# A Cloud-Based Bayesian Smart Agent Architecture for Internet-of-Things Applications

Veselin Pizurica(✉) and Piet Vandaele

Waylay, Gaston Crommenlaan 8, 9050 Ghent, Belgium
{veselin, piet}@waylay.io

**Abstract.** The Internet-of-Things (IoT) connects devices with embedded sensors to the Internet and is expected to grow at a spectacular rate. IoT will drive increased quality of life for individual as well as spur business growth and efficiency gains in industry. Actions based on real-time information, better decision making and remote diagnostics are some key areas where IoT can make a difference. This paper presents a smart agent architecture for the Internet-of-Things based on Bayesian Network technology that can be used for automation, notification, diagnostics and troubleshooting use cases.

**Keywords:** Bayesian networks · Internet-of-Things · Decision making · Artificial intelligence

## 1 Introduction

The Internet-of-Things provides us with lots of sensor data. However, the data by themselves do not provide value unless we can turn them into actionable, contextualized information. Big data and data visualization techniques allow us to gain new insights by batch-processing and off-line analysis of sensor data. Real-time sensor data analysis and decision-making is often done manually but to make it scalable, it is preferably automated. Artificial Intelligence provides us the framework and tools to go beyond trivial real-time decision and automation use cases for IoT.

In this paper, we present a cloud-based smart agent architecture for real-time decision taking in IoT applications. Section 2 reviews the concept of a rational agent. Section 3 describes the architecture of an agent suited for IoT applications. Section 4 explains why Bayesian technology is a good choice as agent logic for IoT applications. Section 5 explains how the agent can be embedded in an overall IoT solution. Finally, Sect. 6 summarizes and concludes the paper.

## 2 Rational Agent

The rational agent is a central concept in artificial intelligence [1]. An agent is something that perceives its environment through sensors and acts upon that environment via actuators. For example, a robot may rely on cameras as sensors and act on its environment via motors.


R. Giaffreda et al. (Eds.): IoT360 2014, Part I, LNICST 150, pp. 42–47, 2015.
DOI: 10.1007/978-3-319-19656-5_6

A rational agent is an agent that does 'the right thing'. The right thing obviously depends on the performance criterion set for the agent, but also on an agent's prior knowledge of the environment, the sequence of observations the agent has made in the past and the choice of actions that an agent can perform (Fig. 1).

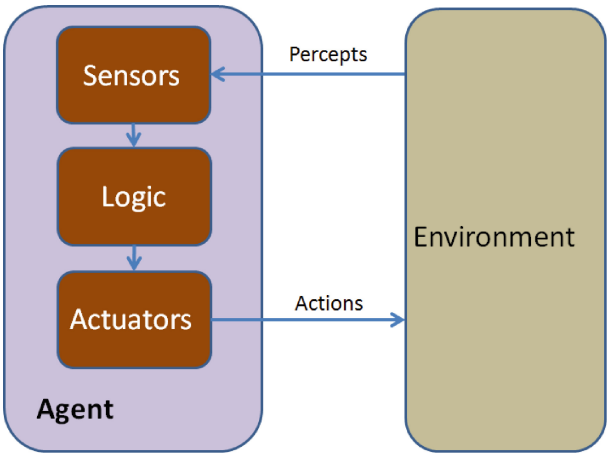


**Fig. 1.** Rational agent architecture [1].

An agent consists of an architecture and logic. The architecture of an agent typically consists of a computing device with physical sensors and actuators. The architecture allows ingesting sensor data, running logic on the data and acting upon the outcome. The logic itself is the heart of the agent and is the agent program that maps percepts of the environment to actions on that environment. It computes and reasons based on the available data and its knowledge of the environment.

## 3 Agent Architecture for IoT Applications

As described above, an agent is composed of an architecture and logic. Each poses specific challenges in the context of IoT. This section describes architectural considerations for the smart agent. Section 4 discusses the choice of logic.

### 3.1 Agent Location

In an IoT context, the agent can be located at different places:

- Close to the sensors and actuators, e.g. on a home gateway, a smartphone or on on-site server or private cloud infrastructure in an industrial environment.
- In a (public) cloud.
- Somewhere in between, such as switches and routers at the network edge. For the latter, Cisco coined the term fog computing [2].

In many respects, a cloud-based architecture is the preferred solution for a smart agent.

A cloud solution is device-agnostic and decouples logic from the presentation layer. So whether logic gets presented on an Android Smartphone, an iPhone, a tablet or as a web app (or any combination of the above), the cloud logic only needs to be developed once and can be used to target all of the above devices. By decoupling the user interface (view) from the logic (model and control), an architectural model arises where logic is built once together with a REST API. It can be changed and updated without necessarily impacting any end-user device software. This model reduces development time and increases business agility. Furthermore, the cloud brings many well-known advantages in terms of software maintenance, upgrades, roll-backs, etc.

