Aix*Marseille université

UNIVERSITÉ AIX-MARSEILLE

# Reasoning about Big Data Flows: TOM4A Recursive Abstraction Based Problem Solving Method

THÈSE

POUR OBTENIR LE GRADE DE

DOCTEUR DE L'UNIVERSITÉ AIX-MARSEILLE

Faculté des Sciences

Discipline: Informatique

PRÉSENTÉE ET SOUTENUE PUBLIQUEMENT PAR

**Fabien VILAR**

Le 21 décembre 2018

Directeur de thèse: Marc LE GOC

École Doctorale en Mathématique et Informatique de Marseille
(ED 184)

Jury

| | | |
|---|---|---|
| M. Marc AIGUIER | Professeur, MICS, Centrale Supélec | Rapporteur |
| M. Laurent FRIBOURG | Directeur de Recherche CNRS, ENS Cachan | Rapporteur |
| M. Jean-Paul COMET | Professeur, Univ. Nice Sophia Antipolis, I3S | Examinateur |
| M. Philippe PREUX | Professeur, Univ. Lille, INRIA | Examinateur |
| M. Eugène ASARIN | Professeur, Univ. Paris 7, IRIF | Examinateur |
| M. Serge LAZZARINI | Professeur, Univ. Aix-Marseille, CPT | Examinateur |
| Mme. Nadia CREIGNOU | Professeur, Univ. Aix-Marseille, LIF | Examinateur |
| M. Marc LE GOC | Professeur, Univ. Aix-Marseille, LIS | Directeur |

Année 2018

*A ma mère Chantal et à mon père Pierre qui ont su me transmettre le goût pour la Science.*

*A Philippe pour tous ses bons conseils durant nos années à TOM4.*

*A Ericka pour son soutien sans faille durant cette période difficile de rédaction.*

*A Marc, pas le directeur de thèse mais l'ami, qui a su me pousser dans mes retranchements afin d'explorer des domaines scientifiques qui ne m'étaient pas familiers. Toutes ces années passées à ses côtés furent une sacrée aventure intellectuelle. Merci.*

*A Jérémy qui fut l'élément déclencheur de cette aventure un soir d'automne 2011 dans un célèbre pub de Salon-de-Provence.*

# Acknowledgements

Je souhaite remercier en premier lieu mon directeur de thèse, M. Marc LE GOC, Professeur des Universités à Aix-Marseille Université et responsable du projet in4Be (Induction for Behavior Modeling) au Laboratoire des Sciences de l'Information et des Systèmes (LSIS) pour m'avoir accueilli au sein de son équipe. Je lui suis également reconnaissant pour le temps conséquent qu'il m'a accordé, ses qualités pédagogiques et scientifiques. J'ai beaucoup appris à ses côtés et je lui adresse ma gratitude pour tout cela.

Je voudrais remercier les rapporteurs de cette thèse, M. Marc AIGUIER et M. Laurent FRIBOURG, pour avoir accepté la lourde tâche de rapporter sur mon mémoire et pour l'intérêt qu'ils ont porté à mon travail.

J'adresse également mes remerciements aux examinateurs, M. Jean-Paul COMET, M. Philippe PREUX, M. Eugène ASARIN, M. Serge LAZZARINI et Mme. Nadia CREIGNOU, pour avoir accepté d'examiner ce mémoire.

21 décembre 2018, Marseille, France.

# Contents

# List of Figures

# List of Tables

CHAPTER 1

# Introduction

## 1.1 General Problematic

This work finds its origin in a concrete problem submitted by a world wide French bank.

It is very easy to formulate the problem under the form of a simple question : how to detect and represent an internal fraud, a flow of bank transactions being given? To understand this simple question, you have to know that an *internal* fraud mostly occurs when the bank manager steals money from his clients. This kind of fraud concerns only rich clients who have an account in the "private bank" section of a bank.

The request of the bank is justified by the fact that internal fraud are *rarely detected*: 70% of internal frauds are identified because the fraud victim, a client of the bank, lodged a complaint and demanded to be reimbursed. Technically speaking, an internal fraud is made of many illegal transactions of *small amounts* during a relatively long period of time. As a consequence, most of the internal frauds are purely ignored. Actually, the annual cost of internal frauds is not negligible.

Besides, the difficulty to *prove* an internal fraud is intrinsic: in average, the evidence of an internal fraud requires 6 man.month for an expert in internal fraud.

From these elements, three important points entail:

1. The fraud being internal, all the banks deny the existence of such frauds. In other words, internal frauds don't exist. This explains why we may not name the bank in this document.

2. As a consequence, each time a client lodges an internal fraud complaint, the bank prime reaction is to immediately reimbursed its client. The analysis of the fraud is delayed later.

3. Despite the huge investments made by the banking industry in Information Technologies since over 50 years, there is no efficient system to detect an internal fraud and to represent it under a juridically form. In other words, detecting and modeling an internal fraud is a very difficult task for humans and machines.

Our works are only concerned with the technical problematic of internal fraud detection and modeling from a bank transactions given flow. The legal aspect of the fraud is the domain of human experts. Nevertheless, according to an expert of the French bank, this technical problematic is the most time consuming task. So, it is necessary to identify the concerned transactions before checking for their legality.

Such a problem could be approached through a real time monitoring problem coupled with a diagnosis problem. Three main facts make such an approach difficult if not impossible: (i) the

size of the bank transaction flow, the *Big Data* problem, (ii) the *real time* the diversity of the problems (real time diagnosis is still a difficult problem) (iii) and, last but not least, the *ingenuity* of the fraudsters. Indeed, the fraudsters creativity drives difficulties for building models of fraud schema that could facilitate the recognition task.

The aim of this work is then to propose a theoretical and technological solution to this technical problematic.

And let us be clear about the practical aspects of this work: a program able to detect and model an internal fraud in real time has been developed and works perfectly, at least on the data provided by the bank. This program is described in a Paper published in 2016 in the proceedings of $8^{th}$ International Conference on Agents and Artificial Intelligence (ICAART 2016, [VLGBR16]) and in chapter 8 of this document.

