

Automatic Decomposition of IoT Aware Business Processes with Data and Control Flow Distribution

Francisco Martins^{1,2} ^a, Dulce Domingos¹ ^b and Daniel Vitoriano¹

¹*LASIGE, Faculdade de Ciências, Universidade de Lisboa, Portugal*

²*University of the Azores, Portugal*

Keywords: Internet of Things, BPMN, Process Decomposition.

Abstract: The Internet of Things (IoT) is generally seen as a distributed information gathering platform when used in business processes (BP). IoT devices have computational capabilities that can and should be used to execute fragments of BP that present benefits for both the devices and the BP execution engine. In fact, executing parts of the BP in the IoT devices may result in the reduction of the number of messages exchanged between the IoT network and the BP execution engine, increasing the battery lifespan of the IoT devices; also, it reduces the workload of the BP execution engine. However, processes are still defined following a centralised approach, making it difficult to use the full capabilities of these devices. In this paper, we present an automatic decomposition solution for IoT aware business processes, described using the Business Process Model and Notation (BPMN). We start from a BP model that follows a centralised approach and apply our decomposition method to transfer to the IoT devices the operations that can be performed there. This transformation preserves the control and the data flows of the original process and reduces the central processing and the number of messages exchanged in the network. The code that IoT devices execute is automatically generated from the BPMN process being decentralised.

1 INTRODUCTION

The Internet of Things (IoT) is one of the key dimensions of the new industrial revolution that is triggering a paradigm shift in business methodologies and in people's daily life. It is a paradigm that makes the things that surround us into active actors of the Internet, generating and consuming information.

Motivated by the rapid development of digital technologies, the IoT is characterised by the presence of things, equipped with RFID tags, sensors, actuators, and mobile devices that, through single address schemes, can interact with each other and cooperate with their neighbours to achieve a common goal. Interconnected things pose several challenges that have been addressed by state of the art literature, such as: energy consumption, the scarcest resource in the IoT; scalability, since the number of devices can be quite high; reliability, since the network can be used to report alarm events; and robustness, as the nodes of the network are susceptible to a wide variety of fail-

ures (Atzori et al., 2010; Moreno et al., 2014; Rault et al., 2014; Lee and Kim, 2010; Zorzi et al., 2010).

A business process model is a conceptual description of how businesses conduct their operations, by specifying activities, events, states, and flow control logic, among other factors. Process models are critical for the optimisation and automation of business processes and are often represented in a graphical notation, such as BPMN (OMG, 2011). These processes can use IoT devices to read and act based on their surrounding environment (Yousfi et al., 2016). They can even take advantage of the IoT to decentralise part of their execution flow when properly decomposed (Haller et al., 2008).

Business processes interact with IoT devices following a request-response or a publish-subscribe scheme to gather information and to trigger actuators. These interaction schemes promote the exchange of messages between IoT devices and the execution engine, which results in a high power consumption profile from the IoT device. Notice that communication is one of the most battery demanding tasks performed by the IoT devices. An alternative scheme would be to transfer part of the business process logic to the IoT

^a <https://orcid.org/0000-0002-2379-7257>

^b <https://orcid.org/0000-0002-5829-2742>

devices and let them make some of the business decisions and actuate on the environment without having to exchange information with the execution engine. This approach offers two benefits: it reduces the number of exchanged messages, increasing battery lifespan, and alleviates the execution engine by decentralising part of the execution of the process to the edges of the network (the IoT devices). Of course this extra execution will consume power of the IoT device, but the energy consumption of the migrated tasks is much smaller than that used on communication.

Decomposition breaks a system into progressively smaller subsystems that implement fragments of the domain problem. Decomposition is also a means to model large systems and to facilitate the reuse of partial models because it reduces coupling, helps in controlling specification complexity, and allows for multiple levels of detail to coexist. Consequently, models become easier to understand that, in turn, facilitates their validation, redesign, and optimisation (Caetano et al., 2010).

Despite the benefits of decentralisation, business processes are still defined following a centralised approach. Our solution automatically decomposes business processes, transforming the original process into one that takes advantage of the computational resources that IoT devices make available. We begin by generating a graph from a BPMN model that captures the control and data flow dependencies of the tasks. Then, we identify paths that only contain BPMN elements capable of being executed in IoT devices and that can be transferred. Then, based on these paths, we check which ones fall within the patterns that we have identified, and apply the corresponding transformation procedure. Finally, we redesign the BPMN model with the achieved solution.

The decomposition procedure is based on partition techniques, which group together activities in sub-processes and assign them to separate participants. These techniques were previously applied to generic workflow models with simplified approaches that omit data dependencies (Sadiq et al., 2006) or require designers to define the possible execution location of activities (Fdhila et al., 2009; Xue et al., 2018). We are using BPMN diagrams to determine data and control dependencies, as well as execution location of activities. This proposal is a step forward from our previous work (Domingos et al., 2014) as it reduces the number of steps of the algorithm and improves the way it identifies the tasks to transfer to IoT devices. Beyond reducing centralised execution of business processes, this solution also reduces the number of communications performed with IoT devices.

