
A Language & an Approach for the Development of IoT Solutions

Taoufik Ben Hassine

Lab Code UR17DN01, Aviation School of Borj ElAmri, Manouba, Tunisia

Email address:

tbenhassine@yahoo.fr, taoufik.benhassine@ensi.rnu.tn

To cite this article:

Taoufik Ben Hassine. A Language & an Approach for the Development of IoT Solutions. *American Journal of Electrical and Computer Engineering*. Vol. 6, No. 1, 2022, pp. 1-14. doi: 10.11648/j.ajece.20220601.11

Received: June 7, 2021; **Accepted:** June 22, 2021; **Published:** January 8, 2022

Abstract: IoT can contribute to the resolution of problems inherent in the control of our environment and to the automation of decisions based on data extracted from that environment. However, the opportunity to contribute to these activities is not offered to everyone. Indeed, according to the majority of decision makers in different fields, IoT solution development is limited to only IoT experts in these fields. According to our opinion this truth is not absolute. In order to demystify IoT and trivialize participation in the IoT solution development effort a language specific to IoT domain has been defined. This language is based on a meta-model of IoT and is made of a textual notation conforming to Backus Naur Form. It allows each participant in a development activity to express and discuss his or her idea of a solution to an IoT problem. An approach build on this language proposes a demarche that starts from the IoT meta-model to arrive at a solution code. Each developer can derive the meta-model to obtain a solution model. Next, she (or he) uses our development environment to concretize this model. Our environment is build using Eclipse/Xtext. It offers technological tools to support our approach. The case study that provides a proof of concept of the approach falls within the domain of smart agriculture.

Keywords: IoT, Language, Development Approach, Code Generation, Technological Tools, Meta-model, Smart Agriculture

1. Introduction

In his quest to live each day a little better and to prevent the future, man has always sought to understand what surrounds him from near or far. To know his environment requires information, the raw material of knowledge. This information is only a set of data forming a meaning and collected on the ground. Instead of inspecting and analyzing the elements in the field, an alternative approach predicted by "Mark Weiser" [5] seeks to provide the environment that is being inspected with the technological means that allow it to "express itself". This approach is made possible by advances in areas such as miniaturization, thanks to a growing integration of microelectronics and wireless communication. With the help of devices made of miniaturized and highly integrated electronic components, we endow the land, we wish to monitor and listen to, with an intelligence that makes it an active object. This active object will participate in man's life. It will be, no longer, a passive object isolated from this life. Indeed, this intelligence allows an element of the terrain to report its state, to be aware of it and to react within the

limits of its competences as soon as it is asked. If this intelligence and competences come to an end, one can claim to have a completely autonomous terrain that one no longer needs to monitor but to support. "Mark Weiser" predicted that these devices would be present in our everyday lives. The problems that the man meets in his life will be solved by the solutions that put in play, the best, the capacities of the man and those of his environment. We propose, in this paper, an approach that helps to conceive and generate such solutions. This approach is based on a language specific to the Internet of Things [25] (IoT) domain.

The remainder of the paper consists of the following sections. Section II briefly describes the development approach we advocate. In Section III, we present the objectives and constituents of the "LIDO" language. Section IV describes the INDIT development environment, which is the concrete and practical counterpart of the theoretical IoT meta-model. Section V presents a case study that serves as a validator of the approach. Finally, in Section VI, we present a summary and perspectives of our work.

2. The Approach

The approach we recommend is based on a language specific to the IoT domain called "LIDO". This language allows the entire development team, including the end users, to discuss and exchange design and implementation ideas about the solution. The approach foresees that, in a first phase, the development team derives a generic meta-model formed by a selection of concepts from the IoT domain. This meta-model constitute the abstract component of "LIDO". The team then obtains a solution model made up of draft and unfinished objects. This is a kind of meta-design. The second phase consists in furnishing the objects of the solution by describing them in detail through a textual notation constituting the concrete part of "LIDO". In a third phase, the noted solution will be transformed by code generation to arrive at a draft of a feasible solution. The different phases are carried out in cascade. Each phase, like a pipeline architecture, represents the source, the raw material, for the phase that follows.

The approach relies on the involvement of users and developers at all levels to enrich and accept the chosen solution as long as they understand the different concepts proposed by "LIDO" and know how to use the textual notation. The meta-model proposes different views that allow modeling the different facets of the solution. Each contributor, according to his skills and concerns, can give his point of view or opinion concerning one or several facets of the solution. Several points of view from different contributors concerning the same facet can be conversed by the development team during a Brainstorming session.

Through the meta-model, the approach tries to provide the development team with a basis for the installation of an IoT culture. This culture is sufficient for an exchange of proposals aimed at advancing the search for solutions in a collective manner. This IoT culture is at the heart of the approach and its philosophy. The meta-model is abstract enough to dispense with implementation specifics and support a horizontal culture. As a result, contributors think of solutions at a high level of abstraction that gives them freedom from restrictive and limiting implementation.

3. The "LIDO" Language

3.1. Reasons and Purposes of LIDO

A development team is usually made up of a multitude of actors with different profiles and concerns. To collaborate, these actors need to understand each other and to share a common language. This language must support an IoT culture that is sufficiently horizontal to bypass the differences in the profiles of the actors. To design and develop this language, we turned to the literature to study the state of the art in IoT solutions. From this study, we drew a set of components of these solutions that we subjected to an extensive abstraction to draw a collection of concepts. From this collection, we built a meta-model called "MODIDO" which we wanted to make a reference model of the IoT. The

development team trained on this meta-model can derive it during the design activity. This derivation results in a collection of objects with sufficient intelligence. This collection will be part of the solution model. At this stage, this solution model is sufficiently abstract for its physical counterpart. This physical counterpart represents the realization of the solution to be imagined in different ways. It consists of choosing from the market the components that will implement the intelligence of the objects in the solution model. To do this, "LIDO" offers a textual notation called "NTIDO" which allows the technological components of the physical solution to be described. An automatic generation of code for this solution is planned.

3.2. Digital Integration, a Topic Common to the Different Points of View

All the definitions of the term "Internet of Things" found in the literature, although different, have in common the fact that the Internet of Things is related to the integration of the physical world in the virtual world of the Internet. Thanks to his vision of the role that computing must play in the integration of man and his environment, Mark Weiser [2] has opened the way for Ubiquitous Computing [26] to integrate the real world, via its physical objects, into the digital world and to make it disappear in our everyday lives.

"Object" here is a generic term that covers all possible combinations of size, shape, material, implementation etc. A physical object is an object in the real world that performs a useful function for humans or in which humans have a particular interest in wanting to integrate it. The Internet of Things (IoT) was initially inspired by members of the RFID (Radio Frequency IDentification) development community who foresaw that most of the objects around us would be on the network in some form or another. They aimed for the ability to discover information about a tagged object by browsing an Internet address or database entry that corresponds to a particular RFID. In this way, everyday real-world objects acquire unique digital identities and can then be integrated into a network and associated with digital information or services.

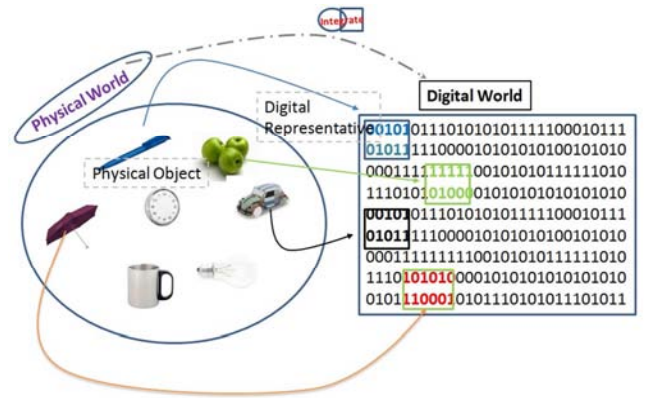


Figure 1. Scanning, a common to the different views.

To integrate a physical object in the digital world, that of

the Internet, it is necessary that this physical object has a digital equivalent (figure 1). This is a kind of digital representative that "digitizes" its characteristics and its knowledge. This digitalization is in relation to the interest and the point of view of the interested stakeholder (man, application...).

