Formal Algebraic Description of a Fog/IoT Computing Environment

Conference Paper · June 2020

DOI: 10.1109/IEEECONF49502.2020.9141602

http://dx.doi.org/10.5755/j01.eee ELEKTRONIKA IR ELEKTROTECHNIKA, ISSN 1392-1215, VOL. XX, NO. X, 20XX

Formal Algebraic Description of a Fog/IoT Computing Environment

Salvador Alcaraz[1], Pedro Juan Roig[1,2], Katja Gilly[1], Sonja Filiposka[3], Noura Aknin[4]
[1]*Department of Physics and Computer Architecture, Miguel Hernández University,*
Avda. Universidad, s/n - 03202 Elche (Alicante), Spain
[2]*Department of Computer Science, Balearic Islands University,*
Ctra. Valldemossa, km 7.5 - 07122 Palma de Mallorca, Spain
[3]*Faculty of Computer Science and Engineering, Ss. Cyril and Methodius University*
Rugjer Boshkovikj 16, 1000 Skopje, North Macedonia
[4]*Faculty of Science, Abdelmalek Essaadi University*
Tétouan, Morocco
{salcaraz,katya}@umh.es, pedro.roig@graduado.umh.es, sonja.filiposka@finki.ukim.mk, noura.aknin@ieee.org

***Abstract*— The society is attending to the integration of the IoT technology in many fields related to urban environments, smart homes and industries. The IoT technology is closely related to other emergent technologies such as cloud and fog computing, which provide virtualization services and mobile facilities. For that reason, the development of correct fog/IoT environments contributes to the IoT technology deployment where one of the most important aspects in the cloud/fog/IoT computing is the service mobility between fog and cloud levels. In this paper, we analyse the communication specification of a scenario composed of mobile IoT devices in a fog and cloud computing system. The devices movement implies the migration of their virtualised services along the fog or cloud system. Our system specification is based on Algebra of Communicating Processes. The system behaviour has been specified through formal algebraic rules which are rooted branching bisimilar.**

***Index Terms*— ACP, fog computing, IoT, formal protocol specification, networking.**

I. INTRODUCTION

Fog computing is an evolution from cloud computing technology [1], being one of the driving forces in computing research in this time and age, along with quantum computing [2]. The development of fog computing facilities is key for the full-scale deployment of IoT environments [3] by means of a range of technologies, such as 5G cellular technology [4] or new WiFi standards [5].

In this paper, we tried to obtain a formal algebraic description of a fog computing environment. Basically, the main interest is to achieve a general expression for the migration of VMs belonging to users getting from the cloud into a fog environment composed by a linear distribution of facilities, such as a long avenue or a pipeline.

The organisation of this paper will be the following: to start with, Section 2 details the related works, Section 3 introduces an overview of the model, then, Section 4 presents an overview on ACP (Algebra of Communicating Processes), after that, Section 5 shows the model specification and verification, and finally, Section 6 draws some conclusions.

II. RELATED WORKS

The model specification and validation issues regarding IoT systems have been analysed through different theoretical and automated tools as it is related in [6]. The IoT systems has been modelled from different point of views, for example, in [7] and [8], we can find several formal models for the interaction between IoT applications, whereas in [9], authors dissert over formal aspects related to IoT protocols.

The ITU-T standard for modelisation, SDL, has been widely used for this purpose. In [10], authors show the specific challenges presented by the IoT systems and the language adaptations in order to address these specific needs. In [11], an integration scheme for IoT solutions based on wireless sensor networks has been proposed.

Promela and Spin model checker is another tool used for modelling and verification. In this sense, [12] uses this tool to model routing functions in IoT environments, whereas [13] proposes and automated validation of IoT devices. Furthermore, High Level Petri Nets is another tool for verification used in [14] to perform model checking in a secure fog deployment over IoT.

Regarding formal algebraic specifications, some specific frameworks have been presented in order to model IoT deployment in [15] and [16], whilst an ACP formal modelling of fog computing environment from the point of view of the virtual machines migration is introduced in [17].

III. MODEL OVERVIEW

The target of this paper is to study a fog computing environment composed of a number of facilities, considered as building blocks, aimed at hosting virtual machines associated with clients within the coverage range of the proposed fog domain.

The particular scheme for the topology proposed is that a user has its computing assets in the cloud environment, and upon getting into the fog environment, those computing assets are going to get allocated within the facility, this is,

Manuscript received April XX, 20XX; accepted April XX, 20XX.

This research was funded by a grant from “Cooperación Universitaria al Desarrollo 2018-Universidad Miguel Hernández de Elche. Convenio UMH-GV”. This research was performed in cooperation with the Institution.

within any of the hosts available, supposing that there are no further restrictions regarding capacity or computing power.

If that is the case, the computing assets may remain located where there are until the client gets out of the coverage range of that building block, and in turn, two things might happen.

On the one hand, if there is a coverage range of another building block on the place, the fog service leaves the previous one, leading the related computing assets to migrate from one facility to another. On the other hand, the computing assets may get out of the fog environment, getting back into the cloud environment, for example, if there are not enough computational resources to perform the migration.

