

<sub>1</sub> A Unified Modeling Framework to Abstract  
<sub>2</sub> Knowledge of Dynamically Adaptive Systems

<sub>3</sub> Ludovic Mouline

<sub>4</sub> April 9, 2019



---

# Abstract

---

**Vision:** As state-of-the-art techniques fail to model efficiently the evolution and the uncertainty existing in dynamically adaptive systems, the adaptation process makes suboptimal decisions. To tackle this challenge, modern modeling frameworks should efficiently encapsulate time and uncertainty as first-class concepts.

*Context* Smart grid approach introduces information and communication technologies into traditional power grid to cope with new challenges of electricity distribution. Among them, one challenge is the resiliency of the grid: how to automatically recover from any incident such as overload? These systems therefore need a deep understanding of the ongoing situation which enables reasoning tasks for healing operations. **Abstraction** is a key technique that provided an illuminating description of systems, their behaviors, and/or their environments alleviating their complexity. **Adaptation** is a cornerstone feature that enables reconfiguration at runtime for optimizing software to the current and/or future situation.

Abstraction technique is pushed to its paramountcy by the model-driven engineering (MDE) methodology. However, information concerning the grid, such as loads, is not always known with absolute confidence. Through the thesis, this lack of confidence about data is referred to as **data uncertainty**. They are approximated from the measured consumption and the grid topology. This topology is inferred from fuse states, which are set by technicians after their services on the grid. As humans are not error-free, the topology is therefore not known with absolute confidence. This data uncertainty is propagated to the load through the computation made. If it is neither present in the model nor not considered by the adaptation process, then the adaptation

1 process may make suboptimal reconfiguration decision.

2 The literature refers to systems which provide adaptation capabilities as dynamically  
3 adaptive systems (DAS). One challenge in the grid is the phase difference between the  
4 monitoring frequency and the time for actions to have measurable effects. Action with  
5 no immediate measurable effects are named **delayed action**. On the one hand, an  
6 incident should be detected in the next minutes. On the other hand, a reconfiguration  
7 action can take up to several hours. For example, when a tree falls on a cable and cuts  
8 it during a storm, the grid manager should be noticed in real time. The reconfiguration  
9 of the grid, to reconnect as many people as possible before replacing the cable, is done  
10 by technicians who need to use their cars to go on the reconfiguration places. In a fully  
11 autonomous adaptive system, the reasoning process should be considered the ongoing  
12 actions to avoid repeating decisions.

13 *Problematic* **Data uncertainty and delayed actions are not specific to smart**  
14 **grids.**

15 First, data are, almost by definition, uncertain and developers always work with  
16 estimates. Hardware sensors have by construction a precision that can vary accord-  
17 ing to the current environment in which they are deployed. A simple example is the  
18 temperature sensor that provides a temperature with precision to the nearest degree.  
19 Software sensors approximate also values from these physical sensors, which increases  
20 the uncertainty. For example, CPU usage is computed counting the cycle used by a  
21 program. As stated by Intel, this counter is not error-prone<sup>1</sup>.

22 Second, it always exists a delay between the moment where a suboptimal state is  
23 detected by the adaptation process and the moment where the effects of decisions taken  
24 are measured. This delayed is due to the time needed by a computer to process data  
25 and, eventually, to send orders or data through networks. For example, migrating a  
26 virtual machine from a server to another one can take several minutes.

27 **Through this thesis, I argue that this data uncertainty and this delay**  
28 **cannot be ignored for all dynamic adaptive systems.** To know if the data un-  
29 certainty should be considered, stakeholders should wonder **if this data uncertainty**

---

<sup>1</sup><https://software.intel.com/en-us/itc-user-and-reference-guide-cpu-cycle-counter>

1 **affects the result of their reasoning process, like adaptation.** Regarding delayed  
2 action, they should verify **if the frequency of the monitoring stage is lower than**  
3 **the time of action effects to be measurable.** These characteristics are common  
4 to smart grids, cloud infrastructure or cyber-physical systems in general.

