A New Formalisation for Wireless Sensor Network Adaptive Context-aware System: Application to an Environmental Use Case

Jie Sun, Gil De Sousa, Catherine Roussey, Jean-Pierre Chanet, François Pinet

Irstea, UR TSCF
9 avenue Blaise Pascal
CS 20085
F-63178 Aubière, France
Email: jie.sun@irstea.fr, gil.de-sousa@irstea.fr, catherine.roussey@irstea.fr
jean-pierre.chanet@irstea.fr, francois.pinet@irstea.fr

Kun-Mean Hou

LIMOS
Campus Universitaire des Cézeaux
1 rue de la Chebarde
TSA 60125 - CS 60026
F-63178 Aubière cedex, France
Email: kun-mean.hou@isima.fr

***Abstract*—Henceforth, new generations of Wireless Sensor Networks (WSN), as part of the Internet of Things (IoT), have to be able to adapt their behaviour to collect, from the study phenomenon (or feature of interest), quality data for long period of time. In this article, we propose a new formalisation for the design and the implementation of context-aware systems. To illustrate this whole proposal, an environmental use case, the study of flood events in a watershed, relying on a WSN for the data collection, is presented.**

Keywords–Context-aware system; formalisation; architecture; Wireless Sensor Network; Internet of Things; environment; phenomenon.

I. INTRODUCTION

The acquisition of heterogeneous data is essential in the era of IoT and Big Data that is just starting. These two research topics have application in numerous fields: industry, "smart home", "smart care", agriculture, environment, etc. WSN technology is now viewed as part of the IoT [1]. The increased use of WSN envisioned at the beginning of the 2000's [2], is now a reality as shown, for example, in environment [3] and agriculture [4]. In these applications, a WSN collects natural phenomenon observations (temperature, humidity, etc.) and sends them to a context-aware system, which may propose adaptation actions based on context. To build a full context adaptation service, information about wireless sensors themselves such as their energy levels are also required. Indeed, despite steady progress in hardware (the development of low energy communication modules for example), a wireless sensor still has scarce resources. It is the case for "scalar" WSN and it is even more for Wireless Multimedia Sensor Networks (WMSN) [5]. Thus, to better use these limited resources, all the components that are part of the data acquisition process have to work together in a cooperative way, from the component that collects raw data to the one that provides indicators to end users. Generally, these components are the wireless sensors, the gateway(s) and the remote Decision Support System (DSS). The acquisition and transmission frequencies required by the DSS, through the gateway, have to be consistent with the energy available at the level of the wireless sensors. For some alert applications such as fire prevention, data transmission is sometimes more important than the "survival" of a node of the network. Thus, all the components implied in the data acquisition process have to adapt their behaviours to the context in order to achieve the best performances. A WSN is also subject to unpredictable events that, without fast interventions, can threaten the stability of the whole system. The combination of the common decisions and actions is the issue addressed in this paper. More precisely, we propose a formalisation to define high level context which, integrated into an adaptive context-aware system, will be used to reduce the number of exchanged communication packets. Our formalisation proposes different reasoning steps in order to build the high level context. Section II presents the main existing concepts related to context-aware systems. Section III of the article explains our proposal of formalisation of context in order to build any context system based on WSN. Section IV shows its application with the design of a context-aware system dedicated to a complex environmental use case. Section V presents our implementation of this formalisation in order to develop a context-aware system and an adaptive context-aware one focused on the previous environmental use case. The second system adapts its behaviour to the context in order to limit packet exchanges. Section VI describes different context systems developed for the same purpose. The last section concludes this article.

II. CONTEXT-AWARE SYSTEM MAIN CONCEPTS

One of the most known and accepted definitions of context is given by [6] as *Context is any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves.* As indicated in this definition, context is focused on one entity. Several contexts can be defined, for example, the context of the user, the context of the device running the application, and so on. As explained in [7], different categories of context exist. **Low level context** corresponds to the raw data acquired by sensors or static data provided by users. **High level context** is computed from the low level one, with more informative data associated to the application and the user. Figure 1 presents the processes associated to an adaptive context-aware system when data are collected by a WSN. It could also be applied to sensor networks or other systems that generate raw data.

