Project Acronym:	**VICINITY**
Project Full Title:	**Open virtual neighbourhood network to connect intelligent buildings and smart objects**
Grant Agreement:	**688467**
Project Duration:	**48 months (01/01/2016 - 31/12/2019)**

1st Open Call. Technical Details

The VICINITY Consortium
1. TU Kaiserslautern (Coordinator) (Germany)
2. ATOS SPAIN SA (Spain)
3. Centre for Research and Technology Hellas (Greece)
4. Aalborg University (Denmark)
5. GORENJE GOSPODINJSKI APARATI D.D. (Slovenia)
6. Hellenic Telecommunications Organization S.A. (Greece)
7. bAvenir s.r.o. (Slovakia)
8. Climate Associates Ltd (United Kingdom)
9. InterSoft A.S. (Slovakia)
10. Universidad Politécnica de Madrid (Spain)
11. Gnomon Informatics S.A. (Greece)
12. Tiny Mesh AS (Norway)
13. HAFENSTROM AS (Norway)
14. Enercoutim – Associação Empresarial de Energia Solar de Alcoutim (Portugal)
15. Municipality of Pylaia-Hortiatis (Greece)

© Copyright 2018, the Members of the VICINITY Consortium

For more information on this document or the VICINITY project, please contact us in the following link

European Platforms Initiative

Co-founded by the Horizon 2020 programme of the European Union

Executive Summary

The VICINITY is built around the concept of connecting different IoT ecosystems through an open gateway API (providing interoperability as a service[1]) which enables interaction with IoT objects (devices and value-added services[2]) from other different ecosystems **as if they were their own**. The VICINITY interoperability services inter-connecting IoT ecosystems creates a common environment where value-added services utilizing different devices can be deployed **and can operate across different domains** (Figure 1).

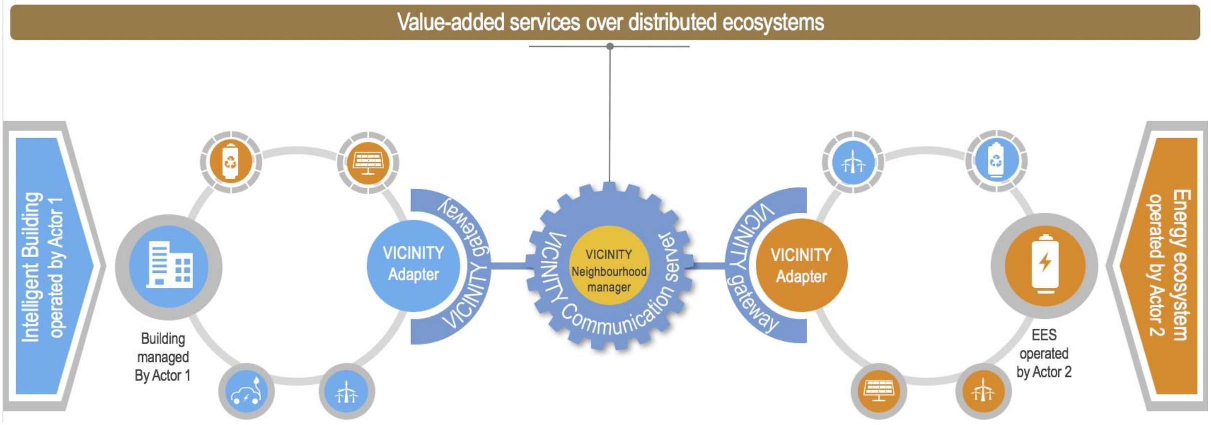
Value-added services over distributed ecosystems
Intelligent Building operated by Actor 1
Building managed By Actor 1
VICINITY Adapter
VICINITY gateway
VICINITY Neighbourhood manager
VICINITY Communication server
VICINITY gateway
VICINITY Adapter
EES operated by Actor 2
Energy ecosystem operated by Actor 2

Figure 1 VICINITY Concept

In Figure 1, two separate ecosystems are presented: intelligent building and energy IoT ecosystems. Each of these ecosystems is integrated with the VICINITY by its VICINITY Adapter through the VICINITY Open Gateway API. Based on the setup of the virtual neighbourhood in VICINITY Neighbourhood Manager, VICINITY Adapters may access remote devices, for example a battery in an Energy ecosystem, and emulate it as a part of their ecosystem. Moreover, IoT objects shared by a VICINITY Adapter within a virtual neighbourhood may be accessed by value-added services to provide cross-domain services using common semantics based on VICINITY ontology (http://vicinity.iot.linkeddata.es/vicinity/ [3]).

