A Graph-Based Cross-Vertical Digital Twin Platform for Complex Cyber-Physical Systems

Thierry Coupaye, Sébastien Bolle, Sylvie Derrien, Pauline Folz, Pierre Meye, Gilles Privat, Philippe Raïpin-Parvedy

A graph-based cross-vertical digital twin platform for complex cyber-physical systems

Thierry Coupaye, *VP Internet of Things Research, Orange Innovation, Grenoble, France*
Sébastien Bolle*, Smart Object Management Program Manager, Orange Innovation, Grenoble, France*
Sylvie Derrien*, Project Manager Thing'in Think, Build, Run & Scale, Orange Innovation, Rennes, France*
Pauline Folz*, Researcher, Orange Innovation, Grenoble, France*
Pierre Meye*, Researcher, Orange Innovation, Rennes, France*
Gilles Privat*, Researcher, Orange Innovation, Grenoble, France*
Philippe Raïpin-Parvedy*, Web of Things Program Manager, Orange Innovation, Rennes, France*

Abstract The intent of this chapter is to demonstrate the value of a *transversal* (i.e. *cross-verticals and multi-actor)* digital twin platform for Internet of Things (IoT) applications and complex cyber-physical systems at large (e.g. large-scale infrastructures such as telecommunication or electricity distribution networks) around the *Thing in The Future* experimental digital twin platform developed at Orange. Several real-life illustrative use cases in various domains — smart building, smart factory, smart city and telecommunication infrastructures — developed by Orange and partners, are introduced. Main design, architectural and technological choices, which sustain this ambition, are discussed: graph-based structural and semantic modelling of systems of systems, large scale graph storage, platform distribution and federation.

***Index Terms*—Digital Twin, Internet of Things, IoT, Systems of Systems, Graph modelling, Graph database, Semantics, Ontology Multisided Platform**

I. Introduction

This chapter reports on experiments with the Orange "*Thing'in the Future"* (*Thing'in* for short in the following) experimental digital twin platform. The intent is to discuss, and hopefully demonstrate the value of a *universal*, *multi* and even *cross-verticals*, multi-actors digital twin platform for Internet of Things (IoT) applications (e.g. in Smart Building, City, Industry, Transports and Logistics, Agriculture…) and complex cyber-physical systems at large (e.g. telecommunication infrastructures).

When looking at the historical emergence of the concept of *digital twin*, essentially in an industrial context associated to computer-assisted design and manufacturing (CAD/CAM) technologies, and at its more recent explosion (cf. other chapters in this book), greatly associated to the advent of the IoT, it is noticeable that most digital twins today are constructed in an ad-hoc fashion, or thanks to specialized products on vertical markets such as manufacturing (e.g. airplanes, automotive, shipbuilding), building construction and technical management (e.g. nuclear reactor design, energy management), city and territories (e.g. transports, gas, water and electrical networks), health and life science. A growing fragmentation of digital twin technologies could hinder their further technological development and preclude the kind of *multi-actors collaboration* that is required when addressing complex multilevel *systems of systems* such as factories, buildings or cities at large. Interoperability and the support of various business interactions, materialized by the exchange of technical (digital twin) data between actors inside a vertical domain, and between different vertical domains, with a single platform, which specialized vertical digital twins platforms cannot support, is at the core of the experimentations with the *Thing'in* platform, and the core subject of this chapter.

This chapter is organized as follows. Section II motivates the need and value of a graph-based, cross-vertical and multi-actor digital twin platform. Section III illustrates this value thanks to several real-life use cases experimented by Orange and partners in different vertical domains: smart building, smart factory, smart city and

All authors are with Orange Labs, France (email: {first_name.last_name}@orange.com)

telecommunication infrastructures. Section IV and V respectively discuss the main technical elements in terms of graph-modelling of digital twins of systems of systems, and then the Thing'in main platform design and implementation choices which sustain this ambition of a cross-vertical and multi-actors digital twin platform.

II. RATIONALE FOR A CROSS-DOMAIN MULTI-SIDED DIGITAL TWIN PLATFORM

A. An experimental natively multi-level graph-based digital twin platform

Orange vision of the future of the Internet of Things goes far beyond the mere extension of the current Internet to so-called *connected objects* (sensors, actuators) but designates a much deeper fusion of the current digital and physical worlds into a brand new world of digital services deployed all around us in the physical world (an "ambient intelligence"), interacting with the physical world and humans in their daily activities at home, at work, in transports, in the city and the countryside, etc. **Digital twins are a cornerstone, of this vision of a cyber-physical world, for they represent a bridge between the physical and digital worlds**. They allow for a digital description of the physical environment in which sensors and actuators are deployed. The initial development of Thing'in was seen as a way to experiment new technologies, use cases, and possibly business models, related to digital twins and the IoT.

Thing'in is an online platform (portal and APIs) that exposes a *graph* (more details in Section IV) of digital twins of entities (objects and systems) of the physical world, where the graph itself constitutes a higher-level "aggregated" ("multi-level") digital twin. Users can create and manipulate (the graph of) these digital twins and associated information (function, properties, state, location, shape, behavior, etc.), as well as get access to the physical objects they represent through sensors and actuators). **Above all, this multi-level graph captures *structural* relationships between these objects and the systems they make up, at multiple levels (single entities, systems, systems of systems), together with corresponding *semantic* knowledge**. Thing'in provides the "kowledge base" which describes in a homogeneous way the physical world (e.g. buildings, cities) in which sensors and connected objects are deployed. The platform also offers an extensible catalog of tools for service developers (e.g. loading of data from other platforms, 2D and 3D visualization, projection on OpenStreetMap and other cartographic supports, reasoning, learning and inference of new information) in order to help them build new vertical services above the platform that will improve the efficiency of the processes and systems considered (e.g. buildings, factories, cities, logistics chains, urban mobility).

B. A multi-actor and cross-vertical platform

An analysis of the market for the ongoing new wave of digital twins devoted to the smart city vertical [16], reveals that the positioning of providers of digital twin technologies is strongly correlated with their original business domain[2]: digital twins are seen as a natural extension of their activities. Actors coming from industry (manufacturing) are typically centered around *3D models* of products, machines, components and systems coming from CAD/CAM and Product Lifecyle Management (PLM) technologies. Actors coming from the building domain are centered around Building Information Models (BIM) which are basically (digital) *plans* of buildings. Actors coming from Geographic Information Systems tend naturally to build digital twins above cartographic representations, i.e. *maps* of the physical world. Actors coming from the *utilities* domain (water, gas, electricity) base their digital twins on infrastructure *planification tools*, often based again on 2D or 3D modelling.

It is worth noticing the current digital twin market is very different from the more mature IoT market where the need for generic IoT platforms that can manage connected devices and raw sensor data independently from the types of devices, data, and protocols, and finally from the applicative/vertical domain concerned (industry, building, city, etc.), has become obvious over time, leading to almost only generic (although rather low level) IoT platforms on the market today.

When experimenting use cases of digital twins in multiple vertical domains in parallel with the Thing'in platform[3], the need for sharing a *common description format* for digital twins, and actually sharing and exchanging digital twins themselves between actors (e.g. different professions in a building or during different phases in the construction and management, maintenance of a building), emerges as a pivotal requirement in these many apparently different use cases. It is this observation that has guided the development of the platform towards **a *transversal* platform that can target and link different verticals domains (e.g. building and city, building and industry, industry and transport/logistics, etc.), and that can enable multiple actors to interact within, or between, vertical ecosystems**

[2] At the exception of IT actors such as Microsoft with its Azure Digital Twins platform. We come to that hereafter.
[3] And perhaps also having IT and IoT background such as Microsoft.

to share digital twins (and associated data). This need is illustrated and further analyzed in Section III, while the technical impacts on the design of the platform is discussed in Sections IV and V.

III. CROSS-VERTICAL AND MULTI-ACTOR USE CASES: BUILDING, INDUSTRY, CITY, TELECOM

A. Digital Twins for Smart Building

1) Context

Smart Building covers many aspects. Among them, logistics and maintenance (from statically scheduled to on-demand organization), building user comfort and wellness (from today building based on simple automatisms to benevolent buildings able to consider contextual situations) and energy management (for example energy consumption in building domain takes 44% of the overall energy consumption in France). Operational building management is currently not as efficient as it could be because each technical team uses its own Building Management System (BMS) independently. Teams work in their own separate "silos", with expertise in one particular area but with poor knowledge of and little to no interaction with the other areas. These silos need to be opened up so that teams can collaborate better and with greater operational efficiency.