 ISBN: 978-1-61208-490-9

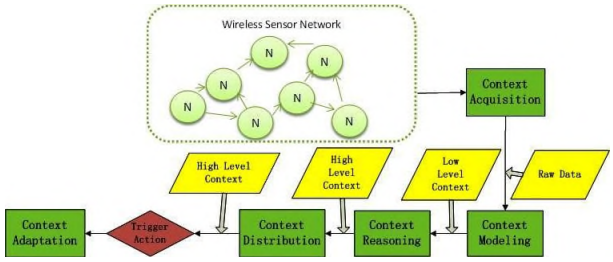

Figure 1. Context cycle of an application based on WSN

In an adaptive context-aware system, different processes are required:

- Context acquisition: collecting raw data and metadata that are useful to build the context.
- Context modeling: organisation of the collected data through a specific context data model. The process gives an interpretation to each raw data. For example, the value 24 becomes the measurement of the outdoor temperature in degree Celsius. This process builds the low level context. This process is also called annotation or tagging [7].
- Context reasoning: the high level context is computed or inferred from the low level one. This process can imply different approaches based on machine learning [8] or rule engine [9].
- Context distribution: diffusion of the high level context to the different consumers, for example, the end user or any system components that are able to adapt their actions according to the current context.
- Context adaptation: actions to adapt any system components according to context changes.

Notice that a context-aware system stops at the context distribution process and sends alert to the end users.

III. PROPOSED CONTEXT FORMALISATION

The work in [10] defines the concept of entity "state" as *a qualitative data which changes over times (summarizing a set of information)*. Based on this definition, we propose a new definition of "context" as *a set of entities characterized by their state, plus all information that can help to derive any state changes of these entities*. In our context-aware system formalisation, we add the definition of two classes of entities:

1) **observable entity**: entity that is directly observed by sensors.
2) **entity of interest**: entity whose characterisation is obtained from one or many other entities and required by the application.

We propose two new reasoning steps to create the high level context in the reasoning process, illustrated in Figure 2. Rules based reasoning is often used to deduce high-level context [7]. As far as we know, our works are the first to promote the division of rules in several reasoning steps in order to make the management of rules easier. Indeed, state of an entity of interest cannot be acquired directly based on low level context. Two levels of reasoning are presented to build the high level context:

- The low level context contains the sensor measurements stored in the context data model. The state of observable entities is inferred from the low level context as indicated by the dotted arc in Figure 2. The high level context contains the state of observable entities.
- The state of an entity of interest is inferred from the state of other entities. The high level context is enriched by the state of the entity of interest.

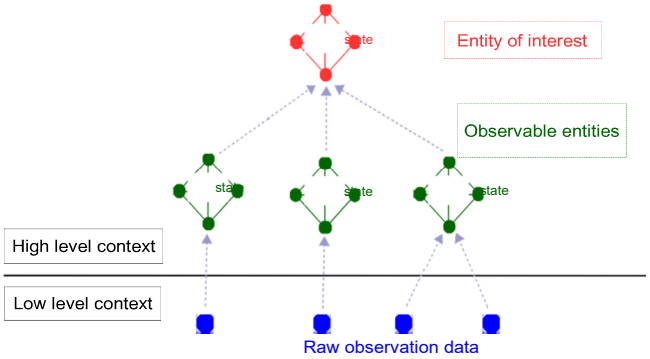

Figure 2. Classes of entities

If we take the example of a wireless sensor management application, we consider a wireless sensor as an entity of interest where one of its associated observable entities is its power supply (a battery). From this observable entity, a sensor measures a charge/an energy level as a raw data observation or low-level context. Based on capacity and charge values, we deduce the percentage of energy remaining in the battery. This percentage is represented by the variable *Energy*. Figure 3 presents an example of finite-state machine used to deduce the energy state (high, middle or low) of the battery which is included in the high level context.