## 1.2 Contributions

Our contribution is then concerned with the theoretical mathematical framework development to provide a technology able to manage *Big Data Flow* problematics, combination of *Big Data* problematics and *real time* constraints inherent to algorithm execution.

*Big Data Flow* are characterized by (i) *temporal properties* and (ii) *high number of dimensions* of the informational space where it is defined. Usually, *continuous data flow* has been studied in the field of *dynamic systems* since the initial works of Henri Poincaré in mathematical fields. On a Computer Sciences point of view, such a problematic is currently referred to *real time software* for embedded systems, which represent less than 20% of the Information Technologies investments. *Real time software* manage *continuous data flow* through a general sampling device called the *Dirac's Sampler*. This device is widely used in practice to work with *Real time* data flow.

Our first main conceptual contribution is to defend the idea that such a sampler is not adequate to deal with *Big Data*. As a consequence, we propose a new kind of sampler, called a *Unary Observer*, which is the basic sampler device of Le Goc's *Timed Observations Theory* (TOT) [LeG06] that implements TOT *spatial discretization principle*.

Our second main conceptual contribution aims at defending the idea that working with *Big Data Flow* requires a *real time abstraction reasoning* process to produce, on line, two dual effects:

1. *Decreasing* the flow data amount, that is to say, coming back to a data flow *normal level* that a common computer can manage.

2. *Increasing* the semantic richness carried by the information flows, that is to say, resuming a lot of *semantically poor* data into an equivalent but *richer* one.

The price to pay for such an information *semantic* enrichment is the loss of *syntactic* data, that is say, to accept a *forgetting* process. This justifies the use of the Category Theory to formalize this process.

Thus, our contributions (i) provide a mathematical formalization of such a *real time abstraction reasoning* process and (ii) propose a methodology, called the TOM4A Methodology (Timed Observations Methodology for Abstraction), method guiding and controlling global abstraction process.

## 1.3 Framework of My Contribution

To manage *Big Data Flows*, it is necessary to combine together at least (i) a theory of *timed data* required to deal with continuous time data flows, (ii) a theory of *abstraction* and (iii) a theory for the *oversight phenomena*.

Developed in 2006, the *Timed Observation Theory* (TOT) of Le Goc [LeG06] has been designed to provide a common mathematical framework to analyse, model and control dynamic processes. To this aim, this theory combines and extends the Markov Chain Theory, the Poisson Process Theory, the Theory of Communication of Shannon [Sha84], and the Logical Theory of Diagnosis [Dag01], and provides operational tools (i.e. the adequate technology) to manage *Big Data Flows* in real time. The applications of this theory are numerous and various (cf. chapter 3), showing its generality and its robustness. At this stage of the document, the important point about the TOT is that it is the mathematical basis of a Knowledge Discovery from Data Base process (KDD process) called Tom4L (Timed Observation Mining for Learning) dedicated to mine flows of timed data, and a Knowledge Engineering (KE) method to model dynamic processes called TOM4D (Timed Observation Modeling for Diagnosis) dedicated to the combination of prior experts knowledge and posterior knowledge discovered from timed data.

Our contributions integrates the *Method of Abstraction* of Floridi [Flo08] (2008) in the mathematical framework of the TOT to build a mathematical basis required by a *real time abstraction reasoning* process. Floridi's theory *provides a detailed and controlled way of comparing analyses and models* [Flo08] with the introduction of multiple levels of abstraction in conceptual analysis. Floridi argued that *for discrete systems, whose observables take on only finitely-many values, the method is indispensable* [Flo08]. It constitutes then a crucial and powerful tool to address the analysis and the modeling of complex phenomenon as dynamic processes. Up to our knowledge, our formalization is the first complete mathematization of Floridi's theory.

Finally, we relied on the Category theory of Samuel Eilenberg and Saunders Mac Lane (1942-45) to provide a mathematical body to the notion of level of abstraction. To our main goal, the major apport of the Category theory is the notion of *functor* because it provides a natural and efficient way to control the *oversight phenomena* inherent to any abstraction reasoning. To this aim, we define the TOT($\mathbb{Z}$) Category to build *modeling* and *abstraction* functors that maps objects from two TOT($\mathbb{Z}$) Categories, of the same level of abstraction for the *modeling* functors and from two different levels of abstraction for *abstraction* functors.

It will be shown that the TOT concept of *timed observation* can be used as a *paradigm* in an abstraction process so that such a process can be automatized. The automatization of a *real time abstraction reasoning* process is indeed the main constraint of our works. This constraint strongly limits the choice of the theories that can be used to this aim.

Up to our knowledge, there is no conceptual or mathematical theories proposing to use another sampler device than Dirac's sampler. It is then necessary to develop a new mathematical framework to reason with the unique alternative, up to our knowledge, the TOT sampler device.

This is the main goal of our contributions: building an analysis, a modeling and a control approach of dynamic process is not an easy task. But doing that around a new kind of sampler is quite disturbing.

On our mind, the best way to manage the trouble of a very complex task is to rely on simple

examples. But one of the main difficulties with our problematic is to find examples as clear as possible but also sufficiently illustrative according to the subject under consideration. So, to facilitate the reading of this document and to illustrate the different aspects of our work, the following examples will be used to illustrate the concepts developed in this work:

1. An example of conversation (cf. http://www.socphilinfo.org/node/150) between three persons, coming from the community of philosophical researchers concerned with the Philosophy of Information. This example will be our running example in order to illustrate the concepts of the TOT in chapter 3. For this reason, our paper published in the proceedings of 9th International Conference on Agents and Artificial Intelligence (ICAART 2017, [LGV17]) has been joined in the appendix B.

2. A real world application of our contributions about the fraud detection in the banking transaction domain, application realized with a world wide French bank which, for obvious reasons, prefers staying anonymous. This example is entirely described in chapter 8 of this document.

