Formal Modeling and Validation of Micro Smart Grids Based on ReDy Architecture

Kaoutar Hafdi
IMS Team, ADMIR Laboratory
Rabat IT Center, ENSIAS
Mohammed V University
Rabat, Morocco
Email: kaoutar.hafdi@gmail.com

Abderahman Kriouile
IMS Team, ADMIR Laboratory
Rabat IT Center, ENSIAS
Mohammed V University
Rabat, Morocco
Email: a.kriouile@um5s.net.ma

***Abstract*—Several cities in the world are moving from traditional power grid to Smart Grids. In order to set up Smart Grids, we should be able to face many challenges related to reliability, scalability, dynamism, technological solutions, security, etc. In this paper, we propose a case study where we model a micro Smart Grid according to the ReDy architecture, which is intended for IoT applications. The ReDy architecture provides a base to implement a scalable, reliable, and dynamic IoT network ready to meet Smart Grid needs. In order to prove those requirements, we opted for formal modeling and validation approach using model checking techniques. This formal analysis is carried out using the CADP toolbox.**

I. Introduction

Humanity is moving more and more towards less polluting energies with sustainable resources called renewable and which are distributed by nature. The distribution of these sustainable resources requires a distribution of their production. For this reason, we speak about distributed generation of energy.

Taking into account this new configuration of the electrical grid, a new complexity is introduced in management and stabilization of this electrical grid. This is why the term Smart Grid is introduced [1]. This intelligent network integrates a data network parallel to the conventional electricity distribution network and offers monitoring, analysis and control tools for the production and consumption of energy at the level of the electrical network [1], [2].

Similarly to many IoT applications, Smart Grid should satisfy a set of requirements, mainly reliability, dynamism and scalability. In this paper, we propose to model the Smart Grid according to our ReDy architecture [3] which is intended for the design of IoT applications in highly dynamic environment in order to build reliable and scalable systems [4], [5], [6]. We focus on a case study of the Smart Grid which consists on a micro Smart Grid allowing to prove the studied requirements. In fact, a Smart Grid needs to be continuously extended. It should not be limited by a certain size, hence the need for scalability. Adding a new entity or disconnecting an entity must not impact the operation of the network, hence the need for reliability and dynamism. The objective of this work is to ensure the balance between consumption and production in the studied micro Smart Grid. In fact, this analysis can be useful to better manage the intelligence of the devices used in Smart Grids.

In order to prove that the system respects the required behavior, we propose a formal model of the micro Smart Grid studied and we use model checking techniques. Formal methods are a special type of math-based techniques for specifying and verifying complicated behavior of systems. The use of formal methods allows us to ensure a good level of reliability and robustness of the design we propose. The formal modeling of our system allows us to express the behavior of the system in an unambiguous way: the formal specification expresses a unique semantic. In addition to that, our formal model can be validated using automatic and exhaustive formal methods which allows us to model and validate our proposed architecture. In our work, our formal model is expressed using the LNT language [7], [8] and the properties of temporal logic are expressed using the MCL language [9]. In order to exploit the formal model for specification and formal validation purposes, we use the CADP toolbox [10].

Outline: The rest of this paper is organized as follows. Section II discusses the general context of Smart Grids. Section III presents the modeling of the micro Smart Grid according to the ReDy architecture. Section IV exhibits the formal model of the studied micro Smart Grid. Section V presents the formal validation of the formal model and exhibits the results of model checking techniques. Section VI surveys related work. Section VII gives concluding remarks and directions of future work.

II. Smart Grids general context

To set up Smart Grids, there are suitable technological solutions [2]. We mainly find AMI systems (Advanced Metering Infrastructure) which is a solution made up of several modules aiming to help electricity companies to take advantage of new technologies for managing the electricity network [1]. Thus, AMI make it possible to introduce smart meters to end customers, to detect energy losses during distribution as well as an interactive exchange with customers. As a result, the quality of services related to energy will improve and savings can be made as a result of the reduction in energy losses, which will also reduce carbon emissions.

There exist several manufacturers offering AMI systems. We mainly cite HUAWEI [11] and INHEMETER. These two companies have started implementing their AMI systems in several countries.