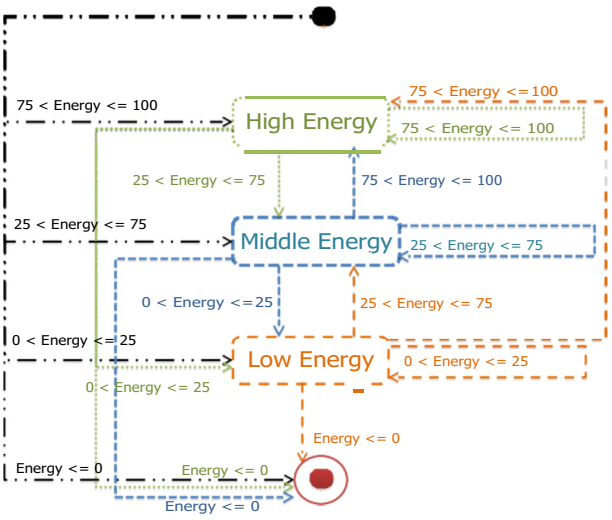

Figure 3. Example of wireless sensor energy finite-state machine

The number of states and entities depends on the application requirements. Any hardware component, such as memories of a wireless sensor, can be added in our context formalisation. It can also be applied to software such as the operating systems. The number of system failures or watchdog calls constitutes raw observation data to derive states of the associated observable entities. The state of a wireless sensor will then be deduced from all the considered (hardware and software) entity states. These different entities and their states enrich the context. However, the complexity of the deduction process increases with the number of relevant entities analysed. Providing entities and states in limited number is essential to have a highly dynamic context-aware system but this should not be at the expense of the quality of the final application decision. In the WSN topic, another possible use of this formalisation is for link quality evaluation application. This problem is well-known in routing protocols. Different metrics can be considered such as the available bandwidth, the latency, the available energy in the neighbourhood nodes and others [11]. The Quality of Service (QoS) of a link can be deduced from different observable entities such as the connected nodes and the bandwidth. Our context formalisation can be improved depending on the complexity of the application requirements. For example, a wireless sensor management application wants to evaluate if a wireless sensor can communicate. The states of the wireless sensor entity (the entity of interest) are "able to communicate" or "not able to communicate". Its state will be deduced from the link entity and the battery entity (enough energy to communicate). For a WSN management application, its entity of interest is the WSN. The state of the WSN could be computed from the states of all its wireless sensors (or nodes). Its connectivity state could be calculated from the QoS of all the links between its wireless sensors. At the end, the application can just need the states of the WSN, established from the states of all its nodes and of the gateway(s). Thus, we divide the context and the reasoning in several parts. Each part can be supported by different components of the context-aware system. If the two steps of the reasoning process is supported by two different components, the first component that deduces the state of an observable entity can communicate it to the second component. For performance reason, it would be better to communicate only the changing state events (with the associated value). In the following section, a context-aware system is built for an environmental use case. We experiment our formalization on it.

IV. ENVIRONMENTAL APPLICATION USE CASE

The considered environmental application is a watershed monitoring system which is able to send alert about flood risk. As shown in Figure 4, the application uses a WSN for data acquisition. This network is composed of wireless sensors, called "Water flow node", equipped with stream gauge measuring the water flow rate. One of these wireless sensors is located on the outlet of the watershed. The network contains also "Precipitation nodes" measuring the precipitation quantity. All the measurements are sent to a DSS. This DSS deduces the risk of the occurrence of a flood and send it to users. One of our assumptions is that the WSN has a star topology: each node communicates directly with the DSS, we do not introduce routing protocol constraints at this step.

In the application, we define four entities:

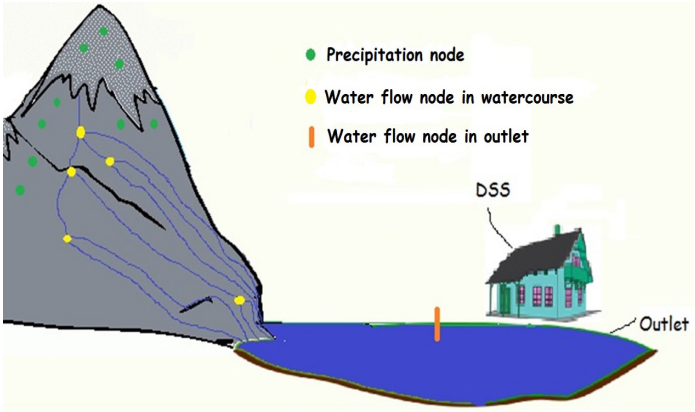

Figure 4. Example of watershed monitoring

1) the Precipitation entity which is an observable one. Its state is calculated from the data collected by the "Precipitation nodes" located at different points of the watershed. The Precipitation entity (P) has two states: high and low.
2) the WaterCourse entity which is an observable one. Its state is calculated from the data collected by the "water flow nodes" located at different points of the tributary stream (water courses). The WaterCourse (W) entity has two states: high and low.
3) the Outlet entity which is also an observable one. Its state is calculated from the data collected by the "water flow node" located on the outlet of the watershed. The Outlet entity (O) has two states: high and low.
4) the Flood entity is the entity of interest of the application. The flood entity is not an observable entity but its state depends on the states of all the observable entities. The Flood entity has four states "Normal", "Rain", "Risk", "Flood". "Normal" state means there is no risk. "Rain" state means that the watershed has received lot of precipitations, but there is no flood. "Risk" state means that flood is coming. "Flood" state means that the flood is there, the main river is overflowing. Application users want to know as soon as possible when a risk state is reached.