Alternatively, it might happen that, at the moment of getting into a fog facility, there is not enough room for those computing assets to allocate them in, so in that case, they will get diverted again to the cloud.

For simplicity purposes, a linear setup is going to be considered, although any other topology might apply, just by changing the available paths within each block and undertaking the proper calculations. However, if that is carried out, results are going to be quite analogous with respect to the input-output relationship.

In general, we consider that each building block in a linear setup will have a common structure, this is, having an entry path coming from the previous block, namely its predecessor, and two exit paths, one going to the following block, namely its successor, as well as another one going straight to the cloud.

There are two exceptions to consider: the first block will not obviously come from a previous block, but from the cloud itself, and likewise, the last block will not have a path going to the following block either, as there will not be more blocks available.

It is to be noted that the case where the clients might get into the fog environment from scratch at any given fog facility has not been taken into account so as to keep it simple. Therefore, it has been considered that the clients getting in are all coming from the cloud environment, and they may go through the fog environment until they come back to the cloud at any given point, this is, at any given building block.

Regarding the number of facilities within each fog domain, the target of the model is to get a general expression for *N* building blocks, and for that purpose, some different values of *N* are going to be studied in order to get a general expression.

Therefore, the different variations of the system proposed always start with an entry point (named r_0) into the building block *1* (named F_1, after fog computing facility) and the following blocks will be named in a sequential manner.

Additionally, for the rest of the blocks, the paths between blocks are named by using their limiting blocks, this is, r_i and s_i for the path between block *i* and the next block *i+1*. Furthermore, the exit points are called $s_{\hat{\imath}}$ with the subscript of its proper block *i*.

IV. ACP OVERVIEW

It is to be remarked that the model is to be built by using an abstract algebra called ACP, which focuses on relationships within processes taking place among all different blocks being part of a system in a distributed manner [18].

In that context, some considerations need to be said, such as ACP does not contemplate the time passing by between states, and the ACP works with process terms describing atomic actions for each of the entities of a system, which, upon applying the proper ACP axioms, will show the external behaviour of that system [19].

Eventually, by comparing the external behaviour of the real system with the external behaviour of the model obtained after reasoning with the processes, it is going to be possible to see that they both are rooted branching bisimilar [20], meaning that they both share the same actions and the same branching structure, so in that case, the model gets verified.

In order to start expressing the model, atomic actions are going to be used to define each building block, meaning read from a channel or send through a channel. In other words, receiving computing assets from a path, or otherwise, sending them to a path.

As per the operations being used among processes [21], sequential operator is going to be defined by the symbol of multiplication (·), whereas alternative operator is stated by the symbol of addition (+).

Furthermore, as all entities being part of the system, namely, the building blocks, are working in a concurrent manner, the operator || is going to be used to show that.

In that sense, it is important to quote the Expansion Theorem by Bergstra and Klop [22], which gives the operations to be undertaken when dealing with the concurrent operator.

$$(X_1 \parallel \cdots \parallel X_n) = \sum X_i \Vdash X^i + \sum (X_i | X_j) \Vdash X^{i,j} \qquad (1)$$

The term X^i means all entities but the one involved, namely $X^i = \{X_1 \cdots X_n\} - \{X_i\}$, and the term $X^{i,j}$ means all entities but the ones involved, namely $X^{i,j} = \{X_1 \cdots X_n\} - \{X_i, X_j\}$. Additionally, the symbol ⊩, called *left merge*, states that the first factor of the proper process, namely *i*, is executed first, and all the rest will be done afterwards in a concurrent manner, whereas the symbol |, called *communication*, states that both first factors of the processes involved, namely *i* and *j*, are executed first all together, whilst all of the rest will be done in a concurrent way afterwards.

Afterwards, *encapsulation* operator is to be used to convert all atomic actions into communications, or otherwise, they will lead to deadlock and may be discarded. It is going to be expressed by the symbol ∂_H, where the set *H* includes all atomic actions all over the system.

At that point, *abstraction* operator is to be employed to hide all internal communications, converting them into silent steps, thus allowing just external communications to show, in a way that the external behaviour of the model is shown. That is going to be stated by the symbol τ_I, where the set *I* includes all internal communications taking place into the system.

Finally, at this point, the comparison between the behaviour of the model and the real system takes place, revealing whether both are rooted branching bisimilar, which leads to the conclusion that the model gets verified [23].

V. MODEL SPECIFICATION AND VERIFICATION

The system is composed by both fog and cloud levels. We suppose an IoT device with a set of virtual services initially allocated in the cloud level. Virtual services migrate from the cloud to the fog level for some performance reasons, such as, latency improvement or bandwidth saving.

Once the virtual services associated to that IoT device get into the fog, through r_0 action, they have to follow the device movement along the system, migrating from one building block, F_x, to the next building block, F_{x+1}, through x action, or otherwise getting back to the cloud level through $\hat{x}$ action.