This means that, although there is usually only one physical object for each digital representative, it is possible for the same physical object to be associated with several different digital representatives. For example, it may be a different representation per application domain. This is done by ignoring the details and retaining only the concepts. Indeed, starting from the end, we like to see the interested party, the human being, interacting with the interesting party, the physical object (figure 2). We draw our models using an UML notation [23].

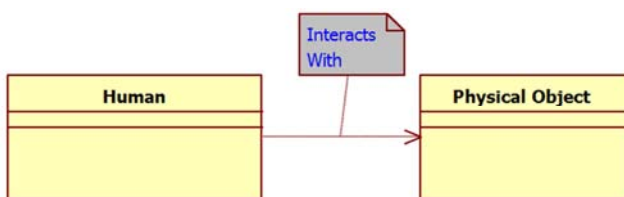


Figure 2. Making the human and the physical object interact.

The "MediaCup" project [3] remains, in spite of the time spent, a pilot project and certainly a pedagogical example of how to augment a physical object with digital capabilities. Indeed, the objective of the development of the MediaCup project was to augment an ordinary coffee cup with sensing, processing power, and communication capabilities. It is then a matter of controlling the acquisition of data from the temperature and motion sensors. When motion is detected, it is recorded as an event. A brief history of these events is used with a heuristic rule base. This allows detecting more abstract events that are related to the cup. These are the cup is stationary, the cup is moving, we are drinking from the cup, and we are playing with the cup.

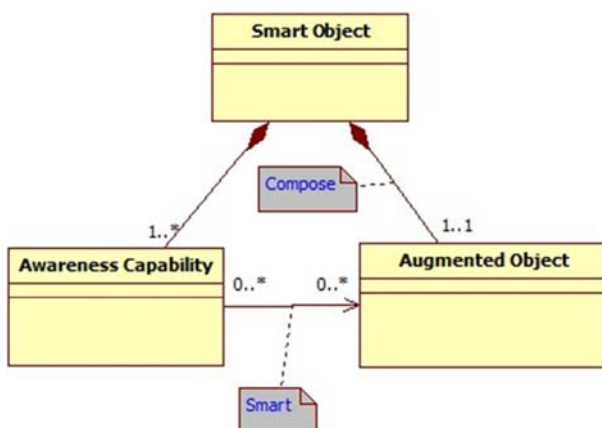


Figure 3. A smart object combines environmental awareness and an augmented object.

When an object possesses an awareness of itself and its surroundings that gives it an ability to understand (i.e.

capture, interpret and react to) human events and activities occurring in the physical world, it is said to be a "smart object" [21]. Everyday smart objects must do without human intervention, establish connections in a spontaneous way, self-organize and self-configure to adapt to a particular environment. Figure 3 shows the articulation and relationship between these objects.

The physical object "coffee cup" in MediaCup could determine whether it (the cup) is moving or stopped and whether the human was drinking or playing with it. The MediaCup experiment teaches us that to integrate a physical object into the Internet of Things, it needs to have a digital representative associated with. This digital representative is proportionally sophisticated to the depth of the projected integration. This relies, by the way, on the devices embedded in or attached to the physical object. Physical objects have different levels of integration. For example, a yogurt can have a RFID tag that is read by other objects like a smart refrigerator. A smart refrigerator [4] is a much more "complex" object than the yogurt; it has a catalog of actions that allows it to interact (see modeling in Figure 4) with people or other things like RFID tagged food. Today, we are not far from the vision of "Mark Weiser" who, already in 1999, defined an intelligent environment as "a physical world richly and invisibly interwoven with sensors, actuators, displays, and computational elements, embedded seamlessly in the everyday objects of our lives and connected through a continuous network" [5].

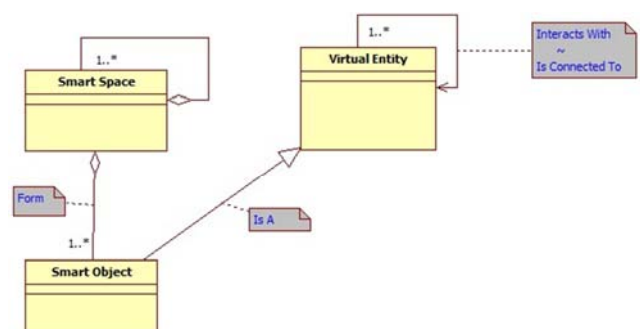


Figure 4. A connected smart object is an interactive environment.

An "intelligent environment" focuses on the intelligence of physical objects in relation to their environment, while an "intelligent space" is a formation of intelligent objects that emphasizes the usefulness and scope of this intelligence. As examples, we speak of smart greenhouses, smart houses and smart cities. In this context, the challenges of research and development to create an intelligent world are enormous. A world where the real, the virtual and the digital converge to create smart environments. A word that make energy, transportation, cities and many other areas smarter is a hoped-for world. In their August 2010 paper T. Lu and W. Neng argue, "The internet of the future will house objects with virtual identities and personalities operating in smart spaces using intelligent interfaces to connect and communicate in user, social and environmental contexts" [6].

3.3. LIDO and the Multiplicity of Views

An IoT project may bring together individuals from different backgrounds and working in different disciplines. "LIDO" allows these individuals to work on different views of the same IoT solution. The textual notation of "LIDO" allows these views to be exchanged and shared. "MODIDO", the reference meta-model of "LIDO", offers six (6) views for an "IoT" solution. Guillemin and Friess recommended these views in their 2009 paper [1]. For them, "The Internet of Things allows people and things to be connected Anytime, Anyplace, with Anything and Anyone, ideally using Any path/network and Any service"[1]. This definition, which we have retained, is sufficiently generic to borrow its meaning from the IoT to any application allowing people and physical objects to be connected and to interact regardless of the time, place and means used for this interaction. However, this definition alone is not sufficient to provide a basis for understanding IoT. "MODIDO" our IoT domain model has been established to shape and build such an understanding.

3.3.1. The "User" View

A user is someone (a human being) or something (application, augmented object) that takes advantage of the digital capabilities of a physical object to interact with it [19]. The human being through a simple browser or through one of its applications will be able to interact with a physical object and take advantage of its services. Examples of interaction consist in accessing the capture services of an object, or accessing the actuation services of an object. The interaction between users is only a sequence of accesses to the services offered by physical objects.

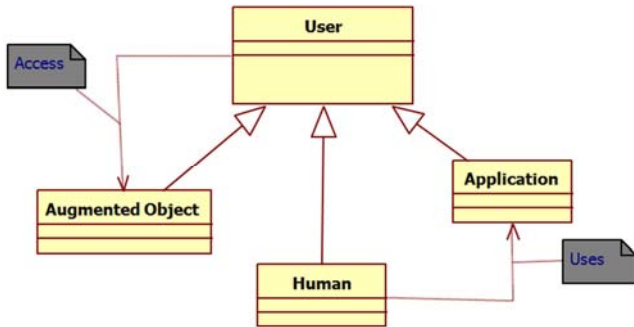


Figure 5. MODIDO user view.

3.3.2. The "Object" View

The progress and convergence of micro-electro-mechanical systems, wireless communications, and digital electronics have allowed the development of miniaturized devices with the ability to sense, compute and communicate wirelessly over short distances. These devices (Figure 6) have the advantage of providing a ubiquitous sensing capability. This is essential to the realization of Weiser's global vision [5]. Indeed, from ambient computing, heir of ubiquitous computing, to wireless communication networks, to sensor networks, to radio frequency identification, man has been able to unite, merge and melt these disciplines in order to make everything around him an object that can be

integrated into the "Internet of Things". By seeking to integrate the objects of the physical world, man seeks that these objects can "augment" him. For this to happen, these objects must be "augmented" first.

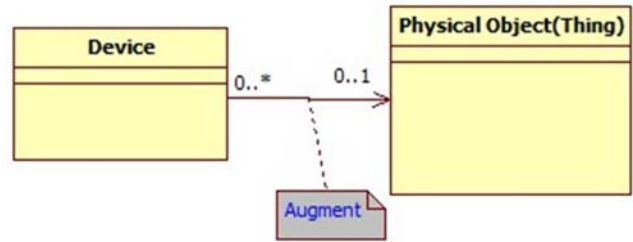


Figure 6. A device enhances the capabilities of a physical object.