2) Aggregating and sharing digital twins between multiple building professions and services

Most recent solutions in the Smart Building domain are based on Building Information Modelling (BIM). BIM is well designed for modelling the infrastructure (walls, rooms, windows, floors, roof, etc.). However, **a building is also a composition of technical systems (energy, HVAC, water, IT, etc.) which needed to be included in the building Digital Twin. These different dimensions of the building cannot be easily captured in a single BIM. What is needed is a multi-dimensional representation of the building including and making explicit the relations between the different dimensions.**

An additional value can also be brought if building is not only considered as a single entity but also integrated and interacting with its environment: other buildings or facilities in the neighborhood (e.g. buildings on an industrial site), city infrastructures the building is connected to, etc. A homogeneous representation of such global digital twin, composed of several "sub digital twins" is needed.

Both multi-dimensional representation of the building itself, and its integration and interaction with its neighborhood, requires a unique platform. As a multi-actor and cross-domain platform, such a platform would enable Digital Twin sharing across the building professions and services.

3) Experimentations

Thanks to the Internet of Things, Smart Buildings are more and more instrumented with various sensors and actuators related to comfort for building users (temperature, shutter or light management, etc.), energy efficiency for building managers, and security for all. Such connected environments provide a first level of automation, with simple static and pre-defined scenarios, based for example on rules designed by a building technical manager. Thanks to a building Digital Twin, more advanced automatic features can be provided, benefiting from the contextual information of the Digital Twin.

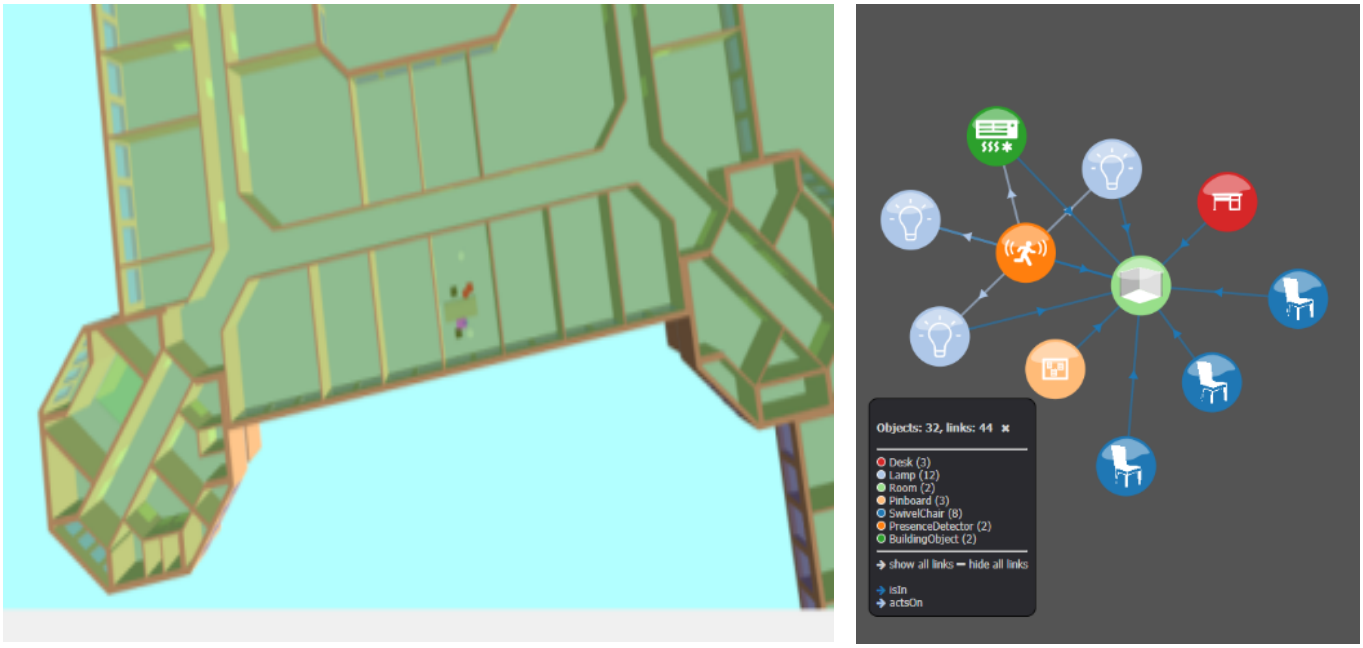

Figure III-1- Objects deployed in a room mapped on 2D map of a building (left) and the graph representation (right)

Several experimentations of a "benevolent building" have been implemented in an Orange Labs building in Meylan, France, with the Thing'In platform:

- Managing security and safety of users: this use case demonstrates how a Digital Twin can be the informational base for ambient intelligence. The security and safety services are based on Artificial Intelligence (AI) software components distributed at the Edge in the building. Contextual information of the Digital Twin can feed these AI, for example information like the rooms to be secured (e.g. close room doors) when an event occurred in such part of the building.
- Monitoring comfort of building users: this use case demonstrates the complementarity between a Digital Twin platform and classical Internet of Things (IoT) platforms dedicated to data collection from sensor. The comfort monitoring service aggregates in the Digital Twin a comfort status for each room. **The status is based on Digital Twin structural graph** (e.g. building topology), contextual information and raw data provided by temperature and humidity sensors through an IoT platform.
- Checking automatically building compliance to norms and regulation: **this use case demonstrates how to take advantage of the semantic graph of the Digital Twin**. Assessing building conformance can be a time-consuming and complex process. Moreover norms are evolving as well as the building itself and its usage. The experimentation has focused on security conformance related to number and capacity of secured waiting areas including accessibility constraints. In the experimentation the automatic checking is performed by reasoning on the knowledge graph extracted from the building Digital Twin.
- Sharing objects for predictive maintenance: this use case demonstrates the benefit of a multi-actor Digital Twin. Currently, most sensors are deployed in a building for a vertical need. For example, presence sensors can be deployed by the team in charge of building energy for an energy efficiency service based on presence information. **This presence information can also be useful for another team, the facility management team, to identify the most used rooms or areas in the building**. Facility management team can optimize its planning based on this information coming from another vertical of the building.

These experimentations are examples of the benefit of the multi-level aspect of Digital Twin graph, either the structural one (e.g. topology of the building), the semantic one (e.g. semantic reasoning) or both.

B. Digital Twins for Smart Industry

1) Context

As the industry 4.0 takes shape, human operators have to deal with complex daily tasks, new environments, and

immersive technologies relying on virtual or augmented reality contents. On the one hand, this can be stressful because it requires from operators agility and flexibility to adapt themselves to this new environment. On the other hand, it can help operators in their daily tasks. They can learn from the digital twin, practice and simulate actions before doing them in reality … Finally, the digital twin helps toperators become better, more efficient and more confident in their daily work in a complex environment.

2) Modelling the smart factory buildings, production chains and flows inside and outside the factory

Equipped with multiple sensors and actuators, a production line is today fully manageable and adaptable to the demand. The digital twin is not only a model of a complete production line, it includes the flows of incoming and outgoing raw materials or goods. The digital twin is not only a model of a factory containing multiple (connected) equipment, it is more generally a composition of digital twins of buildings, furniture, roads, parking lots… i.e. connected and non-connected objects which are linked to each other with different kind of relations. **Modelling homogeneously the smart factory building, its main production areas, its furniture, boxes, pallets, machine tools as well as the stakeholders (i.e. suppliers, subcontractors), is key to be able to get a fully defined digital twin of a global industrial site made of heterogeneous elements.**

A digital twin platform such as Thing'in offers all the required features to model, homogeneously in a pivotal graph-based model, the factory digital twin with its different elements: buildings, the production chains equipment — including the digital equipment e.g. for tracking objects (products and tools) during production cycles, roads and parking lots, etc. Digital twin tracking in Thing'in is time and location-aware. Combining structural and semantic graph with fine-grained Building Information Model capabilities, Thing'in allows for indoor and outdoor device tracking. It detects when the status of a twin changes and notify its operators, for instance when a production piece changes location or when its processing deadline has exceeded. All the locations of the digital twins are historicized in order to improve the traceability of the produced parts. The temporal graph of Thing'in allows the analysis of past events that could make the production more efficient.