Let be A= $\{r, s\}$, the atomic read and send actions, then the fog system is defined as the following tuple:

$$S_i = \langle F, D, U, H, T \rangle \tag{2}$$

Where F is a non-empty set of building blocks, D and U are non-empty sets of communicating channels between the fog and the cloud levels, respectively; H is a non-empty set of communicating actions inside the fog system and $T \subseteq \{A \times \{D, U\}\}$ is the set of communicating actions over the channels, where the actions r_x means an atomic *read* action over channel x, and s_y means an atomic *send* action over channel y.

For the first case, $S_{i=1}$, considering only one building block, the diagram is depicted in Fig. 1, modelled as the cloud and the fog levels, with only two communication channels: channel *0* is used to take down from cloud to fog level, thus hosting taking place in the unique available block; and channel $\hat{1}$, when the virtual device leaves the fog level, hence coming back to the cloud level.

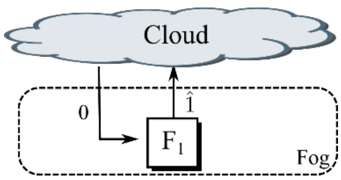

Fig. 1. Fog computing schematic diagram with 1 building block

The system is described by the S_1 tuple as follows:

$$S_1 = \langle \{F_1\}, \{0\}, \{\hat{1}\}, \{\emptyset\}, \{r_0, s_{\hat{1}}\} \rangle \tag{3}$$

Regarding the description of the model, S_1 is specified with the following recursive ACP expression:

$$F_1 = r_0 \cdot s_{\hat{1}} \cdot F_1 \tag{4}$$

Considering that in this first stage of the model there is not any internal communication, therefore $H = \emptyset$, then applying the encapsulation operator, ∂_H over F_1 and resolving from several ACP axioms:

$$\partial_H(F_1) = r_0 \cdot \partial_H(s_{\hat{1}} \cdot F_1) \tag{5}$$

$$\partial_H(s_{\hat{1}} \cdot F_1) = s_{\hat{1}} \cdot \partial_H(F_1)$$

Likewise, the lack of any internal communication makes $I = \emptyset$, so applying the abstraction operator, τ_I over $\partial_H(F_1)$, and resolving from several ACP axioms:

$$\tau_I(\partial_H(F_1)) = r_0 \cdot \tau_I(\partial_H(s_{\hat{1}} \cdot F_1)) \tag{6}$$

$$\tau_I(\partial_H(s_{\hat{1}} \cdot F_1)) = s_{\hat{1}} \cdot \tau_I(\partial_H(F_1))$$

The ACP specification for this first approximation can be modelled as a finite machine as it is depicted in Fig. 2.

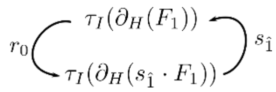

Fig. 2. Finite state diagram for 1 building block

As the final objective of this modelling is to describe the external behaviour of the specification, at this stage, in this model with just one building block, there may be either no items or one item into the system. Therefore, considering X_i with the following semantic value:

$$X_i \Rightarrow i \text{ items inside the system}$$

The external system behaviour can be described through the following recursive equational system:

$$X_0 = r_0 \cdot X_1 \tag{7}$$

$$X_1 = s_{\hat{1}} \cdot X_0$$

From the comparative between the model system and the external behaviour of the real system, we have found that X_0 and X_1 are modelled ($\vDash$) through the following expressions:

$$X_0 \vDash \tau_I(\partial_H(F_1) \tag{8}$$

$$X_1 \vDash \tau_I(\partial_H(s_{\hat{1}} \cdot F_1)$$

Therefore, they are both rooted branching bisimilar, hence, the first model gets verified.

As we must model the internal communications at fog level, we need include the next fog element, F_2, as we can see in Fig. 3:

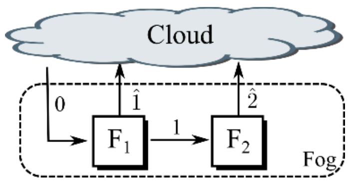

Fig. 3. Fog computing schematic diagram with 2 building blocks

In this second stage of the model, it takes into account the migration from cloud to fog and return back to the cloud, as well as it the movement of the device along the system, this is, the migration of the virtualized service inside the fog level. Therefore, the system (depicted in Fig. 3) is described by the S_2 tuple as follows:

$$S_2 = \langle \{F_1, F_2\}, \{0\}, \{\hat{1}, \hat{2}\}, \{\gamma(s_1, r_1) \to c_1\}, \{r_0, s_1, r_1, s_{\hat{1}}, s_{\hat{2}}\} \rangle$$

