Performance Analysis of Edge-Fog-Cloud Architectures in the Internet of Things

Kurt Geihs
EECS Department
University of Kassel
Kassel, Germany
geihs@uni-kassel.de

Harun Baraki
EECS Department
University of Kassel
Kassel, Germany
hba@uni-kassel.de

Antonio de la Oliva
Telematics Department
Universidad Carlos III de Madrid
Madrid, Spain
aoliva@it.uc3m.es

***Abstract*—We present a general performance model for communication architectures in Internet of Things scenarios. The architecture involves three processing layers, i.e., edge, fog, and cloud computing. According to the objectives of edge and fog computing, we assume that a certain percentage of the data is finally processed at the edge and fog layer. The rest is forwarded to the next higher layer for further processing. Data processing is modeled as a sequence of queueing stations. This allows us to compute performance parameters such as throughput and response times at all layers. The main contribution of this paper is an easy to use method for IoT application designers that provides a very fast and very flexible parametric support for evaluating design trade-offs in edge-fog-cloud configurations.**

Keywords—edge computing, fog computing, cloud computing, Internet of Things, performance analysis, queueing theory

I. Introduction

Applications in the Internet of Things (IoT) process a large amount of data that are generated by a wide variety of sensors. This is where cloud computing came in, because the processing and storage capacities of the IoT devices are not sufficient to run demanding applications. On the other hand, cloud computing based on servers in remote computing centers can lead to execution latencies and operational cost that are unacceptable for many IoT application domains. For example, in novel networks consisting of different kinds of devices such as autonomous ground vehicles, unmanned air vehicles, stationary road-side stations, regional computing facilities etc. the timing requirements cover a wide spectrum because data is processed at different places ranging from processing within the end devices over ground stations up to cloud computing centers. Therefore, edge and fog architectures bring the computing and storage capacity, i.e., the processing of the data, closer to the data sources [1], [2]. Figure 1 illustrates a general 3-layered architecture where application execution may involve the three layers edge, fog, and cloud. In the following we call the cloud layer the highest layer and the edge layer the lowest layer.

Our emphasis in this paper is on the performance of such configurations. We assume that IoT devices produce sensor values that are processed layer after layer, whereby the processing of a certain percentage of sensor values is completed at the current layer and does not incur processing at the next higher layer. This reflects the motivation behind edge and fog computing, i.e., some of the processing moves closer to the data sources avoiding high latencies and cost. The processing capacity of edge devices is typically smaller than that of fog servers whose processing capacity is typically smaller than that of cloud servers. On the other hand, it may be more expensive to use cloud and fog computing resources. The solution to the configuration problem is crucial for designers of large-scale IoT applications. Our performance model provides a means to evaluate these trade-offs between placing computations at the different layers.

The main contribution of this paper is a performance model for such a 3-layer communication architecture. The performance model shall enable designers of IoT applications to estimate the response times of their application architectures for different numbers and processing speeds of processors at the edge-fog-cloud layers and thus to study the various trade-offs. Our goal was to develop an analytical model based on queueing theory that would facilitate a quick and easy to use exploration of a large variety of configurations. This is not to say that we underestimate the value of simulations, but in this paper we use simulations only to validate the analytical model and thus to get confidence in its correctness.

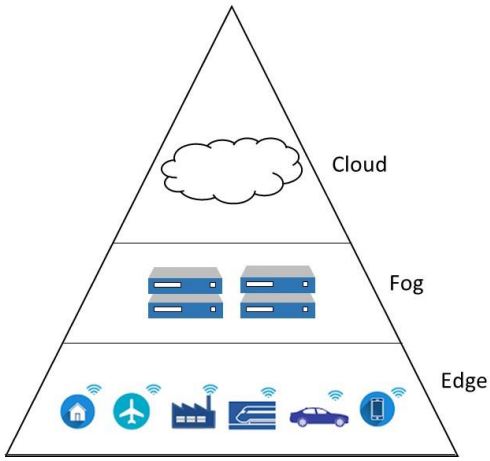

Figure 1. Edge-Fog-Cloud Architecture.

The following Section 2 presents the structure of our performance model and its parameters. In Section 3 results for different configurations obtained with the model are discussed, including cross-validation by simulation studies

978-0-7381-2394-3/20/$31.00 ©2020 IEEE
DOI 10.1109/UCC48980.2020.00059

using the JMT simulation framework. Section 4 presents related work. In Section 5 we conclude the paper and point to future work.

