the Idra Administrator federates a new catalog, then datasets metadata are retrieved, mapped and translated to the internal DCAT-AP representation, through dedicated connectors. Thus, the datasets are stored to the internal database and indexed in a dedicated cache (Apache SOLR Server [20]), in order to respond very quickly during metadata searches.

In a second phase, the Marketplace Administrator request to retrieve a subset of federated dataset from Idra. Platform APIs are called, in order to (optionally) retrieve the list of federated Catalogs and then to perform metadata search. To search among datasets, the Marketplace Administrator specifies the metadata filters to be used as search criteria. Idra returns the resulting datasets in the specified format and profile.

These interactions allow to import open datasets into the marketplace, by searching on specific fields and categories and then receiving all the metadata fields, including the dataset permalinks. In particular, the connection between the marketplace and Metadata Search API enables to define automatically a data source specification for each subset of resulting datasets. Once the specification is created, the Marketplace Administrator will be able to define a data offering, which will be the asset exposed in the catalog.

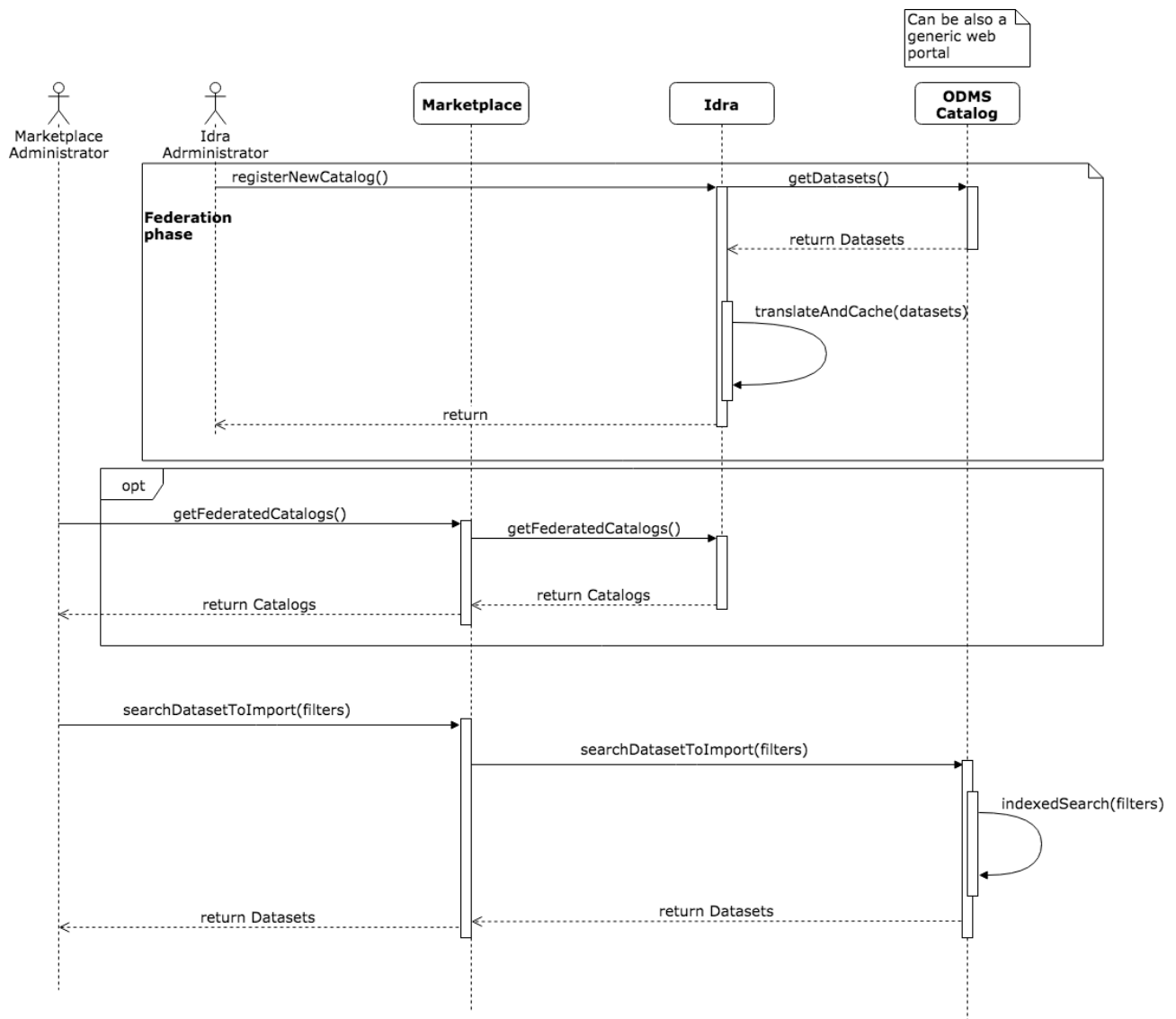


Figure 9. Sequence diagram: Import datasets into the marketplace.