In this case, the system, composed by two fog elements, F_1 and F_2 is modelled through the following ACP expressions:

$$F_1 = r_0 \cdot (s_1 + s_{\hat{1}}) \cdot F_1 \tag{9}$$

$$F_2 = r_1 \cdot s_{\hat{2}} \cdot F_2$$

Considering the communication between F_1 and F_2, defined in $H = \{\gamma(s_1, r_1) \rightarrow c_1\}$, applying the encapsulation operator, ∂_H over both F_1 and F_2, and resolving from several ACP axioms:

$$\partial_H(F_1 \parallel F_2) = r_0 \cdot \partial_H\big((s_{\hat{1}} + s_1) \cdot F_1 \parallel F_2\big) \qquad (10)$$

$$\begin{aligned}
&\partial_H\big((s_{\hat{1}} + s_1) \cdot F_1 \parallel F_2\big) = \\
&\quad s_{\hat{1}} \cdot \partial_H(F_1 \parallel F_2) + \\
&\quad c_1 \cdot \partial_H(F_1 \parallel s_{\hat{2}} \cdot F_2)
\end{aligned}$$

$$\begin{aligned}
&\partial_H(F_1 \parallel s_{\hat{2}} \cdot F_2) = \\
&\quad r_0 \cdot \partial_H\big((s_{\hat{1}} + s_1) \cdot F_1 \parallel s_{\hat{2}} \cdot F_2\big) + \\
&\quad s_{\hat{2}} \cdot \partial_H(F_1 \parallel F_2)
\end{aligned}$$

$$\begin{aligned}
&\partial_H\big((s_{\hat{1}} + s_1) \cdot F_1 \parallel s_{\hat{2}} \cdot F_2\big) = \\
&\quad s_{\hat{1}} \cdot \ \partial_H(F_1 \parallel s_{\hat{2}} \cdot F_2) + \\
&\quad s_{\hat{2}} \cdot \ \partial_H\big((s_{\hat{1}} + s_1) \cdot F_1 \parallel F_2\big)
\end{aligned}$$

Likewise, the abstraction operator is to be used to hide all internal communications, τ_I over $\partial_H(F_1||F_2)$, making them into silent steps, and resolving from several ACP axioms:

$$\begin{aligned}
&\tau_I\big(\partial_H(F_1 \parallel F_2)\big) = \\
&\quad r_0 \cdot \tau_I(\partial_H(s_{\hat{1}} + s_1) \cdot F_1 \parallel F_2)
\end{aligned} \qquad (11)$$

$$\begin{aligned}
&\tau_I(\partial_H(s_{\hat{1}} + s_1) \cdot F_1 \parallel F_2) = \\
&\quad s_{\hat{1}} \cdot \tau_I\big(\partial_H(F_1 \parallel F_2)\big) + \\
&\quad \tau_I\big(\partial_H(F_1 \parallel s_{\hat{2}} \cdot F_2)\big)
\end{aligned}$$

$$\begin{aligned}
&\tau_I\big(\partial_H(F_1 \parallel s_{\hat{2}} \cdot F_2)\big) = \\
&\quad r_0 \cdot \tau_I(\partial_H(s_{\hat{1}} + s_1) \cdot F_1 \parallel s_{\hat{2}} \cdot F_2) + \\
&\quad s_{\hat{2}} \cdot \tau_I\big(\partial_H(F_1 \parallel F_2)\big)
\end{aligned}$$

$$\begin{aligned}
&\tau_I(\partial_H(s_{\hat{1}} + s_1) \cdot F_1 \parallel s_{\hat{2}} \cdot F_2) = \\
&\quad s_{\hat{1}} \cdot \tau_I\big(\partial_H(F_1 \parallel s_{\hat{2}} \cdot F_2)\big) + \\
&\quad s_{\hat{2}} \cdot \tau_I\Big(\partial_H\big((s_{\hat{1}} + s_1) \cdot F_1 \parallel F_2\big)\Big)
\end{aligned}$$

The ACP specification for this second approximation can be modelled as a finite machine as it is depicted in Fig. 4:

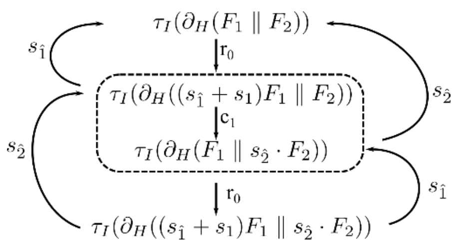

Fig. 4. Finite state diagram for 2 Building Block

The external system behaviour can be described through the following recursive equational system:

$$X_0 = r_0 \cdot X_1 \qquad (12)$$

$$X_1 = r_0 \cdot X_2 + (s_{\hat{1}} + s_1) \cdot \ X_0$$

$$X_2 = (s_{\hat{1}} + s_1) \cdot \ X_1$$

At this point, the formal model description has been deduced (Fig. 4) and the external behaviour of the system can be accommodated through the recursive equational system by X_0, X_1 and X_2, considering X_1 the case where the item is rightmost, as the transition from F_1 to F_2 is internal. That way, F_1 is free and a new item may get into it:

$$X_0 \vDash \ \tau_I\big(\partial_H(F_1 \parallel F_2)\big) \qquad (13)$$

$$X_1 \vDash \ \tau_I\big(\partial_H(F_1 \parallel s_{\hat{2}} \cdot F_2)\big)$$

$$X_2 \vDash \ \tau_I\Big(\partial_H\big((s_{\hat{1}} + s_1) \cdot F_1 \parallel s_{\hat{2}} \cdot F_2\big)\Big)$$

Therefore, both are rooted branching bisimilar, thus, the model gets verified.