II. Performance Model

In this section, we present our performance model which has been developed with the objective to provide the application developer with a quick and effective design support based on queueing theory.

A. Assumptions

The general communication architecture for edge-fog-cloud computing is shown in Figure 1. In order to make the model even more flexible we add a layer with sensors below the edge layer. Figure 2 shows the general structure of the performance model that captures the essence of Figure 1:

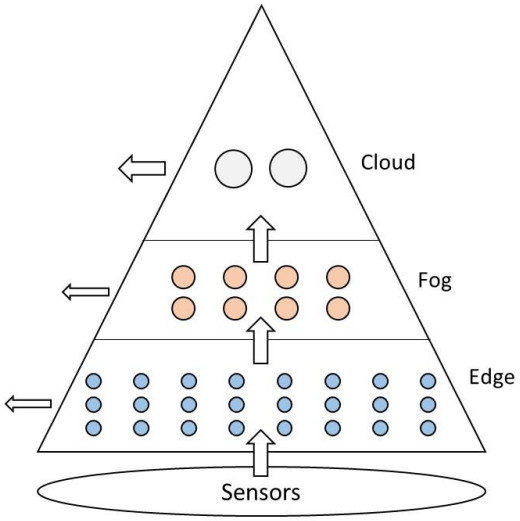

Figure 2. Overall performance model with sensors and processors on the edge / fog / cloud layers.

Sensors produce values that are pre-processed by a number of edge processing devices. The number of sensors may be greater than the number of edge processors, i.e., an edge device may have several sensors that feed data into the system. For a certain percentage of sensor values, the edge processing is all that is required and the processing of these values is finished at the edge layer – indicated in Figure 2 by the arrow pointing to the left. For example, a certain class of sensor values is filtered out by the edge devices. The rest of the values require more processing at higher layers; they are forwarded to the fog layer. The fog layer operates in the same mode as the edge layer: A certain percentage of requests obtain their final processing, and the rest is forwarded to the cloud layer. For example, this split can be used to model the aggregation of sensor values such that only the aggregate value is forwarded to the cloud layer for further data analytics and long-term storage.

Obviously, including the sensors in the model does not preclude a set-up where we have the same number of sensors and edge devices, i.e., every edge device is a single data source. Likewise, one or two of the three layers in the model can be removed easily by adjusting the finishing / forwarding probabilities and the processing times accordingly.

On each layer there operates a certain number of processors. Edge and fog processors have a queue for finishing and a queue for forwarding values. The split is determined by given finishing/forwarding probabilities, as explained in the next subsection. Thus, the processing sequence is modeled as a sequence of queues. The set of sensors produces a stream of sensor values. These values are processed layer by layer by the available processors in a load balancing fashion, i.e., the values coming from a lower layer are evenly distributed to the available processors on the next higher layer. At the edge and fog layers the data stream is split into values that obtain their final service and into values forwarded to the next layer. The final processing of values and the forwarding of values in the edge and fog layers may take different service time distributions. The forwarding service time is assumed to include the communication processing, e.g. transmission time, error handling, retransmissions, etc. The cloud layer is the highest layer; there is no splitting of the input stream at this layer and all values are processed finally according to the specified service time distribution of the cloud.

Figure 3 shows an example for the queues per processing station on the three layers for a mixed M/M/1 and M/D/1 configuration. The queueing order is First-Come-First-Served. While the structure of the queueing models for the edge and fog processing nodes is identical, the overall

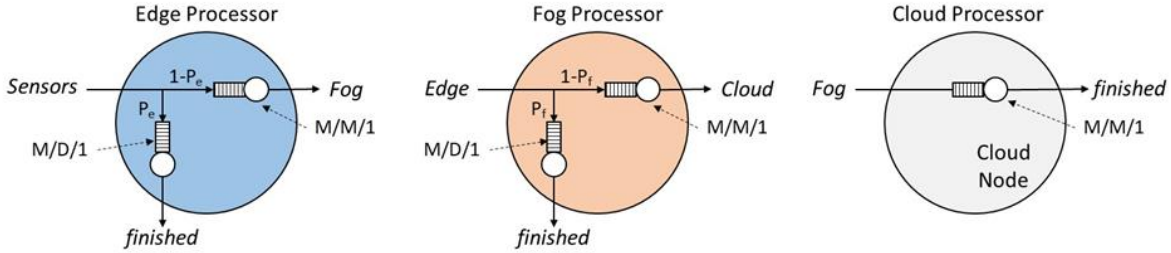

Figure 3. Queueing model for a mix of M/M/1 and M/D/1 processing stations

performance model is parametric in terms of the number of nodes and their processing capacities at the different layers.