This paper is organised as follows: the next sec-

tion presents the related work; Section 3 presents our approach to process decomposition illustrated via a case study; Section 4 shows the developed prototype, and the last section concludes the paper and discusses future work.

2 RELATED WORK

The Mentor project presents one of the first proposals on process decomposition (Wodtke et al., 1996). To model processes, the authors use state diagrams and partition processes taking into account control and data flows.

The growing use of the Web Services Business Process Execution Language (WS-BPEL) (OASIS, 2007) justified the development of decomposition proposals of such processes. Nanda et al. (2004) create new subprocesses for each service of the original process, which communicate directly, reducing the synchronisation and message exchange that the central process execution engine has to perform. However, these authors do not consider the possibility of grouping services in the same subprocess.

Sadiq et al. (2006) and Fdhila et al. (2009) decompose generic process models by grouping various activities into subprocesses and by distributing the execution of these subprocesses among different execution engines. The former only considers control flow dependencies, while the latter also considers data flow dependencies. A subsequent work by Fdhila et al. (2014) optimise subprocesses composition taking into account several quality of service parameters, such as cost, time, reliability, and availability. Yu et al. (2016) use genetic programming to create partitions (subprocesses) of WS-BPEL processes that intensively use data.

To make use of the advantages offered by the cloud to execute fragments of business processes, Duipmans et al. (2012) divide business processes into two categories: those that run locally and those that can run in the cloud. With this division, the authors intend to perform the most computationally intensive tasks in the cloud, whose data is not confidential. The identification of these tasks is performed manually. Povoa et al. (2014) propose a semi-automatic mechanism to determine the location of activities and their data based on confidentiality policies, monetary costs, and performance metrics. Hoenisch et al. (2016) optimise the distribution of activities taking into account some additional parameters such as the cost associated with delays in the execution of activities and the unused, but paid, time of cloud resources.

In (Domingos et al., 2014) we present a prelimi-

nary approach to the decomposition problem of IoT aware BPMN business processes. The decomposition is based on dependency tables, as in (Sadiq et al., 2006; Fdhila et al., 2009) and takes into account control flow as well as data flow. Unlike related work, the partitions that IoT devices can execute are identified automatically, taking into account the capabilities of these devices.

The work we present in this paper goes a step forward by avoiding the generation of dependency tables and by using a more fine-grained selection algorithm for candidate partitions (or sub-paths). Our proposal is based on the graph we generate from BPMN process definitions, which captures control and data flow restrictions. Then, we identify sub-paths that only contain BPMN elements that IoT devices can execute and we transform them with the three patterns that we have identified, as detailed in the next section.

3 DECOMPOSITION OF BUSINESS PROCESSES

This section describes our business process decomposition method focused on reducing communications between IoT devices and the central system by moving fragments of the business process to IoT devices. We illustrate it using a simplified automatic irrigation business process.

3.1 Case Study

The irrigation system we describe automatically determines when to irrigate, based on the soil moisture. The water used for the irrigation comes from tanks, whose water level is also controlled by the system. It is possible to check the water level of the tanks and fill them whenever necessary.

The water level values are stored in a historical record file for future expenses audit. In addition, the system periodically contacts IoT devices to gather soil moisture levels. If the level is below a given threshold, a signal is sent to the IoT network that triggers the irrigation process. Soil moisture values are also kept in a historical record file.

Figure 1 illustrates the simplified BPMN model of our case study. It consists of two pools: the Irrigation Process and the IoT network (WSN). The irrigation process represents the central server responsible for executing the business processes. It contains two execution flows: the top describes tank refilling; the other specifies the irrigation process itself.

The top execution flow is triggered manually (S1 start event) and starts by requesting the tank's water

level (sendT1), sending a message to the IoT pool (SM1). The sensors read the tank's water level (T11) and forward this information back to the irrigation process (sendT12). The Receive Water Level task (receiveT2) blocks until a message arrives. Upon message arrival, it sends a message (sendT3) with the received information to the actuator. This information is transmitted to the Refill Water Tank task that sets the tank's water level to the top. Meanwhile, Save Refill Record task (T4) stores this occurrence by writing it to the Historical Record data store (H1).

The bottom execution flow executes periodically (ST1), starting by determining if the soil moisture levels provided by the IoT device are acceptable. For this, it requests this information (sendT5) to the IoT pool that starts the process (SM3), reads the soil moisture (T13), and sends a message with this information (sendT14) back to the irrigation process. Then, the process checks if it is necessary to start an irrigation cycle. For this, the exclusive gateway Check Moisture Values (IF1) forces the process to follow only one of its paths. If the moisture value is below the defined threshold, it computes the irrigation time based on the moisture level received and signals the actuators (sendT8) to start irrigating (T15). The purpose of the irrigation intermediate timer event (ST2) is to wait for the irrigation time before sending a signal (sendT9) to stop the actuator (T16). Finally, the process records the soil moisture level (T10) into the historical record data store (H2). The purpose of the converging gateway is to forward the process to task T10, regardless of the path taken by the process.