To make IoT infrastructure comply with VICINITY (such as building or energy ecosystems presented on previous figure) the following major requirements needs to be fulfilled:

1. map internal information model of devices and services exposed to VICINITY to the VICINITY common format;
2. integrate IoT infrastructure through the VICINITY Adapter to the VICINITY Open Gateway API to expose IoT objects.

An IoT ecosystem provider can receive the following benefits from being VICINITY compliant:

- access shared/exposed devices from other IoT ecosystems in different domains or based on different technologies;

[1] Interoperability as a service translates integrated IoT ecosystem information model into common VICINITY model based on semantic description of connected devices and services.

[2] Value-added service in context of VICINITY is defined as a piece of software that implements an algorithm (from a simple calculation/data processing to some advanced techniques such as clustering/big data analytics, data storage and auditing etc.). These services may provide the User interfaces to end-user in order to view notifications, statistical data, processed data etc. over collected data from the available integrated IoT infrastructure (IoT devices, sensors etc.).

[3] Note, that due implementation stage of the Project and evolution of the VICINITY ontology even after the life time of the project, VICINITY ontology is subject of change.

- expose device capabilities towards cross-domain value-added services to extend the benefits of the connected devices for the end-user(s).

Content

1. VICINITY Architecture

The objective of the VICINITY architecture[4] is to facilitate interoperability between different IoT infrastructures' devices and to software-enable value-added services through a peer-to-peer (P2P) network of VICINITY Nodes. Each VICINITY Node provides access to IoT infrastructure and/or value-added service. Once the IoT infrastructure is integrated into the VICINITY neighbourhood through the VICINITY Node, devices connected to the infrastructure become accessible through the VICINITY Neighbourhood Manager in the VICINITY Cloud. In VICINITY Neighbourhood Manager IoT infrastructure owner can define sharing access rules of devices and service (i.e. has direct control over his or her devices). Based on these rules he or she creates social network of devices and service called "virtual neighbourhood".

The VICINITY Nodes create a controlled VICINITY Peer-to-peer (P2P) Network based on these rules defined by VICINITY Neighbourhood Manager (Figure 2 – yellow and blue arrows) in VICINITY Cloud. In VICINITY P2P Network, VICINITY Nodes communicate user data directly between each other (Figure 2 – red arrows). Moreover, the VICINITY P2P Network support VICINITY Node with security services (such as end-to-end encryption, data integrity, etc.) to ensure security and privacy of exchanged user data.