All the digital twins are managed by a set of interconnected Thing'in servers that are geo-distributed in the network infrastructure (cf. Section V). **The digital twin platform enforces cross-enterprise collaboration through sharing (parts of) digital twins between users of different Thing'in instances. It allows the digital twin owners to share their private objects with their sub-contractors or their suppliers by extending their visibility.**

3) Experimentation

An experimentation is in progress in real industrial context within a factory located in Brittany (France). The main goals to achieve with the digital twin during the experimentation, are:

- improve the productivity, optimize the quality and On Time Delivery (OTD KPI), improve availability ratio (predictive maintenance, optimized maintenance cycle durations),
- help the operators and all the people who work in the factory in their daily tasks without adding stress related to the digitization of the factory.
- share easily the digital twin information with many actors (sub-contractors, suppliers of raw materials, suppliers of mechanical parts for machine tools, customers, recyclers)

Before launching a production, the production manager has to check that the required materials for manufacturing are available (in stock). Once it is launched, he must be able to keep an eye on the *work orders* at all times and locate the produced objects whether they are being manufactured in the factory or at a subcontractor level. Thanks to the digital twin of the production plant, he is able to search and locate the workorders on a 2D plan, receive notifications whenever an object stays for too long in an area. As the digital twin is part of a temporal graph, each event is recorded with a timestamp. For example, the beginning and the end time of the cleaning cycle of a production machine are stored in its digital twin so that optimized maintenance cycle durations can be planned. By counting the cumulative machining times, it's possible to organize some predictive maintenance and improve the availability ratio. Touch pads have been introduced in the plant to allow operators to easily view, modify or annotate the work orders, consult the plan of the part that is being machined, check the program to use and keep up to date all the documentations. Alarms generated by the production machines are managed in real time by an IoT platform and the digital twin status is updated accordingly. A list of reported breakdowns is stored with their causes in order to improve the quality. Operators can use the tablets to read the list of faults that had occurred. The list of failures can

possibly be manually enriched by the operators.

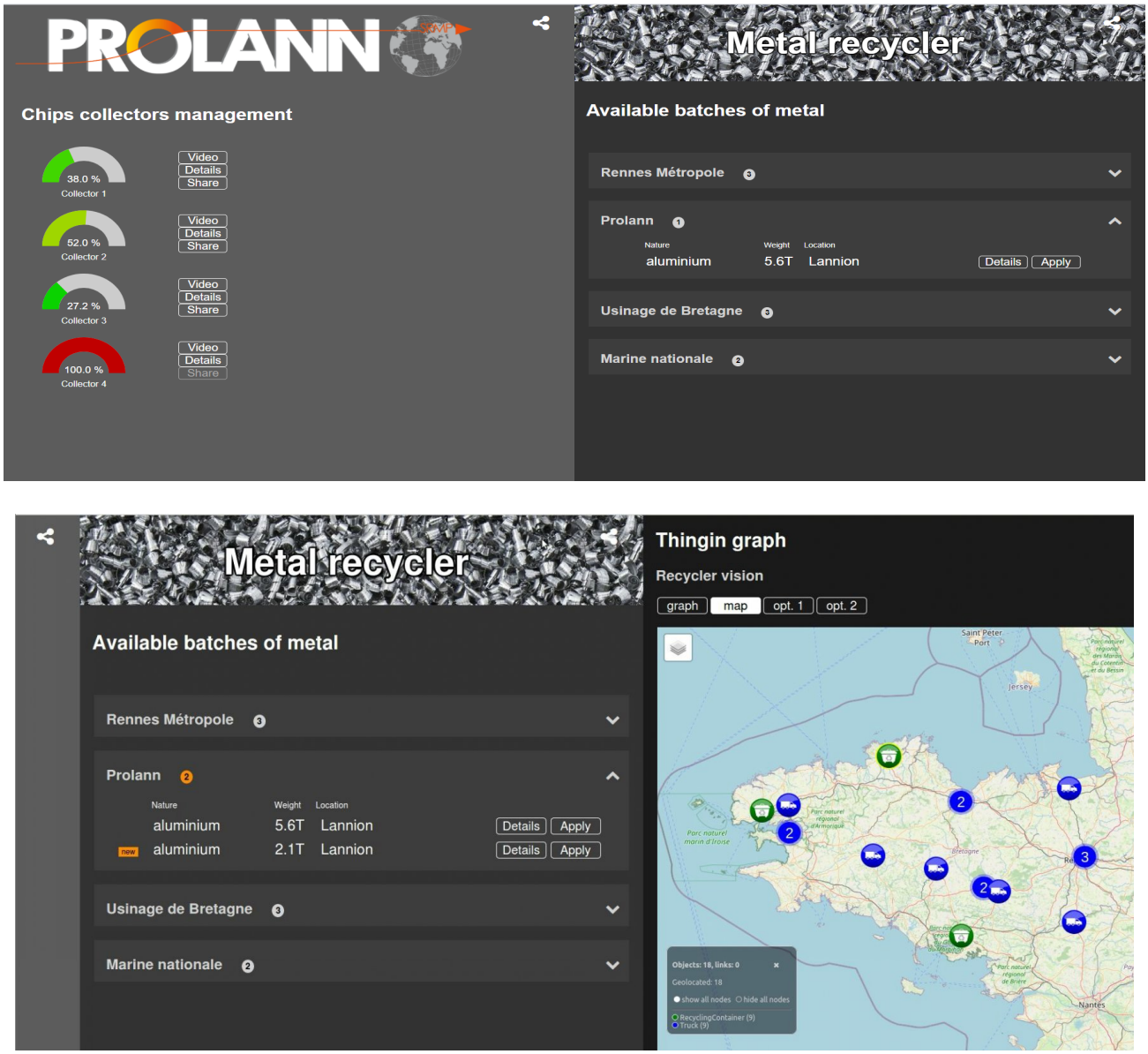

Figure III-2 – Dashboards (applicative view) presenting information about waste containers as seen by the factory (left) and by recyclers (right) based on shared data within the digital twin

The experimentation also includes the collection of waste containers by recyclers (e.g. metal chips collector) in a multi-actors and cross-domain scenario (manufacturing and recycling) — cf. Figure III-2**.** As soon as the digital twins of the waste containers are 80% full, the information is shared by the production manager with its recyclers so that they are immediately notified they have some containers to collect together with all useful data (e.g. location of the containers on site) thanks to the shared digital twin.

C. Digital Twins for Telecommunication infrastructures

1) Context

As a telecom operator, Orange is in charge of various fixed and mobile networks, in various countries. Altogether Orange manages millions of pieces of telecom equipment. With the evolution of technologies, networks are more and more complex and so their management complexity is increasing. The evolution of the ecosystem brings an additional complexity with several telecom operators per country, new actors or competitors and local partnerships between telecom operators and public organizations to provide broadband connectivity to all in the territories they are in charge of. Managing networks is no more the exclusive concern of one national operator but the collaboration

between various actors either operator internally or in an ecosystem gathering telecom operators, public organizations and territories, private service providers or contractors. New approaches and systems are needed to face this increasing complexity.

2) Aggregating and sharing digital twins in a telecom infrastructure ecosystem

An example of sharing of information on network between different actors comes with the access network which connects customer premises to network services. Orange is in charge of the management of a subset of the French access network. It is composed of multiple equipment such as telephone poles, network cabinets, trapdoors for underground chambers…

Figure III-3- Telephone pole (left), network cabinet (middle) and underground chamber and its trapdoor (right)

About 100 telephone poles fall every day. In cities, fallen poles are quickly signaled, but it may not the case in the countryside. Moreover, depending where the fallen telephone pole is located it can have severe consequences. For example, a fallen telephone pole on a u-curve on the road can cause car accident. In the same way a telephone pole connecting a hospital is more critical than a telephone pole connecting a private residential house. Orange share information about network cabinets with other Internet Service Providers (ISP) or subcontractors. Currently cabinets have mechanical locks which can be open with a key. When Orange implant a fiber optic street cabinet, it gives a copy of the key to other ISP so they or they subcontractor can access the cabinet. However, given the multiplicity of intervention from different companies it is difficult to know who performed the last operation. Without considering incivilities or malicious acts in 2020, around 200 have been signaled in a French department. With trapdoors Orange faces the same kind of issue: the difficulty to know if a trapdoor is open or not. With trapdoors, technicians have underground access to copper and fiber cables. They can access it to do maintenance operation or to add new cables. However, once a trapdoor is open everyone can have access to it and some thieves take this opportunity to steal copper cable. These robberies generate millions of euros of damage and client's loose access to the service.