Finally, the S_3 system is specified, depicted in Fig. 5, where there are two internal communications between F_1 and F_2, defined by $\gamma(s_1, r_1) \rightarrow c_1$ and another defined by $\gamma(s_2, r_2) \rightarrow c_2$. Therefore, the system S_3 is defined by the following tuple:

$$S_3 = \left\langle \begin{matrix} \{F_1, F_2, F_3\}, \{0\}, \{\hat{1}, \hat{2}, \hat{3}\}, \{c_1, c_2\}, \\ \{r_0, s_1, r_1, s_{\hat{1}}, s_2, r_2, s_{\hat{2}}, s_{\hat{3}}\} \end{matrix} \right\rangle \qquad (14)$$

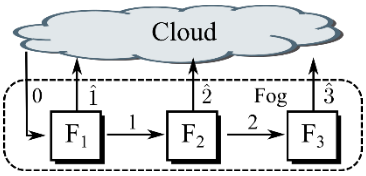

Fig. 5. Fog computing schematic diagram with 3 building blocks

The description of the model S_3 is specified with the following recursive ACP expression:

$$F_1 = r_0 \cdot (s_{\hat{1}} + s_1) \cdot F_1 \qquad (15)$$

$$F_2 = r_1 \cdot (s_{\hat{2}} + s_2) \cdot F_2$$

$$F_3 = r_2 \cdot s_{\hat{3}} \cdot F_2$$

Considering that at this stage of the model the internal communication set is defined by $H = \{c_1, c_2\}$, then applying the encapsulation operator, ∂_H over F_1, F_2 and F_3, and resolving from several ACP axioms:

$$\begin{aligned}
&\partial_H(F_1 \parallel F_2 \parallel F_3) = \\
&\quad r_0 \cdot \partial_H\big((s_{\hat{1}} + s_1) \cdot F_1 \parallel F_2 \parallel F_3\big)
\end{aligned} \qquad (16)$$

$$\begin{aligned}
&\partial_H\big((s_{\hat{1}} + s_1) \cdot F_1 \parallel F_2 \parallel F_3\big) = \\
&\quad s_{\hat{1}} \cdot \partial_H(F_1 \parallel F_2 \parallel F_3) + \\
&\quad c_1 \cdot \partial_H(F_1 \parallel (s_{\hat{2}} + s_2) \cdot F_2 \parallel F_3)
\end{aligned}$$

$$\begin{aligned}
&\partial_H(F_1 \parallel (s_{\hat{2}} + s_2) \cdot F_2 \parallel F_3) = \\
&\quad s_{\hat{2}} \cdot \partial_H(F_1 \parallel F_2 \parallel F_3) + \\
&\quad c_2 \cdot \partial_H(F_1 \parallel F_2 \parallel s_{\hat{3}} \cdot F_3) + \\
&\quad r_0 \cdot \partial_H\begin{pmatrix} (s_{\hat{1}} + s_1) \cdot F_1 \parallel \\ (s_{\hat{2}} + s_2) \cdot F_2 \parallel F_3 \end{pmatrix}
\end{aligned}$$

$$\begin{aligned}&\partial_H(F_1 \parallel F_2 \parallel s_{\hat{3}} \cdot F_3) = \\ &\quad s_{\hat{3}} \cdot\ \partial_H(F_1 \parallel F_2 \parallel F_3) + \\ &\quad r_0 \cdot \partial_H\big((s_{\hat{1}} + s_1) \cdot F_1 \parallel F_2 \parallel s_{\hat{3}} \cdot F_3\big)\end{aligned}$$

$$\begin{aligned}&\partial_H\big((s_{\hat{1}} + s_1) \cdot F_1 \parallel (s_{\hat{2}} + s_2) \cdot F_2 \parallel F_3\big) = \\ &\quad s_{\hat{2}} \cdot \partial_H\big((s_{\hat{1}} + s_1) \cdot F_1 \parallel F_2 \parallel F_3\big) + \\ &\quad c_{\hat{2}} \cdot \partial_H\big((s_{\hat{1}} + s_1) \cdot F_1 \parallel F_2 \parallel s_{\hat{3}} \cdot F_3\big)\end{aligned}$$

$$\begin{aligned}&\partial_H((s_{\hat{1}} + s_1) \cdot F_1 \parallel F_2 \parallel s_{\hat{3}} \cdot F_3) = \\ &\quad s_{\hat{1}} \cdot \partial_H(F_1 \parallel F_2 \parallel s_{\hat{3}} \cdot F_3) + \\ &\quad s_{\hat{3}} \cdot \partial_H((s_{\hat{1}} + s_1) \cdot F_1 \parallel F_2 \parallel F_3) + \\ &\quad c_1 \cdot \partial_H(F_1 \parallel (s_{\hat{2}} + s_2) \cdot F_2 \parallel s_{\hat{3}} \cdot F_3)\end{aligned}$$