An AMI system mainly consists of the following modules (figure 1):

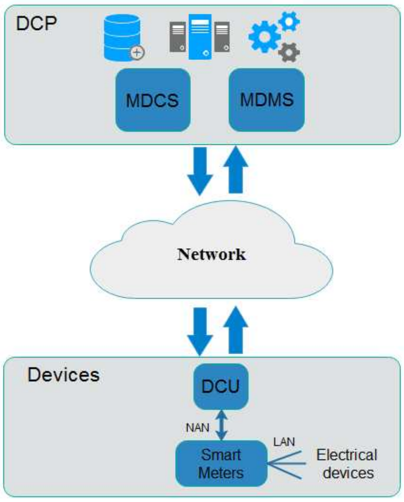

Fig. 1: AMI systems description

1) Devices: this module contains mainly three types:

- Electronic devices equipped with consometers or smart sockets: allowing to control the consumption of each electronic device and to transmit it to the smart meter.

The consometer consists of an electrical outlet and a display screen used to capture the consumption of an electrical device. It can be connected to the local network. The smart socket can be connected to an electrical device and can be controlled by an Android / iOS application. This application is used to display the consumption of the electrical device, to stop, start or program the operation of a device. The connected outlet does not have a display screen.

- Smart meters: replaces the unconnected analog meter. It is connected to all of the customer's electronic devices. This meter sends the state of electricity use from a customer directly to the supplier. There is no longer any need for estimated invoices or manual reading of the meter. There are mainly two types of smart meters: Single-phase meters intended for residential use with low consumption, and three-phase meters intended for industrial and commercial customers with high consumption.

- Data Concentrator Unit (DCU): This is a unit responsible for collecting data relating to the energy consumption sent by smart meters. This unit is equipped with chips implementing communication protocols in order to be able to connect the smart meters to the data center of the energy company.

Communication between the smart meter and electronic devices takes place in a short range network called Local Area Network (LAN). Communication between DCUs and smart meters takes place within a Neighborhood Area Network (NAN) which has a greater range than the LAN.

2) Communication network: The devices used must support the appropriate communication protocols to ensure communication between the DCUs and the centralized DCP platform that collects the data.

3) Data Collection Platform (DCP): this platform is responsible for collecting and analyzing the data sent by DCUs and smart meters. The DCP platform is made up of two systems:

- Meter Data Collection System (MDCS): This system is responsible for interacting with the devices by managing the communication protocols and for collecting and storing data related to electrical consumption.

- Meter Data Management System (MDMC): This system is responsible for analyzing data related to electricity consumption, carrying out statistics, generating reports, etc.

III. Micro Smart Grid modeling using ReDy architecture

In this section we start by presenting the proposed ReDy architecture, then we give the hierarchical organization of AMI systems. After that we present the model of the micro Smart Grid that we study in our paper.

A. ReDy architecture

ReDy architecture [3], [5] is a generic architecture intended for reliable and dynamic IoT systems. ReDy architecture proposes a hybrid composition of their components by combining centralized and decentralized solutions in order to design a dynamic, reliable and scalable IoT system. A ReDy system designed according to a ReDy architecture is composed of several subsystems. Each subsystem contains many detection units (sensors) and action units (actuators) communicating in a centralized mode with one governance unit. The subsystems are nodes of a Peer-to-Peer decentralized overlay.

There exists two variants of the proposed ReDy architecture:

- *The hierarchical ReDy architecture* where the different nodes are organized according to many levels (n). In this variants nodes belonging to the same level are of the same type. Nodes belonging to different levels are of different types.

- *The basic ReDy architecture* where all nodes are of the same type (n=1) and belong to one level (the unic that exists).

B. AMI hierarchical organization

Smart Grids can follow a hierarchical organization of its different elements. This hierarchical organization can be presented as follows:

- Each group of electronic devices is linked to a smart meter (via consumption meters or smart sockets).

- Several smart meters are linked to a DCU.

- Several DCUs are connected to the DCP.

- DCPs can be linked to higher units to offer application or business level services.