5 *Challenge* These problematics come with different challenges concerning the represen-  
6 tation of the knowledge for DAS. The global challenge address by this thesis is: **how**  
7 **to represent the uncertain knowledge allowing to efficiently query it and to**  
8 **represent ongoing actions in order to improve adaptation processes?**

9 *Vision* **This thesis defends the need for a unified modeling framework which**  
10 **includes, despite all traditional elements, temporal and uncertainty as first-**  
11 **class concepts.** Therefore, a developer will be able to abstract information related to  
12 the adaptation process, the environment as well as the system itself.

13 Concerning the adaptation process, the framework should enable abstraction of the  
14 actions, their context, their impact, and the specification of this process (requirements  
15 and constraints). It should also enable the abstraction of the system environment and its  
16 behavior. Finally, the framework should represent the structure, behavior and specifi-  
17 cation of the system itself as well as the actuators and sensors. All these representations  
18 should integrate the data uncertainty existing.

19 *Contributions* Towards this vision, this document presents two approaches: a temporal  
20 context model and a language for uncertain data.

21 The temporal context model allows abstracting past, ongoing and future actions  
22 with their impacts and context. First, a developer can use this model to know what the  
23 ongoing actions, with their expect future impacts on the system, are. Second, she/he  
24 can navigate through past decisions to understand why they have been made when they  
25 have led to a sub-optimal state.

26 The language, named Ain'tea, integrates data uncertainty as a first-class concept. It  
27 allows developers to attach data with a probability distribution which represents their  
28 uncertainty. Plus, it mapped all arithmetic and boolean operators to uncertainty prop-  
29 agation operations. And so, developers will automatically propagate the uncertainty

1 of data without additional effort, compared to an algorithm which manipulates certain  
2 data.

3 *Validation* Each contribution has been evaluated separately. The language has been  
4 evaluated through two axes: its ability to detect errors at development time and its  
5 expressiveness. Ain'tea can detect errors in the combination of uncertain data earlier  
6 than state-of-the-art approaches. The language is also as expressive as current ap-  
7 proaches found in the literature. Moreover, we use this language to implement the load  
8 approximation of a smart grid furnished by an industrial partner, Creos S.A.<sup>2</sup>.

9 The context model has been evaluated through the performance axis. The disser-  
10 tation shows that it can be used to represent the Luxembourg smart grid. The model  
11 also provides an API which enables the execution of query for diagnosis purpose. In  
12 order to show the feasibility of the solution, it has also been applied to the use case  
13 provided by the industrial partner.

14 **Keywords:** dynamically adaptive systems, knowledge representation, model-driven  
15 engineering, uncertainty modeling, time modeling

---

<sup>2</sup>Creos S.A. is the power grid manager of Luxembourg. <https://www.creos-net.lu>

---

# 1 Table of Contents

---

2	<b>1 Introduction</b>	<b>1</b>
3	1.1 Use case: Luxembourg smart grid . . . . .	2
4	<b>2 TKM: a temporal knowledge model</b>	<b>3</b>
5	2.1 Introduction . . . . .	4
6	2.2 Knowledge formalization . . . . .	4
7	2.2.1 Formalization of the temporal axis . . . . .	5
8	2.2.2 Formalism . . . . .	6
9	2.2.3 Application on the use case . . . . .	8
10	<b>Bibliography</b>	<b>i</b>





1

---

## 2 Introduction

---

### 3 Contents

4	1.1 Use case: Luxembourg smart grid . . . . .	2
---	---	---

---

6

8

9     **Abstract:** *Model-driven engineering methodology and dynamically adaptive systems*  
10 *approach are combined to tackle new challenges brought by systems nowadays. After*  
11 *introducing these two software engineering techniques, I give one example of such sys-*  
12 *tems: the Luxembourg smart grid. I will also use this example to highlight two of the*  
13 *problematics: uncertainty of data and delays in actions. Among the different challenges*  
14 *which are implied by them, I present the global one addressed by the vision defended in*  
15 *this thesis: modeling of temporal and uncertain data. This global challenge can be ad-*  
16 *dressed by splitting up in several ones. I present two of them, which are directly tackled*  
17 *by two contributions presented in this thesis.*