All the measurements are stored in the context data model in order to build the low level context. Several reasoning steps will be proposed in order to build the high level context of the Flood entity:

1) The precipitation measurements from the various "Precipitation nodes" are aggregated. One threshold should be set on the aggregation value in order to determine when the Precipitation entity moves from the low to the high state and vice versa.
2) The water flow measurements from the various stream gauges, which equip "Water flow nodes" located in the water courses, are aggregated. One threshold should be set on the aggregation value in order to determine when the WaterCourse entity moves from the low to the high state and vice versa.
3) Based on the measurements of the stream gauge that equips the "Water flow node" located at the Outlet,

 ISBN: 978-1-61208-490-9

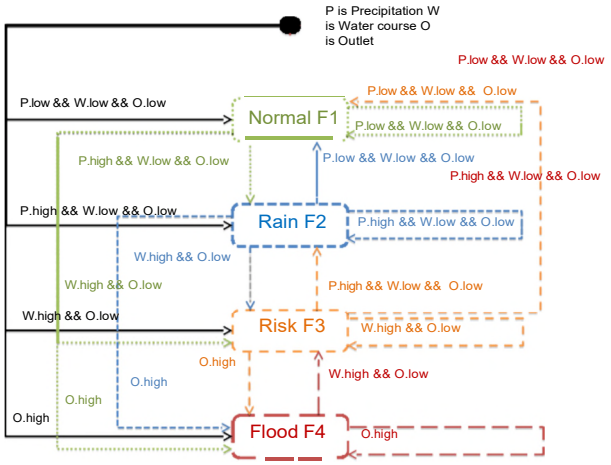

Figure 5. Example of flood finite-state machine

one threshold should be set in order to determine when the Outlet entity moves from the low to the high state and vice versa.

4) Figure 5 presents the finite-state machine that deduces the state of the Flood entity from the states of the three other observable entities. This diagram follows every step of the emergence of a flood. Usually, when a flood event occurs, the Flood entity will move from the "Normal (F1)" state to the "Rain (F2)" one, proceed to the "Risk (F3)" one and finish with the "Flood (F4)" one.

V. FORMALISATION USE IN SIMULATION

To implement our formalisation, we extend the simulation system based on the multi-agent system JADE (Java Agent DEvelopment Framework) [12] as introduced in [10]. JADE is implemented in Java. Three main features of JADE are:

1) **Agent communication**: exchange of messages between agents.
2) **Message content modeling by ontologies**: use of ontologies to model the exchanged message contents between agents.
3) **Integration with other tools**: possible use of tools like Jess rule engine [13] as a decision component of an agent.

However, current implementations based on JADE do not use different levels of reasoning. The work in [14] [15] only realize exchange of messages between agents. It does not care about the content of message modeling and other tools integration. In our simulation, we implement the features from Figure 6. We use ontologies to model the content of the messages exchanged between agents. Thus, we can build the low level context from observable entities (e.g. rainfall amount). Then, we can use Jess and a set of rules to infer the states of observable entities (e.g. Precipitation state) and build the high level context. The state of the entity of interest (e.g. Flood state) is also inferred by Jess and another set of rules from the state of the observable entities. Then the high level context is expanded. As mentioned in [7], the context modeling is often based on ontologies. Ontologies are defined by [16] as *an formal explicit specification of a shared conceptualization*. According to World Wide Web Consortium (W3C), ontologies are vocabularies that define the concepts and relationships used to describe and represent an area of concern. Thus, ontologies provide meaning to data (as data model do).