## 1.4 Running Example

The example of the conversation is given here *in verbatim* from its source:

Suppose we join Alice, Bob, and Carol earlier on at the party. They are in the middle of a conversation. We do not know the subject of their conversation, but we are able to hear this much:

- Alice observes that its (whatever "it" is) old engine consumed too much, that it has a stable market value but that its spare parts are expensive;
- Bob observes that its engine is not the original one, that its body has been recently repainted but that all leather parts are very worn;
- Carol observes that it has an anti-theft device installed, is kept garaged when not in use, and has had only a single owner.

These three points constitute the database of our running example. The Reader is invited to go to the pages www.socphilinfo.org/node/150 to have details about the way such an exchange can be analyzed.

The only point that we ask the Reader to admit is that this exchange can be then summed up with the ten following observations:

1. Alice observes that its engine is old;
2. Alice observes that its engine consumed too much;
3. Alice observes that it has a stable market value;
4. Alice observes that its spare parts are expensive;
5. Bob observes that its engine is not the original one;

6. Bob observes that its body has been recently repainted;
7. Bob observes that all leather parts are very worn;
8. Carol observes that its anti-theft device is installed;
9. Carol observes that it is kept garaged when not in use;
10. Carol observes that it has had only a single owner.

## 1.5 Document Structure

This document is made of 9 chapters, a bibliography and 2 appendix:

- The first chapter is the present introduction.
- Chapter 2, the Preliminaries chapter, is dedicated to the introduction of Newell's notion of abstraction level, Floridi's theory of abstraction and the Category Theory.
- Chapter 3 is dedicated to the introduction of the TOT. In that way, the third chapter is an extension of the Preliminaries chapter. But considering that the Timed Observation's Theory (TOT) is not as recognized as Floridi's theory and the Category Theory, it seems to be necessary to devote a whole chapter to introduce its basis. This chapter aims at providing the Reader the concepts and the theorems that are required to analyse our contributions.
- Chapter 4 introduces the theoretical concepts of the TOT framework. This chapter is crucial because it provides solid basis in order to formalize, in the TOT framework:
  - concepts of algebraic structures and observable spaces,
  - process of abstraction,
  - existence of a new sampling device which differs from the *Dirac's Sampler*,
  - building of TOT categories.
- Chapter 5 formalizes concepts of algebraic structures, observable spaces and process of abstraction in the TOT framework.
- Chapter 6 is one of our first contribution. It demonstrates the existence of a morphism between Dirac's sampler and the TOT sampler devices. As consequence, this chapter demonstrates that the *Unary Observer* concept of the TOT plays the role of a sampler. A concrete specification of a complete and operational sampler is given in the Appendix A.
- Chapter 7 is our second contribution. This chapter provides the basis of *abstraction* and *reification* reasonings according to Category Theory. It is dedicated to the building of the TOT($\mathbb{Z}$) Category and the definition of the *Modeling* and *Abstraction* functors that allows the definitions of *Level of Abstraction* and *Gradient of Abstraction* according to Floridi's theory.

- Chapter 8 is our last contribution. This chapter describes the principles of a concrete *real time abstraction reasoning* process under the form of the *TOM4A* Methodology (Timed Observations Methodology for Abstraction). A concrete application of the *TOM4A* Methodology is made in order to discover and model potential internal frauds in a stream of banking transactions.

- Finally, the chapter 9 provides a conclusion about our work with a critical analysis of our contributions and the drawing of the main features of the potential prolongation of this thesis.

An important point to our contributions is the fact that they have been implemented in a Java software platform called Tom4K (Timed Observations Management for Knowledge). This explains why the appendix A has been joined to the document: it provides the specification of the concrete TOT sampler implemented in the Tom4K plateform.

The other appendix contains the paper about the running example of the conversation (Appendix B, [LGV17]).

## 1.6 Conclusion

Our works are concerned with the management of *Big Data Flows*, that is to say, a combination of the *Big Data* problematic and *real time* constraints about the execution of algorithms.

Our contributions are mainly mathematics. This explains why we insist on the fact that *all* the propositions formulated in this documents *have been implemented* in the Tom4K Java platform. This means that we apply the TOM4A Methodology to various *real world problems*, only one of them having been described in this document: the internal fraud problem.

There is multiple reasons for that. But the main and the strongest reason is the fact that it is the only *real world* example that has been the object of a publication because the solution described in chapter 8 requires the implementation of *three levels of abstraction*.

It is then a true chance that such an example can be introduced by the Author of this document.

CHAPTER 2

# PRELIMINARIES

## 2.1 Introduction: The Knowledge Level of Newell

In his paper [New81], Newell proposes to analyse the current description of computer systems with four levels of abstraction, summed in Table 2.1 :

- level 1, the *device level*: basic electronic components, transistors;
- level 2, the *circuit level*: collection of components;
- level 3, the *logic level* with its two sublevels: combinatorial or sequential circuits and the register transfert level;
- level 4, the *configuration level* (also called the PMS or Processor-Memory Switch level).

The important point for this document is that from this description, Newell identifies the concept of a *level of abstraction* as a collection of *medium*, object that is to be processed; *components*, transforming media in other media; *composition laws*, allowing components to be assembled into systems; and *behavior laws*, describing the way the system behavior depends on its components and structure behaviors.

| | **Medium** | **Components** | **Composition laws** | **Behavior laws** |
|---|---|---|---|---|
| **Level 5: Knowledge** | **Assertions** | **Knowledge, inferences, tasks** | **Logical consequence** | **Principle of rationality** |
| Level 4: Configuration | Symbolic expressions | Memory, operators | Naming and structuring memory zones | Sequential interpretation: Von Neumann and Turing machines |
| Level 3: Logic | Bits, bit vectors | Registers, functional units | Data transfer path (bus), electronic cards | Logical operations |
| Level 2: Circuit | Current, voltage | Transistors, resistance, capacity | Electronic chips, welding | Electricity laws |
| Level 1: Device | Electrons, electromagnetic waves | Junctions | Silicon sandwich | Electromagnetism laws |

Table 2.1: Description of the Knowledge Level

A level of description is then a specialization of the class of systems that can be described at the next lower level. Any instantiation of a level can be used to create an instantiation at the

next higher level. A hierarchy of the description is possible within a level, but it is in no way an additional level of abstraction.