An augmented object (Figure 7) in the IoT domain is an object that has its information capture, computation or communication capabilities augmented [20]. An embedded or attached device provides these capabilities (Figure 8).

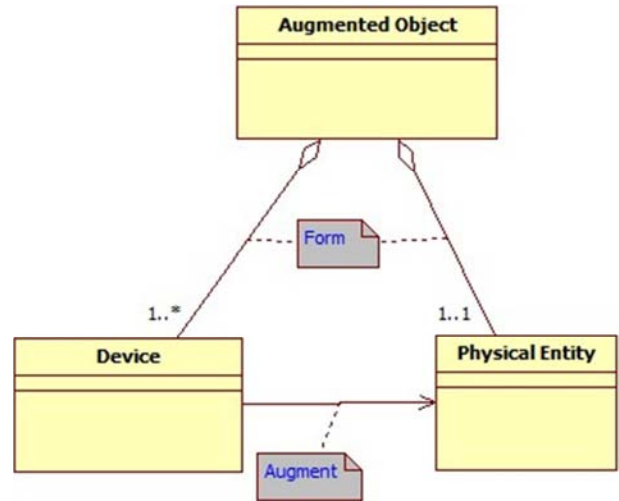


Figure 7. An augmented object, association of a physical entity and devices.

With respect to the level of detail, the domain model should separate what does not vary much from what does. For example, in the IoT domain, the concept of device is expected to remain relevant in the future, even though the types of devices used will change over time and/or vary depending on the application context. To this end, sensors, actuators, tag readers and other devices with specific functionalities and compositions are not represented as such but abstracted in the IoT domain model. Thus, we abstract the details of a device's composition. Understanding the characteristics of a single chip requires an understanding of the internal schema of the entire device, which is too complex to include in the model. A device is considered, here, as a black box whose composition details are hidden. Figure 8 shows, thus, that a device can have information capture, processing and communication capabilities that allow it to augment a physical object and that it can itself be part of a more complex device.

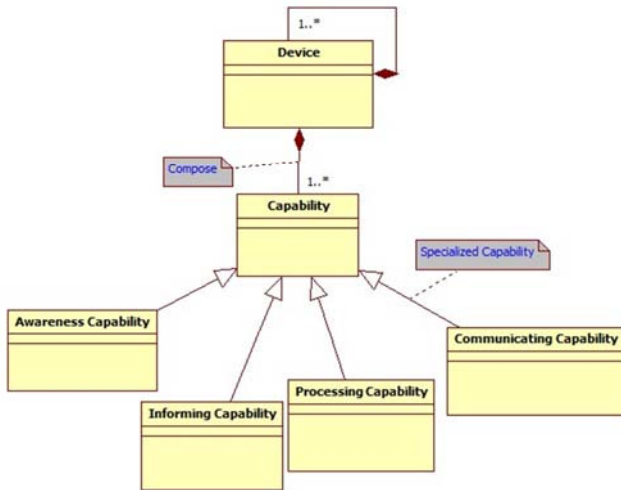


Figure 8. A device has ICT capabilities.

To return to the modeling of the "object view", figure nine takes up, groups and connects all the "concept objects" seen so far.

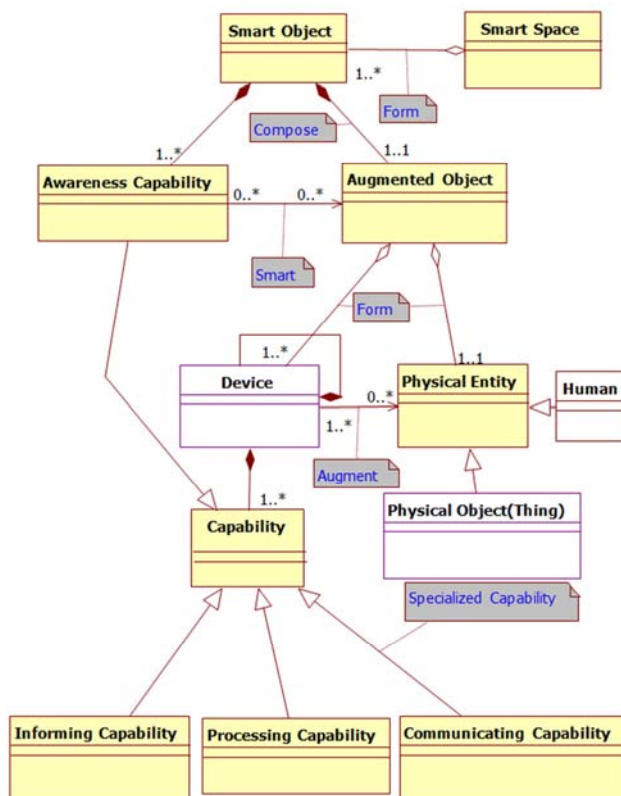


Figure 9. "Object" view of MODIDO.

3.3.3. The "Service" View

The service view is concerned with the provision in the form of services, by devices embedded in or attached to objects in the physical world, of functions for capturing information, processing this information, and performing various actions on these objects. A service can, moreover, be invoked by any automated business process or a process having an automated interface. Thus, physical objects

augmented by devices providing ICT¹ capabilities are at the service of any other connected object that knows how to invoke them. The concept of "Service" is abstract. It "ignore" its consumers and their ways of invoking it. It "ignore" its providers and their ways of exposing and serving it. It "ignore" any other intermediary of any kind, be it a protocol, a language, a mediator, a facilitator or any other. To do this, we can rely on the promise of "service-oriented computing" which aims at a world of loosely coupled services to flexibly create dynamic business processes and agile applications. These applications can span heterogeneous organizations and computing platforms. They can, quickly adapt to the changing mission requirements of those organizations.

Service-oriented computing (SOC) [7] uses services as building blocks in support of rapid, low-cost development and simple composition of distributed applications. SOC reflects a service-oriented approach to programming based on the idea of composing applications. This is achieved by discovering and invoking services available on the network rather than building new applications. Services are autonomous and platform independent computational entities. They are most often build in a way that is independent of the context in which they are used. This means that the service provider and consumers are loosely coupled. Realizing the promise of SOC, as mentioned above, requires the development of service-oriented architectures (SOAs) and corresponding middleware that enable the discovery, use, and composition of interoperable services to support virtually any business process.

(i). A form of Integration for More Services

Nowadays, software are programmed using high-level languages such as C++, Java or C#. Reuse remains limited despite significant progress in related techniques and concepts such as data abstraction, separation of specification from implementation, object-oriented classes, design patterns, code templates and object-oriented frameworks. SOC can be seen as an extension of these concepts. SOC takes the modularity of the object-oriented paradigm, the opacity (black box, interface/implementation separation), the location independence and the transport protocol neutrality of the "component-oriented" paradigm. The SOC paradigm [22] takes the service as a fundamental element for the development of applications/solutions. Current software has to cope with the heterogeneous nature of the Internet, addressing complex aspects such as distributed applications, integration of applications running on various platforms using different protocols and exploiting various devices. Service Oriented Architecture (SOA) is based on the interaction of loosely coupled software components that provide services regardless of how these components are constructed (development environment, languages, runtime environment...).

(ii). Large-scale Integration for More Services

The problem addressed by a service may be small or large. Therefore, the size and scope of the logic represented by the

¹ ICT: Information and Communication Technology.

(iii). ICT Integration for More Services

(iv). For an Urbanization of Services

global management have arisen. These challenges of integration on an Information System (IS) scale have been addressed with varying degrees of success by Information Technology (IT) departments through urbanization projects and IT master plans. The concept of Service Oriented Architecture (SOA) [8] has made it possible to crystallize best practices in terms of IS urbanization and integration and, by naming them, to provide a horizon for major rationalization and integration projects. ESBs (Enterprise Service Buses) [11] are middleware that allow, on the one hand, to ensure interconnection through transport and transformation facilitators and, on the other hand, to manage the mediation of communications and interactions between services and applications of an information system via adapters and interfaces based on standards (figure 10: Mediator). Although not essential, ESBs are nonetheless a high value-added component in the implementation of a mature service-oriented architecture (SOA).