In our example, we deal with three levels of hierarchy. Figure 2 illustrates this hierarchical organization according to three levels. The nodes belonging to each level are of the same type. Nodes belonging to two different levels are of different types.

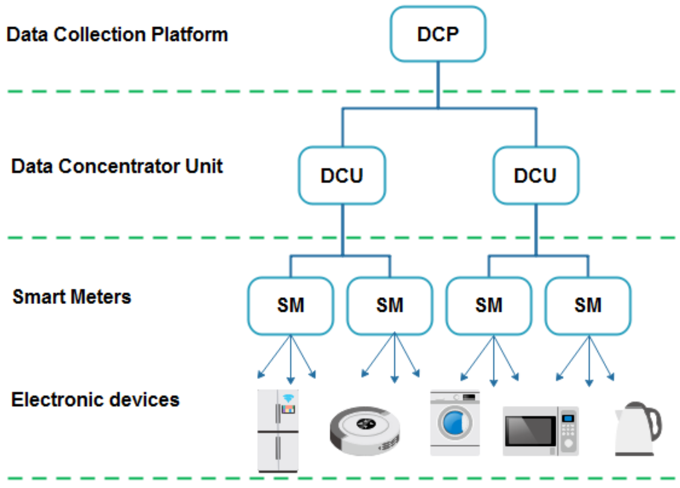

Fig. 2: Hierarchical organization of AMI components

C. Micro Smart Grid according to the ReDy architecture

In addition to the hierarchical ReDy architecture, the micro Smart Grid can be modeled according to the basic ReDy architecture. In the following, we present the model of the micro Smart Grid according to the basic ReDy architecture which will be used for formal validation purposes. In fact, the modeled architecture is composed of four nodes (figure 3). Each node is presented as follows:

- Node 1: a mainly consumer node. It consists on a smart house integrating a small energy production source (solar panels). This node contains one smart meter, two consumption sensors, two consumption actuators, one production sensor and one production actuator.

- Node 2 and node 3 : a consumer node. It consists on normal house that consume energy without having any production source. Such a node contains one smart meter, two consumption sensors and two consumption actuators.

- Node 4 : a mainly producer node. It consists on a power plant producing energy (solar and combustible). This node contains one smart meter, one consumption sensor, one consumption actuator, two production sensors and two production actuators.

IV. Formal Modeling of Micro Smart Grid

In this section, we present the formal model of the micro Smart Grid architecture that we presented above. We start by presenting the modeling language and tools, then we present the formal model of the studied architecture.

A. CADP toolbox

CADP [10] is a scientific toolbox for the design of asynchronous concurrent systems. CADP is developed by VASY team then CONVECS team of Inria. CADP offers a wide set of functionalities, ranging from step-by-step simulation to massively parallel model-checking. This toolbox offers high-level descriptions written in various languages such as LNT [7], [8], a modern and user-friendly variant of E-LOTOS (International Standard ISO-15437:2001). The LNT language is based on process algebra. LNT presents two notions of process algebra which are not possible in classical functional programming languages:

- The parallel composition of processes (with an asynchronous communication scheme).

- The non-deterministic choice within a process.

In addition, LNT contains all the classic formalisms of programming languages, namely loops, tests (if ... then ... else ...), recursive calls, variables, functions, data types as well as data structures.

CADP offers also several model-checkers for various temporal logic and mu-calculus, such as EVALUATOR 4.0 that uses the MCL language [9]. In our work, the validation phase is carried out using EVALUATOR 4.0 and logic temporal properties are expressed using the MCL language. CADP uses several verification algorithms combined together such as enumerative verification and on-the-fly verification [12].

B. Structure modeling

Using the LNT language, we propose a formal model of the studied micro grid architecture. Each component of the architecture is modeled by an LNT process. The overall structure of the architecture is modeled by interactions between LNT processes that we call rendez-vous on communication gates. In the studied architecture, we have two modes:

- Local mode: consists on interactions between components of the same node i.e between smart meter and different sensors and actuators.

- Distant mode: consists on interactions between components belonging to different nodes i.e between smart meters of the micro Smart Grid.

In the following, we present the model of a node among the micro Smart Grid then we present the parallel composition used in order to obtain the model of the overall micro Smart Grid studied.