3 Validation

A first validation of the basic enablers included in the initial PoC prototype of the IoT data marketplace has been presented in D2.4. In this section, we present the current PoC prototype of the marketplace, including the advanced enablers and validate some of the new features using real data deployed in Manchester and Santander RZs.

As part of this validation, we have extended the PoC prototype presented in D2.4 by integrating it with Manchester RZ's IoT infrastructure. It is worth noting that the prototype does not represent a federated marketplace even if for its development we have used the federation feature provided by the Orion Context Broker for the open data. In fact, as already stated in Section 2.5, a real federation among marketplaces is not possible yet as the SynchroniCity framework does not rely on a single IdM and therefore only open data can be federated.

In D2.4, we have shown how a data consumer creates a data offering and how a data consumer purchases a data offering. The aim of the validation in this Deliverable is to show how the transparency and accountability service allows for storing the agreements associated with a data offering in a blockchain after a data consumer purchases the offering.

3.1 Data sets and data models

Santander and Manchester RZs' IoT facilities are based on real IoT deployment in urban settings. They target different key areas of the city infrastructure, ranging from public transport, logistics facilities, public places and buildings, environment, workplaces and residential areas, thus creating the basis for development of a smart city. These deployments exhibit the diversity, dynamics and scale that are essential for the evaluation of advanced protocol solutions.

Santander RZ's IoT data is available in several formats, including SynchroniCity data models [21] based on OMA NGSI meta-model and offered through an NGSIv2 RESTful API. In our PoC, we have published the following data offerings:

- **Air Quality – SAN.** Collection of air quality data in various locations formatted according to the *AirQualityObserved* data model.
- **Noise Level – SAN.** Noise level data in Calle Albericia formatted according to the *NoiseLevelObserved* data model.
- **Parking – SAN.** Number of parking spots available in the Carrefour market. Data formatted according to the *OffStreetParking* data model.

Manchester RZ's IoT data deployment has been adapted to be SynchroniCity compliant by converting the original data models in the SynchroniCity ones. In our PoC, we have published the following data offerings:

- **Air Quality – MAN.** Collection of air quality data in various locations formatted according to the *AirQualityObserved* data model.
- **Traffic Flow – MAN.** Traffic counting for cars at Oxford Street and Whitworth Street formatted according to the *TrafficFlowObserved* data model.

- **People Counting – MAN.** Count of people at Portland Street. Data formatted according to the CrowdFlowObserved data model.

3.2 Marketplace in action

Figure 10 and Figure 11 show the lists of data offerings published in the marketplace filtered by the catalog. In the current prototype, we have created two catalogs, one for Santander RZ and one for Manchester RZ. The data offerings published in the two catalogs refer to real data coming from IoT devices deployed in the two RZs. The user can easily browse through the different catalogs and also filter the data offerings according to them. As stated above, we have published three data offerings in each of the catalogs.