(v). An Operable Synthetic Service Model

So far, we have studied the concepts related to a service without taking into account the operability or the programmability of the model to be produced. Nevertheless, thinking about the usefulness of concepts for the design and implementation of an IoT solution, some concepts will be hidden and others will appear in relation to the composition of the software to be made and its deployment. To this end, we propose a model that is lightened of some concepts and enriched by new ones. This model illustrated by the figure 10, takes again the concept of physical object increased by a device whose functions are encapsulated by a service. This device service extends a generic service. It can be a service that operates independently of enterprise business services. Such a service can be bundled with other device services to form an independent "mega-service" as well as being part of an integrated service that bundles device services, enterprise business process services, and communication mediators or facilitators. Both independent and integrated services are deployable on nodes (processor machines).

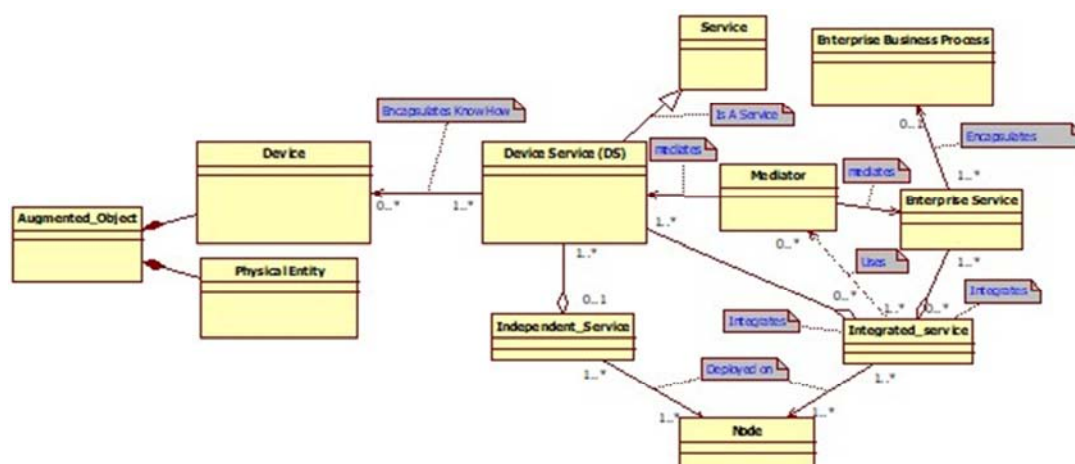


Figure 10. Operable Service Model.

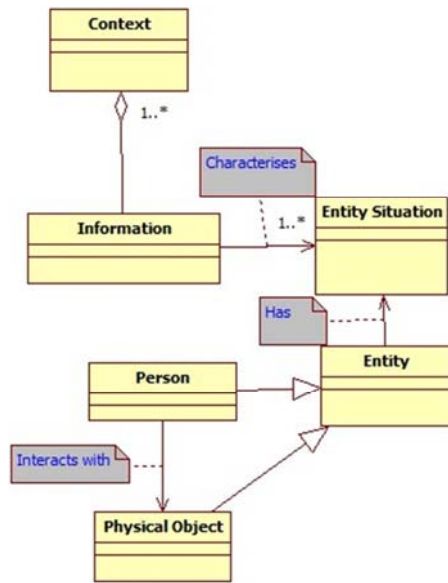


Figure 11. The context is any information that can be used to characterize the situation of an entity.

3.3.4. The “Context” View

To understand any scene, such as an interaction between a human and a system, one must understand the rules that govern this interaction and the behaviors of the human and the system. To deduce these rules and the reasons for the behaviors or attitudes of the participants in an interaction, one must know the context in which this interaction took place. At this stage, we can confuse context with information to situate an interaction. In order to evaluate the situations in which these participants act, one must know the context. This evaluation depends qualitatively (accuracy) and quantitatively (completeness) on all the information characterizing a situation. "The context is any information that can be used to characterize the situation of an entity. An

entity is a person, a place or an object that is considered relevant to the interaction between a user and an application, including the user and the applications themselves" [12]. This definition, which we retain, considers information in general as the main element in the identification of a context. Figure 11 shows this definition of context.

Given the diversity of context information, it is useful to attempt to categorize it to make the context easier to understand in a systematic way. To this end, A. K. Dey, G. D. Abowd, and D. Salber [13], introduce a simple classification of context information based on the entities against which the context is evaluated. They also classify the categories of context information. They introduce four essential categories or features of context information: identity, location, state (or activity) and time. Identity refers to the ability to assign a unique identifier to an entity. The identifier must be unique in the namespace used by applications. Location is more than positional information in 2-D space. It is extended to include orientation and elevation, as well as any other information that can be used to infer spatial relationships between entities, such as co-location, proximity, or containment. Status (or activity) identifies the perceived intrinsic characteristics of the entity. Thus, for a place, it may be the current temperature, light level, or ambient noise. Similarly, for a group of people, status is a characteristic of the group such as their enthusiasm or overall mood, or a description of their activity, such as attendance at a conference or meeting.

A context attribute is an element of the context model describing the context. A context attribute has an identifier, a type and a value, and possibly a collection of properties describing specific characteristics [14]. This introduces the notion of template, which considers that a context model is formed of context attributes. Figure 12 diagrams a context model and its template.

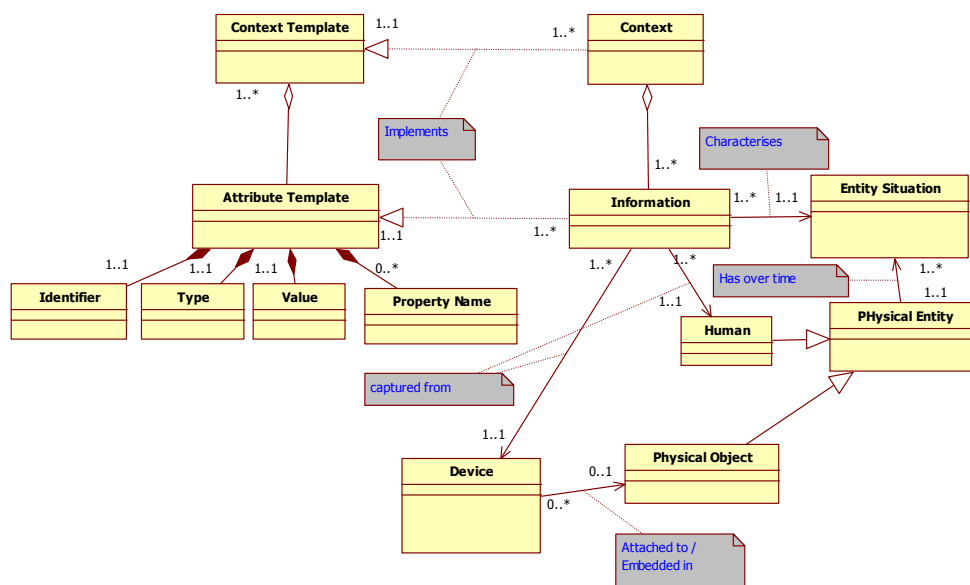


Figure 12. A context model and its template.

3.3.5. The “Network” View

The Internet of Things can be seen as a network of connected objects interacting with each other or with users of the network (human, application). Augmented objects (by devices: sensors and actuators attached or embedded) use communication protocols (figure 13).

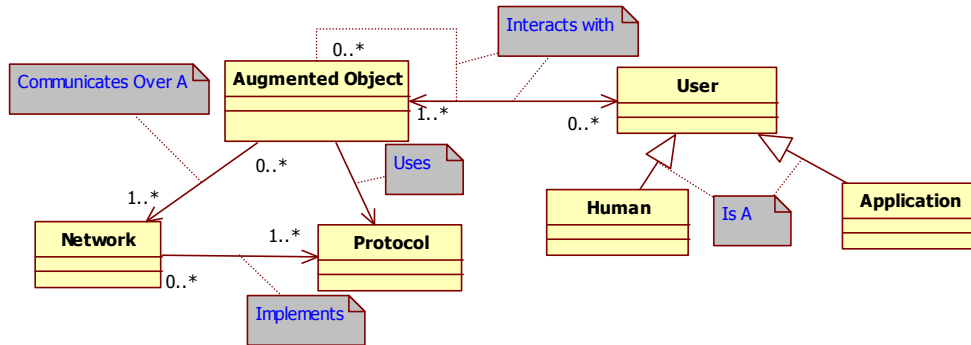


Figure 13. The IoT can be seen as a network of connected objects.