The key points of Newell's notion of level of abstraction are the followings:

1. A level can be defined *autonomously.* The specification of a single-level system always completely determines a particular behavior for the system at this level of description (given the initial conditions and the boundary conditions). The overall system behavior results from the local effects of each of its components when used to transform system inputs into outputs. The immense variety of behavior arises from the structure of the system, i.e. the variety of ways of assembling a small number of component types. As a consequence, *the nature of one level differs radically from the others.*

2. Levels of description are *approximations.* The design is the operation of interpreting the specification against the capabilities of the selected technologies. Levels of description are achieved by technologies, and by construction, technologies are imperfect: they impose constraints that *limit the size and complexity of the systems* that can actually be built.

3. Each level can always be lowered. In other words, each of the aspects of a level (medium, components, laws of composition and behavior) can be defined in terms of the system described in the next lower level. But *errors at lower levels propagate at higher levels, producing incomprehensible behaviors within the higher level.*

From these key points, Newell formulated the *Knowledge Level Hypothesis*: *there exists a distinct computer systems level lying immediately above the symbol level which is characterized by knowledge as the medium and the principle of rationality as the law of behavior.* Newell calls this level the *Knowledge Level* where *Knowledge* is all that can be imputed to an *agent* so that its *behavior* can be evaluated in terms of the *principle of rationality*:

- Knowledge must be characterized *functionally*, in terms of what it does (the *role*), and not structurally, in terms of physical objects endowed with particular properties and relationships.

- Knowledge is intimately linked to the concept of rationality: agents for which it is possible to state a rationality, it is possible to say that they possess knowledge.

- Knowledge is a concept similar to the concept of competence, that is to say, a potentiality for the generation of actions, a potential for action.

- Because *representations of knowledge* exist at the *symbolic level*, the level of *Formal Systems*, data structures and algorithms provide a *body* for knowledge expressed at the *Knowledge Level.* Formal Logics are then only a particular way to represent knowledge.

A fundamental point in Newell's acception of knowledge is that *any description* at the *Knowledge Level* is an *approximation.* Although very practical (and very pragmatic), an approximation is always imperfect, not only in degree but in perimeter. In other words, *there is no guarantee that the entire behavior of a system has been described at the Knowledge Level.* But the fact is that, according to Newell, a *description at the Knowledge Level is definitively incomplete.*

We consider that the *incompleteness property* of any description at the *Knowledge Level* is a key point of any abstraction reasoning. This property is also required by Floridi's Method of Level of Abstraction for which, as it will be seen in the next section, Newell description of computer systems is a *nested Gradient of Abstraction*.

But where Newell uses an *ontological* point of view about the *Knowledge Level*, Floridi's uses an *epistemological* point of view to develop its *Method of Levels of Abstraction* in [Flo08, Flo10].

## 2.2 Floridi's Method of Level of Abstraction

According to Floridi, a *level of abstraction (LoA) is a finite but non-empty set of observables* [Flo08, p. 10].

The word *system* refers to the object of study, a process in science or engineering or a domain of discourse. The *behaviour of a system, at a given LoA, is defined to consist of a predicate whose free variables are observables at that LoA. The substitutions of values for observables that make the predicate true are called the system behaviours.* A *Level of Abstraction* is then a particular organization of *variable*, *observable*, *behavior* and *transition rules* between *values. A moderated LoA is defined to consist of a LoA together with a behaviour at that LoA*, [Flo08, p. 11].

The *Method of Levels of Abstraction* organizes LoA's in *Gradient of Abstraction* (GoA). A GoA allows to vary the LoA to make observations at different *granularity levels.* Two kinds of GoA are distinguished: *disjoint GoAs*, where the LoA are independent together, and *nested GoAs* where each LoA incrementally describes the same phenomena. The running example introduced in the chapter 1 is useful to expose the Method of Level of Abstraction.

Alice, Bob and Carol describe whatever "it" is according to their own points of view. They all talk about the same object but in very different ways: they focus on different features of this object. Alice, Bob and Carol have their own way to observe this object, to describe it: each of them have their own *Level of Abstraction (LoA).*

Luciano Floridi in [Flo08] have provided notions of the basic concepts useful to formalise the concept of LoA.

Let us focus on Alice's speech. She describes this object from the angle of its engine's *age*, its engine's *consumption*, its *market value* and its spare parts *cost.*

At this stage, Floridi's notions of a *system* and a *typed variable* can be given:

**Notion 1** System
*The **system** is the object of the study.*

Here, the *system* is the unknown object designed by "it" or "its" in the conversation between Alice, Bob and Carol.

**Notion 2** Typed variable
A **variable** is a uniquely named conceptual entity.
The **type** of a variable is set of all possible values of the entity.
A **typed variable** is the couple made of a variable and its type.

An *ill typed variable* is a *typed variable* with no well defined values. Floridi denotes a *typed variable* as *x:X* where $x$ is the variable and $X$ its type.