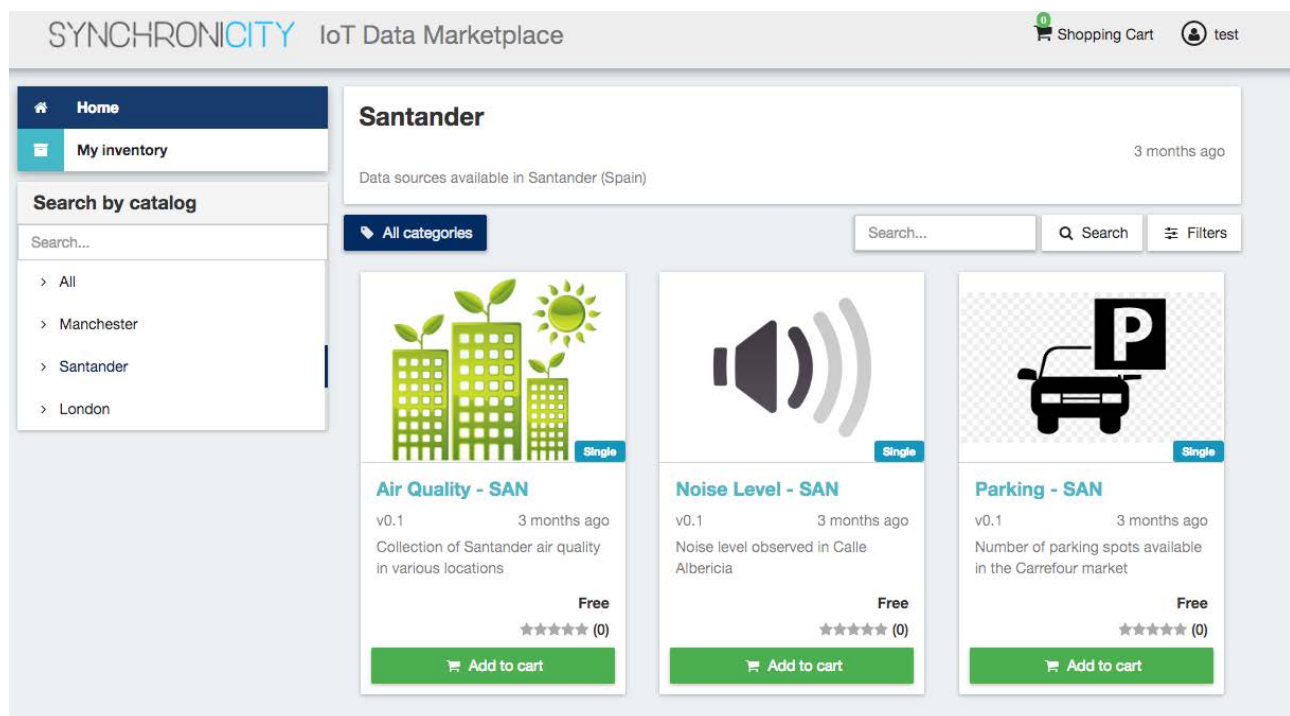


Figure 10. Marketplace portal – Catalog Santander.