Issues presented above with telecom equipment can be first addressed with connected objects to monitor object status: tilt sensors for telephone poles inclination, connected locks for network cabinets to notify when the door is open and have a log of open actions, a door sensor for trapdoor to know when the trapdoor is opened. **But sharing this information between Orange and partners or contractors is needed. This can be done thank to a shared/multi-actor telecom digital twin storing the description of the network equipment and their status. Moreover, if a city shares with Orange the digital twin of the city (cf. III.D), the combination of those two data sources can enrich the context of the telecom equipment. For example, it can help in evaluating the impact of an issue on a telephone pole near a hospital or a school.**

The approach can of course be extended to other telecom operators. Previously Orange was France Telecom, the historical national telecom operator, but today telecom networks can be shared between several telecom operators. Therefore, Orange is no longer in charge of all the telecom equipment in the French network, the responsibility is shared between the different telecom operators. However, for historical reason, an incident on a telecom equipment is often reported to Orange, even if the telecom equipment is not under the responsibility of Orange. Being able to share information between telecom operators through digital twins managed by the Thing'in platform could help solving this issue. In addition, sharing digital twins of telecom equipment can benefit to both other telecom operators and city stakeholders, cf. next section III.D on shared digital twins for incidents handling in a city.

D. Digital Twins for Smart City and Territory

1) Context

Equipment from the public space in French cities, like roads, trees, benches, telephone poles, electricity poles, etc., relies on numerous actors/operators: public sector entities at different administrative layers (city council, metropolis, country, region), private sector providers, such as telecom operators, electricity, water and gas operators, etc. Usually in cities the roads are under the competency of the metropolis or the county. Depending on their location, trees are in charge of the city, the metropolis, public or private parks, whereas telephone poles or electricity poles are maintained by telecom operators and electricity operators. When someone wants to report an issue on an equipment from the public space, because of the multiplicity of actors, it is difficult for citizens or even for some providers to know who is in charge of what and to whom address their requests.

2) Aggregating and sharing digital twins between multiple sectorial actors

A generic and multisided platform such as the Thing'in platform can help to solve the issue of multi actor responsibilities. It can help to break silos between actors on a same territory. Indeed, each actor is able to share information about equipment it is in charge of with other actors on the same territory, and even broader (to user to whom access rights on the Thing'in platform are given). Equipment from the public space are represented within a city digital twin on

Figure III-. A digital twin of an equipment will describe the characteristics of the equipment, for example a bench will have properties such as composition (wood, stone, metal…) and state of use (good, medium, bad, …). Each equipment's digital twin also has a property about the it's manager. Beyond that, this shared digital representation of the city allows third parties to develop services on a territory, rather than having a service by provider or by usage.

3) Experimentation

An experimentation is going on in the city of Meylan in France. The city of Meylan provide the Thing'in digital twin platform with data from their Geographic Information System (GIS) describing equipment in the city such as: street lights, trees, benches … A prototype was developed to ease incident reports by citizen of the city. Digital twins of the equipment provided was created in Thing'in, with for each digital twin, the actor in charge of the equipment. Data about the same territory coming from our GIS describing telephone poles (more detailed in next section) was also injected inside Thing'in. Therefore, citizen living or working in Meylan have a unique entry point to report any incident on equipment in the public space, as shown in Figure IV-2, the user can directly see all the equipment around him. Thanks to the data stored in Thing'in risen incidents are directly routed to the right actor

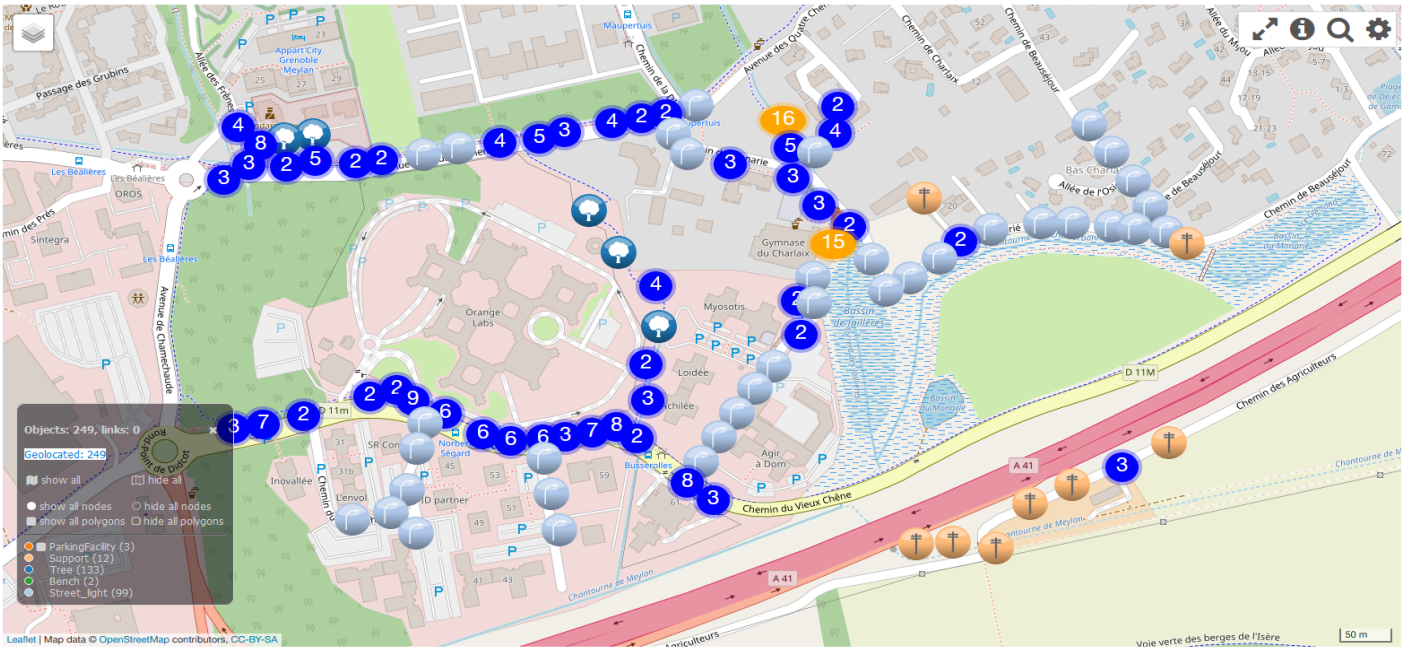

Figure III-4- Equipment data in an area in Meylan's city (street lamps in light blue, and telephones poles in orange) coming from information systems of multiple actors and aggregated in a shared digital twin of the city.

IV. ADVANCED GRAPHS MODELLING FOR SYSTEMS OF SYSTEMS DIGITAL TWINS

A. NGSI-LD as the basis for digital twin graphs

With classical (3D-inspired) digital twins in sight, graphs might not appear as an obvious modelling choice. Yet they have a storied and multifarious record, extending over almost three centuries, as *structural* models for all kinds of systems, across many branches of science and engineering. A more recent development, of truly revolutionary import, has been their use, in both natural and social sciences, as models for *complex systems*, spawning the new transdisciplinary field of network science [3]. Their use for **capturing multi-level structural models of *systems of systems* [4], viewed as the engineering counterparts of complex systems, is a key element of the approach proposed here.** Taking this broad view of graph models, they are the common denominator model of choice for Digital Twins, assuming the emphasis on open systems and cross-silo information sharing, as outlined in the previous sections of this chapter. For this we need to draw upon a wide-ranging and long-established lore of graph-modeling know-how, in order to represent and *match directly* the *structure of physical systems* as multi-level digital twins. These *cyber-physical graphs* [5] capturing system structure should make up the core of the graph and platform, yet they should still just be a kind of skeleton for holding and giving access to more classical pieces of data, meta-data, attached or referenced information that come to "flesh out the skeleton". These requirements set out the framework for choosing the best graph model for capturing digital twins in the sense we envision.