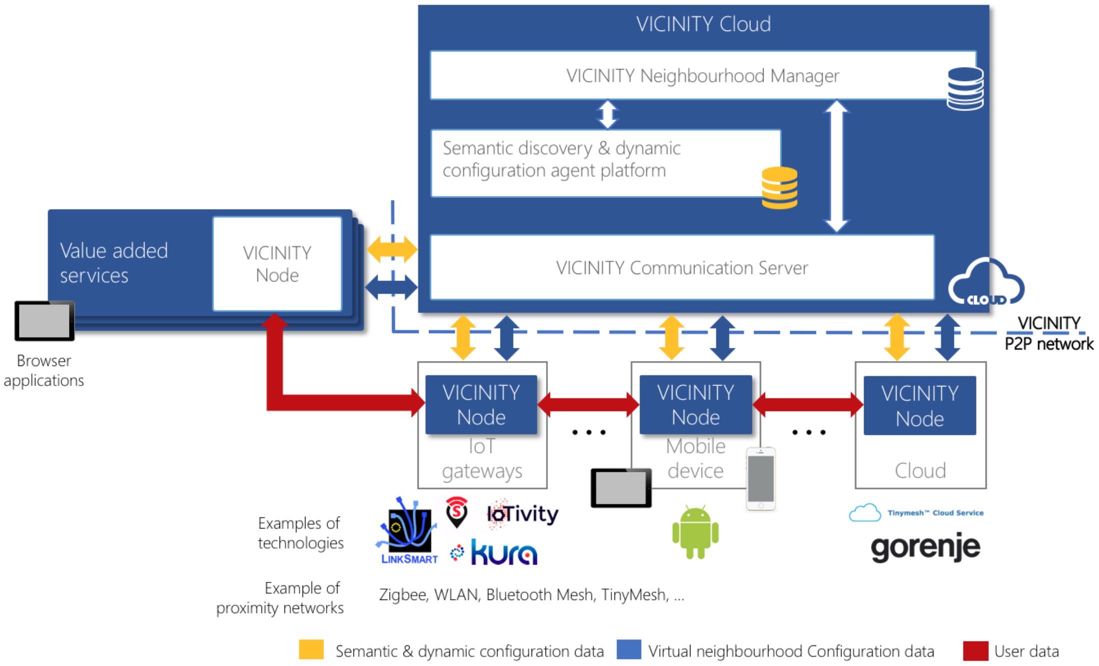
VICINITY Cloud
VICINITY Neighbourhood Manager
Semantic discovery & dynamic configuration agent platform
VICINITY Communication Server
Value added services
VICINITY Node
Browser applications
VICINITY P2P network
VICINITY Node
IoT gateways
VICINITY Node
Mobile device
VICINITY Node
Cloud
Examples of technologies
LinkSmart
IoTivity
kura
Tinymesh Cloud Service
gorenje
Example of proximity networks
Zigbee, WLAN, Bluetooth Mesh, TinyMesh, ...
🟨 Semantic & dynamic configuration data
🟦 Virtual neighbourhood Configuration data
🟥 User data

Figure 2 High-level architecture of the VICINITY

The VICINITY provides semantic interoperability features to facilitate exchange of user data between IoT devices and value-added services to overcome technology differences between each connected IoT ecosystem. Thus, communication with each device or service via VICINITY P2P Network is standardized regardless of the technology the device or service is connected to a VICINITY Adapter. This semantic interoperability approach is based on the work being done by the Web of Things (WoT) WG[5], where a proposal for describing, exposing and consuming *web things* by leveraging Semantic Web technologies is in development. Such web things are things that can be accessed through the Web, either physically or abstract.

[4] For detail description of the VICINITY architecture see: http://vicinity2020.eu/vicinity/content/d16-architectural-design-10

[5] https://www.w3.org/WoT/IG/

One of the pillars of the W3C WoT is the Thing Description (TD), which aims to be a standard frame to support the description of *web things* semantically to make them interoperable. Thus, TDs are expected to cover the following aspects:

- Semantic meta-data, so to explicitly specify the semantics of a web thing;
- Thing's interaction resources: property, action and event;
- Security including concrete prerequisites to access things are stated;
- Communications, i.e., what kind of protocols and data exchange formats are supported, and which endpoints are exposed to give access to the existing interaction resources of a web thing.

Example of how the integrated IoT infrastructure will be accessed by other peers in VICINITY P2P Network is elaborated in following sections (1.4).

1.1. VICINITY Cloud

The VICINITY Cloud enables IoT infrastructure operators and Service providers to configure a virtual neighbourhood of connected devices and value-added services including the setup of sharing access rules between them through the user-friendly interface of VICINITY Neighbourhood Manager. Configuration of the virtual neighbourhood and sharing access rules are used by VICINITY Communication Server to setup communication channels between each VICINITY Node to control exchange of user data. IoT infrastructure operators and Service providers can search for devices and services in virtual neighbourhood based on semantic description of device properties, actions, events and service products & required inputs stored in semantic repository. Moreover, IoT operators, System integrators and Services providers can register the VICINITY Nodes (registration of the application API) to communication in peer-to-peer.