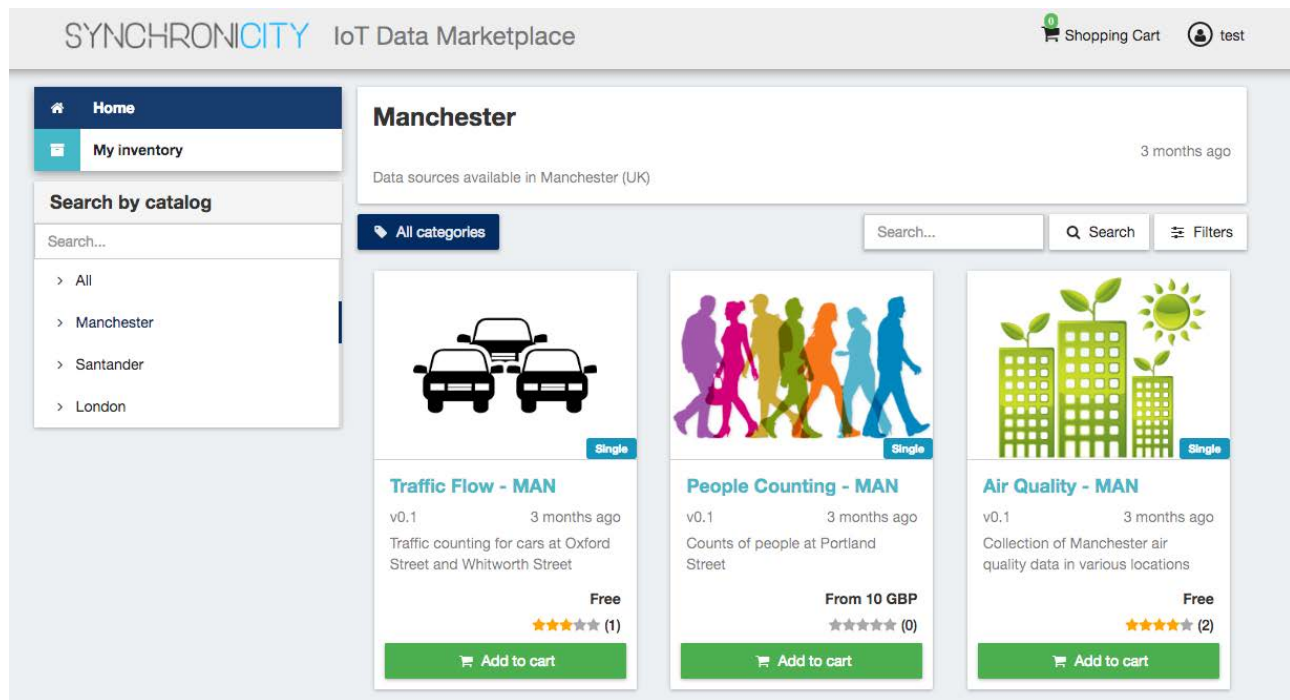


Figure 11. Marketplace portal – Catalog Manchester.

By clicking on each of the data offerings, the user can retrieve the information associated to them. Besides the main characteristics and the license agreements already shown in our first validation in D2.4, a new tab now shows the SLA set for the data offerings (Figure 12).

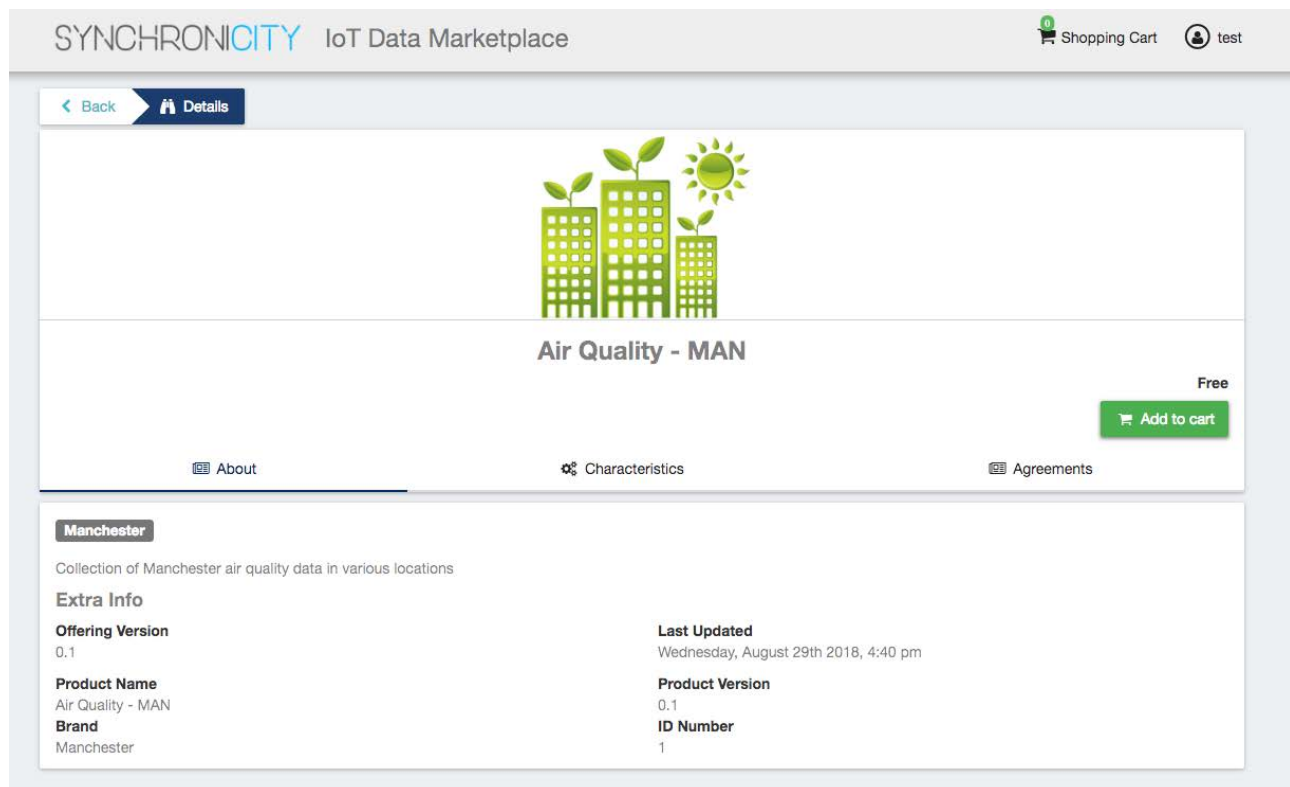


Figure 12. Details page of a data offering.