This process, despite using IoT devices, follows a centralised approach. In the following sections, we use it to illustrate the various steps of our decomposition procedure.

3.2 Decomposition Patterns

This section presents the patterns that we have identified as ineffective uses of the computational capabilities of IoT devices. The main concern on the identification of the patterns, and their respective transformations, is to minimise the number of communications between the central process and to preserve the execution flow of the initial business process. This means that the tasks still execute in the original order after the transformations.

Figure 2a presents an example of the first pattern taken from our running example. The pattern is identified by a receive task (receiveT2) followed by a send task (sendT3) in the central pool that are, respectively, preceded and followed by a send and a receive task or start message event belonging to the same IoT

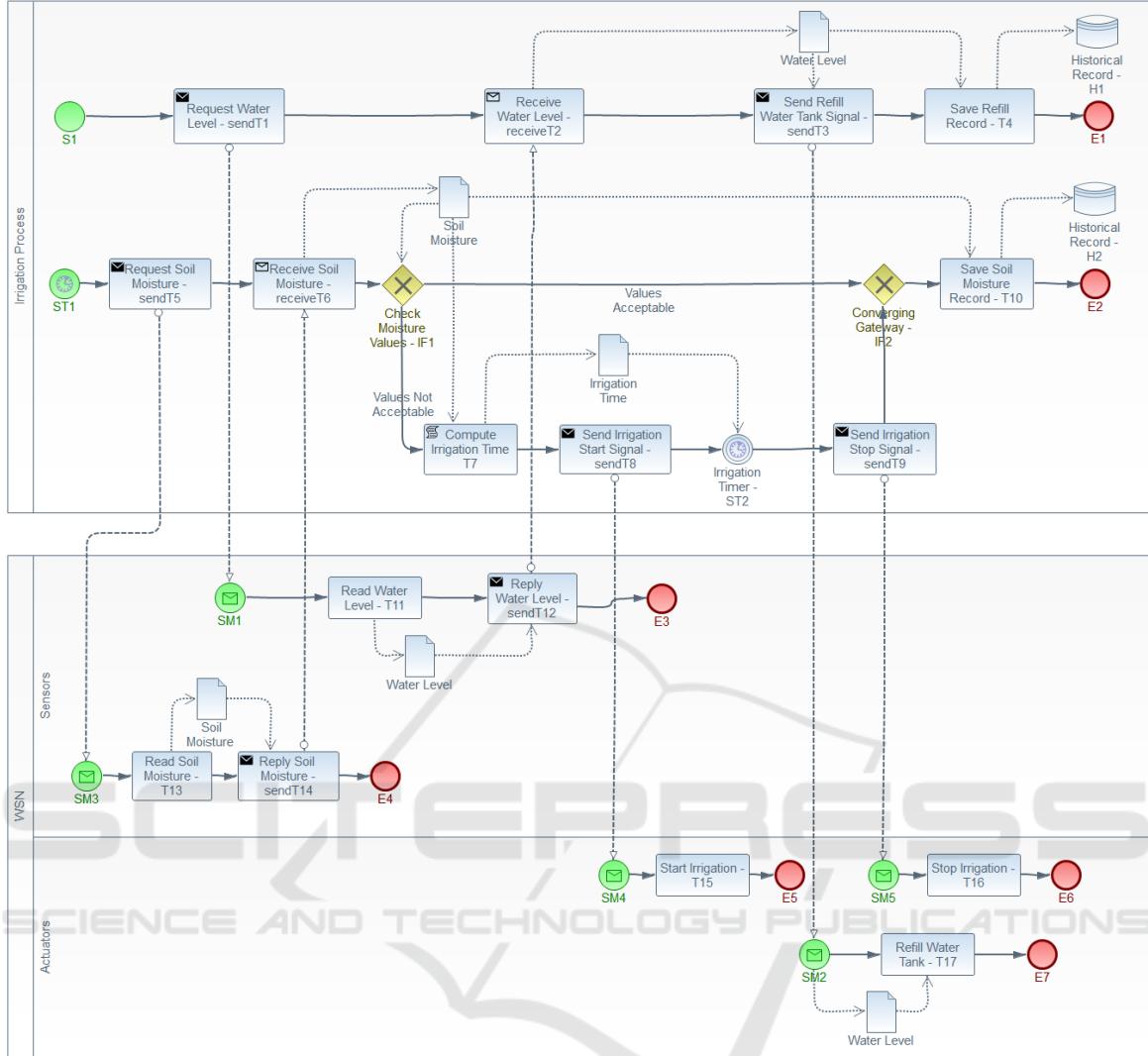


Figure 1: BPMN model of an automatic irrigation system - a case study.

pool. This includes an unnecessary communication between the two pools: the transformation eliminates SendT3 and SM2. To enforce the original control flow, receiveT2 is connected to T4 and sendT12 is connected to T17, as Figure 2b illustrates.