The reader should note that while the finishing stations can have any M/G/1 service time distribution, all forwarding stations are modeled as M/M/1 queues. This simplification eases the performance calculation substantially, because a probabilistic split of a Poisson process creates two arrival processes that are still Poisson processes. Similarly, the joining of two Poisson streams preserves the Poisson property for the combined stream. Thus, arrivals across all layers are all Poisson distributed and we can apply well known closed form solutions for computing the utilization, waiting time, and response time [3].

B. *Model parameters*

Table 1 shows the model parameters that can be set by the user of the performance model according to the application environment configuration.

TABLE 1. MODEL PARAMETERS.

s / e / f / c	number of sensors / edge processors / fog processors / cloud processors
p_e / 1-p_e	probability of finishing / forwarding at an edge processor
p_f / 1-p_f	probability of finishing / forwarding at a fog processor
λ_s	generation rate per sensor
S_e / S_f / S_c	mean service time for finishing a value per edge / fog / cloud processor
$S_{e\text{-}not}$ / $S_{f\text{-}not}$	mean service time for forwarding a value per edge / fog processor

The most relevant result values that we compute based on these model parameters and thus the main evaluation criterion for a given communication architecture configuration are the mean response times $E[R_x]$ for finished jobs at the edge / fog / cloud layers, i.e., x = e(dge), f(og) or c(loud). Note that the response time of a finished job at the fog layer includes the response time for forwarding the job at the edge layer. Likewise, the response time of a finished job at the cloud layer includes the response times for forwarding the job through the edge and fog layers.

It is easy to see from Figure 3 that all utilizations, waiting times, and response times at the three layers can be computed using standard formulas for M/M/1 and M/G/1 queues [3], taking into account that arrivals are split into finished and forwarded values according to given probabilities. For M/G/1 queues we have the well-known general solutions:

$$\rho = \lambda\, E[S]$$
$$E[W] = \lambda\, E[S^2] / 2\,(1-\rho)$$
$$E[R] = E[W] + E[S]$$

where λ denotes the arrival rate, ρ the traffic intensity, S the service time, W the waiting time in the queue, and R the overall response time.

The use of closed-form solutions makes the performance model very efficient, even with the simplifying load balancing assumption, as discussed above. The parametric model provides results very quickly for a large variety of configurations. Clearly, simulating these configurations would take much more time.

III. RESULTS

Obviously, the degree of potential variability based on the set of input parameters is huge, and there is no point in presenting too many diagrams here. In the following, we show only a few examples for the mean response times of different configurations.

A. *Example configurations*

The first example is based on the following input parameters (Table 2):

TABLE 2. EXAMPLE 1 – INPUT PARAMETERS.

s	**e**	**f**	**c**	**p_e**	**p_f**	**λ_s**
10..100	10	5	1	0.25	0.75	0.05

S_e	**$S_{e\text{-}not}$**	**S_f**	**$S_{f\text{-}not}$**	**S_c**
0.25	0.1	0.5	0.1	0.75

The number of sensors, i.e., the load presented to the systems, varies from 10 to 100. The arrival distribution is Poisson. The finishing takes a constant service time, the forwarding service time is exponential. Hence, we have a mix of M/D/1 for finishing in edge and fog processors and M/M/1 for forwarding and for cloud processing, just like shown in Figure 3 above. Figure 4 shows the results, i.e. the mean response times for jobs that are finished at the edge, fog, and cloud layers.

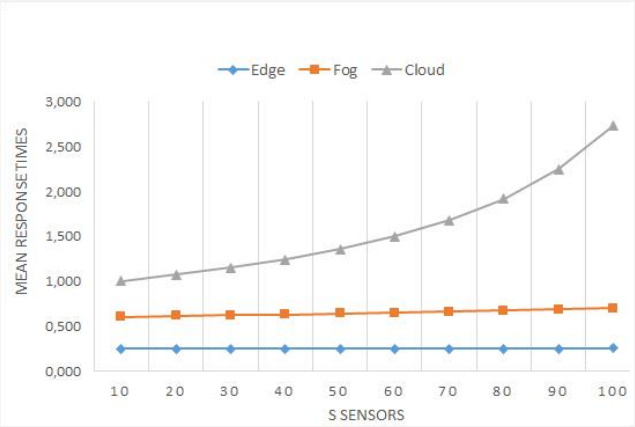

Figure 4. Example 1 – response times for M/M/1 and M/D/1 configuration.