1.2. VICINITY Node

A VICINITY Node is the set of software components which maintains the user data exchange between peers in the VICINITY P2P network based on configuration of the virtual neighbourhood and sharing rules received from VICINITY Communication Server. For that purpose, VICINITY Node consists of the following 3 main components:

- VICINITY Gateway API and Communication Node;
- VICINITY Agent;
- VICINITY Adapter.

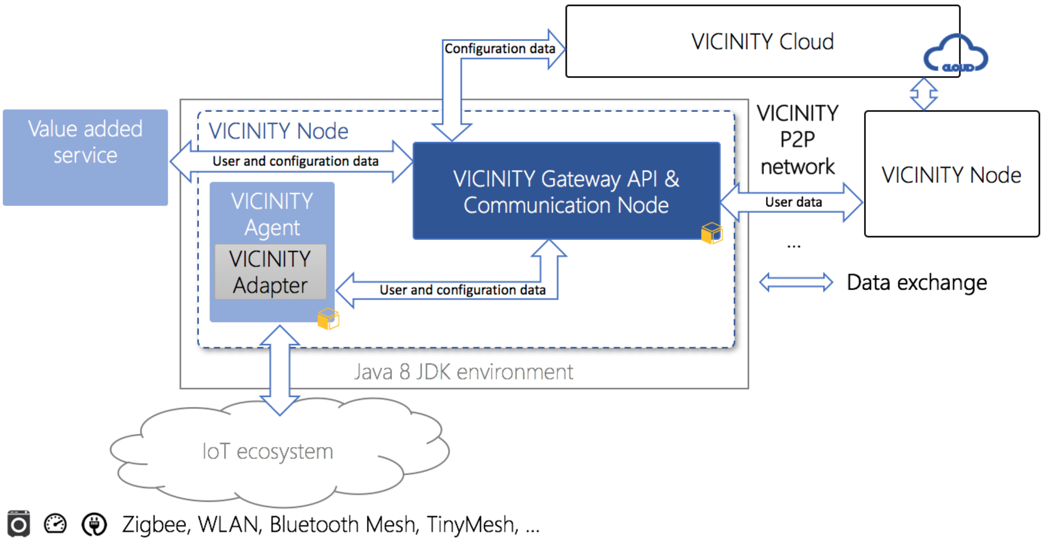

Figure 3 Logical architecture of the VICINITY Node

The *VICINITY Gateway API and Communication Node* service provides the following set of JSON HTTP REST[6] services in the VICINITY common format and data model:

- Registry service to register devices and value-added services using their simple semantic description as JSON document in VICINITY;
- Search service in virtual neighbourhood of connected IoT infrastructures and value-added services;
- Services to read/write properties, performing actions and process events from shared devices device/service from connected IoT infrastructure or value-added service;
- Expose properties, actions and events of device/service to VICINITY through simple REST API;
- Authentication services for the IoT infrastructure (authentication of the application) and devices and value-added services;
- Status check service.

The *VICINITY Adapter* is a component provided by IoT infrastructure owner or respective system integrator. The VICINITY Adapter provides simple API, which has to be implemented for every adopted infrastructure. The core responsibilities of the Adapter API are:

- to provide the description of IoT objects (devices, services) of infrastructure in common VICINITY format, which enables VICINITY to create internal models of used IoT objects in uniform way;
- to provide access to properties, actions and events of devices provided by infrastructure.

The purpose of the VICINITY Adapter is to simplify the adoption of IoT infrastructure in as simple way as possible. The Adapter is used just to discover IoT objects in infrastructure and to communicate with IoT objects. Once the IoT infrastructure owner/system integrator provides the Adapter component implementing simple prescribed API, the IoT infrastructure can be easily adopted. The full VICINITY functionality is then managed by VICINITY Agent component.

The *VICINITY Agent* is the wrapper for the VICINITY Adapter. The VICINITY Adapter provides the full VICINITY specific functionality, as managing communication via P2P network, semantic discovery of IoT objects, semantic search of IoT objects, communication with IoT objects within the infrastructure, where each VICINITY specific interaction with IoT objects is translated in the VICINITY Adapter calls.
Simply speaking, the Adapter handles only very basic communication with IoT infrastructure by implementing simple API. Agent manages full VICINITY interactions with IoT infrastructure by translating this interaction into the Adapter API services.