1) Micro Smart Grid node modeling: In order to model a node, we start by modeling different components of the architecture by processes: Smart Meter (SM), Consumption Sensor (CS), Consumption Actuator (CA), Production Sensor (PS), Production Actuator (PA).

In the micro Smart Grid node, communication is ensured using communication gates. We define four communication gates used in local mode (i.e within the same node). Figure 4 illustrates different processes and communication gates defined in our formal model. ConsA is the communication gate between SM and CA. ProdA is the communication gate between SM and PA. ConsS is the communication gate between SM and CS. ProdS is the communication gate between SM and PS.

Each node of the micro Smart Grid is defined by a module called MSGNode. This module uses five other modules : SM, SC ,PS, CA, PA. Then we define the process MSGNode by a parallel composition between SM process and all other processes on the defined communication gates (ConsS, ProdS, ConsA, ProdA). Then we define the composition of the node

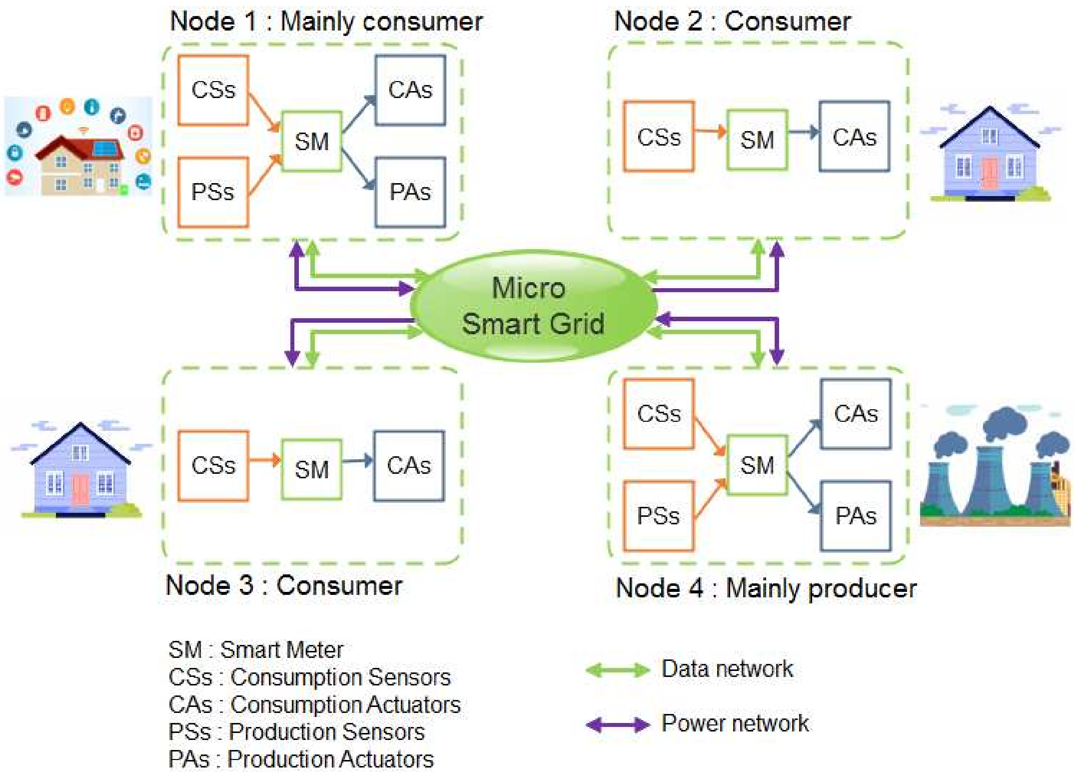

Fig. 3: Micro Smart Grid model according to the ReDy architecture

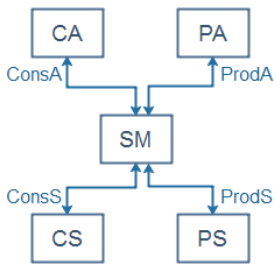

Fig. 4: Communication gates in local mode

i.e. the number of each sensor/actuator in the node. In our example, we can define a node whith one or two consumption sensors (CS). Using the same method, we can define the number required of each sensor/actuator in the micro Smart Grid node by giving different values to the concerned variables (idNode, nbCS, nbPS, nbCA, nbPA).

```
module MSGNode(SM, CS, PS, CA, PA) is

process MSGNode [ConsS, ProdS, ConsA,
                 ProdA, DisComm : any]
 (idNode:index_Node, nbCS:index_CS,
  nbPS:index_PS, nbCA:index_CA,
nbPA:index_PA)

is
```

```
    par ConsS, ProdS, ConsA, ProdA in
    --Smart Meter
      SM[ConsS, ProdS, ConsA, ProdA,
DisComm]
   (idNode)
      ||
        par
        -- Consumption Sensors
           if (nbCS==index_CS(1)) then
CS[ConsS](idNode,index_CS(1))
           elsif (nbCS==index_CS(2)) then
           par
             CS[ConsS](idNode,index_CS(1))
             ||
             CS[ConsS](idNode,index_CS(2))
             end par
           end if
          ||
             -- Production Sensors
          ||
             -- Consumption Actuators
          ||
             -- Production Actuators


          end par
      end par
end process

end module
```

2) Micro Smart Grid modeling: Once we model the node of the studied micro Smart Grid, we can now model the overall micro Smart Grid by executing a parallel composition between four nodes on a communication gate named DisComm (Figure 5).