$$\begin{aligned}&\partial_H(F_1 \parallel (s_{\hat{2}} + s_2) \cdot F_2 \parallel s_{\hat{3}} \cdot F_3) = \\ &\quad s_{\hat{2}} \cdot \partial_H(F_1 \parallel F_2 \parallel s_{\hat{3}} \cdot F_3) + \\ &\quad s_{\hat{3}} \cdot \partial_H(F_1 \parallel (s_{\hat{2}} + s_2) \cdot F_2 \parallel F_3) + \\ &\quad r_0 \cdot \partial_H\begin{pmatrix} (s_{\hat{1}} + s_1) \cdot F_1 \parallel \\ (s_{\hat{2}} + s_2) \cdot F_2 \parallel s_{\hat{3}} \cdot F_3 \end{pmatrix}\end{aligned}$$

$$\begin{aligned}&\partial_H\begin{pmatrix} (s_{\hat{1}} + s_1) \cdot F_1 \parallel \\ (s_{\hat{2}} + s_2) \cdot F_2 \parallel s_{\hat{3}} \cdot F_3 \end{pmatrix} = \\ &\quad s_{\hat{1}} \cdot \partial_H(F_1 \parallel (s_{\hat{2}} + s_2) \cdot F_2 \parallel s_{\hat{3}} \cdot F_3) + \\ &\quad s_{\hat{2}} \cdot \partial_H\big((s_{\hat{1}} + s_1) \cdot F_1 \parallel F_2 \parallel s_{\hat{3}} \cdot F_3\big) + \\ &\quad s_{\hat{3}} \cdot \partial_H\big((s_{\hat{1}} + s_1) \cdot F_1 \parallel (s_{\hat{2}} + s_2) \cdot F_2 \parallel F_3\big)\end{aligned}$$

The application of the abstraction operator, τ_I over $\partial_H(F_1||F_2||F_3)$, cancels all internal communication factors (terms with c_1 and c_2), leaving the rest of the expressions as they are above, although each term is enclosed by $\tau_I(\,)$. This leads to the conclusion that the following expressions are equivalent, meaning that they all have 1 block occupied, no matter which one:

$$\begin{aligned}&\tau_I\Big(\partial_H\big((s_{\hat{1}} + s_1) \cdot F_1 \parallel F_2 \parallel F_3\big)\Big) \Leftrightarrow \\ &\tau_I\big(\partial_H(F_1 \parallel (s_{\hat{2}} + s_2) \cdot F_2 \parallel F_3)\big) \Leftrightarrow \\ &\tau_I\big(\partial_H(F_1 \parallel F_2 \parallel s_3 \cdot F_3)\big)\end{aligned} \tag{17}$$

Likewise, the following expressions are equivalent, related to the cases where there are 2 blocks occupied, regardless which one:

$$\begin{aligned}&\tau_I\left(\partial_H\begin{pmatrix} (s_{\hat{1}} + s_1) \cdot F_1 \parallel \\ (s_{\hat{2}} + s_2) \cdot F_2 \parallel F_3 \end{pmatrix}\right) \Leftrightarrow \\ &\tau_I\big(\partial_H((s_{\hat{1}} + s_1) \cdot F_1 \parallel F_2 \parallel s_3 \cdot F_3)\big) \Leftrightarrow \\ &\tau_I\big(\partial_H(F_1 \parallel (s_{\hat{2}} + s_2) \cdot F_2 \parallel s_3 \cdot F_3)\big)\end{aligned} \tag{18}$$

Therefore, the last one for each case is going to be considered as the canonical one, as it makes clear that new items may get into the system.

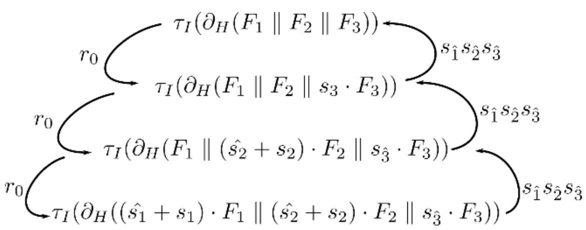

Fig. 6. Finite state diagram for 3 building blocks

When there are three building blocks, the number of items into the system ranges from zero to three, regardless in which block they are located. Therefore, and considering the semantic meaning of X_i, the external behaviour of the real system may be described with the following recursive expressions:

$$X_0 = r_0 \cdot X_1 \tag{19}$$

$$X_1 = r_0 \cdot X_2 + (s_{\hat{1}} + s_{\hat{2}} + s_{\hat{3}}) \cdot X_0$$

$$X_2 = r_0 \cdot X_3 + (s_{\hat{1}} + s_{\hat{2}} + s_{\hat{3}}) \cdot X_1$$

$$X_3 = (s_{\hat{1}} + s_{\hat{2}} + s_{\hat{3}}) \cdot X_2$$

The aforesaid variables match those expressed in the state diagram of the model within this section, in upside down order, hence both the real system and the model proposed are rooted branching bisimilar, thus, the model gets verified.