1) Knowledge graphs and Entity-Attribute Values: not expressive enough for digital twins

In view of the huge spectrum of applications addressed by graph models, it is somewhat baffling that their most recent and narrow derivatives used for knowledge representation may have come to acquire such a pregnancy across information technology. RDF graphs have emerged as the de facto common-denominator model for this, especially through their recommended use for linked open data datasets, the "web of data". Yet it should be clear that **RDF graphs do not model physical systems, as a digital twin should: instead they just store weakly structured information *about* these systems**, as sets of logical predicates. RDF graphs cannot as such make up the core of a digital twin graph, yet they may and should be associated to DT graphs as a "semantic overlay", in a way that we describe in subsection C below (Section dealing with Semantics). Coming from either basic IoT or lowly physical models side, data semanticization is a "low-hanging fruit" for information interoperability, which should already be taken for granted. This means that, at the very least, all devices, things and systems being represented must be categorized by reference to classes formally defined in shared ontologies, not with ad hoc types or labels. This two-level overlay amounts, as presented in subsection C, to the association of generic non-contingent knowledge (about classes/concepts as defined in a "Terminology box"/Tbox) to contingent information (about the association of individuals/instances to these classes, as laid out in an "Assertion box"/Abox).

Starting again from the IoT side, the Entity-Attribute-Value (EAV) meta-model was an early step towards basic structuration of IoT information, as it was captured by, e.g., the first versions of the OMA NGSI information model of FIWARE [4]. Lowly and weak as these models may appear, they are still a qualitative improvement from the binary payloads used in some IoT devices and networks.

An evolutionary path can be traced from this kind of object-like modeling to the properties attached to property graphs, which are just a different way to capture attributes in key-value style. It may appear paradoxical to call these "Property Graphs", because their properties, akin to `owl:datatypeProperties` in the RDF model, are not represented as arcs of the graph proper: they are embedded as inner structuration within both vertices and relationships of Property Graphs, which would thus better deserve to be called "Propertied Graphs". The values of these properties may be defined in structured types, such as arrays, thus richer than mere RDF literals. The skeleton of PGs corresponds to relationships between vertices which stand for all kinds of entities. There is another evolutionary path from the relations used in the long-established Entity-Relation (ER) model, where entities correspond to entire tables in relational data models), whereas PG entities would correspond to individual rows in these tables. PG relationships have some similarity to `owl:ObjectProperties` in the RDF model, but a crucial difference is that PG relationships are instantiated and identified individually on a per-instance basis, contrary to RDF properties which are un-instantiated and identified only as generic logical predicates. They are, just like vertices/entities, "first-class citizens" of the PG data model. PGs allow properties for both vertices (nodes that stand for entities) and arcs that stand for physically-based relationships: this corresponds to the minimal "fleshing out of the skeleton" supported by the PG model.

2) NGSI-LD graphs: the best choice for digital twins

The NGSI-LD information model [9][10], as standardized by the ETSI CIM (Context Information Management) group, provides a formalized basis for Property Graphs (PG), with a few extensions on the customary use of the PG model by most existing graph databases. As such, though defined based on RDF/RDFS/OWL, the NGSI-LD graph model has a higher-expressivity than Description Logics (DL) or First-Order Logic (FOL), bringing it closer to second-order logic. The counterpart is that it may lead to undecidability, and it should not be used with formal reasoning tools geared to DL or FOL. From our viewpoint, **NGSI-LD brings the best of three worlds: a "structural skeleton" inherited from entity-relationship models, key-value properties attached to both entities and relationships and, crucially, a semantic web grounding that makes it possible to overlay an NGSI-LD graph with an RDF knowledge graph**, as explained in subsection C. As laid out in [5][6], an NGSI-LD graph supports properties of relationships and properties of properties, which do not exist in RDF. It may be converted into an RDF graph, serialized and exported as a linked-data dataset, after applying reification. NGSI-LD graphs are the best common denominator for DT graphs as envisioned here, we explain in the following how they are actually used for this.

B. Multi-level structural twinning of cyber-physical systems

1) Thing-twins (TTs): atomic devices & physical entities as graph vertices

Most IoT platforms do already maintain some kind of very minimal "digital twin" — understood as a mere *proxy* — for the devices they take charge of. These "thing-twins" make sense as the lowest rung of a ladder, nested within higher-level and larger-scale graph-based-twins of the environments they fit into. These "atomic" systems/entities/devices are represented directly through graph vertices because they need not or cannot be decomposed further internally into sub-systems. They will have an assortment of fixed and variable attributes of their own, possibly capturing part of their state in the sense of dynamical system theory. For the connected devices most IoT platforms deal with, these proxies may provide a network interface supporting direct interaction of applications with the device.

Now, the **Internet of Things should not be limited to connected devices, sensors and actuators: it may, viewed as a graph, extend to the non-connected, passive or legacy "thing" that is sensed by sensors or acted upon by actuators** [11][12]. These non-connected things are also represented by vertices in the Thing'in graph, with corresponding "phenotropic" or "stigmergic" [12] links to the sensors and actuators, respectively, which may be used as network intermediaries to them. These atomic devices and things are deemed to be well-characterized by their external relationships with other entities at the same level, or with the larger systems which they are part of. In a revealing analogy with social networks, just as a person called John Smith can be made unique by a tiny number of relationships of his own, a few external graph links may be enough to make a standard-issue entity (like a home appliance, or a piece of furniture) unique among many identical copies, without the need to characterize it internally.

2) System-Twins (STs): self-contained systems as rooted subgraphs

Well-defined subgraphs of the overall graph will capture the key structural links that make up the scaffolding of a "classical" self-contained system, in the intuitive sense of a physically-enclosed contraption, tacking together a set of parts/subsystems which are its direct constituents. These constituent subsystems may either be captured as "thing-twin" atomic vertices as described before, or decomposed further, recursively, into subsystems which may themselves be described by the same kind of rooted subgraphs.

Figure 1 gives an example of this for an *apartment* captured as a subsystem of a *building*, decomposed further into *rooms*, which are themselves composed of entities considered here as atomic and captured as "thing-twins". These subgraphs have a "root" node, with type `NGSI-LD:system`, standing for the subsystem being described by the subgraph (building, apartment, room), but the overall connection pattern is *not* limited to a directed rooted tree (a.k.a. arborescence). Typically, they will "look like" rooted trees, with added transversal links, mostly undirected edges, between the "vertical" branches connected to the root, as shown in Figure IV-1- (with NGSI-LD relationships drawn as diamonds, à la ER diagram, and entities as rectangles).

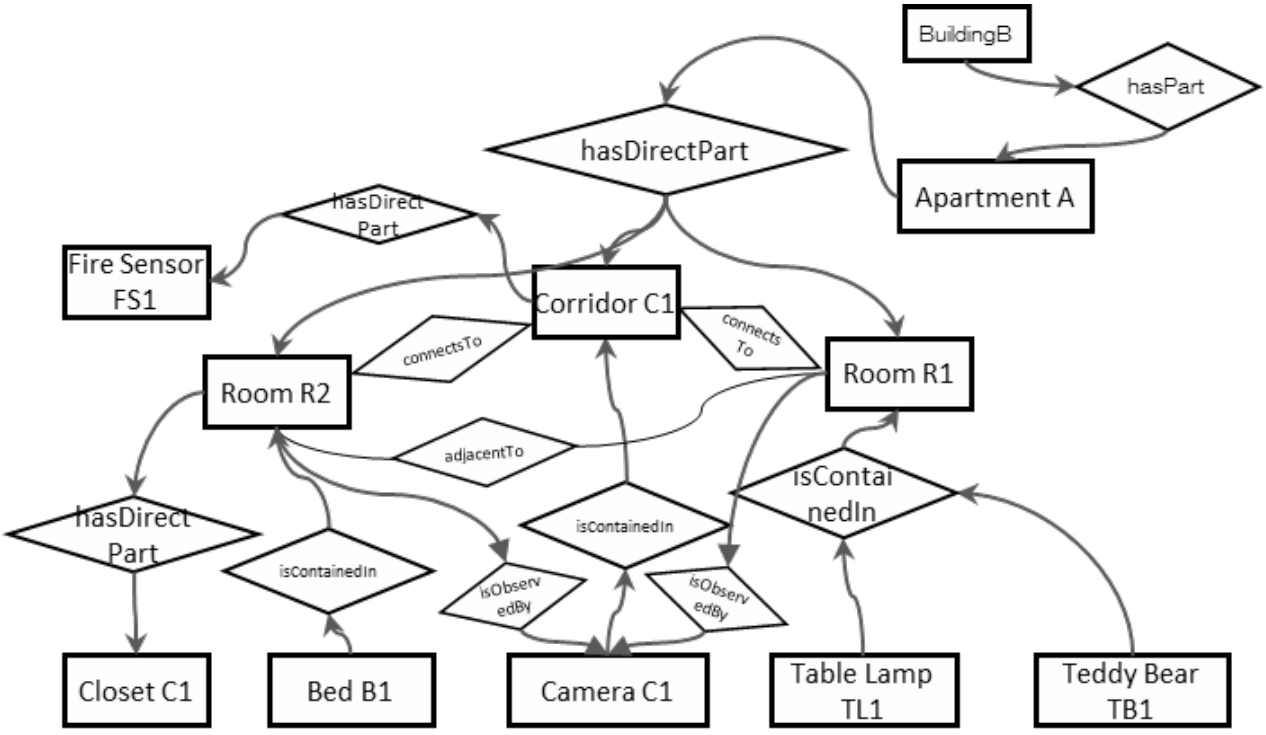

Figure IV-1- Description of a building as self-contained system, with two-level nesting of subsystems