An example of the second pattern is shown in Figure 3a. The central pool has a process that starts with a timer event (ST1) and is followed by a send task (sendT5) and a receive task (receiveT6), both to/from the same IoT pool. By moving the timer to the IoT pool, we eliminate one communication between sendT5 task and SM3 (typically, IoT devices have timer operations), as Figure 3b shows.

Figure 4a contains an excerpt of our running example that illustrates the third pattern. Typically, IoT devices have sufficient computational capabilities to perform logical and mathematical operations, so it is

possible to transfer gateways to the IoT network, as long as the data for making the decision is available. Also script tasks that compute mathematical expressions can be moved to IoT devices as it is the case for task 7. This pattern is characterised by a message flow from the IoT pool to the central process (from sendT14 task to receiveT6 task), which afterwards branches (IF1 gateway) based on the received data. The goal is to transfer as much tasks as possible to the IoT pool, while preserving control and data flow dependencies. Figure 4b illustrates the application of this pattern on our running example. The message flow from the IoT pool to the central process is postponed as long as the central process has BPMN elements that can be moved to the IoT pool. This set of BPMN elements includes exclusive gateways, script tasks that only compute mathematical expres-

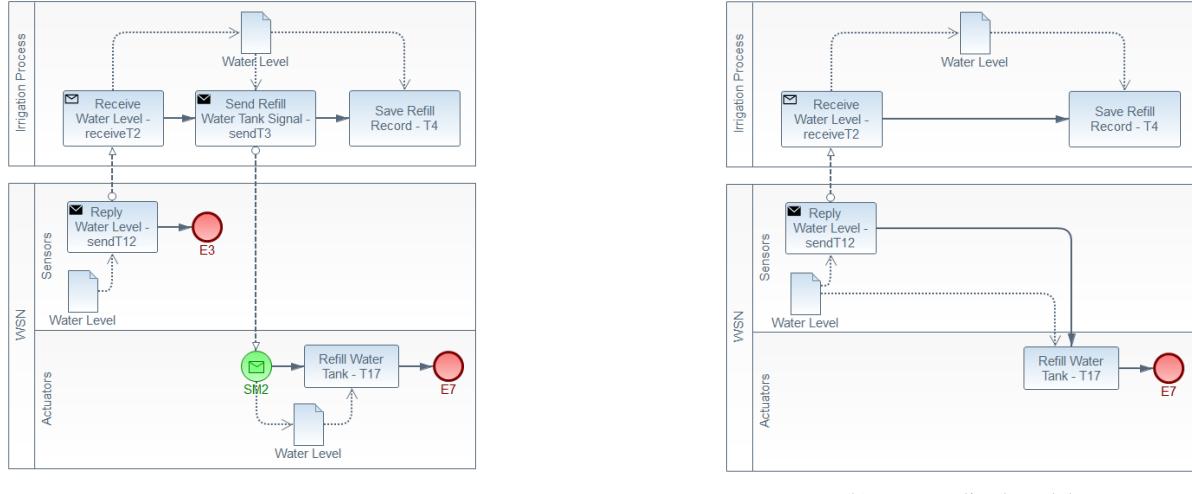


Figure 2: Pattern 1 example taken from the case study.

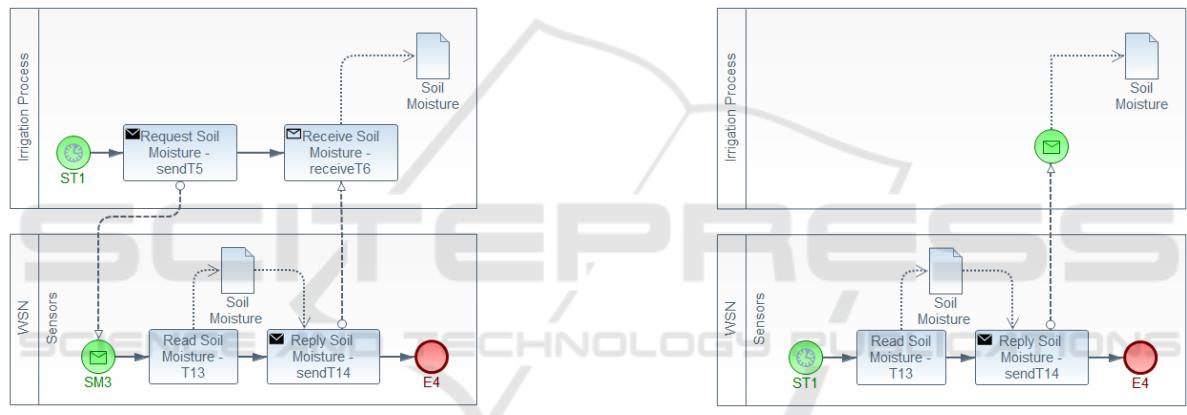


Figure 3: Pattern 2 example taken from the case study.

sions, timer events, and send tasks targeted at the IoT pool.