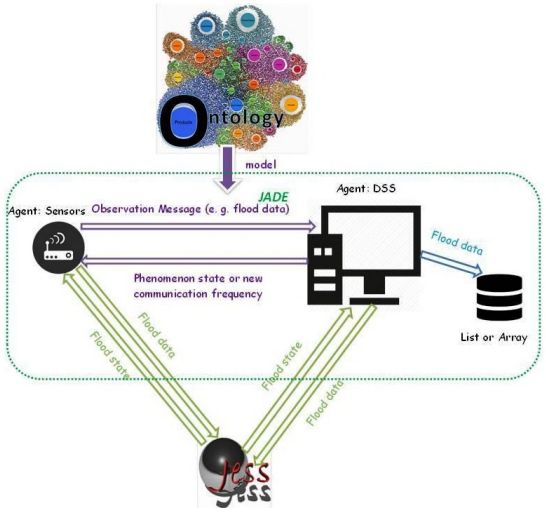

Figure 6. Simulation architecture

Our ontology is based on Semantic Sensor Network ontology (SSN) proposed by the W3C [17]. This ontology is a nucleus on which other ontologies can be connected, in order to develop a full context data model. The main concepts of SSN that we reused are "Sensor", "FeatureOfInterest", "Property", "Observation". Our observable entities or entity of interest are defined in SSN as "FeatureOfInterest". Possible entities can also come from some dedicated ontologies such as the Climate and Forecast ontology [18] or the SWEET ontology [19]. To describe time stamp, we reuse the Time ontology proposed by the W3C [20]. We use the QU ontology to define the unit [21]. To describe the state of our entities, we reuse the ontology proposed in [10].

```
(defrule floodState NormalF
1(declare (salience 10))
?p <- (pluvio {state      == low})
?w <- (WaterCourse{state== low})
?o <- (outlet{state       == low})
?f <- (flood)
=> ( modify ?f (state f1))
)
```

Figure 7. Rule deducing the Normal state of Flood entity

Concerning the reasoning process, we use the Jess rule-based engine [13] as indicated above. Jess is also implemented in Java language. We define several rules sets. Some are dedicated to infer the state of observable entity based on predefined thresholds and aggregation values. Others are dedicated to infer the state of the Flood entity. For example, the rule presented in Figure 7 deduces the state "Normal (F1)" of the Flood entity from states of observable entities.

 ISBN: 978-1-61208-490-9

We implement two systems where the DSS receives all the measurements and performs all the reasoning processes. These systems use the same WSN composed of heterogeneous wireless sensors to collect precipitation quantities, the water courses and the outlet flow rates. Figure 8 is a UML sequence diagram that presents the operating mode of "system 1", a context-aware one that send the Flood entity state to end user. Figure 8 presents the four processes of a context-aware system: context acquisition, context modeling, context reasoning and context distribution. The three different types of wireless sensors, previously mentioned, are represented by: "PrecipitationNode", "WaterCourseNode" (for the "water flow nodes" located in the tributary stream), "OutletNode" (for the "water flow node" located in the outlet). In "system 1", the acquisition and the transmission frequencies are equal.

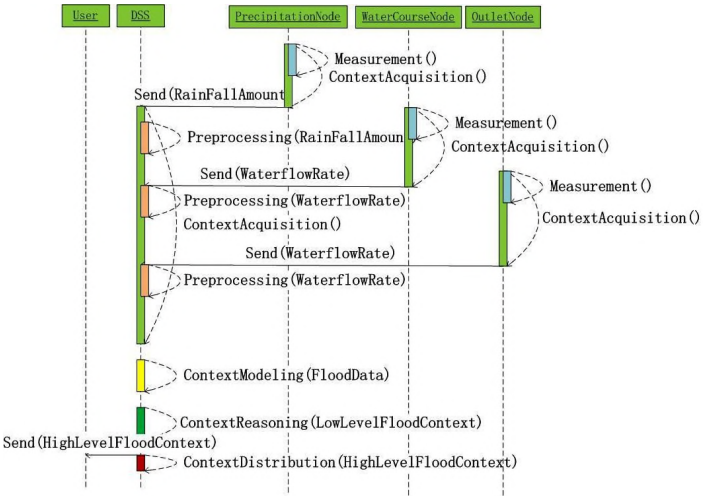

Figure 8. Sequence diagram of the Flood context-aware system (system 1)

Based on the previous system, we develop an adaptive context-aware one, "system 2", presented in Figure 9. The adaptation decision is implemented by the DSS. It deduces a new transmission frequency for each wireless sensor based on the Flood entity state.

Using our simulation architecture, we have compared these two systems at the level of the total amount of exchanged communication packets, using one-month data collected on a watershed, provided by [22]. Three "PrecipitationNodes", two "WaterCourseNodes" and one "OutletNode" are considered in our simulation. In JADE, we define as many agents as nodes. We also add a DSS agent. Each node agent acquires raw observation data and sends them to DSS. The sample frequency is of one measurement every minute. In the "system 2", the transmission frequency is modulated (calculated by the DSS) as shown in Figure 9. The table presented in Figure 10 shows how the transmission frequencies are computed based on the Flood entity state. The DSS agent processes the context modeling in order to build the low level context. It infers the high level context from the low level one using Jess rule engine. Figure 11 shows the obtained results. The nodes of the "system 1" has transmitted near 250000 packets. With the "system 2", the number of transmitted message is reduced to less than 100000 packets. In terms of the phenomenon monitoring quality, the two systems detect the same number of state changes.