For instance, the type of the variable *age* is the set $AGE$ containing the values *old* and *not_ old*:

$$AGE = \{old, not\_old\} \tag{2.1}$$

The type of the variable *consumption* is the set $CONSUMPTION$ containing the values *too_ much* and *not_ too_ much*:

$$CONSUMPTION = \{too\_much, not\_too\_much\} \tag{2.2}$$

The type of the variable *market value* is the set $MARKET\_VALUE$ containing the values *stable* and *not_ stable*:

$$MARKET\_VALUE = \{stable, not\_stable\} \tag{2.3}$$

The type of the variable *cost* is the set $COST$ containing the values *expensive* and *not_ expensive*:

$$COST = \{expensive, not\_expensive\} \tag{2.4}$$

Doing so with Bod and Carol, the set of the *typed variables* contained in the conversation can be extracted and summed up in tables 2.2 and 2.3:

| **Variables** | **Types** |
|---|---|
| engine | ENGINE = $\{original, not\_original\}$ |
| body | BODY = $\{repainted, not\_repainted\}$ |
| leather parts | LEATHER_PARTS = $\{very\_worn, not\_very\_worn\}$ |

Table 2.2: Typed variables extracted from Bob's speech

| **Variables** | **Types** |
|---|---|
| anti-theft device | ANTI_THEFT_DEVICE = $\{installed, not\_installed\}$ |
| park status | PARK_STATUS = $\{garaged, not\_garaged\}$ |
| utilisation status | UTILISATION_STATUS = $\{in\_use, not\_in\_use\}$ |
| owner | OWNER = $\{single, not\_single\}$ |

Table 2.3: Typed variables extracted from Carol's speech

Notion of *typed variable* being introduced, let us now focus on Floridi's concept of *observable*. An *observable* is a feature of the *system* that the observer chooses to focus on while a *typed variable* is used to measure or to evaluate the state of that *observable*. For instance, the *typed variable market value* is used to evaluate the value of the market of the *system* "it" observed by Alice. If Alice had not been interested by this *system*'s specific feature, i.e. its market value, the *typed variable market value* would not have been considered as an *observable* of the *system*.

Floridi says that an *observable* is an *interpreted typed variable* and gives the following notion:

**Notion 3** Observable
The **observable** is a *typed variable* together with a statement of what feature of the system under consideration it represents.

An **observable** is said to be:

- **discrete** iff its **type** is a finite set;
- **analogue**, else.

This notion given, the *typed variables* previously introduced are all *discrete observables* of the *system*.

Alice, Bob and Carol have then their own sets of observables:

- Alice: {age, consumption, market value, cost};
- Bob: {engine, body, leather parts};
- Carol: {anti-theft device, park status, utilisation status, owner}.

Those sets highlight which particular features of the *system* are considered by Alice, Bob and Carol. As a consequence, Alice, Bob and Carol forget and ignore many other features of the *system*: they make an abstraction of these latest to describe the system according to their own point of view. Alice, Bob and Carol describe the *system* under the angle of their own *level of abstraction*:

**Notion 4** Level of abstraction
The **level of abstraction** is a finite but non empty set of **observables**.

A **level of abstraction** is said to be:

- **discrete** iff all **observables** are discrete;
- **analogue** iff all **observables** are analogue;
- **hybrid**, else.

Alice, Bob and Carol all use *discrete levels of abstraction* to describe the *system*.

Let us now focus on one particular Carol's observations:

- Carol observes that it is kept garaged when not in use.

The use of the word "when" creates a strong link between the values of the *observables park status* and *utilisation status*:

- If it is not in use then it is kept garaged.

This clearly forbids an assertion of the type:

- If it is in use then it is kept garaged.

This example shows that some combinaisons of the values of the *observables* at a given LoA are allowed and some others are not. Here, the combinaison:

- (anti-theft device, park status, utilisation status, owner) = (installed, garaged, not_in_use, single)

is allowed while combinaison:

- (anti-theft device, park status, utilisation status, owner) = (installed, garaged, in_use, single)

is forbidden.

Describing all the allowed combinations of values of the *system* at a given LoA leads to determine all the possible *behaviours* of such a *system*:

**Notion 5** Behaviour of a system
The **behaviour** of a system, at a given LoA, is defined to consist of a **predicate** whose free variables are observables at that LoA.
The substitutions of values for observables that make the **predicate** true are called the **system behaviours**.
A **moderated LoA** is defined to consist of a LoA together with a behaviour at that LoA.

For instance, let us denote $p^C$, the *predicate* associated with Carol's *system behaviour* whose free variable are anti-theft device, park status, utilisation status and owner:

$$p^C \equiv p^C(anti-theft\ device, park\ status, utilisation\ status, owner) \tag{2.5}$$

The table 2.4 gives the possible combinaisons generated by the substitution of these variables by their values.

| **anti-theft device** | **park status** | **usage status** | **owner** | $p^C$ |
|---|---|---|---|---|
| installed | garaged | in_use | single | False |
| installed | garaged | in_use | not_single | False |
| installed | garaged | not_in_use | single | True |
| installed | garaged | not_in_use | not_single | True |
| installed | not_garaged | in_use | single | True |
| installed | not_garaged | in_use | not_single | True |
| installed | not_garaged | not_in_use | single | False |
| installed | not_garaged | not_in_use | not_single | False |
| not_installed | garaged | in_use | single | False |
| not_installed | garaged | in_use | not_single | False |
| not_installed | garaged | not_in_use | single | True |
| not_installed | garaged | not_in_use | not_single | True |
| not_installed | not_garaged | in_use | single | True |
| not_installed | not_garaged | in_use | not_single | True |
| not_installed | not_garaged | not_in_use | single | False |
| not_installed | not_garaged | not_in_use | not_single | False |

Table 2.4: Carol's system behaviours

For example:

- the substitution of the variable *anti-theft device* by its value *installed* and
- the substitution of the variable *park status* by its value *garaged* and
- the substitution of the variable *usage status* by its value *not_in_use* and
- the substitution of the variable *owner* by its value *single*

make the *predicate* $p^C$ *true*.