3.3 Decomposition Procedure

This section describes the procedure we propose to decompose IoT aware business processes by transferring fragments of the process to be executed by IoT devices. This procedure includes the following steps:

1. Generate a graph, from a BPMN business process model, that captures control and data flow dependencies of the BPMN elements. Figure 5 illustrates the graph that captures the control and data flow dependencies of the BPMN model of our running example. Each node contains the identification of the BPMN element its pool name.
2. Generate a list of sub-paths (that we call candidate sub-paths) that can be transferred to IoT devices.

For each initial node of the graph, the algorithm traverses it to list sub-paths that: (1) contain, at least, one message flow exchanged between the central pool and an IoT pool, (2) only include BPMN elements that are part of the IoT pool or BPMN elements (events, send tasks, receive tasks, gateways, script tasks computing mathematical expressions, data objects) that can be executed on IoT devices. At the end, paths that are sub-paths of other paths are removed.

In our running example, we obtain the following candidate sub-paths:

- [T1, SM1, T11, WL, T12, T2, WL, T3, SM2, WL, T17, E7],
- [ST1, T5, SM3, T13, SM, T14, T6, SM, IF1, T7, IT, ST2, T9, SM5, T16, E6],
- [ST1, T5, SM3, T13, SM, T14, T6, SM, IF1, T7, T8, ST2, T9, SM5, T16, E6],

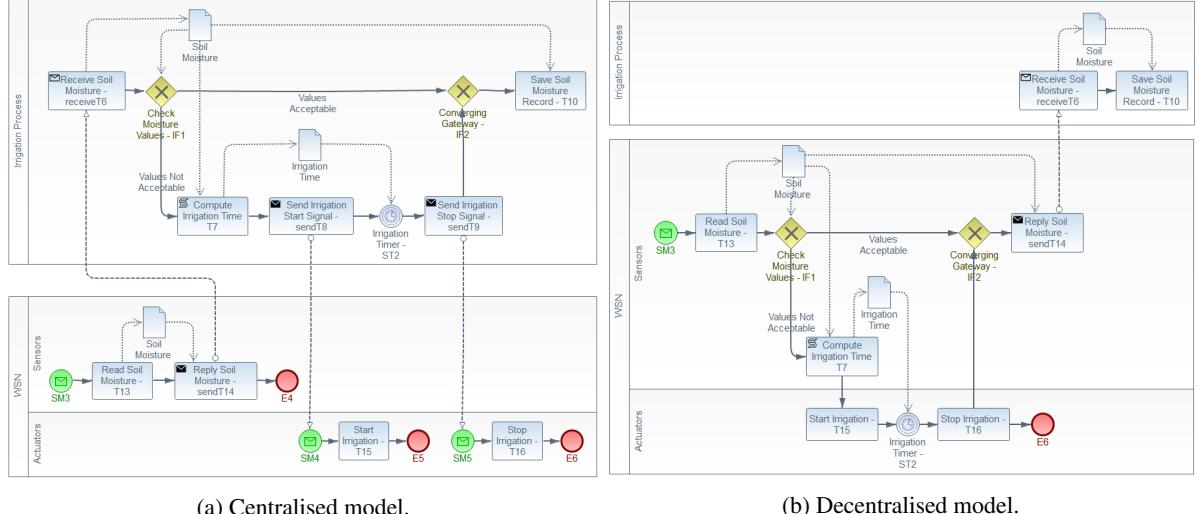


Figure 4: Pattern 3 example taken from the case study.

[ST1, T5, SM3, T13, SM, T14, T6, SM, IF1, T7, T8, SM4, T15, E5].

For instance, path [T1, SM1, T11, T12, T2, T3, SM2, WL, T17, E7] is removed because it is a sub-path of [T1, SM1, T11, WL, T12, T2, WL, T3, SM2, WL, T17, E7].

3. Redefine pools based on the candidate sub-paths. Process pools are redefined when transferring activities to the IoT pool or by eliminating communications. The redefinition occurs based on the patterns that we previously described.
4. Create a new BPMN model.

We use Graphviz to generate the coordinates of the elements of the modified BPMN model. Figure 6 illustrates the final solution of our use case, after several iterations of our decomposition procedure. We performed some manual layout adjustments in order to make the overall process more readable than that generated automatically by our tool.

It is possible to observe that, considering the seven message flows that the original model has between the irrigation process and the IoT pool, our solution shows that only three are in fact necessary if we take advantage of the capabilities that IoT devices offer.

4 PROTOTYPE

We conceived a prototype that performs the decomposition procedures described in the previous section, whose source code is available

at [github \(<https://github.com/fcmartins/bpmn-decomposition.git>\)](https://github.com/fcmartins/bpmn-decomposition.git).