## <sup>1</sup> **1.1 Use case: Luxembourg smart grid**

<sup>2</sup> Should contain: -

1

---

## TKM: a temporal knowledge model to represent actions, their contexts and their impacts

---

4

### Contents

5

6

---

**2.1 Introduction** . . . . . 4

7

**2.2 Knowledge formalization** . . . . . 4

---

8

10

**Abstract:** *The evolving complexity of adaptive systems impairs our ability to deliver anomaly-free solutions. Fixing these systems require a deep understanding on the reasons behind decisions which led to faulty or suboptimal system states. Developers thus need diagnosis support that trace system states to the previous circumstances –targeted requirements, input context– that had resulted in these decisions. However, the lack of efficient temporal representation limits the tracing ability of current approaches. To tackle this problem, we first propose a knowledge formalism to define the concept of a decision. Second, we describe a novel temporal data model to represent, store and query decisions as well as their relationship with the knowledge (context, requirements, and actions). We validate our approach through a use case based on the smart grid at Luxembourg. We also demonstrate its scalability both in terms of execution time and consumed memory.*

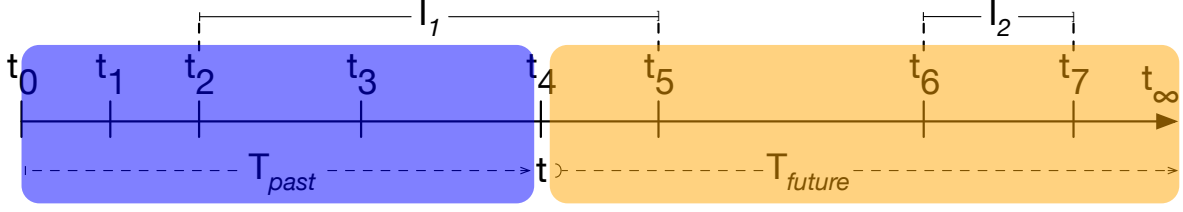


Figure 2.1: Time definition used for the knowledge formalism

## 2.1 Introduction

should define: decision, action, context, knowledge

## 2.2 Knowledge formalization

As discussed previously, I consider **knowledge** to be the association of **context** information, **requirements**, and **action** information, all in one global and unified model. While **context** information captures the state of the system environment and its surroundings, the system **requirements** define the constraints that the system should satisfy along the way. The **actions**, on the other hand, are means to reach the goals of the system.

In this section, I provide a formalization of the **knowledge** used by adaptation processes based on a temporal graph. Indeed, due to the complexity and interconnectivity of system entities, graph data representation seems to be an appropriate way to represent the **knowledge**. Augmented with a temporal dimension, temporal graphs are then able to symbolize the evolution of system entities and states over time. We benefit from the well-defined graph manipulation operations, namely temporal graph pattern matching and temporal graph relations to represent the traceability links between the **decisions** made and their **circumstances**.

Before describing this formalism, I describe the semantic used for the temporal axis. Then, I exemplify the knowledge formalism using the Luxembourg smart grid use case.

### 2.2.1 Formalization of the temporal axis

The formalism describe below has been made with two goals in mind. First, the definition of the time space should allow the distinction between past and future. Doing this distinction enable the differentiation between measured data and estimated (or predicted data). Second, it should permit the definition of the life cycle of an element of the **knowledge**, which can be seen as a succession of states with a validity period that should not overlap each other.