The marketplace is connected to a blockchain network of two nodes, one for Manchester RZ and one for Santander RZ. The integration with the Hyperledger Fabric application is realized through a simple REST interface (see Figure 13). When an offering is acquired on the marketplace, a REST call carries out the sign agreement transaction. The post data contains the URL of the data offering purchased, the data consumer and the data provider itself. By calling the function, the agreement is signed by both the data consumer and the data provider. This way, participants can be held accountable in the future to adhere to the signed agreement, as it is stored on the blockchain in a tamper-proof way.



Figure 13. REST interface of Hyperledger Fabric.

Figure 14 shows the output logs of the two blockchain nodes which allows us to see behind the curtain of what happens when signed agreement is stored on the chain. The sign agreement transaction triggers the consensus mechanism of hyperledger. When the two blockchain nodes (Manchester and Santander) are receiving the new transaction, they are verifying that the signatures are corresponding to the correct parties involved in buying/selling the data offering. Only after a consensus is reached, the ledger is updated with the newly signed agreement stored tamper-proof on the blockchain.

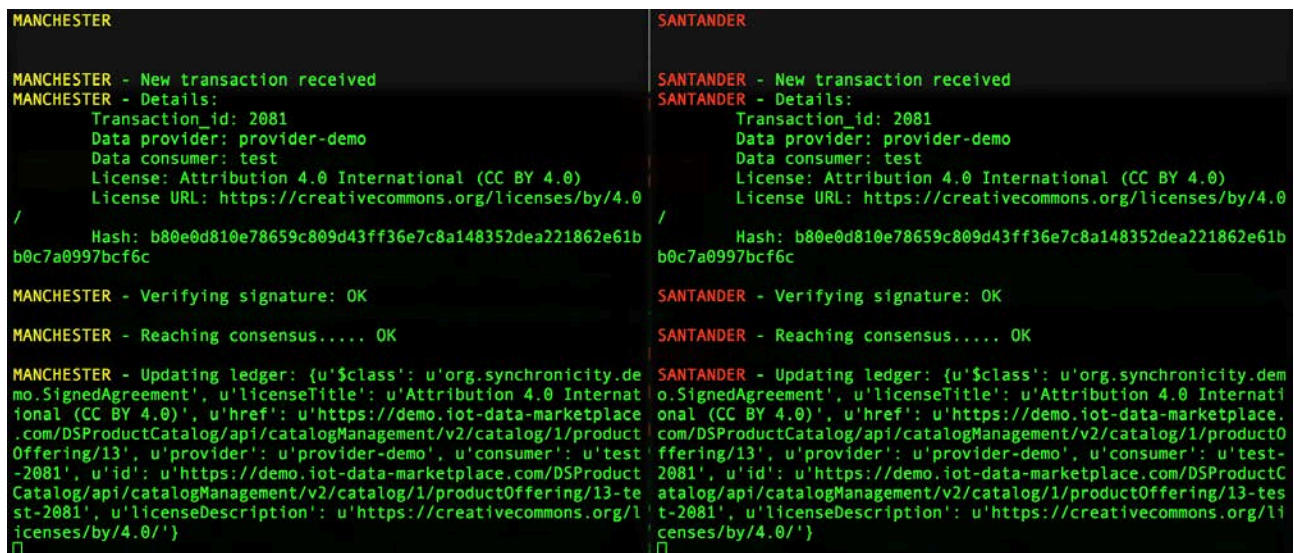


Figure 14. Output logs of the blockchain nodes.

4 Conclusion

In this deliverable, we have presented the advanced data marketplace enablers that extend the basic enablers – introduced in D2.4 – and complete the picture of the SynchroniCity IoT data marketplace platform. The latter is a key element of the SynchroniCity ecosystem transaction management and is complemented by the marketplaces for hardware and services. Together, they underpin the SynchroniCity's vision to create a global market for IoT-enabled urban services with standardized interfaces and common information models and is aligned with the principles defined by the Open & Agile Smart Cities (OASC) community, which already now encompasses more than 100 cities across the world.