Our development environment uses the Java programming language and the following tools:

- jBPM (version 6.3.0);
- Eclipse Luna (version 4.4.2) with BPMN2 Modeller and SonarLint plug-ins;
- Graphviz.

This prototype builds on top of previous tools we already developed that translate BPMN into CALLAS (a high-level sensor programming language (Lopes and Martins (2016))) and that automatically transform the BPMN model to communicate with the IoT network either using request-response or publish-subscribe architectures (Domingos and Martins (2017a,b)).

jBPM is a JavaEE application that implements the BPMN standard. This application is associated with the Luna version of Eclipse. BPMN2 Modeller is a BPMN graphical visualisation plug-in and SonarLint is a plug-in for enforcing source code quality. Graphviz is a tool for drawing graphs specified in the DOT language.

In order to execute the prototype, we provide a BPMN file with the model to be decomposed.

One of the challenges of the implementation, besides the decomposition of the model, was to calculate the coordinates of the BPMN elements on the new model generated by the prototype. For this, we use Graphviz. We generate a DOT file for each pool of the BPMN new model and use Graphviz that, through its graphing algorithm, generates the coordinates that we use as reference to lay down the BPMN elements of decomposition. Then, we apply geometric trans-

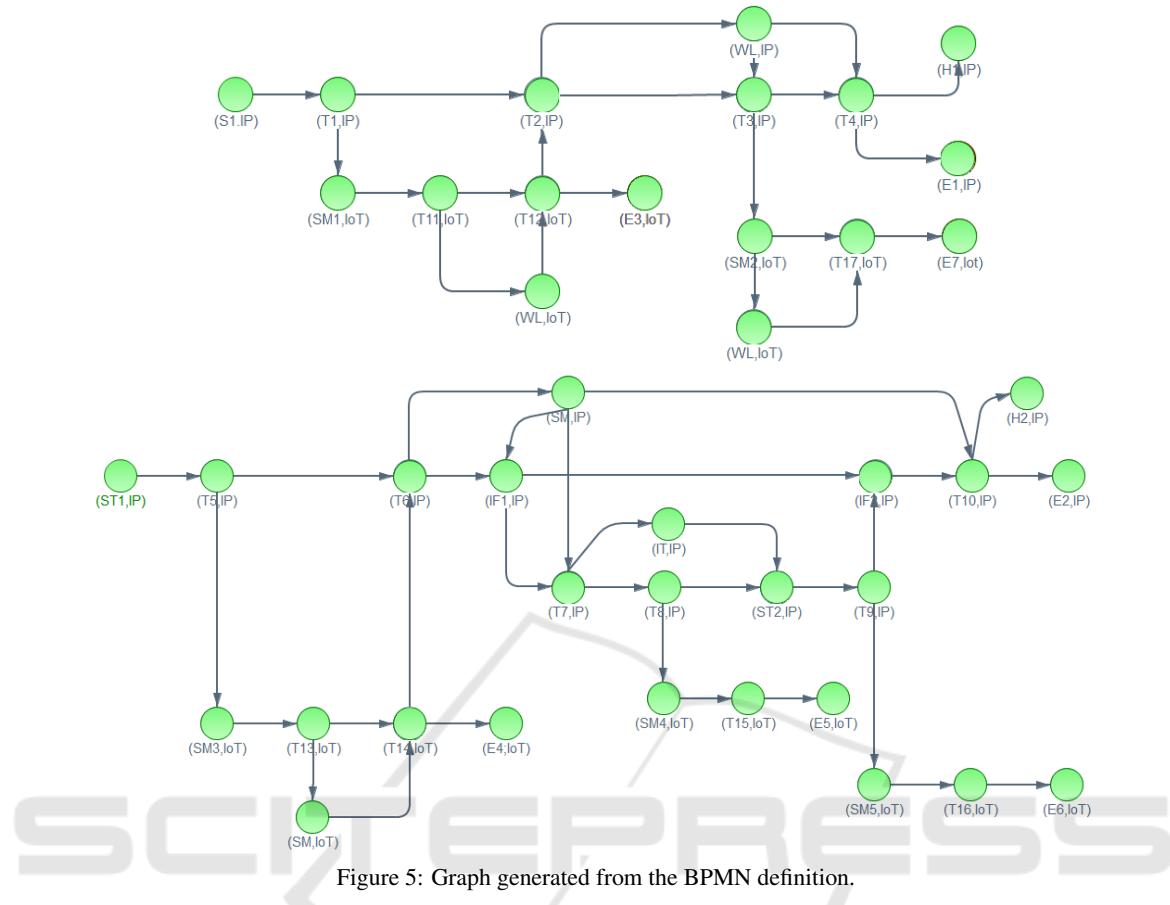


Figure 5: Graph generated from the BPMN definition.

formations to the coordinates generated so they are aligned with the other components of the model.

This prototype also translates the IoT behaviour into Callas bytecode to support the execution of all the BPMN model, as detailed in (Domingos and Martins (2017b)).