Let us denote $p_1^C$ the instance of the *predicate* $p^C(installed, garaged, not\_in\_use, single)$:

$$\begin{array}{l} p_1^C \equiv p^C(installed, garaged, not\_in\_use, single) \\ p_1^C \equiv (anti-theft\ device = installed) \wedge (park\ status = garaged) \\ \wedge (usage\ status = not\_in\_use) \wedge (owner = single) \\ p_1^C = true \end{array} \tag{2.6}$$

Among those 16 combinaisons, 8 make Carol's *predicate* true, defining then 8 instances $p_i^C, i \in [1;8]$ of such a *predicate*:

- $p_1^C \equiv p^C(installed, garaged, not\_in\_use, single) = true$;
- $p_2^C \equiv p^C(installed, garaged, not\_in\_use, not\_single) = true$;
- $p_3^C \equiv p^C(installed, not\_garaged, in\_use, single) = true$;
- $p_4^C \equiv p^C(installed, not\_garaged, in\_use, not\_single) = true$;
- $p_5^C \equiv p^C(not\_installed, garaged, not\_in\_use, single) = true$;
- $p_6^C \equiv p^C(not\_installed, garaged, not\_in\_use, not\_single) = true$;
- $p_7^C \equiv p^C(not\_installed, not\_garaged, in\_use, single) = true$;
- $p_8^C \equiv p^C(not\_installed, not\_garaged, in\_use, not\_single) = true$.

This also defines a set $B^C = \{b_i^C, i \in [1;8]\}$ representing the 8 *behaviours* $b_i^C$ moderating Carol's *level of abstraction*:

- $b_1^C = (installed, garaged, not\_in\_use, single)$;
- $b_2^C = (installed, garaged, not\_in\_use, not\_single)$;
- $b_3^C = (installed, not\_garaged, in\_use, single)$;
- $b_4^C = (installed, not\_garaged, in\_use, not\_single)$;
- $b_5^C = (not\_installed, garaged, not\_in\_use, single)$;
- $b_6^C = (not\_installed, garaged, not\_in\_use, not\_single)$;
- $b_7^C = (not\_installed, not\_garaged, in\_use, single)$;
- $b_8^C = (not\_installed, not\_garaged, in\_use, not\_single)$.

The way the *system* moves from one of its *behaviour* to another must be described thanks to *transitions rules*. For example, moving from *behaviour* $b_1^C = (installed, garaged, not\_in\_use, single)$ to *behaviour* $b_2^C = (installed, garaged, not\_in\_use, not\_single)$ must be done by defining a *transitions rule* denoted $t_{12}^C$ such as:

$$\begin{array}{rccc} t_{12}^C & : & B^C & \rightarrow & B^C \\ & & b_1^C & \mapsto & b_2^C \end{array} \tag{2.7}$$

describing the way how the *observable owner* switches from value *single* to value *not_ single*.

Concrete illustrations of *system behaviours* and *transitions rules* can be found in [PLG14] and [LGFCT13].

At this stage of the reasoning, we can affirm that Alice, Bob and Carol have their own *moderated LoAs*, i.e. their own *level of abstraction* with their own *behaviours*, *predicates* and *transitions rules*, reflecting their own views of the *system*.

Nevertheless, a way describing interactions between LoAs is needed. Floridi's introduced the concept of *gradient of abstraction* providing *a way of varying the LoA in order to make observations at different levels of abstraction*. The observations at each LoA must be explicitly related to those at the others. To do so, relations between LoAs must be introduced.

Let us then consider a *level of abstraction* denoted $L^p, p \in \mathbb{N}$. By definition, $L^p$ is a finite (but non empty) set of $n^p \in \mathbb{N}$ observables denoted $x_i, i \in [1; n^p]$ whose type is denoted $X_i, i \in [1; n^p]$:

$$L^p = \{x_i : X_i, i \in [1; n^p]\} \tag{2.8}$$

This *level of abstraction* is *moderated* by a set $B^p$ of $n_{B^p} \in \mathbb{N}$ *behaviours* denoted $b_i^p$:

$$B^p = \{b_i^p, i \in [1; n_{B^p}]\} \tag{2.9}$$

By construction, any *behaviour* $b_i^p$ makes the *predicate*, denoted $p^p$, at this *level of abstraction* true:

$$\forall i \in [1; n_{B^p}], p^p(b_i^p) = true \tag{2.10}$$

In a similar way, let us then consider a *level of abstraction* denoted $L^q, q \in \mathbb{N}, q \neq p$. $L^q$ is also a finite (but non empty) set of $n^q \in \mathbb{N}$ observables denoted $y_j, j \in [1; n^q]$ whose type is denoted $Y_j, j \in [1; n^q]$:

$$L^q = \{y_j : Y_j, j \in [1; n^q]\} \tag{2.11}$$

This *level of abstraction* is *moderated* by a set $B^q$ of $n_{B^q} \in \mathbb{N}$ *behaviours* denoted $b_j^q$:

$$B^q = \{b_j^q, j \in [1; n_{B^q}]\} \tag{2.12}$$

By construction, any *behaviour* $b_j^q$ makes the *predicate*, denoted $p^q$, at this *level of abstraction* true:

$$\forall j \in [1; n_{B^q}], p^q(b_j^q) = true \tag{2.13}$$

Figure 2.1 gives an illustration of two *moderated LoAs*, $L^p$, made of three *observables* $x_1 : X_1$, $x_2 : X_2$, $x_3 : X_3$ and one *predicate* $p^p$, and $L^q$, made of two *observables* $y_1 : Y_1$, $y_2 : Y_2$ and one *predicate* $p^q$.