By definition, the relationships between the nodes that make up self-contained systems like these are physically local; they may correspond to:

- vertical *top-down* links between the root system and its constituent parts (with type `NGSI-LD:hasPart/hasDirectPart`), when these parts are designed to be included in the overall system and this system cannot work if these parts are removed
- vertical *bottom-up* links (with type `NGSI-LD:isContainedIn`) between parts and systems that would not always be captured as such, like the sets of all things (furniture, appliances, etc.) contained inside a room. This type of relationship may also be used to capture an even more informal and purely contingent set-based location [13], without implying a "systemic" relationship.
- transversal (possibly undirected) links (with type `NGSI-LD:ConnectsTo)` to capture the way through which one may go from one room to another, or a room is near another (with type `NGSI-LD:AdjacentTo)`transversal directed links between a sensor and what it observes (with type `sosa:isObservedBy`), or an actuator and what it acts upon (with type `sosa:isActedOnBy`)

3) Sy*stems of Systems Twins (SoSTs): capturing distributed and complex systems*

This level of representation does also capture systems as subgraphs of an overall reference graph. It is used for a less obvious, but critically important type of systems that we label here "Systems of Systems" (SoS) for short, even if not all of them, by far, are SoST in a strict sense [4]. For these more complex systems, it would make no sense, or be impractical, to have a regular direct relationship between a "root" node and the constituent parts of the systems, as proposed for the simple self-contained systems described before. The reasons why this kind of system has to be captured in this special way may be one or several (*but not all*) of the following:

- **the system is physically distributed**, potentially on a very large scale, with a large number of direct constituents,
- the system belongs to the broad category of ***physical infrastructure networks***, whose connections correspond to actual physical links: road/street networks, electrical grids, water distribution networks, gas distribution and transport networks, telecom networks (at the physical infrastructure level), etc.
- the system is a "system of systems" in the strict sense of the customary definition [4] : **a bottom-up assemblage of subsystems which are *operationally independent* and *have not been designed to work***

together*, but which do happen to *work together to provide a functionality that is more than the sum of those provided by the individual subsystems separately; like the Internet at large, or a city,

- the system exists mostly as an informational abstraction, **grouping subsystems that are physically independent, or have a loose connection**, like e.g. a logistical network, a waste collection network, or, in different vein, a sharing system that federates a set of physical assets for rental or lending,
- relationships between the parent system and its constituent subsystems correspond neither to the **"`NGSI-LD:hasPart`" relationship** characteristic of a classically-engineered top-down system, nor the more informal "`NGSI-LD:isContainedIn`" relationship of a bottom-up, "informal" system.

These subgraphs are clusters of nodes, together with the relationships that bind them and the properties that characterize them. They are impersonated by higher-level "hypernodes" with type "`NGSI-LD:graph`". The relationship between constituent nodes and their parent hypernode are captured by a special "`NGSI-LD:isNodeOfGraph`" relationship. The graph "hypernodes" may themselves be matched to nodes within this "hyper-structural" graph, as subgraphs of this graph, with a relationship of type "`NGSI-LD:isSubGraphOf`".

An example of this type of systems-of-systems grouping is illustrated in Figure IV-2- below for the infrastructure of a smart city, with the hyper-structural graph shown in red and the structural graph in black, showing obviously a tiny sample of the kinds of nodes that each of these systems would comprise.

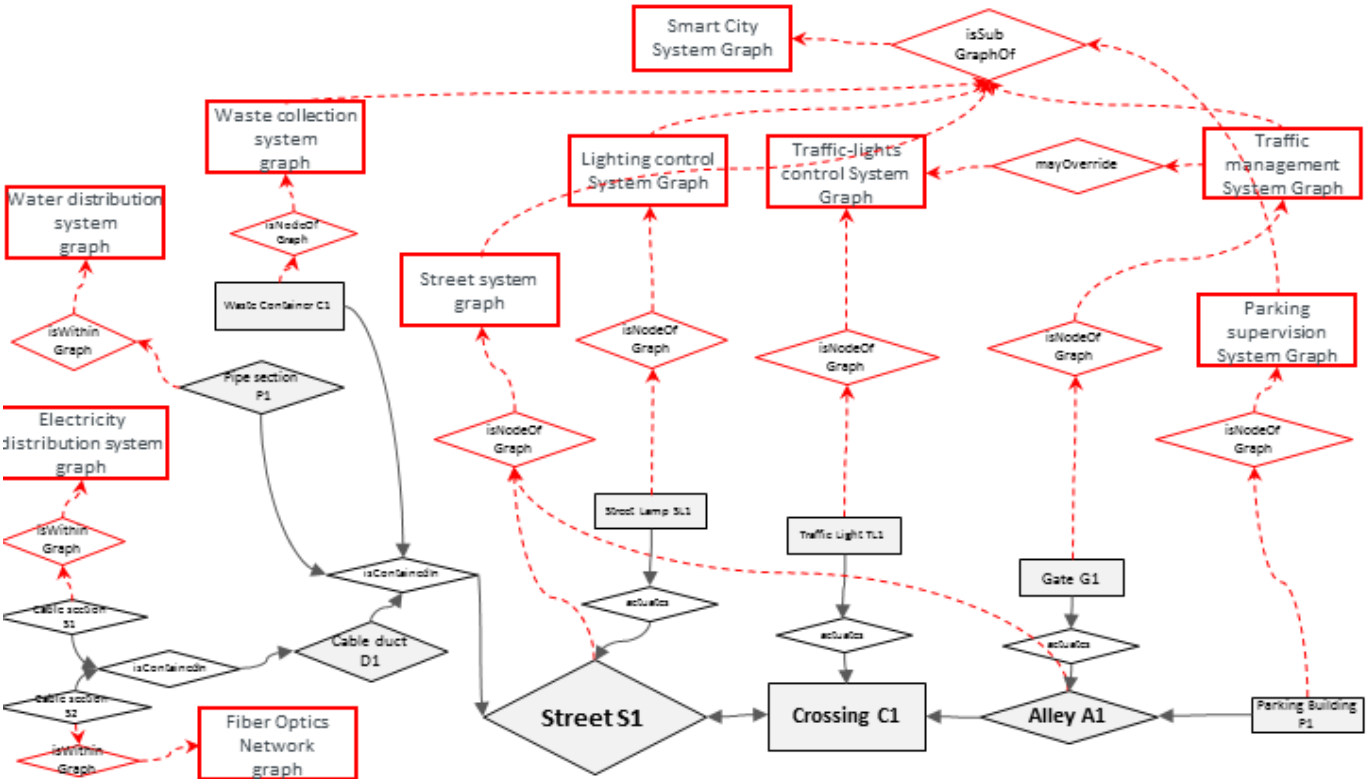

Figure IV-2- Subsystems of city infrastructure, captured as separate subgraphs & represented by graph "hypernodes"

C. Semantics for digital-twin graphs

As stated in subsection IV.A, knowledge graphs are fundamentally distinct from the kind of physically-matched DT graph models presented before. Yet, even if semantics as expressed by RDF graphs is not at the core of the proposed vision, RDF graphs and semantics have a role to play:

- in supporting the interoperation of these models with third party data sources,
- in supporting the actual use of the information maintained in these graphs by applications which do not, natively, understand PG/NGSI-LD models.