As explained in the previous section, the displayed response times at the fog and cloud layers are accumulated including the response times of the layers below.

In Example 2 we use the same input parameters, but with M/M/1 queues at all service stations, i.e., we replace constant finishing times by exponential finishing times. Figure 5 shows the results. The response times are only marginally higher.

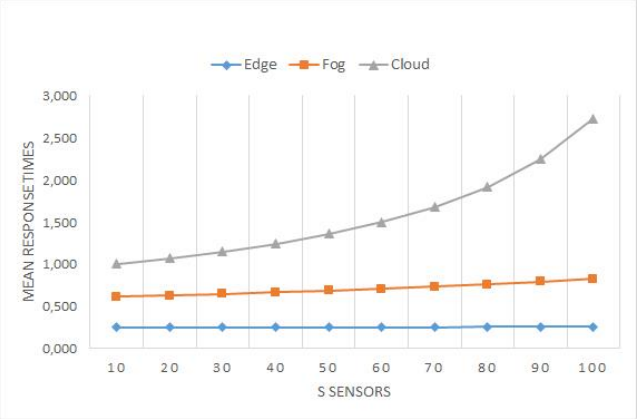

Figure 5. Example 2 – response times for M/M/1 queues all over, same input values as in Example 1.

Let us look at one more configuration which is substantially different from the previous two configurations (Table 3):

TABLE 3. EXAMPLE 3 – INPUT PARAMETERS.

s	e	f	c	p_e	p_f	λ_s
10..100	10	5	1	0.75	0.75	0.05

S_e	$S_{e\text{-not}}$	S_f	$S_{f\text{-not}}$	S_c
2.5	0.1	2.0	0.1	1.75

In this example the finishing probability on the edge and fog layer is rather high (i.e., 0.75) and the finishing service times are much bigger than in the previous examples. Finishing takes a constant service time, the forwarding service time is exponential.

From Figure 6 it is obvious that in this example the edge processors carry much more load and thus their response time is much higher, while the response time of fog and cloud processors is mainly determined by their mean service times.

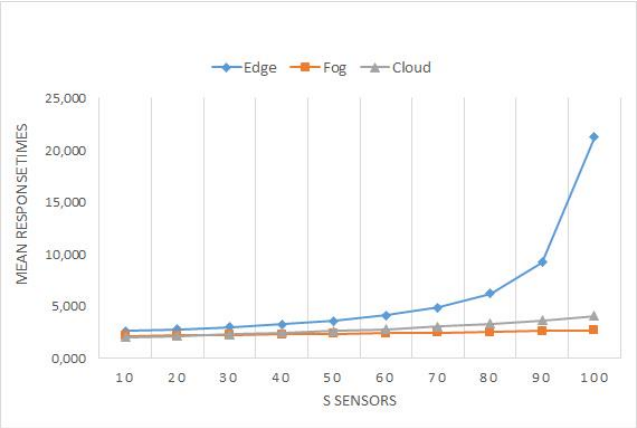

Figure 6. Example 3 – mixed M/M/1 and M/D/1 configuration.

B. *Validation by simulation*

In order to cross-validate the analytical performance model we used the Java Modelling Tools (JMT) simulation kit and performed a number of simulations for different configurations. JMT is a suite of applications developed by Politecnico di Milano and Imperial College London [4]. It offers a comprehensive set of tools for performance evaluation, system modeling with analytical and simulation techniques, capacity planning and workload characterization studies. The software and documentation are available at http://jmt.sourceforge.net/.

The simulation results show a high degree of agreement with the results of the analytical performance model and thus increase the confidence in the correctness of the analytical results. For example, Table 4 shows both sets of results, i.e., simulation and analytical, for the configuration in our Example 1 above. The confidence interval for the simulations is set to 99%.

TABLE 4. AVERAGE RESPONSE TIMES FROM SIMULATION AND ANALYTICAL MODEL (FOR PARAMETERS IN TABLE 2 AND S=100).

	R-edge	R-fog	R-cloud
Simulation	0.255	0.704	2.763
Analytical Model	0.254	0.702	2.732

The JMT simulation model for this specific configuration is shown in the Appendix.

We would like to emphasize here that in this paper the simulation model only serves as a means to validate the results of the analytical model. We did not intend to use the simulation experiments for the exploration of more complex configurations, which is left for future work.