The Internet of Things can otherwise be seen as a collection of resources accessible through paths (Figure 14). Each resource is identified by a URI (Unified Resource Identifier). This URI defines a path used by a user (human, application) to access the resource. The access to the resource

can be in the form of a service that makes the possibilities of this resource available. In the case of a physical object, the service can offer the possibilities of information capture and actuation of devices embedded or attached to the physical object.

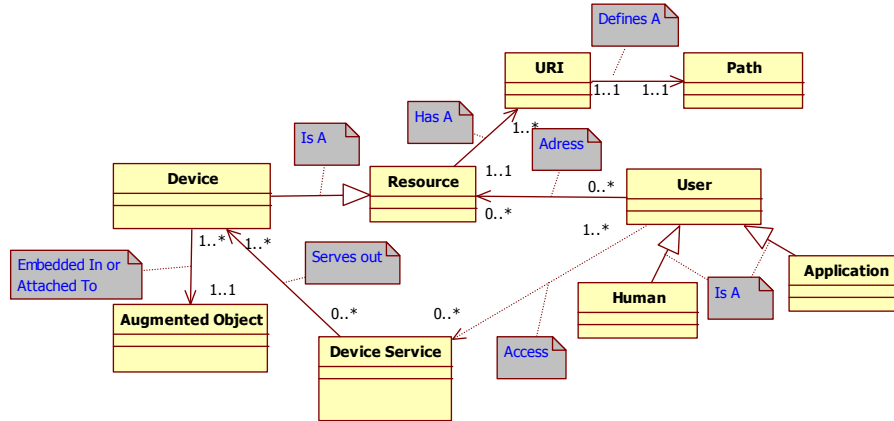


Figure 14. The IoT is a collection of resources accessible through a path.

To schematize the two sides of the IoT as a network of connected objects and as a network of resources, it is sufficient to consider the IoT as an extension of the internet

(global network) to physical objects accessible by different technological means. Figure 15 illustrates the integration of the two sides (Figures 13 and 14) mentioned above.

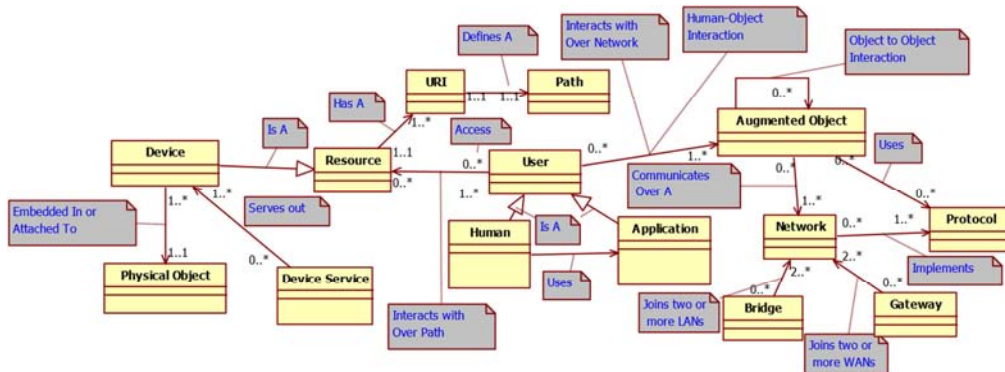


Figure 15. The network view model, supporting interactions.

3.3.6. The Location View

Location plays a key role in determining the type and nature of human activity. Location can determine consumers' information needs and their choices of products and services. If a mobile service provider knows the exact location of the user, and is able to target useful (and billable) information based on the user's location and time space, the interest can be shared. Thus, one of the contributions of location is the targeting of specific needs and the offering of the studied solutions.

The techniques and means of location are many and varied. The Global Positioning System (GPS) is the most widely used satellite positioning system, which offers maximum coverage. GLONASS is the Russian equivalent of the American GPS. Galileo is the European version. Today, many cellular or wireless networks exist. These networks communicate with mobile equipment by radio. Location-based services are any service that takes into account the geographical location of an object. The METRO Group [15], based in Germany, after a pilot project "Tag it Easy" with

suppliers, warehouses and retail stores using different generations of RFID (Radio Frequency IDentification) to track pallets throughout the process chain, has completed its deployment at 180 locations. RFID technologies aim to identify something using a passive tag that responds to a radio frequency signal.

These location techniques and means are intended to provide a location service.

A device embedded in or attached to a physical object can move because the physical object is mobile. Some devices can be made to change their positions, autonomously. The service that a device can provide is information capture and actuation. The subcontracted system is a positioning system that allows determining the position of a physical object or a human being. This system provides location information. This information may or may not influence the nature or characteristics of an interaction. This information can be used by a location-based service to provide the human being with information related to his context (figure 16).

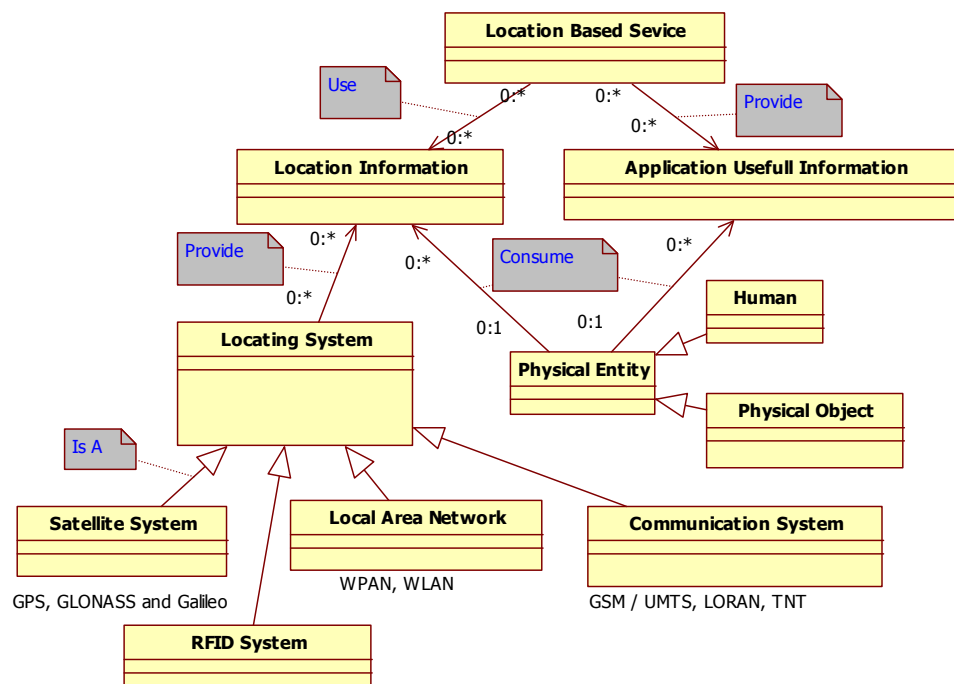


Figure 16. The location View Model.

3.4. LIDO's Textual Notation

Once our meta-model, MODIDO, has been adopted to constitute the abstract syntax of our language (LIDO), we must seek to define its concrete syntax (Figure 17). This syntax proposes a grammar that specifies the allowed constructs according to a textual notation in which an IoT solution can be written. A graphical notation is visual and efficient for representing relationships between elements, which was chosen for the abstract syntax. A textual notation makes it easier to describe and edit details. We opt for a textual notation and we call it NTIDO.

The first objective of NTIDO is to offer to a developer a language means that allows him to express his solution derived from MODIDO. NTIDO also serves as a gateway for communication and exchange of solution descriptions between developers of the same or different environments. It represents, again, a language intermediary between a design and a code generation.

NTIDO as a language has been designed using the Backus-Naur Form (BNF²) metalanguage.

² Backus-Naur Form: www.w3.org/Notation.html.