1.3. How to connect IoT infrastructure to VICINITY

The VICINITY support following VICINITY Node deployment scenarios (Figure 4):

- VICINITY Node local IoT gateway deployment – IoT infrastructure A – VICINITY and IoT infrastructure are integrated in location of deployment of infrastructure. For this scenarios software and hardware constraints should apply (see section 2.4);
- VICINITY Node cloud deployment – IoT infrastructure B – VICINITY and IoT infrastructure are integrated on the level of cloud services. For this scenario software constraints should apply (see section 2.4).

[6] VICINITY Gateway API is described using Open API standard and due implementation stage of the VICINITY project it is subject of change, for conceptual understanding of the API see: https://app.swaggerhub.com/apis/intersoft.sk/vicinity-adapter/1.0.0

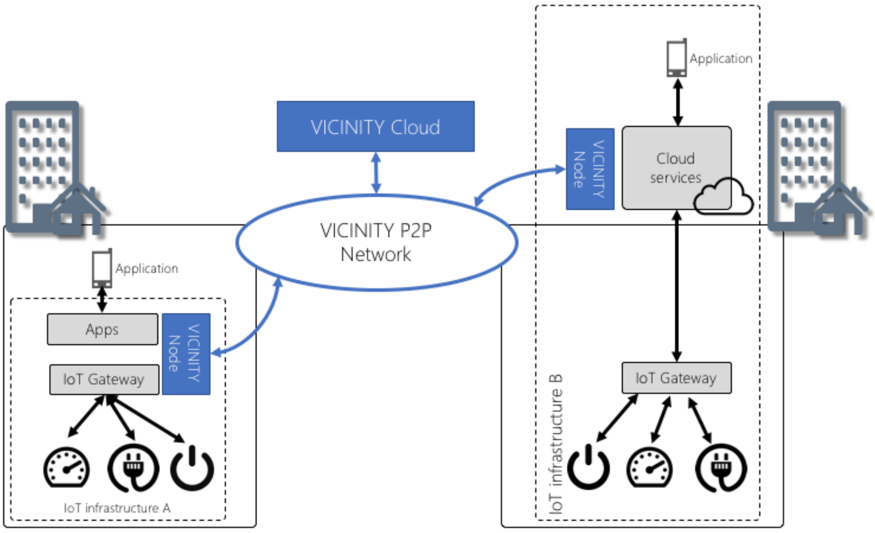

Figure 4 Conceptual approach to connect IoT infrastructure to VICINITY

1.4. How semantic interoperability works in VICINITY

VICINITY builds on W3C WoT standard frame of Thing Ecosystem Description (TED) which describes the set of Thing. The purpose of bringing the TED frame into the VICINITY's interoperability scenario is to support the description of *ecosystems* of Things, i.e., sets of Things that *coexist* in the same environment. In VICINITY, an IoT infrastructure makes up an ecosystem of IoT objects whose *grounding* environment is the infrastructure itself. However, the VICINITY interoperability approach also considers as ecosystems those sets of IoT objects whose common environment is defined by the scope of a query context for discovery and/or accessing. Such ecosystems made up of query-relevant IoT objects are described in what we call Virtual TEDs (VTEDs).