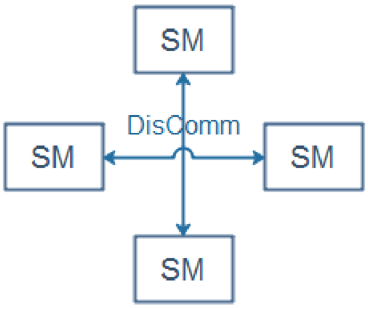

Fig. 5: Communication gates in distant mode

The model of the micro Smart Grid is defined by a parallel composition between four processes. Each process represents one node of the micro Smart Grid. For each process we define communication gates and variables values. The first process (MSGNode of index 1) contains two CSs (Consumption Sensors), one PS (Production Sensor), two CAs (Consumption Actuators) and one PA (Production Actuator). The second/third process (MSGNode of index 2/3) contains two CSs (Consumption Sensors) and two CAs (Consumption Actuators). The forth process (MSGNode of index 4) contains one CS (Consumption Sensors), two PSs (Production Sensor), one CA (Consumption Actuators) and two PAs (Production Actuator).

```
module main(MSGNode) is

process MAIN [ConsS, ProdS, ConsA,
             ProdA, DisComm : any]
is

 par DisComm in

 MSGNode [ConsS, ProdS, ConsA, ProdA,
DisComm]
  (index_Node(1), index_CS(2), index_PS(1),
   index_CA(2), index_PA(1))
 -- Node number 1 with 2 CSs 1 PS 2 CAs 1
PA

  ||
 MSGNode [ConsS, ProdS, ConsA, ProdA,
DisComm]
  (index_Node(2), index_CS(2), index_PS(0),
   index_CA(2), index_PA(0))
 -- Node number 2 with 2 CSs 2 CAs

  ||
 MSGNode [ConsS, ProdS, ConsA, ProdA,
DisComm]
  (index_Node(3), index_CS(2), index_PS(0),
   index_CA(2), index_PA(0))
 -- Node number 3 with 2 CSs 2 CAs

  ||
 MSGNode [ConsS, ProdS, ConsA, ProdA,
DisComm]
  (index_Node(4), index_CS(1), index_PS(2),
   index_CA(1), index_PA(2))
 -- Node number 4 with 1 CS 2 PSs 1 CA 2
PAs

 end par

end process

end module
```

C. *Formal model generation examples*

In the following, we give generation examples of the formal model of a system designed according to the ReDy architecture. The table I summarizes the characteristics of the examples generated from the formal model expressed in LNT. The generation result of this model is a BCG file representing an LTS. The column *SM* represents the number of smart meters (and thereafter the number of nodes) present in the system. The column *S* (respectively *A*) represents the number of sensors (respectively of actuators) in each subsystem generated. The number of states and transitions of the LTS generated are expressed in the column *without minimization*. To this generated LTS, we apply a minimization in order to have an equivalent LTS of reduced size. The number of states and transitions of this reduced LTS are expressed in the column *with minimization*.