We evaluated the prototype using various models we built. In particular, the decomposition we present here results from the application of the prototype.

5 CONCLUSIONS AND FUTURE WORK

Business processes are increasingly using information made available by IoT devices to provide timely responses according to their context. In addition, IoT devices have enough computational power to execute parts of business processes, thus reducing central processing and the number of messages exchanged, and, consequently, increasing the energy autonomy of these devices.

Considering that business process modelling usually follows a centralised approach, the work pro-

posed in this paper allows for automatic decomposition of IoT-aware business processes. The decomposition takes into account constraints imposed by both control flow and data flows. This proposal reduces the number of messages exchanged between a central pool and IoT pools.

As for future work, we intend to generalise the patterns we propose in this paper, to conclude the work of checking whether script task can be executed by IoT devices, and to use additional information of the business processes history logs in the decomposition procedure. We also plan to extensively assess our approach, in particular, by measuring the amount of time, energy, and exchanged messages saved when using our algorithm.

ACKNOWLEDGEMENTS

We thank the anonymous reviewers for their comments and suggestions on an earlier version of this work. Some of their valuable comments will be definitely considered in future developments of our work. This work is partially supported by

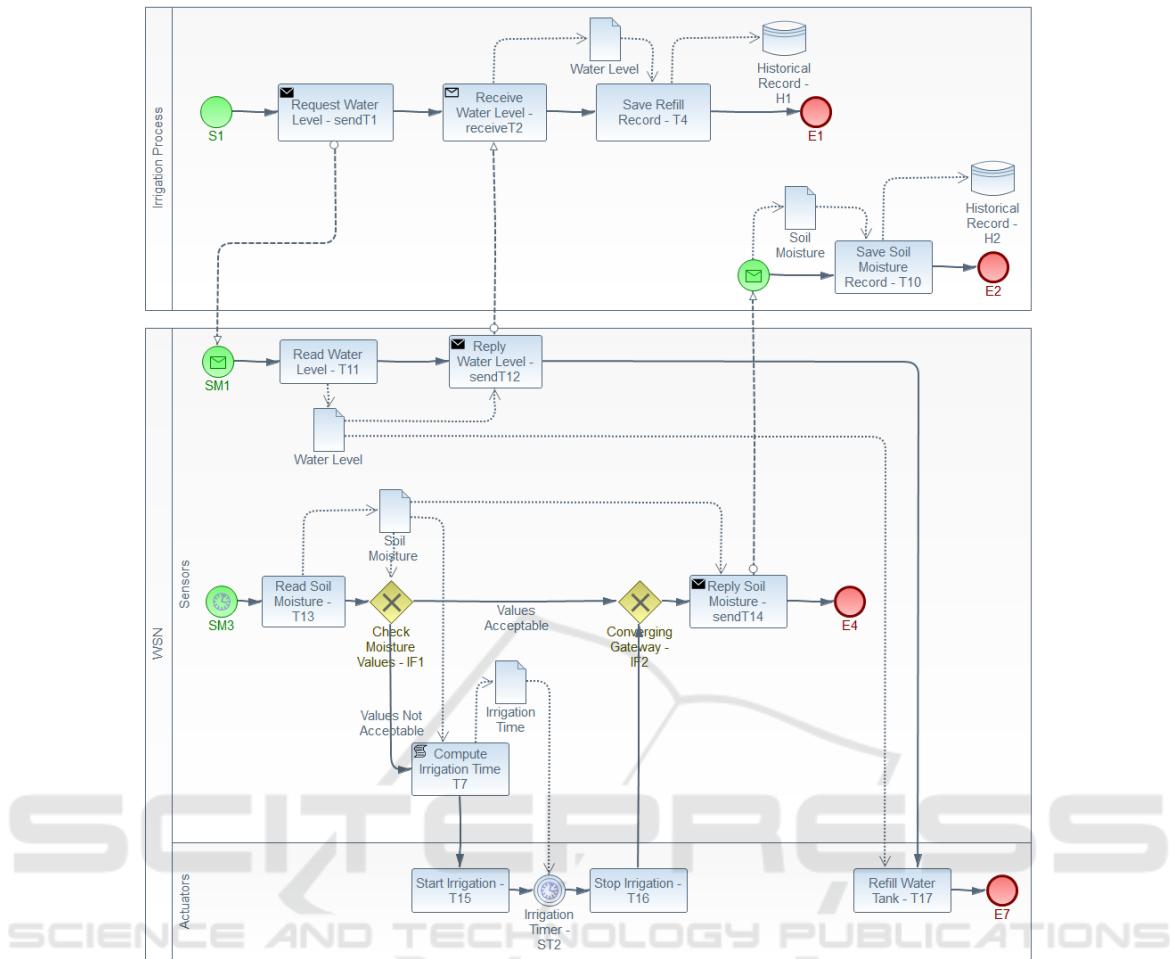


Figure 6: BPMN model of the automatic irrigation system after decentralisation.