Cloud logic scales well: cloud-capacity scales horizontally, while distributed HW often needs to be swapped when HW resources are no longer sufficient. In that sense, cloud logic is also more future-proof.

Cloud capacity is a shared resource and hence per-customer costs are substantially lower than for equivalent distributed HW capabilities. This can make a big difference in market segments that are CAPEX-sensitive. Solutions that rely primarily on cloud may be better suited towards a subscription model with smaller initial CAPEX investment for customers. Gateways are usually considered to be price-sensitive.

In case sensor information is combined with information from the Internet such as social media data, weather forecast, generic API information, or user profile information (which is by definition not bound to any particular device), the cloud is the natural place for these things to come together.

Similarly when integrating multiple IoT vertical solutions, e.g. the integration of an alarm system from vendor A with a home automation system of vendor B, that integration is likely to happen via APIs exposed in the back-end, rather than via direct integration between devices in the home. Using battery-powered devices to retrieve all this information from the cloud for local execution of the logic may drain battery resources.

Finally, cloud intelligence also allows easy generation of analytics regarding the usage of the logic itself. Which rules fired and why? How often?

However, in some application domains a cloud-based model is not optimal. In some cases, bandwidth is expensive or the amount of data is so large that connectivity and bandwidth are the bottleneck, i.e. data does not get easily into the cloud. You can think of applications relying on camera video feeds or certain industrial applications, e.g. Virgin Atlantic reported that Boeing 787 s will create half a terabyte of data per flight [3].

In those cases, local, on-premise processing or preprocessing, compression or buffering of data conserves bandwidth and associated costs. In a multi-tiered architecture, local preprocessing complements core logic that still resides in the cloud.

In other use cases, availability and latency are crucial aspects of a service, e.g. in process control in industry. For time-critical applications that require response times in the order of milliseconds, cloud-based logic can be replaced by fog or onsite logic.

In sum, there are compelling arguments for a cloud-based architecture but some verticals have boundary conditions that do not allow doing that.

### 3.2 Software-Defined Sensors and Actuators

Besides the location of the agent, the agent architecture also specifies how the agent interacts with sensors and actuators in its environment. In the context of IoT, applications can benefit from enriching sensor data with other types of information such as location, social media data, wearables, big data, open data, data from the API economy etc. The usage of additional sources of information leads to increased precision, personalization and contextualization of IoT applications, see e.g. [4].

Therefore, a smart agent architecture requires a generalized sensor and actuator concept that allows not only accepting raw sensor data but also other types of information. We call this concept a 'software-defined-sensor' and 'software-defined-actuators'.

## 4 Rational Agent Logic for IoT

The choice for a particular type of agent logic is influenced by the characteristics of the environment in which an agent needs to operate [1].

The simplest agent relies on 'if-then-else' constructs that are available in any programming language or rules engine. These agent implementations belong to the class of simple reflex agents. Their advantage is simplicity and familiarity; the main drawback is that they are only applicable if the action to be taken depends only on the current observation.

A second option is flowchart models such as used in business process modeling systems. These agents belong to the class of model-based reflex agents. They can maintain state of past observations, but can become very complex when the number of inputs and states rises. Moreover, when uncertainty is involved, flowchart based models may not come to satisfactory conclusions.

A third option is to use graph modeling technology, such as Bayesian networks, which is an implementation of goal- or utility-based agents. Below we illustrate why Bayesian networks match IoT environment characteristics.

Bayesian logic is the technology of choice when working in environments that cannot be completely observed, i.e. when not all aspects that could impact a choice of action are observable. In IoT, due to the complexity of the environment or due to cost constraints on the number of sensors, the environment may not be fully observable.

Bayesian logic is suited for applications with unreliable, noisy or incomplete data or when domain knowledge is incomplete such that probabilistic reasoning is required. In IoT, sensors may not be responding, out-of-service or quality of sensor data may be low or inaccurate. In all these cases, Bayesian technology is the best choice.

Bayesian logic is suited for use cases where the number of causes for a particular observation is so large, that it is nearly impossible to enumerate them explicitly. For example, when a door pressure sensor observes that a door is open, somebody may have forgotten to close the door, somebody may still be standing at the door while having a conversation, somebody may have briefly left open the door but will return soon, etc. Bayesian networks allow you to pick the most relevant options and model all others as 'other causes' via so-called leaky nodes.

Bayesian networks are well suited to model expert-knowledge together with knowledge that is retrieved from accumulated data [5]. In a data-driven approach, humans can specify the main attributes of Bayesian logic while the exact (conditional) probabilities can be learned in an automatic way from the available data.

From an implementation perspective, Bayesian networks allow more compact modelling in case the number of variables and states rises. The modeling of the logic and as-well-as the ongoing maintenance and updates to the logic will be simpler than for alternative technologies.