TABLE I: Generation examples of the ReDy architecture formal model

	SM	S	A	without minimization		with minimization	
				States	Transitions	States	Transitions
Example 1	2	2	2	163	627	37	141
Example 2	3	3	3	2087	8955	171	714
Example 3	3	6	6	2087	13079	171	1116
Example 4	3	9	9	2087	17203	171	1518
Example 5	4	4	4	43708	251975	1257	7004
Example 6	5	5	5	798292	5760947	8403	58520
Example 7	6	6	6	13305262	115204621	52605	439590

V. Formal Validation of Micro Smart Grid

From the formal model of the micro Smart Grid presented above, we can apply model checking techniques to verify a set of temporal logic properties expressed in MCL. There are two types of properties: liveness properties express that something good happens well, and safety properties express that something bad never happens.

A liveness property can detect two types of problems:

- The deadlock that occurs due to a mutual wait for two processes.

- The livelock which occurs when the system enters a loop which does not advance the state of the system.

In the following, we present an MCL property which ensures the absence of livelocks and deadlocks in our model. This property is called Order property which expresses that any detection made in the system must be followed by an action corresponding to it. In order to express this property, we define a classic MCL macro in temporal language which expresses inevitability.

```
macro inevitable (L) =
   mu X . ( < true > true and [ not L ] X
)
end_macro
```

The first MCL property checks that each time an overconsumption is detected by the ConsS label at the node of index IdNode, inevitably there is at least one flexible production which is activated with the ProdA label:

```
[ true * . {ConsS ?IdNode:Nat ...} ]
inevitable ( {ProdA !IdNode ...} )
```

The second MCL property checks that each time an overproduction is detected by the ProdS label at the node of index IdNode, inevitably flexible consumption is activated with the ConsA label:

```
[ true * . {ProdS ?IdNode:Nat ...} ]
inevitable ( {ConsA !IdNode ...} )
```

If the system contains deadlocks or livelocks then at least one of these two properties will not be satisfied.

By using the model checking techniques, we were able to detect deadlocks and livelocks in previous version of the formal model. All deadlocks and livelocks detected were corrected in the latest version of the model. As a result, our model is free of deadlocks and livelocks.

In the following, we present one example of a livelock detected while verifying the first MCL property by using the model checker EVALUATOR 4.0. Indeed, this model checker returned a counterexample which proves that a livelock exists. This counterexample represents the classic problem of starvation, *i.e.* a process monopolizes execution and does not let the other processes go ahead. The figure 6 represents this counterexample. We note that in this counterexample, the smart meter of index 2 monopolizes communications and no longer let other governance units execute their actions, which brings the global system to a livelock.

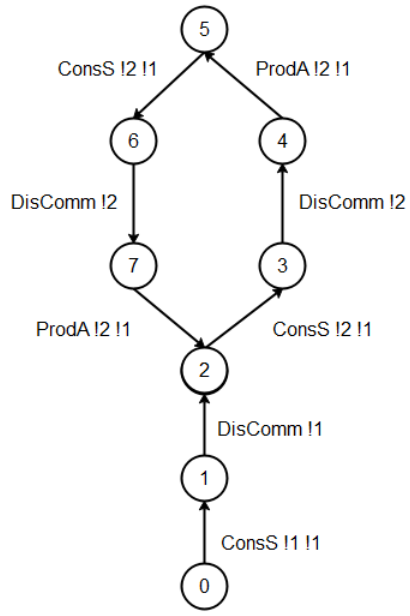

Fig. 6: Livelock detection-Starvation problem

VI. Related work

In this section, we focus on different existing approaches of formal analysis of IoT architectures.