We described each of the new individual components, including a detailed API specification in the Appendix, and illustrated the added value they bring to the platform. Together, these bring features that in their entirety were not available like this in one data marketplace. These new enablers are:

- **SLA and license management** – Building on the definitions of license agreements we presented in D2.4 we have finalized the SLA and license management. It allows data providers and data consumers to agree on metrics that guarantee a level of quality of the data as well as defining the conditions for the consumption of the data by the data license attached to an offering.
- **Feedback and reputation** – A data marketplace needs mechanisms to increase the confidence of the potential consumers to the available data offerings. A feedback and reputation service incentivizes data providers to provide the highest quality data and ensures that consumers can make informed decisions what data set to acquire.
- **Transparency and accountability service** – A key element of the advanced enablers is the transparency and accountability service, a component that allows to create a visible and immutable audit trail of the agreements between data providers and data consumers as well as to track the adherence to SLA during data provisioning. The goal of this component is to promote fair behavior among the platform users as they are incentivized to build up a good reputation naturally.
- **Federation management** – Federation among closed data deployed in multiple RZs is still not possible within the current SynchroniCity framework because user identities are managed locally by the RZs. However, we have integrated the IoT data marketplace with an open data federation platform that that allows to easily import open data into the marketplace and vice versa.

We validated a new PoC prototype of the marketplace – which includes the new enablers as well as data from Manchester and Santander RZ's IoT infrastructures – by showing how the transparency and accountability service allows to store the agreements associated with a data offering in a blockchain after a data consumer purchases the offering.

The advanced enablers provide unique features to the marketplace platform and we believe they can foster the establishment of a trusted and transparent ecosystem that should eventually attract stakeholders and grow the business around the exchange of IoT data.

5 References

1. Guillaume Gailard, Dominique Barthel, Fabrice Theoleyre, and Fabrice Valois. SLA Specification for IoT Operation – The WSN-SLA Framework. [Research Report] RR-8567, INRIA. July 2014.
2. E. Androulaki et al. Hyperledger fabric: a distributed operating system for permissioned blockchains. In Proceedings of the Thirteenth EuroSys Conference (EuroSys 2018). New York, NY, USA. 2018.
3. Idra - Open Data Federation Platform. Web page – <https://github.com/OPSILab/Idra>. Accessed 5 Nov. 2018.
4. CKAN. Web page – <https://ckan.org>. Accessed 5 Nov. 2018.
5. DKAN. Web page – <https://getdkan.org>. Accessed 5 Nov. 2018.
6. Socrata. Web page – <https://socrata.com>. Accessed 5 Nov. 2018.
7. DCAT Application Profile for data portals in Europe. Web page – <https://joinup.ec.europa.eu/release/dcat-ap-v11>. Accessed 5 Nov. 2018.
8. W3C. Data Catalog Vocabulary (DCAT). Web page – <http://www.w3.org/TR/vocab-dcat>. Accessed 5 Nov. 2018.
9. FOAF Vocabulary Specification. Web page – <http://xmlns.com/foaf/spec>. Accessed 5 Nov. 2018.
10. SKOS Simple Knowledge Organization System. Web page – <https://www.w3.org/2004/02/skos>. Accessed 5 Nov. 2018.
11. vCard Ontology - for describing People and Organizations. Web page – <https://www.w3.org/TR/vcard-rdf>. Accessed 5 Nov. 2018.
12. Dublin Core Metadata Initiative. Web page – <http://dublincore.org>. Accessed 5 Nov. 2018.
13. Idra Open Data Apiary. Web page – <https://idraopendata.docs.apiary.io>. Accessed 5 Nov. 2018.
14. Idra - End User Api. Web page – <https://idraopendata.docs.apiary.io/#reference/end-user-api>. Accessed 5 Nov. 2018.
15. EuroVoc Thesaurus. Web page – <http://eurovoc.europa.eu>. Accessed 5 Nov. 2018.
16. RDF 1.1 XML Syntax. Web portal – <https://www.w3.org/TR/rdf-syntax-grammar>. Accessed 5 Nov. 2018.
17. JSON for Linking Data. Web page – <https://json-ld.org>. Accessed 5 Nov. 2018.
18. RDF 1.1 Turtle. Web page – <https://www.w3.org/TR/turtle>. Accessed 5 Nov. 2018.
19. Notation3 (N3): A readable RDF syntax. Web page – <https://www.w3.org/TeamSubmission/n3>. Accessed 5 Nov. 2018.
20. Apache SOLR. Web page – <http://lucene.apache.org/solr>. Accessed 5 Nov. 2018.
21. SynchroniCity Data Models. Web page – <https://gitlab.com/synchronicity-iot/synchronicity-data-models>. Accessed 01 Nov. 2018.