Floridi introduces a family of relations needed to link elements associated with both LoAs:

1. a relation, denoted $R_{p,q}$ defined over $L^p \times L^q$, linking the *variables* of each LoA:

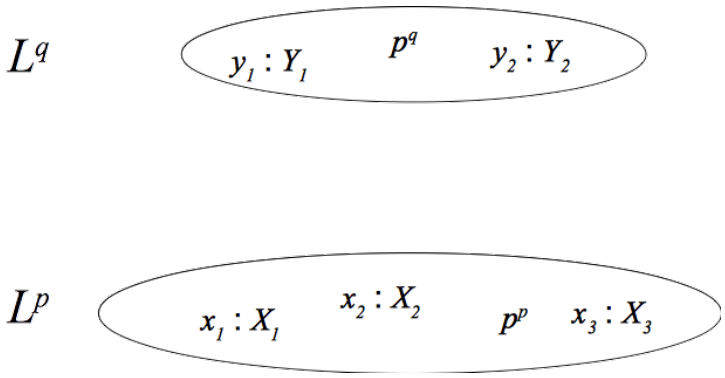


Figure 2.1: Moderated levels of abstraction $L^p$ and $L^q$

$$\forall x_i \in L^p, \exists y_j \in L^q, \quad \begin{array}{rccc} R_{p,q} & : & L^p & \rightarrow & L^q \\ & & x_i & \mapsto & y_j \end{array} \tag{2.14}$$

2. such a relation $R_{p,q}$ from $L^p$ to $L^q$ translates any *predicate* $p^p$ on $L^p$ to a *predicate* denoted $p_{R_{p,q}}$ on $L^q$:

$$\forall x_i \in L^p, \exists y_j \in L^q, \quad \begin{array}{rccc} R_{p,q} & : & L^p & \rightarrow & L^q \\ & & p^p & \mapsto & p_{R_{p,q}} \end{array} \tag{2.15}$$

3. relations, denoted $R_{X_i,Y_j}$ defined over $X_i \times Y_j$, linking the *types* of variables of each LoA. The *type* $X_i$ is made of $n_{X_i} \in \mathbb{N}$ possible values denoted $\alpha_k$ that the *variable* $x_i$ can take:

$$X_i = \{\alpha_k, k \in [1; n_{X_i}]\} \tag{2.16}$$

The *type* $Y_j$ is made of $n_{Y_j} \in \mathbb{N}$ possible values denoted $\beta_l$ that the *variable* $y_j$ can take:

$$Y_j = \{\beta_l, l \in [1; n_{Y_j}]\} \tag{2.17}$$

The relation $R_{X_i,Y_j}$ maps then any value $\alpha_k$ of $X_i$ with a value $\beta_l$ of $Y_j$:

$$\forall \alpha_k \in X_i, \exists \beta_l \in Y_j, \quad \begin{array}{rccc} R_{X_i,Y_j} & : & X_i & \rightarrow & Y_j \\ & & \alpha_k & \mapsto & \beta_l \end{array} \tag{2.18}$$

Figure 2.2 illustrates relations linking of *moderated LoAs* $L^p$ and $L^q$:

- a relation $R_{p,q}$ linking the *variables* and translating the predicate $p^p$ such as:

$$\begin{array}{rccc} R_{p,q} & : & L^p & \rightarrow & L^q \\ & & x_1 & \mapsto & y_1 \\ & & x_2 & \mapsto & y_1 \\ & & x_3 & \mapsto & y_2 \\ & & p^p & \mapsto & p_{R_{p,q}} \end{array} \tag{2.19}$$

- relations denoted:

$$\begin{array}{rcccl} R_{X_1,Y_1} & : & X_1 & \rightarrow & Y_1 \\ R_{X_2,Y_1} & : & X_2 & \rightarrow & Y_1 \\ R_{X_3,Y_2} & : & X_3 & \rightarrow & Y_2 \end{array} \tag{2.20}$$

  linking *types* of the variables.

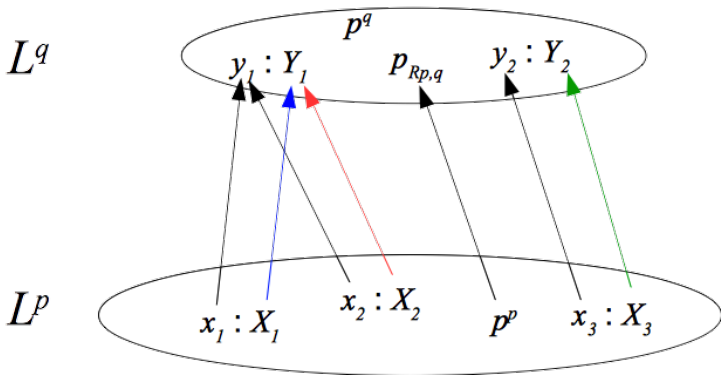


Figure 2.2: Relations linking moderated level of abstractions $L^p$ and $L^q$

All elements have now been introduced to give Floridi's notion of a *gradient of abstraction*:

**Notion 6** Gradient of abstraction
A **gradient of abstraction (GoA)** consists of a finite set of $N$ **moderated LoAs** $\{L^p, p \in [0; N[, N \in \mathbb{N}\}$ and a family of relations $R_{p,q} : L^p \rightarrow L^q, p \neq q$ relating **observables** of $L^p$ with **observables** of $L^q$ such as:

1. $\forall p, q, p \neq q, R_{p,q}$ is the reverse relation of $R_{q,p}$;
2. the **behaviour** $p^q$ at $L^q$ is at least as strong as the translated **behaviour** $p_{R_{p,q}}$;
3. for each **observable** $x_i : X_i$ of $L^p$ and $y_j : Y_j$ of $L^q$ linked by a relation $R_{p,q}$, there exists a relation $R_{X_i,Y_j}$ linking elements of their types $X_i$ and $Y_j$.

A *gradient of abstraction* is said to be *discrete* iff all *LoAs* are discrete; *disjoint* iff LoAs are pairwise disjoint (i.e. they have no observable in common) and the relations are all empty; and *nested* iff only non empty relations are those between two successive LoAs $L^p$ and $L^{p+1}$ and the reverse of each relation $R_{p,p+1}$ (i.e. $R_{p+1,p}$) is a surjective function from observables of $L^{p+1}$ to observables of $L^p$. A concrete example of *disjoint* and *nested* GoAs can be found in [LGV17], reproducted in the Appendix B.