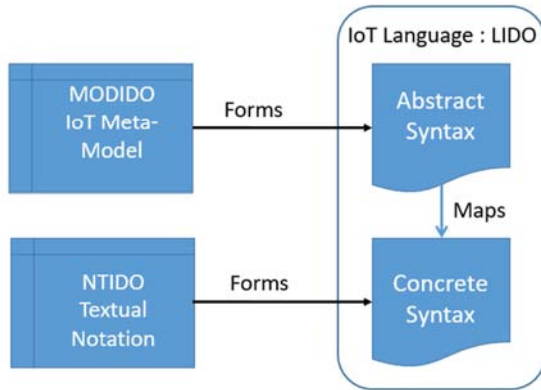


Figure 17. MODIDO and NTIDO component of LIDO.

4. The INDIT Development Environment

4.1. Purpose and Objectives of INDIT

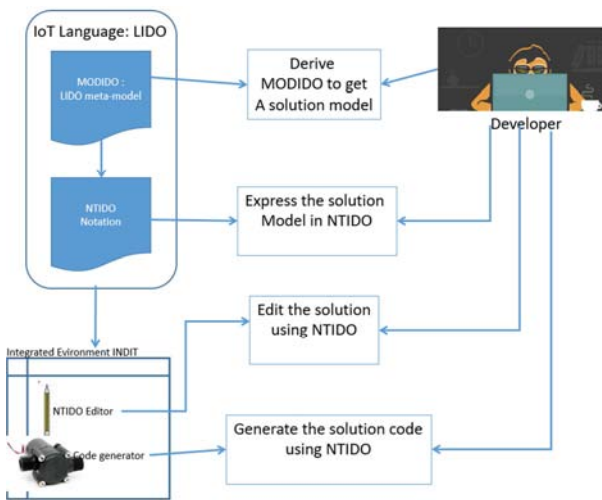


Figure 18. From the LIDO language to the INDIT environment.

In order to support our development process, we have created a development environment called INDIT. This environment makes it possible to use the LIDO language through its textual notation to reinforce the research and realization of IoT solutions by a non-homogeneous team of stakeholders. In fact, these stakeholders have different profiles and concerns. What connects them is none other than their knowledge and sharing of LIDO. Once the solution model is derived from MODIDO, each of the participants scores their part of the solution in NTIDO. To help each participant write down his or her part of the solution, the INDIT environment offers an NTIDO language editor. This tool allows describing in detail the different components of the solution. The editor offers syntax highlighting. Typed keywords are displayed in color. It also offers auto-completion. Once the description of the solution is edited, it is saved in a text file that can be exchanged to share parts of the solution or even complete solutions. Another tool is launched as soon as the "ctrl-c" key is pressed. This tool is a code generator of the solution already noted in NTIDO.

When the noted description does not conform to NTIDO, error messages are generated to warn the participant. Given the popularity of the Java language, we have chosen to generate code in Java. Figure 18 shows the place of INTID in the IoT solution development process.

In order to enable code transformation or generation, we need to provide developers with an integrated development environment (IDE) that implements our LIDO language. To this end, we share Bettini's thinking and approach to the implementation of a programming language [16].

4.2. Development of INDIT

The implementation of a language, even if only in prototypical form, is still very time consuming. Writing the parser using a compiler generator (such as Flex & Bison [17] and ANTLR [18]), building the Abstract Syntax Tree (AST), performing all the visits to the AST (e.g., type checking), and finally generating code in a target language or building the interpreter are all time consuming tasks. To save this time, we have chosen to use Eclipse and Xtext Plug-in. Eclipse/Xtext is a general framework for the development of programming languages and domain specific languages. It covers all aspects of a complete language infrastructure, from parsers, linkers, compilers or interpreters to the complete Eclipse IDE integration.

5. Application & Validation: Case Study

We have presented the INDIT integrated development environment that provides a practical component to the approach. In this part, we are interested in experimenting the approach. We have chosen a case study from the agricultural domain, in particular precision agriculture (smart agriculture [24]), which we present in the following paragraph.

5.1. The Case Study Description

Italy is the second largest producer of kiwifruit after China with a saleable production of about 448,000 tons a year. The two countries, together with New Zealand, Chile and Greece, account for 93% of world production, according to research by the Centro Servizi Ortofrutticoli³ (CSO).

CSO informed in 2015 that the growing area of kiwifruit in this Mediterranean country of Italy was 24,440 ha, mostly in the regions of Lazio, Piedmont and Emilia Romagna.

FAMOSA⁴ is an Italian company that works on technological support for agriculture, offering solutions for monitoring and management of crops. They have developed wireless sensor networks, based on Libelium⁵ technology, in a kiwi plantation with GPRS and Sigfox to develop accurate irrigation strategies for farmers.

Kiwifruit is one of the most sensitive fruits in terms of quality, which is given by size and sweetness. To achieve the best quality, agricultural organizations consider it essential

3 CSO- www.csoservizi.com/.

4 FAMOSA- <http://www.famosasrl.com/en/>.

5 Libelium-www.libelium.com.

for farmers to develop a good irrigation strategy to obtain marketable products and reduce product losses.

Agricultural organizations are deploying advanced technologies in their plantations to improve crops, maintain high quality standards, and make farmers' daily work easier. For this project, the company installed two wireless sensors to monitor soil water status to schedule irrigation in a kiwi orchard. Data was recorded with the same system but information was transmitted to the platform through two different wireless connections: GPRS and Sigfox. The main objective was to test the reliability and costs during the kiwi-

growing season in 2016.

Two Waspote Plug & Sense! Smart Agriculture were deployed with sensors at different depths to monitor soil moisture with a fruit diameter sensor (dendrometer) to measure fruit size and temperature and humidity sensors to monitor environmental conditions (Figure 19). One of the sensor platforms is connected to a GPRS display and the other with Sigfox. The former represents the widely used conventional data communication network and the latter represents the rapidly diffusing LPWAN technologies (Figure 19).