Appendix

FORMAT: 1A

TITLE: SynchroniCity IoT data marketplace APIs Specification

DATE: 10 November 2017

VERSION: v0.1

APIARY_PROJECT: synchronicityiotdm

SPEC_URL: <https://synchronicityiotdm.docs.apiary.io/#>

SynchroniCity IoT data marketplace APIs (new components)

The SynchroniCity IoT data marketplace Open API is an extension of the Business API Ecosystem made up of the FIWARE Business Framework and a set of APIs (and its reference implementations) provided by the TMForum

(<https://catalogue.fiware.org/enablers/business-api-ecosystem-biz-ecosystem-ri>).

This set of APIs allows the monetization of digital assets during the whole service life cycle, from offering creation to its charging, accounting and revenue settlement and sharing.

It exposes functionalities such as catalog management, ordering management, federation management, revenue management, customer management, SLA and license management, feedback and reputation, and transparency and accountability service.

In this Appendix, we report the specification of the APIs for the new components included in the advanced marketplace enablers.

Editors

- Angelo Capossele, Digital Catapult
- Andrea Gaglione, Digital Catapult
- Daniel Puschmann, Digital Catapult

Copyright

- Copyright © 2017 - 2018 by Digital Catapult

Specification

Authentication

Each HTTP request against the SynchroniCity IoT data marketplace Open API requires the inclusion of specific authentication credentials.

The specific implementation of this API may support multiple authentication schemes (OAuth, Basic Auth, Token) and will be determined by the specific provider that implements the data marketplace.

Please contact the provider to determine the best way to authenticate against this API. Remember that some authentication schemes may require that the API operates using SSL over HTTP (HTTPS).

The reference implementation of the SynchroniCity IoT data marketplace Open APIs provides support for Cookie, and OAuth2 authentication.

Synchronous Faults

Error responses will be encoded using the most appropriated `content-type` in base to the `Accept` header of the request.

JSON Example

```
{  
  "error": "No JSON object could be decoded",  
}
```

XML Example

```
<?xml version="1.0" encoding="utf-8"?>  
<error>No JSON object could be decoded</error>
```

Used HTTP Codes

HTTP Code	Type	Description
200	OK	Your request has been completed properly
201	Created	Your resource has been created.
204	No content	Your request has been processed, but a response is not available. Generally used when deleting entities
400	Bad Request	The content of your request is not correct
401	Unauthorized	You are not logged in or the Authorization token you are providing is not valid

403	Forbidden	You have no rights to perform the request
404	Not Found	The resource you are looking for does not exists
409	Conflict	The resource you are trying to create already exists
422	Unprocessable Entity	The entity included in the request cannot be processed (e.g., it includes invalid fields)
500	Internal server error	There was an internal error in the system so your request cannot be completed

Group SLA Specification Management API

API for the management of SLA Specifications. A SLA Specification is a detailed description of the Service Level Agreement related to a particular data offering available in the marketplace.

The SLA Specification Management API uses the following fields:

- **id** - Unique identifier of the SLA specification
- **offerId**: The id of the corresponding offer that the SLA has been attached to
- **description** - Narrative text that explains the body of the SLA specification
- **services** - List of metrics that have been defined for the SLA specification. They contain the following fields:
 - **type**: The type of the metric (can be one of the following values: "Update rates", "Response time" or "Delay")
 - **threshold**: The threshold value set for the metric
 - **unitMeasure**: The unit of measurement for the threshold value
 - **description**: Narrative text that explains the metric

Create SLA Specification [POST]

- Request (application/json)
 - Headers
 - Authorization: Bearer YOUR_OAUTH2_TOKEN
 - Body:

```
{
  "offerId": "223",
  "description": "This Service Level Agreement defines response time and update frequency",
  "services": [{
    "type": "Updates rate",
    "description": "Expected number of updates in the given period.",
    "threshold": "100",
    "unitOfMeasure": "ms"
  }, {
    "name": "Response time",
    "description": "Total amount of time to respond to a data request (GET).",
    "threshold": "100",
    "unitOfMeasure": "pkt/day",
  }]
}
```

- Response 201

SLA Specification Entry

[/DSLALicenseManagement/api/SLAManagement/v1/SLASpecification/{id}]