Formal modeling of IoT systems is a very interesting field, since it allows IoT systems to acquire maturity by ensuring the correct behavior of the system before implementation. In the literature, the need for formal modeling of IoT systems is obvious and clearly expressed [13], [14], [15], [16]. However, research on this paradigm generally relates to just one aspect among the different aspects of IoT systems. Indeed, in existing work, the use of formal modeling is mainly applied to specific protocols used in IoT systems [14], [17], [18], without proposing a complete modeling of the architecture representing the IoT system.

Several formal tools allow to formally model IoT systems and thus exploit the formal model by analyzing it, simulating behavior and verifying properties. In this context, work has proposed the PRISM tool [19] to model the behavior of real-time IoT systems.

Petri-Nets have been proposed [20] to model an IoT service using a multi-agent approach, then the model is used in simulation and verification.

The timed synchronous process algebra TPi is used [14] to formally model an MQTT communication protocol for IoT applications. This model is used to study and analyze properties concerning the security of the MQTT protocol.

The probabilistic model checking techniques [17] have been used to study properties of reliability and correctness and to detect ambiguities in publish/subscribe protocols. We also find work that has been carried out to verify the behavior of service composition protocols in service-oriented architectures using probabilistic model-checking techniques [18].

Other works have proposed to use model-based testing techniques to test the safety and reliability aspects of devices in an IoT system in the context of smart city applications [21] and smart houses [22]. The idea is to formally model the system and then use the formal model to generate tests to run on the IoT system then use model-based coverage criteria to ensure that you have tested all the aspects of the model [23].

Semi-formal modeling techniques using multi-agent systems [24] were used to model a smart factory in the context of Industry 4.0. The analysis concerns flexibility and self-organization. This modeling make it possible to avoid deadlocks thanks to the simulation using a semi-formal model.

For the specific case of Smart Grids, other existing semi-formal approaches are proposed in order to validate Smart Grids architecture such as UML langage [25].

VII. Conclusion

In this paper, we present the formal modeling and validation of the specific case study of micro Smart Grid. Our objective is to apply the ReDy architecture to a practical case study in order to ensure a high degree of reliability and dynamism of the designed system. Scalability is ensured as well. The ReDy architecture is intended principally for IoT applications. In this paper we prove that this architecture can be applied for micro Smart Grid. Our proposals are proven by using formal modeling and model checking techniques for validation purposes. This work ensures the balance between consumption and production at the level of the micro Smart Grid studied. This analysis can be useful to better manage the intelligence of the devices used in the Smart Grids of tomorrow: the goal is to have a more dynamic management of the components of a Smart Grid, while keeping a good level of reliability.

References

[1] G. Dileep, "A survey on smart grid technologies and applications," *Renew. Energy*, vol. 146, pp. 2589–2625, 2020.

[2] R. Bayindir, I. Colak, G. Fulli, and K. Demirtas, "Smart grid technologies and applications," *Renewable and Sustainable Energy Reviews*, vol. 66, pp. 499–516, 2016.

[3] K. Hafdi and A. Kriouile, "Designing redy distributed systems," in *International Conference on Autonomic Computing (ICAC), 2015 IEEE*. IEEE, 2015, pp. 331–336.

[4] K. Hafdi, A. Kriouile, and A. Kriouile, "Formal modeling and validation of redy architecture intended for iot applications," *International Journal of Innovative Research in Computer Science and Technology*, vol. 5, pp. 339–349, 07 2017.

[5] ——, "Iot redy architecture for smart grid management," *Computer and Information Science*, vol. 11, no. 4, pp. 36–44, 2018. [Online]. Available: https://doi.org/10.5539/cis.v11n4p36

[6] ——, "Overview on internet of things (iot) architectures, enabling technologies and challenges," *JCP*, vol. 14, no. 9, pp. 557–570, 2019. [Online]. Available: http://www.jcomputers.us/index.php?m=content\&c=index\&a=show\&catid=216\&id=2965

[7] D. Champelovier, X. Clerc, H. Garavel, Y. Guerte, C. McKinty, V. Powazny, F. Lang, W. Serwe, and G. Smeding, "Reference manual of the lnt to lotos translator (version 6.1)," *Inria/Vasy and Inria/Convecs*, vol. 131, 2014.