FCT funding through LASIGE Research Unit, ref. UID/CEC/00408/2018, and by project DoIT, ref. PTDC/EEIESS/5863/2014.

REFERENCES

- Atzori, L., Iera, A., and Morabito, G. (2010). The internet of things: A survey. *Computer networks*, 54(15):2787–2805.
- Caetano, A., Silva, A. R., and Tríbolet, J. (2010). Business process decomposition—an approach based on the principle of separation of concerns. *Enterprise Modelling and Information Systems Architectures*, 5(1):44–57.
- Domingos, D. and Martins, F. (2017a). Modelling iot behaviour within bpmn business processes. *Procedia computer science*, 121:1014–1022.
- Domingos, D. and Martins, F. (2017b). Using bpmn to model internet of things behavior within business process. *IJISPM-International Journal of Information Systems and Project Management*, 5(4):39–51.
- Domingos, D., Martins, F., and Caiola, L. (2014). Decentralising internet of things aware bpmn business processes. In *International Conference on Sensor Systems and Software*, pages 110–119. Springer.
- Duijmans, E. F., Pires, L. F., and da_Silva Santos, L. O. B. (2012). Towards a BPM cloud architecture with data and activity distribution. In *Proceedings of the 2012 IEEE 16th International Enterprise Distributed Object Computing Conference Workshops*, pages 165–171. IEEE.
- Fdhila, W., Dumas, M., Godart, C., and García-Bañuelos, L. (2014). Heuristics for composite web service decentralization. *Software & Systems Modeling*, 13(2):599–619.
- Fdhila, W., Yıldız, U., and Godart, C. (2009). A flexible approach for automatic process decentralization using dependency tables. In *Proceedings of the 2009 IEEE International Conference on Web Services, (ICWS)*, pages 847–855. IEEE.
- Haller, S., Karnouskos, S., and Schroth, C. (2008). The Internet of Things in an enterprise context. In *Future Internet Symposium*, pages 14–28. Springer.
- Hoensche, P., Schuller, D., Schulte, S., Hochreiner, C., and

- Dustdar, S. (2016). Optimization of complex elastic processes. *IEEE Transactions on Services Computing*, 9(5):700–713.
- Lee, G. M. and Kim, J. Y. (2010). The internet of things problem statement. In *Proceedings of the 2010 International Conference on Information and Communication Technology Convergence (ICTC)*, pages 517–518. IEEE.
- Lopes, L. and Martins, F. (2016). A safe-by-design programming language for wireless sensor networks. *Journal of Systems Architecture*, 63:16–32.
- Moreno, M., Úbeda, B., Skarmeta, A. F., and Zamora, M. A. (2014). How can we tackle energy efficiency in IoT based smart buildings? *Sensors*, 14(6):9582–9614.
- Nanda, M. G., Chandra, S., and Sarkar, V. (2004). Decentralizing execution of composite web services. In *ACM SIGPLAN Notices*, volume 39, pages 170–187. ACM.
- OASIS (2007). Web services business process execution language version 2.0. URL: <http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html>.
- OMG (2011). Business Process Model and Notation (BPMN), Version 2.0.
- Povoa, L. V., de Souza, W. L., Pires, L. F., and do Prado, A. F. (2014). An approach to the decomposition of business processes for execution in the cloud. In *Proceedings of the 2014 IEEE/ACS 11th International Conference on Computer Systems and Applications (AICCSA)*, pages 470–477. IEEE.
- Rault, T., Bouabdallah, A., and Challal, Y. (2014). Energy efficiency in wireless sensor networks: A top-down survey. *Computer Networks*, 67:104–122.
- Sadiq, W., Sadiq, S., and Schulz, K. (2006). Model driven distribution of collaborative business processes. In *Services Computing, 2006. SCC'06. IEEE International Conference on*, pages 281–284. IEEE.
- Wodtke, D., Weißenfels, J., Weikum, G., and Dittrich, A. K. (1996). The mentor project: Steps towards enterprise-wide workflow management. In *Proceedings of the Twelfth International Conference on Data Engineering*, pages 556–565. IEEE.
- Xue, G., Liu, J., Wu, L., and Yao, S. (2018). A graph based technique of process partitioning. *J. Web Eng.*, 17(1&2):121–140.
- Yousfi, A., de Freitas, A., Dey, A. K., and Saidi, R. (2016). The use of ubiquitous computing for business process improvement. *IEEE Transactions on Services Computing*, 9(4):621–632.
- Yu, Y., Ma, H., and Zhang, M. (2016). A genetic programming approach to distributed execution of data-intensive web service compositions. In *Proceedings of the Australasian Computer Science Week Multiconference*, page 29. ACM.
- Zorzi, M., Gluhak, A., Lange, S., and Bassi, A. (2010). From today's Intranet of Things to a future Internet of Things: a wireless and mobility-related view. *IEEE Wireless communications*, 17(6).