In both cases, the RDF metamodel may be used as a lowest common denominator on which to fall back if the higher expressivity of PG/NGSI-LD graphs cannot be directly addressed. This is used by:

- the import of data in RDF serialization formats (Turtle) from our Thing'in platform,
- the NGSI-LD API (itself used by FIWARE) based on JSON-LD, another RDF serialization format, with semantic matching captured and compressed in context files.

Even if DT graphs have a semantics of their own, which is global to the whole graph and not reducible to "per resource" semantics as used by RDF, RDF-style semantics may still be applied in a very rich and detailed way to individual NGSI-LD entities and relationships (as already exemplified above), because their formal definition is itself grounded in the RDF/RDFS/OWL meta model.

Besides their physical matching, another distinction between DT graphs as proposed here and generic knowledge graphs should be clear: knowledge graphs, inasmuch as they capture ontologies or taxonomies in a description-logics ("Tbox") are meant to capture generic knowledge as relationships between the concepts/classes of these ontologies, whereas DT graphs capture a more contingent type of information at the individual/instance level. The kind of information captured by an "Abox" (link between an instance and its concept/category/class), is closer to what DT graphs capture, and would correspond to an `rdf:type` property arc between a vertex or relationship of an NGSI-LD graphs and a class-level resource drawn from a relevant ontology. As we had asserted previously [5], RDF graphs may and should be used to complement and *overlay* the Property Graphs capturing multi-level digital-twins with such an Abox. In doing this, we propose to maintain the distinction between instance-level information (the NGSI-LD graph) and class/concept-level knowledge corresponding to the Abox with relevant parts of the TBox. This distinction may have been obscured by the use of RDF graphs and triple stores to capture both, jointly and indistinctly. Figure IV-3 shows how this pans out, with classes drawn from relevant ontologies (pictured as green, rounded rectangles) overlaid on an NGSI-LD graph with `rdf:type` typing links pictured as dotted green lines connecting the two. These Abox arcs or the green RDF graph represent "raw" RDF properties, and are of a completely different nature from the relationships and properties of an NGSI-LD graph because, contrary to those, *they are not identified as individual instances, but as generic predicates.* Additional Tbox arcs (dashed green lines) describe subclassing (inheritance) relationships between these classes. This thin Tbox overlay, a minuscule subset of the overall graph in terms of number of nodes, is the only part which fits the strict definition of a knowledge graph as capturing conceptual (non-instance-related) information.

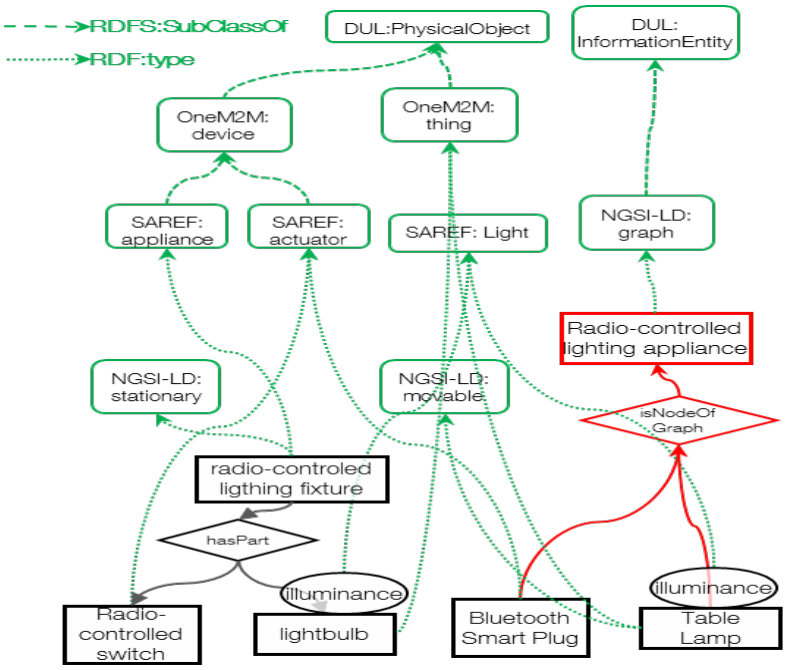

Figure IV-3- Semantic graph overlaid upon NGSI-LD graph (Abox with dotted arcs, Tbox with dashed arcs)

V. THING'IN DIGITAL TWIN PLATFORM DESIGN AND STRUCTURING IMPLEMENTATIONS CHOICES

A. Platform functional architecture overview

The platform functional architecture is showed in Figure V-1-Functional architecture (Thing'in instance).

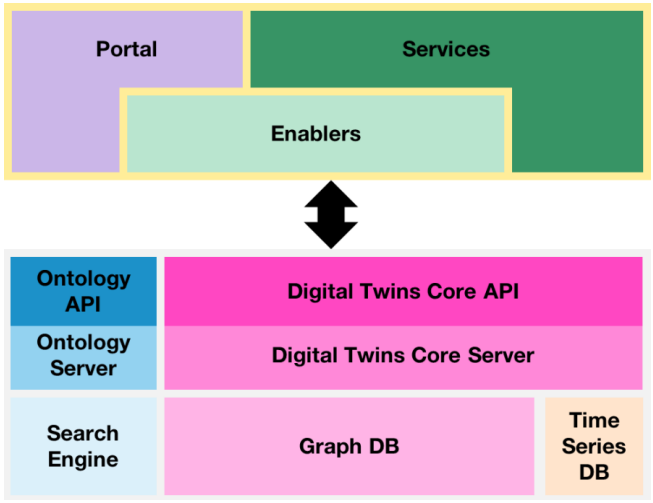

Figure V-1-Functional architecture (Thing'in instance)

The bottom layer is composed of infrastructure fundamental technical services: graph database, time series database, search engine. The search engine stores the ontologies defined and used in the platform, the abilities of the search engine ease the lookup of the different concepts according to their syntax and semantics. The Ontology server converts an ontology described in RDF to documents for the search engine. The Ontology API provides to the user an HTTP Rest interface to query the Ontology Server.

The Digital Twins Core Server manages digital twin storage and management (Create, Read, Update, Delete). It provides advanced search functionalities based on semantics, structure, context or geolocation of the digital twins. It can manage also the historization of some properties of the digital twins. The core server is warrantor of the security and the privacy of the digital twins: it manages the identity, the roles and the rights of the users inside the platform. The Digital Twins Core API provides to the user an HTTP Rest interface to request the Digital Twins Core Server.

The three bottom layers components collectively constitute the *Thing'in Core*. Above this core are built some *enablers* facilitating the management and usage of digital twin applications. Examples are components that convert information about physical objects from open data or from Orange LiveObjects IoT platform into digital twins, graph discovery and graph traversal functionalities, viewer components such as 2D/3D visualizations and projections on geographic maps.

B. Graph storage: mapping NGSI-LD graphs onto graph databases

The Thing'in core property graph introduced in the previous section is stored and manage thanks to a graph database management system (DBMS) (currently the ArangoDB open source multi-model DBMS).

As the targeted physical objects/systems are very heterogeneous — since Thing'in seeks to be agnostic towards vertical application domains — the data model needs to be very flexible and extensible. Thing'in relies on the property graph flexibility provided by a schema-less DBMS to avoid to be locked inside a data model. Thing'in supports semantics description of digital twins, in addition to structural description, through the use of ontologies. Thing'in stores RDF entities inside the property graph (these entities can be exported back from Thing'in in a RDF format). As explained in previous section, the property graph of Thing'in is more than just a RDF graph, this is why Thing'in does not use a RDF Store. **Thing'in properties of nodes and edges can store any type of information, and not just concepts from ontologies-related information as in RDF stores. For instance, all the information about ownership, access control security, attached contents (like icon, plan, file …) could be part of the graph.**

Furthermore, the use of a graph DBMS allows for functionalities of graph traversal and graph pattern matching. Graph traversal allows a query to reach any vertices that are connected to a starting vertex by taking steps. This is very adapted to discover a digital twin and its context. Graph pattern matching allows the finding all the parts of the graph that respect a pattern. This can be linked up this with the semantic description, i.e. allow for semantic graph

pattern matching. For instance, in the global graph, one can make a query such as *"find all the connected objects owned by Alice located in a room having a window"*. The figure Figure V-2-Example of graph pattern. illustrates this pattern.