There is a pattern clearly spotted as the number of building blocks grows, in a way that the number of possible states in the system goes from 0 to N (depicted in Fig. 7), this is, there are $N + 1$ states in all, coming from the one having none items inside, all the way to the one being full.

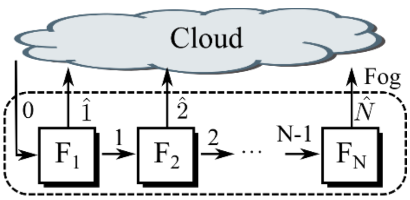

Fig. 7. Fog computing schematic diagram for N building blocks

Additionally, for each of the states, there is a number of equivalent expressions, denoting all possible combinations of a given value along all available building blocks. Those numbers are given by the *i*-th row of the Pascal Triangle, as in Figure 8.

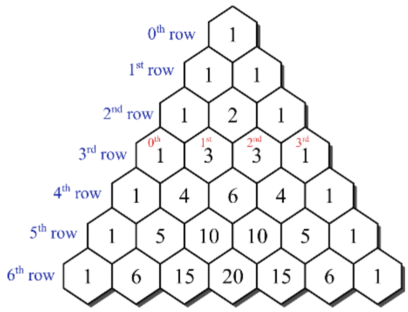

Fig. 8. Pascal Triangle for the first 6 rows

The values appearing therein are obtained by calculating the number of combinations for the *n*-th row and the *k*-th column of the triangle, meaning the number of combinations of n items taken k at a time, according to the following expression:

$$c(n,k) = \binom{n}{k} = \frac{n!}{k!\,(n-k)} \tag{20}$$

Regardless of the number of those equivalent expressions, a canonical representative for each state is to be chosen, and that is going to be the one with the items situated rightmost.

On the other hand, three logic rules may be spotted in the cases presented above for describing the behaviour of the fog computing system, which might be generalised by applying natural deduction rules on set *V* of variables, where the upper part is the antecedent and the lower one is the consequent and the predicate $x \to \surd$ represents successful termination after the execution of action *x*.

The first logic rule (*I*) is related to a new item coming into the system through the r_0 action; the second one (*II*) has to do with an item passing from one block to its successor neighbour through an internal communication action (c_i); and finally, the third one (*III*) is about an existing item going out of the system through the exit action ($s_{\hat{i}}$).

$$I: \frac{r_0 \to \surd}{\partial_H(F_1 \parallel \cdots \parallel F_N) \to \partial_H\big((s_{\hat{1}} + s_1)F_1 \parallel \cdots \parallel F_N\big)} \tag{21}$$

$$II: \frac{i \in [1, N-1] \mid c_i \to \surd}{\partial_H(\cdots \parallel (s_{\hat{i}} + s_i)F_i \parallel \cdots \parallel F_N) \to (\cdots \parallel F_i \parallel (s_{\widehat{i+1}} + s_{i+1})F_{i+1} \parallel \cdots \parallel)}$$

$$III: \frac{\hat{i} \in [1, N] \mid s_{\hat{i}} \to \surd}{\partial_H(\cdots \parallel (s_{\hat{i}} + s_i)F_i \parallel \cdots) \to (\cdots \parallel F_i \parallel \cdots)}$$

Taking all of that into account, and after applying all the operators considered above, the relationship among the possible states and the number of items in the system goes like this:

$$\begin{aligned} X_0 &\vDash \tau_I(\partial_H(F_1 \parallel \cdots \parallel F_{N-1} \parallel F_N)) \\ X_1 &\vDash \tau_I(\partial_H(F_1 \parallel \cdots \parallel F_{N-1} \parallel s_{\hat{N}} \cdot F_N)) \\ X_2 &\vDash \tau_I(\partial_H(F_1 \parallel \cdots \parallel (s_{\widehat{N-1}} + s_N) \cdot F_{N-1} \parallel s_{\hat{N}} \cdot F_N)) \\ &\vdots \\ X_N &\vDash \tau_I(\partial_H((s_{\hat{1}} + s_1) \cdot F_1 \parallel \cdots \parallel (s_{\widehat{N-1}} + s_N) \cdot F_{N-1} \\ &\qquad \parallel s_{\hat{N}} \cdot F_N) \end{aligned} \tag{23}$$

Therefore, when considering a given number of items *i* inside the system, X_i, the concurrent operator $N - i$ within the system gets multiplied by the factor $(s_{\widehat{N-i}} + s_{N-i})$, and the rest of the concurrent operators are the same as X_{i-1}.

However, it is to be noted that the last block only has one exit point to the cloud, whereas the rest of blocks have also an extra exit point to the following block, and that makes a difference for the $i = 1$ case, related to $i > 1$.

Alternatively, structural induction might be applied in order to get the expression for *N* blocks, considering the base case $N = 1$, supposing the case *N*, and calculating the result for $N + 1$ to be sure that it meets the expectations. It clearly does, therefore, the expression for *N* blocks is right. Furthermore, the verification process goes the same way as in the previous cases, so it is clearly achieved.