IV. Related Work

Resource management techniques for edge / fog / cloud computing architectures have been a subject of intense research in recent years. A variety of techniques has been applied to evaluate and optimize the performance of configurations. The authors of references [5] – [12] use queueing theory models to study service performance in cloud-based IoT architectures.

In [5] the performance of a specific car parking scenario is analyzed where one cloud service is involved. The focus of [6] is on the optimization of energy consumption in edge devices. In [7] and [8] only one layer, i.e., the cloud layer, is analyzed. The authors of [9] view fog computing and edge computing as synonyms. Consequently, their queueing model, which in principle is quite similar to our model, has only two layers. Moreover, only exponential inter-arrival times and service times are considered throughout the whole system. The network architecture studied in reference [10] comes closest to the setup that we address in this paper. However, their performance model abstracts the fog and cloud servers as single queues, and requests may circle back to the fog server after they left the cloud server. Reference [11] considers user mobility and focusses specifically on the placement of fog processors in order to reduce the costs associated with their deployment and maintenance. In [12] a fine-grained performance model for the components of IoT middleware, such as orchestration, controlling, and request forwarding, is presented assuming a 3-layer IoT communication architecture. In summary, none of the discussed works are comparable to our study in terms of generality, flexibility, and efficiency of evaluating possible configurations.

V. Conclusions

We have presented an analytical performance model for a 3-layer communication architecture in IoT scenarios consisting of edge, fog, and cloud layers. The performance model can serve as an effective supporting method for designers of IoT applications in order to estimate the processor utilizations and response times of their application architectures for different traffic volumes as well as different numbers and processing speeds of processors at the three layers. Thus, appropriate configuration dimensions and trade-offs can be studied. Our analytical model is based on queueing theory and facilitates a quick and easy to use exploration of a large variety of configurations. Simulations demonstrate its accuracy.

There are two notable simplifications in our approach: The sensor output and all forwarding traffic are Poisson streams, and the traffic from a lower layer to the higher layer is load-balanced evenly to all processors on the receiving layer. These two assumptions make it possible to compute the performance results using straightforward closed-form solutions for M/G/1 queueing systems.

For the sake of clarity, we assumed in the examples above that all sensors and all processing stations on a layer are identical in terms of their performance parameters. This assumption can easily be relaxed such that sensors produce values at different rates and processors have different service times. Thus, you can also model an uneven distribution of the sensor data streams, e.g. if many devices are located in a certain region. Likewise, multi-cloud architectures can be modeled. Moreover, it is straightforward to eliminate a layer of the layered communication architecture completely by adjusting the finishing / forwarding probabilities and service times. This flexibility is one strength of our modeling approach.

The presented analytical performance model will be integrated into a comprehensive model-driven application development framework for edge / fog / cloud computing applications. The framework is work in progress.

Acknowledgment

Parts of this study were performed while author Kurt Geihs was a guest scientist at Universidad Carlos III de Madrid (UC3M). He gratefully acknowledges the support from the Banco Santander Chairs of Excellence program and the insightful discussions with researchers from UC3M and IMDEA Networks. The authors thank the anonymous reviewers for their constructive and inspiring comments.

References

[1] W. Yu, F. Liang, X. He, W. Hatcher, C. Lu, and J.L.X Yang. A survey on the edge computing for the Internet of Things. IEEE Access 2018, 6, pp. 6901–6919.

[2] C. Mouradian, D. Naboulsi, S. Yangui, R. H. Glitho, M. J. Morrow, and P. A. Polakos. A Comprehensive Survey on Fog Computing: State-of-the-Art and Research Challenges, IEEE Communications Surveys & Tutorials, Vol. 20, No. 1, First Quarter 2018

[3] H. Kobayashi. Modeling and Analysis: An Introduction to System Performance Evaluation Methodology. Addison-Wesley, 1978.

[4] M.Bertoli, G.Casale, G.Serazzi. JMT: performance engineering tools for system modeling. ACM SIGMETRICS Performance Evaluation Review, Vol. 36, No. 4, pp. 10-15, ACM Press, New York, NY, USA 2009.

[5] T. Bures, V. Matena, R. Mirandola, L. Pagliari, and C. Trubiani. Performance modelling of smart cyber-physical systems. In 2018 ACM/SPEC Intl. Conf. Performance Engineering, pages 37–40, ACM, 2018.