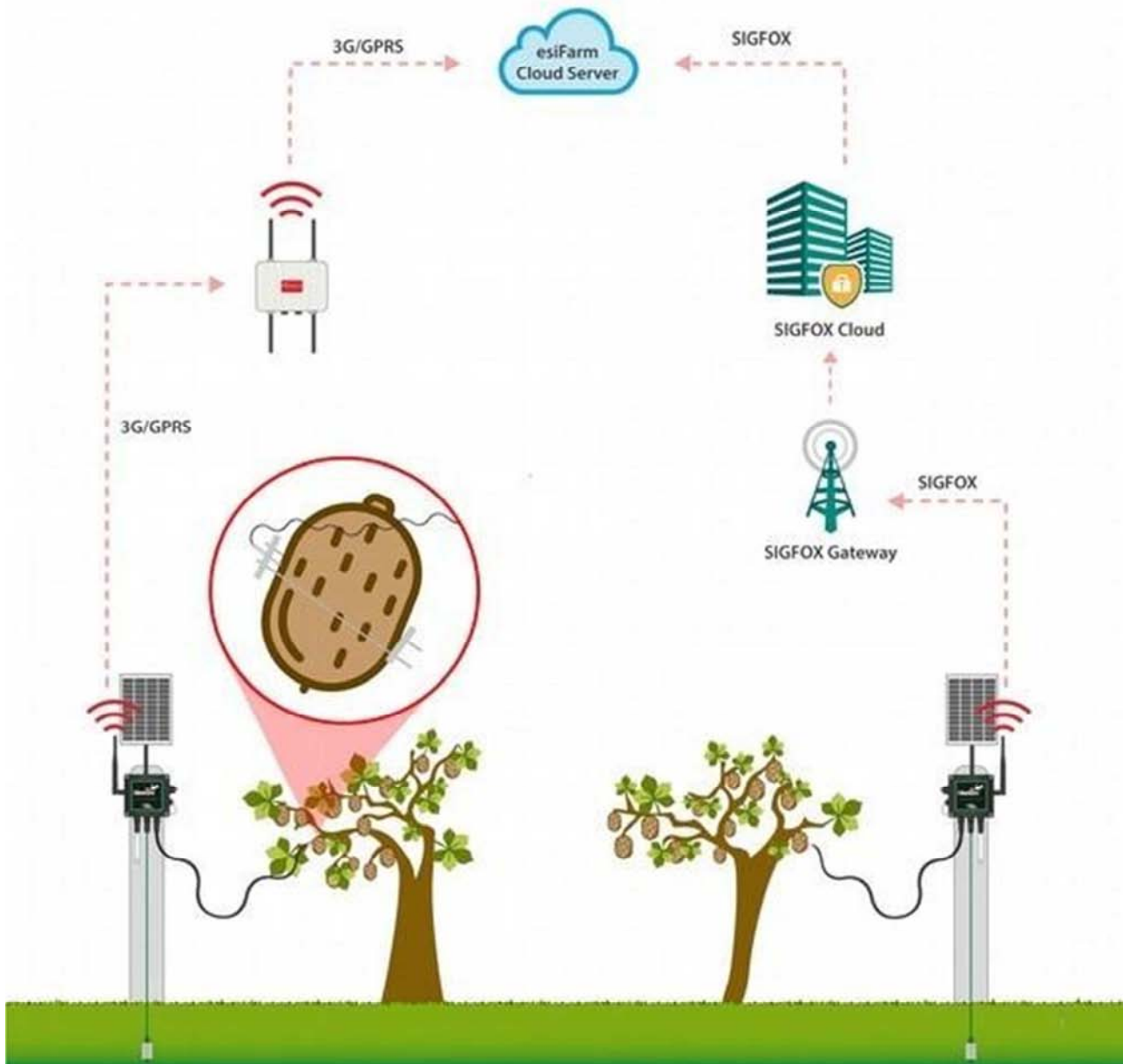


Figure 19. FAMOSA deployment block diagram.

The solution proposed by FAMOSA consists in collecting data from sensors (Temperature, Air humidity, Soil humidity, Dendrometer) integrated in the Waspote Plug & Sense! Smart Agriculture equipment. This data is transported to the

Esifarm server through SIGFOX or 3G-GPRS bridges. Farmers through their PCs or cell phones can access the Esifarm application to check on the status of their fields and schedule irrigation when they deem it necessary. The general

scheme of this hardware and software infrastructure is given in Figure 20.

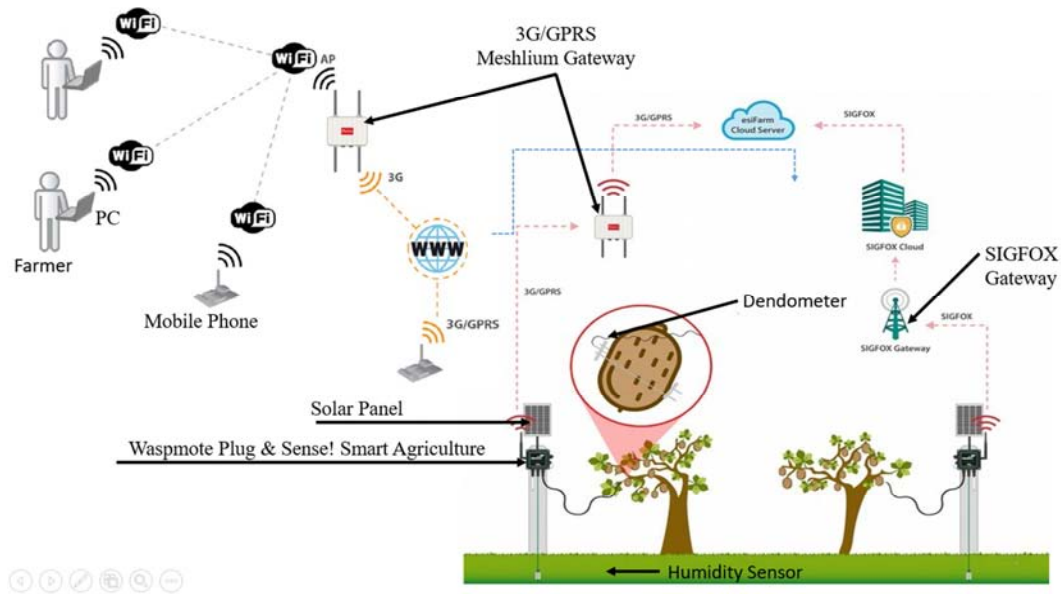


Figure 20. General diagram of the solution.

5.2. Modeling the Solution

According to our approach, the first step to be performed by a developer is to derive MODIDO to create a model of his solution. In what follows, we describe the experimentation of this step for the case study. In order not to overload the article, we will limit ourselves to presenting the models of

the "User" view and the "Network" view.

5.2.1. The "User" View

In this view, EsiFarm being an "Application" and The Farmer, being a "Human" using EsiFarm, both derive from "User". "User" accesses (uses) the augmented objects "Atmosphere", "Soil" and "Kiwifruit" (Figure 21).

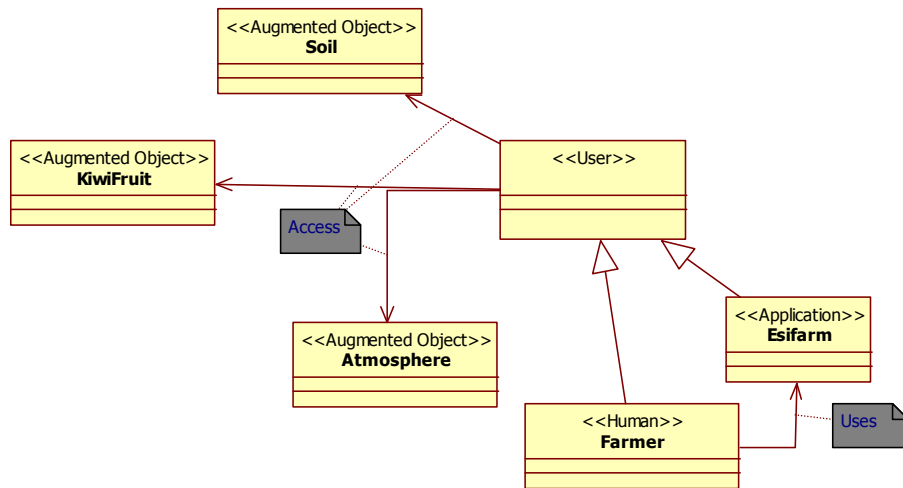


Figure 21. User view model of the solution.

5.2.2. The "Network" View

There are two types of networks: a 3G/GPRS⁶ network and a SIGFOX⁷ network. The local sensor networks are connected to the remote sites via gateways. We distinguish between Meshlium⁸ gateways (3G/GPRS) and SIGFOX gateways. The EsiFarm application accesses the augmented objects connected to these networks (Figure 22).

6 GRPS- <https://www.unescwa.org/general-packet-radio-service>

7 SIGFOX- <https://www.sigfox.com/en>

8 Meshlium-<https://www.libelium.com/iot-products/meshlium/>

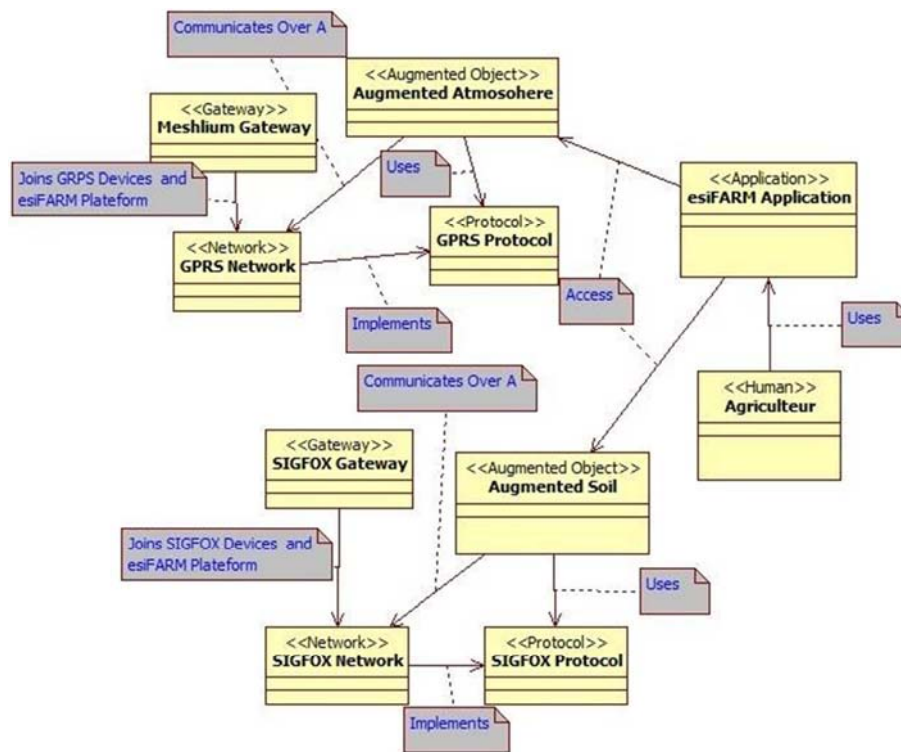


Figure 22. Networks and Gateways view of the solution.