Time space  $T$  is considered as an ordered discrete set of time points non-uniformly distributed. As depicted in Figure 2.1, this set can be divided into 3 different subsets  $T = T_{past} \cup \{t\} \cup T_{future}$ , where:

- $T_{past}$  is the sub-domain  $\{t_0; t_1; \dots; t_{current-1}\}$  representing graph data history starting from  $t_0$ , the oldest point, until current time,  $t$ , excluded.
- $\{t\}$  is a singleton representing the current time point
- $T_{future}$  is sub-domain  $\{t_{current+1}; \dots; t_{\infty}\}$  representing future time points

The three domains depend completely on the current time  $\{t\}$  as these subsets slide as time passes. At any point in time, these domains never overlap:  $T_{past} \cap \{t\} = \emptyset$ ,  $T_{future} \cap \{t\} = \emptyset$ , and  $T_{past} \cap T_{future} = \emptyset$ . The definition of these three subsets reaches the first goal.

In addition, there is a right-opened time interval  $I \in T \times T$  as  $[t_s, t_e)$  where  $t_e - t_s > 0$ . In English words, it means that the interval cannot represent a single time point and should follow the time order. For any  $i \in I$ ,  $start(i)$  denotes its lower bound and  $end(i)$  its upper bound. As detailed in Section 2.2.2, these intervals are used to define the validity period for each node of the graph.

Figure 2.1 displays an example of a time space  $T_1 = \{t_0, t_1, t_2, t_3, t_4, t_5, t_6, t_7\}$ . Here, the current time is  $t = t_4$ . According to the definition of the past subset ( $T_{past}$ ) and the future one ( $T_{future}$ ), there is:  $T_{past1} = \{t_0, t_1, t_2, t_3\}$  and  $T_{future1} = \{t_5, t_6, t_7\}$ . Two intervals have been defined on  $T_1$ , namely  $I_1$  and  $I_2$ . The first one starts at  $t_2$  and ends at  $t_5$  and the last one is defined from  $t_6$  to  $t_7$ . As shown with  $I_1$ , an interval could be defined on different subsets, here it is on all of them ( $T_{past}$ ,  $t$ , and  $T_{future}$ ).

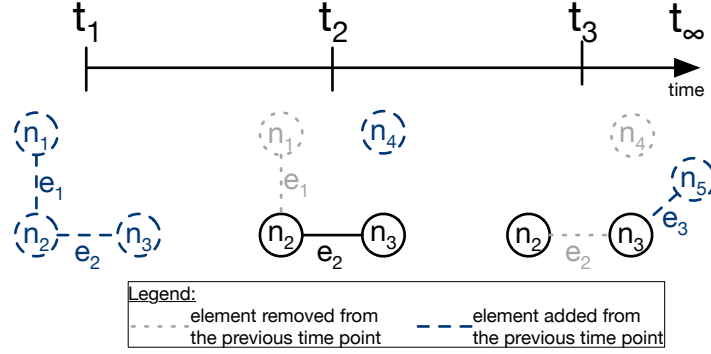


Figure 2.2: Evolution of a temporal graph over time

## 2.2.2 Formalism

**Graph definition** First, let  $K$  be an adaptive process over a system **knowledge** represented by a graph such as  $K = (N, E)$ , comprising a set of nodes  $N$  and a set of edges  $E$ . Nodes represent any element of the knowledge (context, actions, *etc.*) and edges represent their relationships. Nodes have a set of attribute values. An attribute value has a type (numerical, boolean,  $\dots$ ). Every relationship  $e \in E$  can be considered as a couple of nodes  $(n_s, n_t) \in N \times N$ , where  $n_s$  is the source node and  $n_t$  is the target node.

**Adding the temporal dimension** In order to augment the graph with a temporal dimension, the relation  $V^T$  is added. So now the knowledge  $K$  is defined as a temporal graph such as  $K = (N, E, V^T)$ .