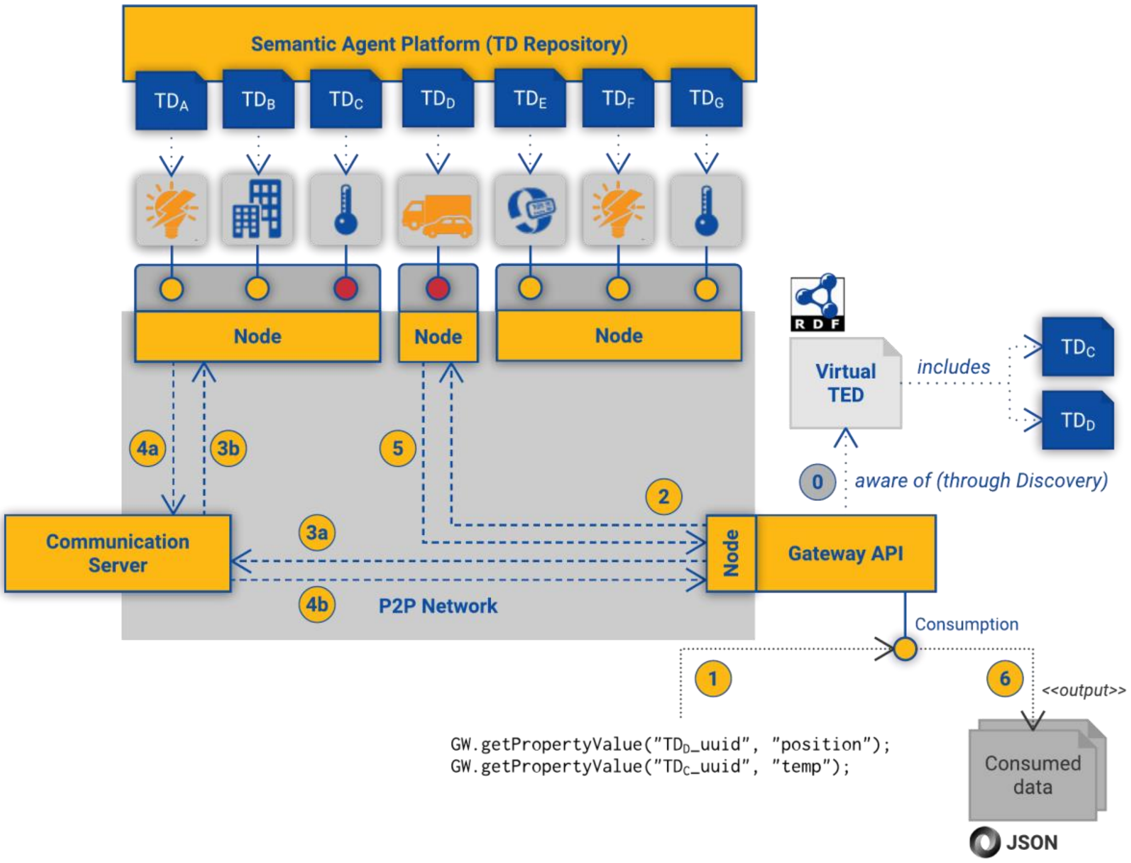

Figure 5 Semantic interoperability approach for Accessing

Figure 5 illustrates an example of how VICINITY supports semantic interoperability for *consumption* of devices from integrated IoT ecosystems to obtain property values. This diagram describes how the approach works by means of a sequence of interactions:

0. Before consuming requests from IoT objects, it is assumed that “Discovery” of IoT objects is performed through the search service of Gateway API. The “Discovery” is performed over semantic description of each IoT object in semantic repository.
1. VICINITY Open Gateway API offers a dedicated interface for Consumption requests, which in the example, receives two *getPropertyValue* commands pointing at different IoT objects (T_C and T_D). It is assumed that the Gateway API was previously given a Virtual TED through a search service of VICINITY Open Gateway API. Thanks to this, the consumer can use the corresponding identifiers of each IoT object and their properties at the time of invoking both commands. Again, the Consumption interface allows consumers to be agnostic of ontologies.
2. The Gateway API processes the first command and sends a message to the Node that hosts the referenced IoT object. In this case, the P2P communication can be directly established between the two involved Nodes.
3. As in the previous step, the Gateway API processes the received command, but in this case, the recipient Node is not directly reachable due to some network constraint in-between. The Communication Server takes care of the issue and makes sure that the message finally reaches its addressed Node.
4. Once the corresponding Gateway API of the recipient Node gets the message, it queries the TED that describes its own infrastructure to determine the specific endpoints that must be invoked. Having all raw data collected from its underlying infrastructure, the recipient Gateway API composes a response message and sends it back to the requester (through the Communication Server).

5. Same process as described in step 4, except for the P2P communication does not require an intermediary.
6. The message from resulted from VICINITY Node in previous steps are provided as result of the consumption request.