5.3. Solution Scoring

After deriving MODIDO to obtain the solution model, we described this solution using Libelium⁹ technology hardware and scored it as NTIDO. We then used the INDIT environment editor to edit it.

5.4. Code Generation

The INDIT technological environment that we imagined designed and developed (prototyped) was able to generate twelve (12) packages comprising seventy-five (75) classes. The classes specific to the case study are 32 in number. Each class has a number of lines of code ranging from 20 to 128.

6. Conclusion

The development approach we propose is based on a language specific to the IoT domain. It proposes a development demarche that leads to a skeleton implementation starting from a design model derived from the domain meta-model and expressed in a textual notation. Our language and our approach would not be useful without a practical counterpart materialized by an integrated development environment (IDE).

The case study is entitled "Intelligent irrigation system for the cultivation of Kiwi fruit in Italy". The objective of the project was to optimize the use of water for the irrigation of Kiwi plantations. This optimization was aimed at limiting the quantity of water and dosing it for a better cultivation of

Kiwifruit in terms of both quality and quantity. The solution consisted in equipping these plantations with digital supplies forming networks of sensors accessible through the Internet by means of a computer application.

We were able to experiment with the language, approach and tool for this case study. We found that the concepts are not difficult to grasp and to use to build a model of a solution. However, the practice of the approach is within the reach of everyone only if a learning to a textual notation close to the one of Backus Naur Form is ensured. As for the generation of code for the solution, the integrated development environment was able to provide satisfaction in terms of the quantity of code generated and the quality of the data structures it was able to provide. However, further development of the code generation should be planned if we aspire to go beyond the prototypical nature of what is generated.

Following on from what has been achieved in this work, we plan to look at a new domain model and integrated development environment specific to the agriculture sector using artificial intelligence, ambient intelligence and humanized computing. We believe that digital has an important role to play in the modernization of agriculture. The example we have experimented in this work concerns the optimization of water usage and dosage for the cultivation of Kiwi fruit.

References

- [1] Guillemin P. and Friess P. "Internet of things strategic research roadmap" The Cluster of European Research Projects, Tech. Rep, September 2009.

⁹ Libelium: www.libelium.com.

- [2] Weiser Mark. "The Computer for the 21st Century" - Scientific American Special Issue on Communications, Computers, and Networks, September 1991.
- [3] Beigl, M., Gellersen, H.-W. and Schmidt, A. MediaCup. Experience with Design and Use of Computer Augmented Everyday Objects. *Computer Networks*, Vol. 35, No. 4, Special Issue on Pervasive Computing, Elsevier, March 2001, pp. 401-409.
- [4] Espada J. P., Martinez O. S., Garcia-Bustelo B. C. P. & Lovelle, J. M. C. Virtual Objects on the Internet of Things. *International Journal of Artificial Intelligence and Interactive Multimedia* 2011, 1 (4): 24-30.
- [5] Weiser Mark, Gold R. The origins of ubiquitous computing research at PARC in the late 1980s, *IBM Systems Journal* (1999).
- [6] Lu T. and Neng W. Future internet: The internet of things, in 3rd International Conference on Advanced Computer Theory and Engineering (ICACTE), vol. 5, August 2010, pp. V5-376-V5-380.
- [7] Michael P. Papazoglou, Paolo Traverso, Schahram Dustdar, Frank Leymann "Service-Oriented Computing: A Research Roadmap" *International Journal of Cooperative Information Systems* Vol. 17, No. 2 (2008) 223-255.
- [8] Thomas Erl, "Service-Oriented Architecture: Concepts, Technology, and Design", Pearson Education, 2005.
- [9] L. M. S. de Souza, P. Spiess, D. Guinard, M. Koehler, S. Karnouskos, and D. Savio. Socrades: A web service based shop floor integration infrastructure. In *Proc. of the Internet of Things*. Springer, 2008.
- [10] Xebia IT Architects SAS: White Paper - Understanding and Using an ESB in an SOA.
- [11] David A Chappell, *Enterprise Service Bus: Theory in Practice*, "O'Reilly Media, Inc.", June 25, 2004.
- [12] G. D. Abowd, A. K. Dey, P. J. Brown, N. Davies, M. Smith, and P. Steggles, "Towards a better understanding of context and context-awareness," in *Proceedings of the 1st international symposium on Handheld and Ubiquitous Computing*, ser. HUC '99. London, UK: Springer-Verlag, 1999, pp. 304-307.
- [13] A. K. Dey, G. D. Abowd, and D. Salber, "A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications," *Hum.-Comput. Interact*, vol. 16, pp. 97-166, December 2001.
- [14] K. Henriksen, "A framework for context-aware pervasive computing applications," *Computer Science*, School of Information Technology and Electrical Engineering, The University of Queensland, September 2003.
- [15] ElAmin, A. Metro Group completes Europe's largest RFID rollout, 2007.
- [16] Bettini, L.: An Eclipse-based IDE for Featherweight Java implemented in Xtext. In: *ECLIPSE-IT*, pp. 14-28 (2010).
- [17] J. Levine. *flex & bison*. O'Reilly Media, 2009.
- [18] T. Parr. *The Definitive ANTLR Reference: Building Domain-Specific Languages*. Pragmatic Programmers, May 2007.
- [19] T. Ben Hassine, O. Khayati and H. Ben Ghezala, "An IoT domain meta-model and an approach to software development of IoT solutions," *2017 International Conference on Internet of Things, Embedded Systems and Communications (IINTEC)*, Gafsa, Tunisia, 2017, pp. 32-37. doi: 10.1109/IINTEC.2017.8325909.
- [20] T. Ben Hassine, O. Khayati and H. Ben Ghezala, "LIDO a modeling language for developing solutions for the internet of things," *Digital Tools & Uses Congress*, Paris, 3-5 October 2018.
- [21] pp Kortuem G., Kawsar. F., Sundramoorthy V. & Fitton D. Smart objects as building blocks for the internet of things. *IEEE Internet Computing* 2010, v 14, n 1, (44-51).
- [22] Papazoglou, M. P., *Service-Oriented Computing: Concepts, Characteristics and Directions*, in *Fourth International Conference on Web information Systems Engineering (WISE)*, pp. 3, 2003.
- [23] Fowler Martin "UML Distilled: A Brief Guide to the Standard Object Modeling Language", Addison-Wesley Professional, 3rd Edition, 2003.
- [24] M. Ayaz, M. Ammad-Uddin, Z. Sharif, A. Mansour and E. -H. M. Aggoune, "Internet-of-Things (IoT)-Based Smart Agriculture: Toward Making the Fields Talk," in *IEEE Access*, vol. 7, pp. 129551-129583, 2019, doi: 10.1109/ACCESS.2019.2932609.
- [25] Kumar, S., Tiwari, P. & Zymbler, M. Internet of Things is a revolutionary approach for future technology enhancement: a review. *J Big Data* 6, 111 (2019). <https://doi.org/10.1186/s40537-019-0268-2>.
- [26] S. A. Alshqaqi, A. T. Zahary and M. M. Zayed, "Ubiquitous Computing Environment: literature review," *2019 First International Conference of Intelligent Computing and Engineering (ICOICE)*, 2019, pp. 1-8, doi: 10.1109/ICOICE48418.2019.9035157.