VI. CONCLUSIONS

In this paper we undertook a study in order to get a formal algebraic description of a fog/IoT computing environment composed by *N* fog facilities where each of them are considered as building blocks, interconnected in a linear manner, such as in a daisy chain.

The paper has been carried out in an incremental manner, starting with the study of a system with just one building block, where a specification of the model has been achieved, and in turn, verification of the model has been obtained by comparing the external behaviour of both the real system and the model itself, in a way that if both are rooted branching bisimilar, according to ACP principles, the verification of the model is granted.

After that, the number of building blocks has been increased up to two and three, carrying out analogous studies, and finally, a general expression has been reached for *N* building blocks, obtaining its verification. Therefore, it has been shown that a fog computing system may be formally described and modelled in an algebraic manner.

REFERENCES

[1] O. Osanaiye, S. Chen, Z. Yan, R. Lu, K.R. Choo, M. Dlodlo, *From Cloud to Fog Computing: A Review and a Conceptual Live VM Migration Framework*, in RACRAN'2017, Vol. 5, pp. 8284-8300.

[2] F. Arute et al, *Quantum supremacy using a programmable superconducting processor*, in Nature 2019, Vol. 574, pp. 505-510.

[3] C.S.R. Prabhu, *Overview - Fog Computing and Internet-of-Things (IoT)*, in EAI Endorsed Transactions on Cloud Systems 2017, Issue 10, Article 5, pp. 1-24.

[4] P. Sethi, S.R. Sarangi, *Internet of Things: Architectures, Protocols, and Applications*, in JECE'2017, article ID 9324035, pp. 1-25.

[5] A. Biral, M. Centenaro, A. Zanella, L. Vangelista, M. Zorzi, *The challenges of M2M massive access in wireless cellular networks*, in DCN'2015, Vol. 1, Issue 1, pp. 1-19.

[6] A. Souri and M. Norouzi, *State-of-the-Art Survey on Formal Verification of the Internet of Things Applications*, in Journal of Service Science Research 2019, Vol. 11, pp. 47–67.

[7] S. Marir, F. Belala and N. Hameurlain, *A Formal Model for Interaction Specification and Analysis in IoT Applications*, in MEDI 2018, pp. 371-384.

[8] K. Hofer-Schmitz and B. Stojanovic, *Towards Formal Methods of IoT Application Layer Protocols*, in CMI 2019, pp. 1-6.

[9] A. Timenko, *On the aspects of Iot protocols specification and checking*, in Shipbuilding & Marine Infrastructure, 2019, Nº 2 (12), pp. 35-41.

[10] E. Sherratt, *SDL: Meeting the IoT Challenge*. In System Analysis and Modeling. Technology-Specific Aspects of Models. SAM 2016. Lecture Notes in Computer Science, vol 9959. Springer, pp. 36-50.

[11] A.F. Fuentes, E. Tamura, *From SDL Modeling to WSN Simulation for IoT Solutions*, in Applied Computer Sciences in Engineering. WEA 2018. Comms in Computer and Inform. Science, Vol. 916. Springer.

[12] A. J. Vattakunnel, N. S. Kumar, *Modelling and Verification of CoAP over Routing Layer Using SPIN Model Checker*, in Procedia Computer Science, 2016, Vol 93, pp. 299-308.

[13] Y. Choi, *Automated Validation of IoT Device Control Programs Through Domain-Specific Model Generation*, in Software Engineering and Formal Methods. SEFM 2018. Lecture Notes in Computer Science, vol 10886. Springer.

[14] Z. Samman et al., *Fog Computing Over IoT: A Secure Deployment and Formal Verification*, in IEEE Access 5 (2017), pp. 27132-27144.

[15] S. Ouchani, *Ensuring the Functional Correctness of IoT through Formal Modeling and Verification*, in MEDI 2018. Lecture Notes in Computer Science, Vol. 11163, Springer, 2018.

[16] S. Lee, Y. Choe and M. Lee, *dT-Calculus: A Formal Method to Specify Distributed Mobile Real-Time IoT Systems*, IntechOpen, 2018.

[17] P.J. Roig, S. Alcaraz, K. Gilly and C. Juiz, *Modelling VM Migration in a Fog Computing Environment*, in Elektronika Ir Elektrotechnika 2019, Vol. 25, No.5, pp. 75-81.

[18] D.A. Padua, *Encyclopedia of Parallel Computing*, Springer, 2011.

[19] M. Gazda, W. Fokkink, V. Massaro, *Congruence from the operator's point of view*, in Acta Informatica 2019, pp. 1-23.

[20] J.F. Groote, M.R. Mousavi, *Modelling and Analysis of Communicating Systems*, MIT Press, 2014.

[21] W. Fokkink, *Introduction to Process Algebra*, Springer, 2007.

[22] J.A. Bergstra, J.W. Klop, *Algebra of communicating processes with abstraction*, in TCS'1985, Vol. 37, pp. 77-121.

[23] L. Lockefeer, D.M. Williams, W. Fokkink, *Formal specification and verification of TCP extended with the Window Scale Option*, in FMICS'2014, pp. 63-77.