Finally, Bayesian networks are more amenable to use cases where there are asynchronous information flows, i.e. when not all information streams are synchronized, since the inference (i.e. the computation of posterior probability given observations on other variables) can be executed as new evidence comes in, for any node. As noted above, contextualized IoT applications will require the combination of many input sources, and therefore Bayesian networks may also be a good choice.

A couple of authors have suggested the usage of Bayesian Network technology as a valuable technology for IoT [6–10], however without implementation architecture as defined in this paper.

## 5 Embedding the Smart Agent in an IoT Solution

In many cases, the Bayesian logic will form part of a larger IoT application that has other components such as dedicated user interfaces, device management, big data analysis etc. Therefore, it is required that the Bayesian logic can be integrated in an overall solution. This can be done by exposing the architecture as a REST service, which means the agent, sensors and actuators can be controlled from the outside, and the intelligent agent can be integrated as part of a bigger solution. Figure 2 summarizes the cloud-based smart agent architecture proposed in this paper.

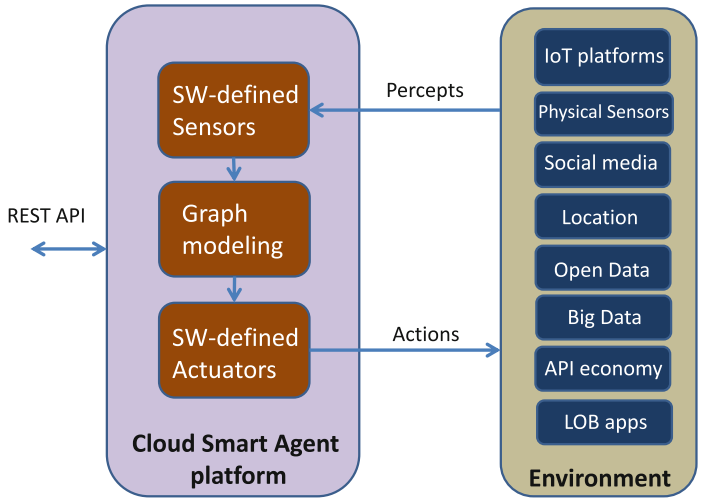


**Fig. 2.** A cloud-based Bayesian smart agent architecture for IoT applications.

## 6 Conclusion

This paper has presented a smart agent architecture for real-time decision making in IoT applications. It is based on a cloud architecture with flexible, generalized sensor and actuator capabilities via a software-defined-sensor and software-defined-actuator layer. At its core it uses Bayesian network technology. This paper has demonstrated that Bayesian technology matches challenges and environmental conditions of IoT. Its API-driven architecture makes it compatible with modern SaaS development practices and allows integration in broader IoT solutions.

## References

1. Russell, S., Norvig, P.: Artificial Intelligence: A Modern Approach, 3rd edn. Pearson, London (2014)
2. Cisco: Fog Computing, Ecosystem, Architecture and Applications. http://www.cisco.com/web/about/ac50/ac207/crc_new/university/RFP/rfp13078.html
3. Finnegan, M.: Boeing 787 s to create half a terabyte of data per flight, says Virgin Atlantic, ComputerWorldUK (2013). http://www.computerworlduk.com/news/infrastructure/3433595/boeing-787s-create-half-terabyte-of-data-per-flight-says-virgin-atlantic/
4. Scoble, R., Israel, S.: Age of Context: Mobile, Sensors, Data and the Future of Privacy. CreateSpace Independent Publishing Platform, Seattle (2013)
5. Koller, D., Friedman, N.: Probabilistic Graphical Models Principles and Techniques. Massachusetts Institute of Technology, Cambridge (2009)
6. Ko, K.-E., Sim, K.-B.: Development of context aware system based on Bayesian network driven context reasoning method and ontology context modeling. In: International Conference on Control, Automation and Systems, 2008, ICCAS 2008, pp. 2309–2313, October 2008
7. Park, H.-S., Oh, K., Cho, S.-B.: Bayesian network-based high-level context recognition for mobile context sharing in cyber-physical system. Int. J. Distrib. Sens. Netw. **2011**, 10 (2011)
8. Perera, C., Zaslavsky, A., Christen, P., Georgakopoulos, D.: Context aware computing for the internet of things: a survey. http://arxiv.org/pdf/1305.0982.pdf
9. Renninger, H., von Hasseln, H.: Object-Oriented Dynamic Bayesian Network-Templates for Modelling Mechatronic Systems. http://www.dbai.tuwien.ac.at/user/dx2002/proceedings/dx02final21.pdf
10. Carner, P.: Beyond home automation: designing more effective smart home systems. In: 9th IT & T Conference School of Computing, 1 October 2009