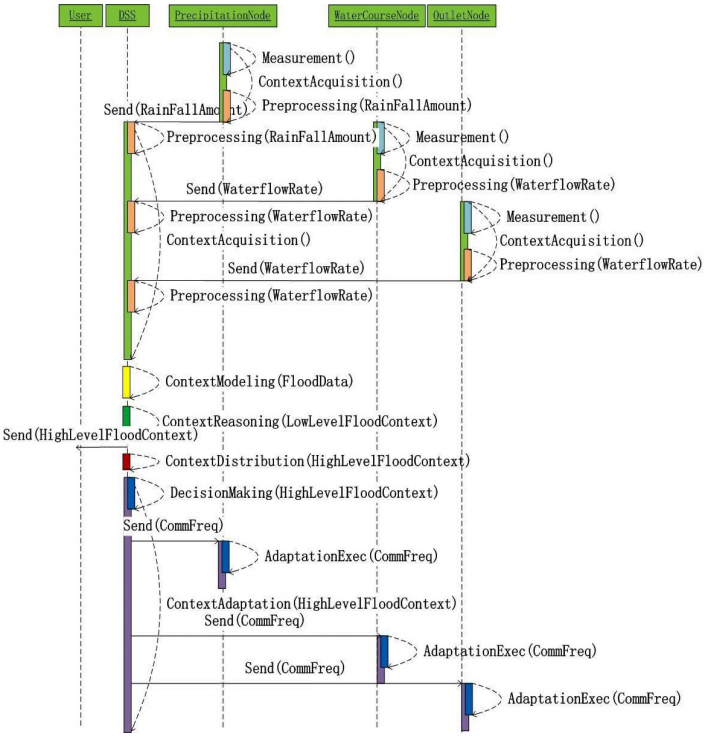

Figure 9. Sequence diagram of the Flood adaptive context-aware system (system 2)

Flood State	NORMAL	RAIN	RISK	FLOOD
Transmission Frequency	1/3*Acquisition Frequency	1/2*Acquisition Frequency	Acquisition Frequency	Acquisition Frequency

Figure 10. Table of transmission frequency based on Flood entity state

VI. RELATED WORK

No system of this type dedicated to flood monitoring was found. However, a context-aware system for water quality management exists. The InWaterSense project proposes a context-aware system to deduce the water quality of any water bodies (lake, river) [9] [23]. Their system is totally built using Semantic Web technologies. SSN ontology is used as a nucleus in order to build the context model. They also use the Jess rule based engine. Their rule format is based on SQWRL language. It is able to build aggregation value using rules. Thus, their rules merge the characteristics of observable entities and those of the entity of interest. Their rules infer the

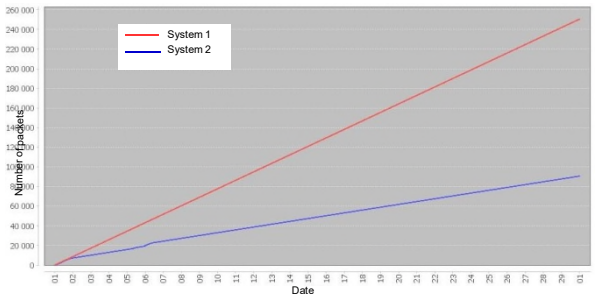

Figure 11. Number of exchanged communication packets

state of the water body without intermediate steps. Compared to our approach, their rules are much more difficult to manage due to their complexity. Our formalisation eases the reasoning process by splitting it into several steps: deduction of the states of observable entities; then, deduction of the state of the entity of interest.

The work of [24] proposes a WSN architecture called "Sepsen" in order to integrate, in nodes, several components: semantic annotator based on fragments of ontology, rule-based engine and a knowledge base that stores events. The goal is to decrease the number of event messages between sensors by classifying them as share, forward or discard event. The share events are sent to other sensor nodes to update their knowledge base. The forward events are sent to the gateway. The discarded events are removed. However, the semantic annotation is done manually. The rule indicates that a sensed value should be above a threshold in order to become a share or forward event. Using the PowerTOSSIM environment, the "Sepsen" architecture is applied on a simulation scenario showing the energy saving which this kind of approach can bring.