[8] D. Champelovier, X. Clerc, H. Garavel, Y. Guerte, F. Lang, C. McKinty, V. Powazny, W. Serwe, and G. Smeding, "Reference manual of the lnt to lotos translator," 2017.

[9] R. Mateescu and D. Thivolle, "A model checking language for concurrent value-passing systems," in *International Symposium on Formal Methods*. Springer, 2008, pp. 148–164.

[10] H. Garavel, F. Lang, R. Mateescu, and W. Serwe, "Cadp 2011: a toolbox for the construction and analysis of distributed processes," *International Journal on Software Tools for Technology Transfer*, vol. 15, no. 2, pp. 89–107, 2013.

[11] H. AMI, "Ami architecture," http://support.huawei.com/enterprise/docinforeader!loadDocument1.action?contentId=DOC1000106039&partNo=10052#adc_iot_pd_0012/, 2019, [Online; accessed 23-november-2019].

[12] A. Kriouile and W. Serwe, "Formal analysis of the ace specification for cache coherent systems-on-chip," in *International Workshop on Formal Methods for Industrial Critical Systems*. Springer, 2013, pp. 108–122.

[13] A. Čolaković and M. Hadžialić, "Internet of things (iot): A review of enabling technologies, challenges, and open research issues," *Computer Networks*, 2018.

[14] B. Aziz, "A formal model and analysis of an iot protocol," *Ad Hoc Networks*, vol. 36, pp. 49–57, 2016.

[15] J. A. Stankovic, "Research directions for the internet of things," *IEEE Internet of Things Journal*, vol. 1, no. 1, pp. 3–9, 2014.

[16] I. Bojanova and J. Voas, "Trusting the internet of things," *IT Professional*, no. 5, pp. 16–19, 2017.

[17] F. He, L. Baresi, C. Ghezzi, and P. Spoletini, "Formal analysis of publish-subscribe systems by probabilistic timed automata," in *International Conference on Formal Techniques for Networked and Distributed Systems*. Springer, 2007, pp. 247–262.

[18] S. Gallotti, C. Ghezzi, R. Mirandola, and G. Tamburrelli, "Quality prediction of service compositions through probabilistic model checking," in *International Conference on the Quality of Software Architectures*. Springer, 2008, pp. 119–134.

[19] M. Rahmani, J. Song, and M. Lee, "Prism: A knowledge engineering tool to model collective behaviors of real-time iot systems." in *PrOse@PoEM*, 2017.

[20] S. Yamaguchi, S. Tsugawa, and K. Nakahori, "An analysis system of iot services based on agent-oriented petri net pn2," in *2016 IEEE International Conference on Consumer Electronics-Taiwan (ICCE-TW)*. IEEE, 2016, pp. 1–2.

[21] M. Krichen, O. Cheikhrouhou, M. Lahami, R. Alroobaea, and A. J. Maâlej, "Towards a model-based testing framework for the security of internet of things for smart city applications," in *International Conference on Smart Cities, Infrastructure, Technologies and Applications*. Springer, 2017, pp. 360–365.

[22] M. Martínez, A. I. Esparcia-Alcázar, T. E. Vos, P. Aho, and J. F. i Cors, "Towards automated testing of the internet of things: Results obtained with the testar tool," in *International Symposium on Leveraging Applications of Formal Methods*. Springer, 2018, pp. 375–385.

[23] A. Ahmad, F. Bouquet, E. Fourneret, and B. Legeard, "Model-based testing for internet of things systems," in *Advances in Computers*. Elsevier, 2018, vol. 108, pp. 1–58.

[24] S. Wang, J. Wan, D. Zhang, D. Li, and C. Zhang, "Towards smart factory for industry 4.0: a self-organized multi-agent system with big data based feedback and coordination," *Computer Networks*, vol. 101, pp. 158–168, 2016.

[25] M. Uslar, S. Rohjans, C. Neureiter, F. Pröstl Andrén, J. Velasquez, C. Steinbrink, V. Efthymiou, G. Migliavacca, S. Horsmanheimo, H. Brunner *et al.*, "Applying the smart grid architecture model for designing and validating system-of-systems in the power and energy domain: A european perspective," *Energies*, vol. 12, no. 2, p. 258, 2019.