A node is considered valid either until it is removed or until one of its attributes value changes. In the latter case, a new node with the updated value is created. Whilst, an edge is considered valid until either its source node and target node is valid, or until the edge itself is removed. Otherwise, nodes and edges are considered invalid. The temporal validity relation is defined as  $V^T : N \cup E \rightarrow I$ . It takes as a parameter a node or an edge ( $k \in N \cup E$ ) and returns a time interval ( $i \in I$ , *cf.* Section 2.2.1) during which the graph element is valid.

Figure 2.2 shows an example of a temporal graph  $K_1$  with five nodes ( $n_1, n_2, n_3, n_4$ , and  $n_5$ ) and three edges ( $e_1, e_2$ , and  $e_3$ ) over a lifecycle from  $t_1$  to  $t_3$ . In this

way,  $K_1$  equals to  $(\{n_1, n_2, n_3, n_4, n_5\}, \{e_1, e_2, e_3\}, V_1^T)$ . Let's assume that the graph is created at  $t_1$ . As  $n_1$  is modified at  $t_2$ , its validity period starts at  $t_1$  and ends at  $t_2$ :  $V_1^T(n_1) = [t_1, t_2)$ .  $n_2$  and  $n_3$  are not modified; their validity period thus starts at  $t_1$  and ends at  $t_\infty$ :  $V_1^T(n_2) = V_1^T(n_3) = [t_1, t_\infty)$ . Regarding the edges, the first one,  $e_1$ , is between  $n_1$  and  $n_2$  and the second one,  $e_2$  from  $n_2$  to  $n_3$ . Both are created at  $t_1$ . As  $n_1$  is being modified at  $t_2$ , its validity period goes from  $t_1$  to  $t_2$ :  $V_1^T(e_1) = [t_1, t_2)$ .  $e_2$  is deleted at  $t_3$ . Its validity period is thus equal to:  $V_1^T(e_2) = [t_1, t_3)$ .

**Lifecycle of a knowledge element** One node represents the state of exactly one knowledge element during a period named the validity period. The lifecycle of a knowledge element is thus modeled by a unique set of nodes. By definition, the validity periods of the different nodes cannot overlap. A same time period cannot be represented by two different nodes, which could create inconsistency in the temporal graph.

To keep track of this knowledge element history, the  $Z^T$  relation is added to the graph formalism:  $K = (N, E, V^T, Z^T)$ . It serves to trace the updates of a given knowledge element at any point in time. This relation can also be seen as a temporal identity function which takes as parameters a given node  $n \in N$  and a specific time point  $t \in T$ , and returns the corresponding node at that point. Formally,  $Z^T : N \times T \rightarrow N$ .

In order to consider this new relation in the example presented in Figure 2.2, the definition of  $K_1$  is modified to  $K_1 = (\{n_1, n_2, n_3, n_4, n_5\}, \{e_1, e_2, e_3\}, V_1^T, Z_1^T)$ . In Figure 2.2, let's imagine that  $n_1$ ,  $n_4$ , and  $n_5$  represent the same knowledge element  $k_e$ . The lifecycle of  $k_e$  is thus:

- $n_1$  for period  $[t_1, t_2)$ ,
- $n_4$  for period  $[t_2, t_3)$ ,
- $n_5$  for period  $[t_3, t_\infty)$ .

Let  $t'_1$  be a timepoint between  $t_1$  and  $t_2$ . When one wants to resolve the node representing the knowledge element at  $t'_1$ , she or he gets  $n_1$  node, no matter of the node input ( $n_1$ ,  $n_4$ , or  $n_5$ ):  $Z_1^T(n_4, t_1) = n_1$ . On the other hand, applying the same relation with another node ( $n_2$  or  $n_3$ ) returns another node. For example, if  $n_2$  and  $n_3$  do not belongs to the same knowledge element, then it will return the node given as input:  $Z_1^T(n_2, t_1) = n_2$ .

- <sup>1</sup> Graph nodes for context
- <sup>2</sup> Graph nodes for actions
- <sup>3</sup> Temporal queries for requirements
- <sup>4</sup> Temporal relations for decisions
- <sup>5</sup> **2.2.3 Application on the use case**