None of these systems uses the same formalisation based on observable entities, entity of interest and states. Thus, even if all these systems use a rule-based engine and ontologies, their rules are very complex and hard to maintain.

The Semsorgrid4Env project [25] wants to help coastal flood planning managers to make decisions during coastal flooding events. It proposes a mash-up application that integrates heterogeneous datasets: sensor data stream, historical database. The integration is made possible by a set of ontologies: SSN, SWEET, etc. In this project, the context is not modeled explicitly.

When dealing with complex phenomena like natural disasters, context-aware systems based on WSN become situation awareness system also based on WSN. In this type of system, the data management model is composed of different layers (sensor data, aggregation data, situation representation knowledge) [26]. Our formalisation can be integrated in the situation layer. In the work of [26], a situation is defined as the representation of a "structured part of the reality". It contains all the description of entities involved in the situation. Context is a point of view of one entity about the situation.

VII. Conclusion and Perspectives

In this article, we have proposed a new formalisation for the design and the implementation of context-aware systems. One of its advantages is that our approach can be used for multiple purposes. It can integrate both the monitoring of the studied phenomenon (feature of interest) and the management of the hardware and the software system used to observe it. More generally, it provides a unified way to deal with all the components/entities of an observation process. This formalisation can be used in different application top- ics related to agriculture, environment, "smart care", "smart home", industry. To illustrate its use, we have provided an environmental use case application: the study of flood events in a watershed. In the Irstea institute, we have different data related to this topic and we will continue the implementation and the experiment of our approach in this application field. A simulation architecture is provided to evaluate systems developed using our formalisation. This architecture is based on the ontology concept with the use of the multi-agent system JADE and the rule-based engine Jess. Different scenarios for this environmental application will be proposed in our future work taking into account different states and extended wireless sensors reasoning capabilities. Our application will also be implemented with tools suitable for the limited resources of wireless sensors.

Acknowledgment

This research is partly funded by the grant of the French Auvergne region, and now in the new Auvergne-Rhône-Alpes region (temporary name) and, the grant of the European Regional Development Fund (ERDF). The authors would like to thank the team of the Orgeval observatory for their help.

References

[1] L. Atzori, A. Iera, and G. Morabito, "The internet of things: A survey," Computer Networks, vol. 54, no. 15, 2010, pp. 2787 – 2805.

[2] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," Computer networks, vol. 38, no. 4, 2002, pp. 393–422.

[3] M. F. Othman and K. Shazali, "Wireless sensor network applications: A study in environment monitoring system," Procedia Engineering, vol. 41, 2012, pp. 1204 – 1210, international Symposium on Robotics and Intelligent Sensors 2012 (IRIS 2012).

[4] A. ur Rehman, A. Z. Abbasi, N. Islam, and Z. A. Shaikh, "A review of wireless sensors and networks' applications in agriculture," Computer Standards & Interfaces, vol. 36, no. 2, 2014, pp. 263 – 270.

[5] I. F. Akyildiz, T. Melodia, and K. R. Chowdhury, "A survey on wireless multimedia sensor networks," Computer Networks, vol. 51, 2007, pp. 921–960.

[6] A. K. Dey and G. D. Abowd, "Towards a better understanding of context and context-awareness," in Workshop on The What, Who, Where, When, and How of Context-Awareness, as part of the 2000 Conference on Human Factors in Computing Systems (CHI 2000), The Hague, The Netherlands, April 3, 2000, pp. 204 – 307.

[7] C. Perera, A. Zaslavsky, P. Christen, and D. Georgakopoulos, "Context aware computing for the internet of things: A survey," Communications Surveys Tutorials, IEEE, vol. 16, no. 1, 2014, pp. 414–454.

[8] C. Goumopoulos, B. O'Flynn, and A. Kameas, "Automated zone-specific irrigation with wireless sensor/actuator network and adaptable decision support," Computers and Electronics in Agriculture, vol. 105, 2014, pp. 20–33.

[9] E. Jajaga, L. Ahmedi, and F. Ahmedi, "An expert system for water quality monitoring based on ontology," in 9th Research Conference on Metadata and Semantics Research (MTSR) 2015, Manchester, UK, September 9-11, 2015, 2015, pp. 89–100.