Get SLA Specification [GET]

- Parameters
 - id: 52 - Id of the SLA specification to be retrieved
- Request
 - Headers
 - Authorization: Bearer YOUR_OAUTH2_TOKEN
- Response 200 (application/json):

```
{
  "id": "5bf690e6bcd245e683d555e0",
  "offerId": "233",
  "description": "This Service Level Agreement defines response time and update frequency",
  "services": [{
    "type": "Updates rate",
    "description": "Expected number of updates in the given period.",
    "threshold": "100",
    "unitOfMeasure": "ms"
  }, {
    "name": "Response time",
    "description": "Total amount of time to respond to a data request (GET).",
    "threshold": "100",
    "unitOfMeasure": "pkt/day",
  }]
}
```

Update SLA Specification [PATCH]

- Parameters
 - id: 5bf690e6bcd245e683d555e0- Id of the SLA specification to be updated
- Request - Partial update of the license specification, only the fields to be updated need to be provided - (application/json)
 - Headers
 - Authorization: Bearer YOUR_OAUTH2_TOKEN

○ Body:

```
{
  "services": [{
    "type": "Updates rate",
    "description": "Expected number of updates in the given period.",
    "threshold": "500",
    "unitOfMeasure": "ms"
  }, {
    "name": "Response time",
    "description": "Total amount of time to respond to a data request (GET).",
    "threshold": "300",
    "unitOfMeasure": "pkt/day",
  }]
}
```

- Response 200 (application/json):

```
{
  "id": "5bf690e6bcd245e683d555e0",
  "offerId": "233",
  "description": "This Service Level Agreement defines response time and update frequency",
  "services": [{
    "type": "Updates rate",
    "description": "Expected number of updates in the given period.",
    "threshold": "500",
    "unitOfMeasure": "ms"
  }, {
    "name": "Response time",
    "description": "Total amount of time to respond to a data request (GET).",
    "threshold": "300",
    "unitOfMeasure": "pkt/day",
  }]
}
```

Group Feedback and Reputation Management API

API for the management of reputation scores. It is a simple five-star rating system according to which data consumers can rate data offerings on the basis of a scale from 1 to 5. Data consumers are allowed to add a comment along with the rating score and they are also able to change their rating at a later stage.

The Feedback and Reputation Management API uses the following fields for an aggregated rating:

- **id** - Unique identifier of the reputation score
- **offerId** - The id of the corresponding data offering that the reputation score has been set to
- **avg** - Average reputation score
- **count** - Number of times a reputation score has been given to the specified data offering
- **comments** - List of comments submitted by the data consumers with corresponding consumerID. Comments contain the following fields:
 - **comment**: The comment the data consumer submitted
 - **consumerID**: The consumerId of the corresponding data consumer

Create Reputation Score [POST]

- Request (application/json)
 - Headers
 - Authorization: Bearer YOUR_OAUTH2_TOKEN
 - Body:


```
{
    "offerId": "223",
    "consumerId": "c8d86fdd-2c76-44a5-a366-819f1bf998a6",
    "comment": "I always receive my data on time",
    "rating": "5"
  }
```
- Response 201

Get Reputation Score [GET]

- Parameters
 - offerId: 233 - Id of the offering whose aggregated rating is required
- Request
 - Headers
 - Authorization: Bearer YOUR_OAUTH2_TOKEN
- Response 200 (application/json):

```
{
  "offerId": "233",
  "avg": "4.5",
  "count": "12",
  "avg": "4.5",
  "count": "12",
}
```

```
"comments": [{  
  "comment": "I always receive my data on time"  
  "consumerId": "c8d86fdd-2c76-44a5-a366-819f1bf998a6"  
}]  
}
```

Update Reputation Score [PATCH]

- Parameters
 - offerId: 233 - id of the offering that the consumer wants to change the rating to
 - consumerId: c8d86fdd-2c76-44a5-a366-819f1bf998a6 – id of the data consumer
- Request - Update a reputation score, only the fields to be updated need to be provided - (application/json)
 - Headers
 - Authorization: Bearer YOUR_OAUTH2_TOKEN
 - Body:

```
{  
  "offerId": "233",  
  "consumerId": " c8d86fdd-2c76-44a5-a366-819f1bf998a6",  
  "comment": "Yesterday did not receive any data",  
  "rating": "1"  
}
```

- Response 204