The Gateway API processes both incoming response messages separately and applies the corresponding sharing access rules specified in the VICINITY Cloud.

2. Technical requirements for 1st open call

This section defines the following technical requirements which needs to be fulfilled by applicant to make IoT infrastructure compliant to VICINITY and is the part of the technical delivery of the proposal:

1. to recognize the VICINITY Open Gateway API and VICINITY Agent as VICINITY Interface towards integrated IoT infrastructure;
2. to specify device description for each supported device which will be exposed and access through VICINITY and being use as demonstrator of the integration;
3. to implement and integrate the VICINITY Adapter of the integrated IoT ecosystem which will translate ecosystem communication into common VICINITY communication format;
4. to connect real device and IoT infrastructure into VICINITY;
5. to demonstrate accessibility of exposed device through VICINITY.

2.1. Analysis and study of VICINITY Gateway API and integration manuals

The applicant shall:

- study the VICINITY Gateway API and respective integration manual to integrate IoT infrastructure to VICINITY;
- provide intermediate and final technical report providing high-level design of the solution;
- be able to submit relevant bugs and change requests to VICINITY Gateway API[7] through github issue tracking mechanism provided by VICINITY Gateway API consortium.

The VICINITY Gateway API and integration manual will be available to applicant by providing access to the VICINITY Gateway API github repository which includes the VICINITY Gateway API source code, latest builds and documentation[8]. The VICINITY consortium will be collaborative to provide applicant information necessary to understand the VICINITY Gateway API.

2.2. Specify device description templates for each supported device type

The applicant shall:

- **provide description in VICINITY common format of each testing device** to be exposed to VICINITY through the VICINITY Gateway API.

[7] VICINITY project consortium will keep right to reject a change request to VICINITY Gateway API without explanation (mostly in case if the change request is irrelevant or not in-line with VICINITY Gateway API concept).

[8] Current working version of VICINITY Gateway API is available at https://app.swaggerhub.com/apis/voravec/VICINITY/0.1.1. Note, this version of VICINITY Gateway API will be subject of the change in the course of VICINITY Project.

A device description is defined as a simple JSON document including at a least identification of the device and a list of specifications of the interaction patterns with the devices such as: properties, actions and events. The specification of the interaction patterns defines the REST service, where the VICINITY Gateway API will interact with the VICINITY Adapter to reach exposed specific device.

2.3. Implement and integrate the VICINITY Adapter for the connected IoT infrastructure

The applicant shall:

- **implement the VICINITY Adapter** which will translate the IoT infrastructure information model and its interaction with the VICINITY Common format provided by VICNITY Gateway API;
- **integrate VICINITY Gateway API services** to extend the required by interaction nature between IoT ecosystem and value-added services. However, note that, registry, authentication services and status check service are mandatory. For example, for remote access of shared devices through VICINITY, the services to access devices are required. However, on the other hand, in this example integrate exposing service is not required.

Currently the VICINITY Gateway API supports only REST API communication over the HTTP.

2.4. Connect real device and IoT infrastructure into VICINITY

The applicant shall:

- **connect real devices to IoT infrastructure**. If local IoT gateway is used it shall run on standard HW platforms able running Java 8 virtual machine (such as: Raspberry PI, BananaPRO, Cubieboard, PINE64+, Intel Edison, etc.). If integration with cloud service is selected, then VICINITY Gateway API shall run in JAVA 8 complaint environment. The applicant shall provide environment where VICINITY Node is running;
- **connect the IoT infrastructure to VICINITY** through VICINITY Gateway API and VICINITY Adapter;
- **provide support and cooperation** of the connected devices, IoT ecosystems, the VICINITY Adapters throughout the project.

The VICINITY currently supports the VICINITY Adapters or the VICINITY Adapters examples for Kura IoT platform, SiteWhere, IoTivity and LinkSmart and OpenHAB (work is ongoing).

2.5. Demonstrate accessibility of exposed device through VICINITY

The applicant shall **demonstrate proposed accessibility and controllability** of real devices in selected pilot site locations and/or laboratory.