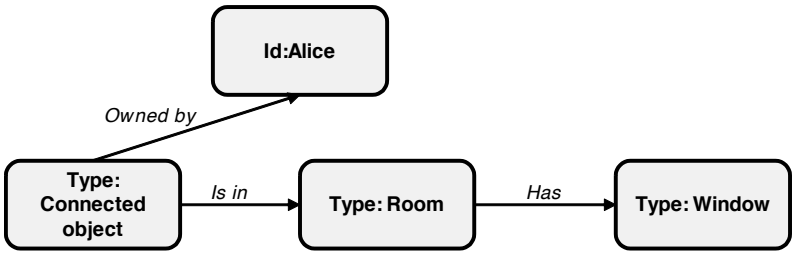

Figure V-2-Example of graph pattern

C. Scalability and Federation of multiple platform instances

Considering the complexity of the target systems considered (building, city, industry, telecommunications infrastructures…), and the multi-actor and cross-vertical ambitions, the implementation of a platform such as Thing'in has to face two major challenges: scalability and governance/control of (possibly shared) digital twins. In the long run, Thing'in may have to manage billions of digital twins provided and used by millions of object owners and service providers, so the platform must be designed to be highly scalable allowing massive storage and intensive IO to support massive queries. Also, some users owner of Digital Twins would better like to host by themselves ("on-premise") their graph of digital twins, be it to ensure more security and privacy, or to improve the latency by bringing closer the digital twins platform to their physical objects, or to have better control over the hosting infrastructure (and associated costs).

To fulfill these requirements, Thing'in offers 2 levels of scalability and governance thanks to a *federated technical architecture*:

1. multiple *instances* (called "Tipods") can be deployed in the cloud or on premise,
2. these instances are linked and coordinated by a *federator*.

At a Thing'in instance level, all the functional components (cf. Figure V-1) **can be replicated to consider more digital twins or more requests to handle local scalability. At the global level, scalability is ensured by the federation architecture itself, i.e. by the number of deployed instances.**

In order to ensure that the platform maintains one single (logical) graph in which relationships may be established between any single Digital Twins, the platform enforces some constraints of the deployment of instances (Tipods), e.g. restrictions on the naming of the digital twins so as to avoid conflicts. Before installing a Tipod, a user should have a certificate produced by the federator (entity that control the identities of the Tipods). Associated to this certificate, the user obtains the right to name his digital twins with a prefix chosen by this user and accepted by the federator. This rule create a trusted graph where only authorized Tipods collaborate to build the unique global graph.

D. Multi-level security

Sharing (parts of) Digital Twins between different actors imply a sophisticated security model. The Thing in platform implements a multi-level security model with security mechanisms at different levels for different purposes.

First, the security architecture lays on identification of the users. Each user who wants to use the Thing'in API must have an *access token* resulting from his authentication. This identification and authentication are compliant with the Oauth protocol [RFC 6749] and some identity providers like Orange, Live Objects or Google are trusted (this list can be extended). The access token forged by a Tipod is a Json Web Token containing some information about the user and the platform: *user id, user role, token expiration time, Tipod id* (that has forged it), and a signature computed with the private key of the Tipod. When a user is enrolled within the platform, the administrator gives him dedicated role(s) (among: basic user, provider, service manager, supervisor, administrator) used to ensure the access to the API according to Role-Based Access Control (RBAC).

At the digital twin level, Thing'in uses Access Control List (ACL) to specify the grants of the users. This ACL is concretely based on an Attribute Based Access Control (ABAC) implementation allowing to define the right at the level of the digital twin properties. For instance, with the access control mechanism, a user can restrict the reading of the geolocation attributes of his digital twins to the users who belong to a specific group or can allow the update of an

attribute to the users who have given a special key (in the request). With this kind of control, the users have free hands to define any security for their services.

Users identity is also federated. A user registered in a Tipod A, can request a Tipod B, since he uses a signed access token to request the Tipods. The verification of this signature can be checked everywhere upon the verifier can retrieve the certificate of the Tipod that has emitted the token. The role of a user registered in a Tipod A is not valid in another Tipod B, he can only request Tipod B as a guest.

VI. Conclusion, ongoing and future works

Digital Twin is quickly emerging as a very important matter in many economic sectors, together with Cloud Computing, Internet of Things (IoT) and Artificial Intelligence (AI). The field of application is immense from buildings, cities and territories, to industry, transport, logistics, energy, telecommunications and other large-scale infrastructures, or medicine, biology and science in general. **The development of Digital Twin is currently fragmented and siloed in terms of technology and usages. The work presented in this chapter advocates for a more *transversal* and *open* vision of digital twins**, underpinned by the development of the Orange Thing'in platform which seeks to experiment the value, and the actual implementation, of **cross-vertical and multi-actor usages of digital twins**. From these objectives and illustrative use cases, this chapter mentions main structuring platform design choices: **advanced multi-level graph modeling and security, graph database core implementation, federative and distributed architecture**. The intent of this chapter was to focus on the use cases rather than on the technology. However, there are many technical works behind the development of the Thing'in Digital Twins platform on the subjects just mentioned here and others, works which cannot be detailed here due to lack of space. Some examples are the ***historicization*** of the digital twin graph or the ***synchronization*** of the digital twins with their physical counterparts, which are major issues tackled by ongoing and future works.

Acknowledgment

The authors wish to thank the whole Thing'in team inside Orange research for contributions to the design and development of the platform and use cases protoytpes and demos — especially Fabrice Blache, Benjamin Chevallier, Fano Ramparany, Thomas Hassan, David Crosson and Alain Dechorgnat, Cyprien Gosttein and Maria Massri, Michel Giordani and Sylvie Tournoud, Yvand Picaud and Regis Esnault… Many thanks also for support and fruitful discussions with Nicolas Demassieux, Roxane Adle, Guillaume Tardiveau, Adam Ouorou, Lyse Brillouet, François-Gaël Ottogalli, Julien Riera, Bruno Gallier, Laurent Marchou, Jean-Marc Lafond, Jean-Michel Ortholland, Alain Chenavier, Emmanuel Routier… any many others. The authors acknowledge the contributions of partners of the ETSI CIM (Context Information Management) Industry Specification Group with whom we have specified the NGSI-LD property graph information model presented in section IV.

References

[1] Orange Thing'in Digital Twin Platform [Online]. Available: http://thinginthefuture.com/

[2] IDATE DigiWolrd. Digital Twin Cities, Jannuary 2020

[3] U. Brandes, G. Robins, A. McCranie and S.Wasserman S. (2013). " What is network science"?. *Network science*, 1(1), 1-15.

[4] G. Privat : "The "Systems of Systems" viewpoint in telecommunications", *Orange Research Blog,* September 2018

[5] Wu, Zonghan, et al. "A comprehensive survey on graph neural networks." *IEEE transactions on neural networks and learning systems* (2020).

[6] Angles, R "The property graph database model" , Coeur-WS.org vol 2100

[7] G. Privat, A. Abbas: "Cyber-Physical Graphs vs.RDF graphs", *W3C Workshop on Web Standardization for Graph Data*, Berlin, 03/2019

[8] W. Li, G. Privat, J. M. Cantera, M. Bauer, F. Le Gall: "Graph-based Semantic Evolution for Context Information Management Platforms". 2018 *Global Internet of Things Summit* (GIoTS), Bilbao, Spain; 06/2018, DOI:10.1109/GIOTS.2018.8534538

[9] ETSI GS CIM 006 V1.1.1 (2019-07) Context Information Management (CIM) Information Model (MOD0)

[10] A. Abbas, G. Privat: “Bridging Property Graphs and RDF for IoT Information Management”. *International Semantic Web Conference* (ISWC 2018), Workshop on Scalable Semantic Web Knowledge Base Systems, Monterey, California, USA; 10/2018

[11] G. Privat: “Extending the Internet of Things”. *Communications & Strategies*, *Digiworld Economic Journal* n° 87, 3d Q 2012, pp101-119

[12] G. Privat: “Phenotropic and stigmergic webs: The new reach of networks”. *Universal Access in the Information Society* 08/2012; 11(3):1-13., DOI:10.1007/s10209-011-0240-1

[13] T. Flury, G. Privat, F. Ramparany: “OWL-based location ontology for context-aware services”. *AIMS 2004, Artificial Intelligence in Mobile Systems*; 09/2004, 1989