[6] P. G. Harrison and N. M. Patel. Optimizing energy-performance trade-offs in solar-powered edge devices. In Proc. 2018 ACM/SPEC Intl. Conf. Performance Engineering, New York, NY, USA. ACM.

[7] T. S. Sowjanya, D. Praveen, K. Satish, and A. Rahiman. The queueing theory in cloud computing to reduce the waiting time. Intl. J. Computer Science Engineering & Technology, 1(3), 2011.

[8] J. Vilaplana, F. Solsona, I. Teixid'o, J. Mateo, F. Abella and J. Rius. A queuing theory model for cloud computing. J. Supercomputing, 69(1):492–507, 2014.

[9] S. El Kafhali and K. Salah. Efficient and dynamic scaling of fog nodes for IoT devices. The Journal of Supercomputing, Vol. 73, pp. 5261–5284, 2017.

[10] U. Tadakamalla and D. Menascé. FogQN: An Analytic Model for Fog/Cloud Computing. 2018 IEEE/ACM International Conference on

Utility and Cloud Computing Companion (UCC Companion), Zurich, pp. 307-313, 2018

[11] R. A. da Silva and N. L. da Fonseca. On the Location of Fog Nodes in Fog-Cloud Infrastructures. Sensors 2019, 19, 2445, https://doi.org/10.3390/s19112445.

[12] D. Rathod and G. Chowdhary, Scalability of M/M/c Queue based Cloud-Fog Distributed Internet of Things Middleware, Int. J. Advanced Networking and Applications, Volume: 11, Issue: 01, pp. 4162-4170 2019.

APPENDIX: JMT SIMULATION MODEL

The simulation model for the configurations discussed in the examples in Section III is illustrated in the following Figure A.1. It shows the general structure of the 3-layered simulation model with finishing and forwarding stations at the edge and fog layers as well as finishing stations only at the cloud layer.

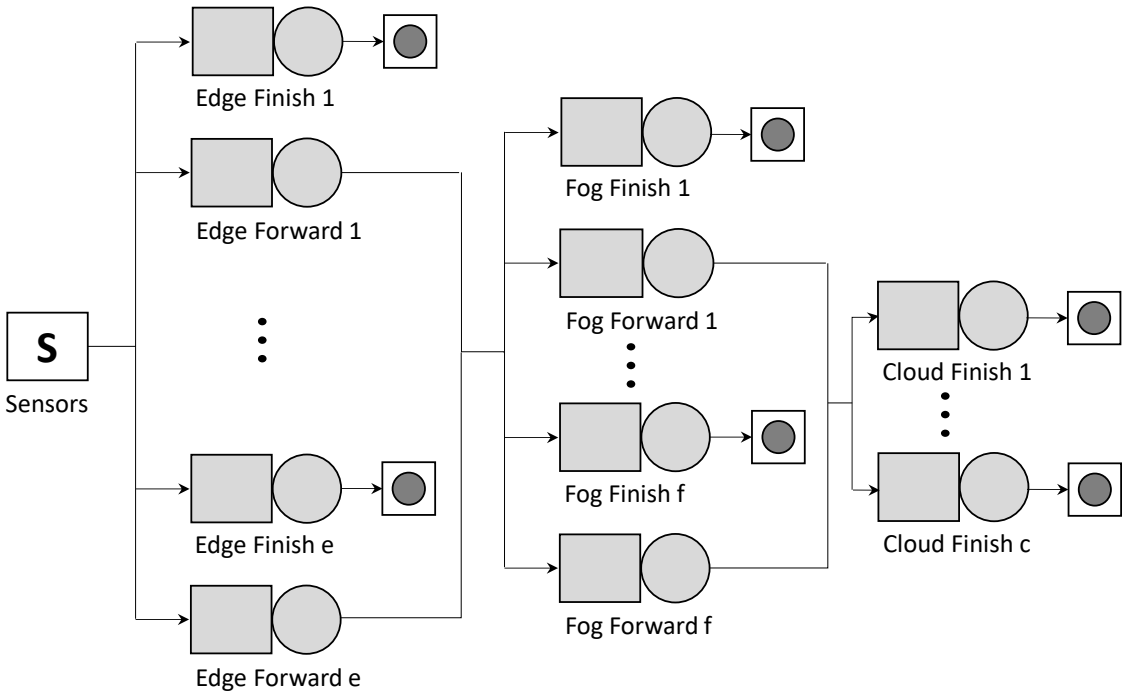

Figure A.1. General structure of the simulation model