[10]R. Bendadouche, C. Roussey, G. D. Sousa, J.-P. Chanet, and K.-M. Hou, "Extension of the semantic sensor network ontology for wireless sensor networks: The stimulus-wsnnode-communication pattern," in 5th International Workshop on Semantic Sensor Networks in conjunction with the 11th International Semantic Web Conference (ISWC), Boston, USA, November 2012, pp. 49–64.

[11] R. Draves, J. Padhye, and B. Zill, "Comparison of routing metrics for static multi-hop wireless networks," ACM SIGCOMM Computer Communication Review, vol. 34, no. 4, 2004, pp. 133–144.

[12] F. Bellifemine, A. Poggi, and G. Rimassa, "Jade–a fipa-compliant agent framework," in Proceedings of PAAM, vol. 99, no. 97-108. London, 1999, p. 33.

[13] E. Friedman-Hill, Jess in Action: Java Rule-Based Systems. Greenwich, CT, USA: Manning Publications Co., 2003.

[14] J. Subercaze and P. Maret, "Programming semantic agent for distributed knowledge management," in Semantic Agent Systems. Springer, 2011, pp. 47–65.

[15] T. Logenthiran, D. Srinivasan, and A. M. Khambadkone, "Multi-agent system for energy resource scheduling of integrated microgrids in a distributed system," Electric Power Systems Research, vol. 81, no. 1, 2011, pp. 138–148.

 ISBN: 978-1-61208-490-9

[16] R. Studer, V. Benjamins, and D. Fensel, "Knowledge engineering: Principles and methods," Data & Knowledge Engineering, vol. 25, no. 12, 1998, pp. 161 – 197.

[17] M. Compton et al., "The ssn ontology of the w3c semantic sensor network incubator group," Web Semantics: Science, Services and Agents on the World Wide Web, vol. 17, 2012, pp. 25 – 32.

[18] W. S. S. N. I. Group, "Climate and Forecast (CF) features," W3C, Tech. Rep., 2011, retrieved: 04, 2016. [Online]. Available: http://www.w3.org/2005/Incubator/ssn/ssnx/cf/cf-feature

[19] NASA, "Semantic Web for Earth and Environmental Terminology (SWEET)," Jet Propulsion Laboratory California Institute of Technology, Tech. Rep., 2016, retrieved: 04, 2016. [Online]. Available: https://sweet.jpl.nasa.gov/

[20] J. R. Hobbs and F. Pan, "Time Ontology in OWL, W3c Working Draft 27 September 2006," W3C, W3C Working Draft, sep 2006, retrieved: 04, 2016. [Online]. Available: https://www.w3.org/TR/owl-time/

[21] H. P. de Koning, N. Rouquette, R. Burkhart, H. Espinoza, and L. Lefort, "Library for Quantity Kinds and Units: schema, based on QUDV model OMG SysML(TM), Version 1.2," CSIRO, Tech. Rep., 2011, retrieved: 04, 2016. [Online]. Available: http://www.w3.org/2005/Incubator/ssn/ssnx/qu/qu

[22] G. Tallec, P. Ansart, A. Gurin, O. Delaigue, and A. Blanchouin, "Observatoire oracle," 2015. [Online]. Available: http://dx.doi.org/10.17180/OBS.ORACLE

[23] L. Ahmedi, E. Jajaga, and F. Ahmedi, "An ontology framework for water quality management," in Proceedings of the 6th International Conference on Semantic Sensor Networks (SSN), ser. CEUR Workshop Proceedings, vol. 1063. Sydney, Australia: CEUR-WS. org, Oct. 2013, pp. 35–50.

[24] M. K. Kasi, A. Hinze, C. Legg, and S. Jones, "SEPSen: semantic event processing at the sensor nodes for energy efficient wireless sensor networks," in Proceedings of the 6th ACM International Conference on Distributed Event-Based Systems. Berlin, Germany: ACM Press, 2012, pp. 119–122.

[25] A. J. G. Gray et al., "A Semantically Enabled Service Architecture for Mashups over Streaming and Stored Data," in The Semantic Web: Research and Applications, G. Antoniou, M. Grobelnik, E. Simperl, B. Parsia, D. Plexousakis, P. De Leenheer, and J. Pan, Eds., vol. 6644. Crete, Greece: Springer, 2011, pp. 300–314.

[26] M. Stocker, M. Rönkkö, and M. Kolehmainen, "Abstractions from Sensor Data with Complex Event Processing and Machine Learning," in IEMSs Conference Proceedings, vol. 2014. USA, California, San Diego: IEMS society, Jun. 2014, pp. 1273–1280.

 ISBN: 978-1-61208-490-9