As a consequence, the key point about Floridi's concept of GoA is the following property, close to the *incompleteness property* of Newell's *Knowledge Level*:

- The higher the level of abstraction, the fewer but richer the information.

Let us cite Floridi [Flo08, p. 18] to expose this point. *The quantity of information in a model varies with the LoA: a lower LoA, of greater resolution of finer granularity, produces a model that contains more information than a model produced at a higher, or more abstract, LoA.*

It is then clear that for Newell as for Floridi, a key point of abstraction is the ability to *forget* some thing to go from a LoA to a more abstract LoA. It is also clear that this *forgetfulness* must be *controlled* to maintain *rational relations* between the different LoA's of a GoA.

The Category Theory of Samuel Eilenberg and Saunders Mac Lane provides the adequate mathematical tools to this aim.

## 2.3 Merker's Introduction to the Category Theory

In years 1942 to 1945, *Samuel Eilenberg* and *Saunders Mac Lane* [ML71] introduced the concepts of categories and functors to reason about the mathematical structure of objets such as sets, rings or groups.

A category is characterized by two fundamental properties: (i) the *associativity* of morphisms' composition and (ii) the existence of an *identity morphism*. These two properties are necessary and sufficient to form a new morphism with an adequate sequence morphisms. To introduce the notion of functor, let us use Merker's lecturer notes of 1983 [Mer83].

Let $X$ be any set and $\top$, an *operation* satisfying some properties in $X$. Such a set provided with its *operation*, $(X, \top)$, is said to be *algebraically structured*.

The most important *algebraic structure* is the *group* structure:

**Definition 2.1** *Group*
*A* ***group*** *is a set $X$ provided with an operation $\top$ satisfying the following axioms:*

1. *associativity:*

$$\forall f, g, h \in X, (f \top g) \top h = f \top (g \top h) \tag{2.21}$$

2. *existence of a neutral element $e$:*

$$\exists e \in X, \forall f \in X, e \top f = f \top e = f \tag{2.22}$$

3. *existence of an inverse element of $f$ denoted $f^{'}$:*

$$\forall f \in X, \exists f^{'} \in X, f \top f^{'} = f^{'} \top f = e \tag{2.23}$$

*If the operation $\top$ satisfies the commutativity axiom:*

$$\forall f, g \in X, f \top g = g \top f \tag{2.24}$$

*then the* ***group*** *$X$ is said to be commutative.*

For instance, the set $\mathbb{Z}$ provided with the addition $+$ is a *group*.

A *homomorphism* aims to compare sets having the same (*homo*) shapes (*morphism*). Let us denote $X$ and $X^{'}$, respectively provided with operations denoted $\top$ and $\perp$, two *algebraically structured* sets.

**Definition 2.2** *Homomorphism*
*A* ***homomorphism*** $H$ *is an application:*

$$\forall f \in X, \begin{array}{rcccc} H & : & X & \rightarrow & X^{'} \\ & & f & \mapsto & H(f) \end{array} \tag{2.25}$$

*such as the following property is satisfied:*

$$\forall f, g \in X, H(f \top g) = H(f) \bot H(g) \tag{2.26}$$

A *homomorphism* transcribes the set $X$ in the set $X^{'}$ and the law $\top$ in the law $\bot$. A *homomorphism* is an application between two sets which preserves the set structure in each set as shown on figure 2.3 where the *homomorphism* $H$ is represented such that:

$$\begin{array}{rcccc} H & : & X & \rightarrow & X^{'} \\ & & f & \mapsto & H(f) \\ & & g & \mapsto & H(g) \\ & & z & \mapsto & H(z) \end{array} \tag{2.27}$$

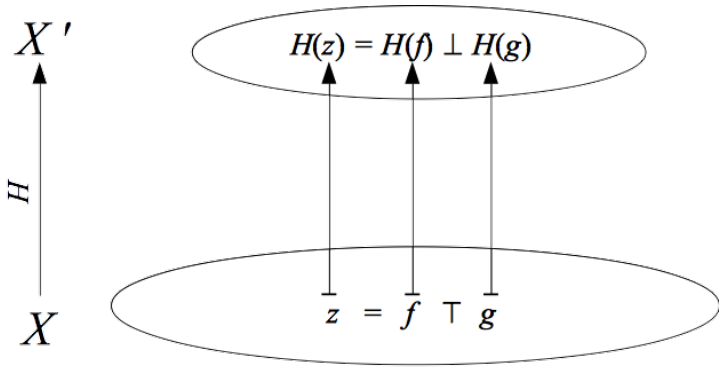


Figure 2.3: A *homomorphism* aims to compare sets and preserves the structure of each set

When a *homomorphism* is a surjective application, some informations available in the input set $X$ are forgotten and are then no more available in the output set $X^{'}$. On one hand, a *homomorphism* induces a loss of information but, on another hand, it simplifies the output set $X^{'}$. Such a simplification may be very useful to make simpler reasonings in $X^{'}$ to solve far more complex problems in $X$. An example of usage of such a *homomorphism* is given in [Mer83] (page 79) to bring a demonstration to the *Fermat's little theorem.*

*Homomorphisms* between *algebraic structures* satisfy the following properties:

1. the identity application is a *homomorphism* (see figure 2.4):
   Let be $X$ an *algebraic structure.* The identity application is a *homomorphism*, denoted $id_X$, such as:

$$\begin{array}{rcccc} id_X & : & X & \rightarrow & X \\ & & f & \mapsto & f \end{array} \tag{2.28}$$

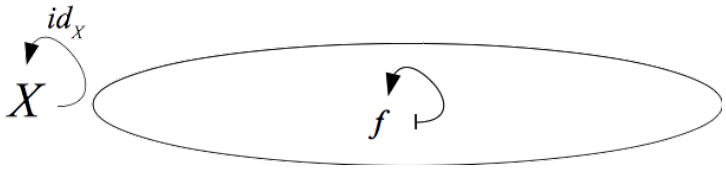


Figure 2.4: *Homomorphism* identity $id_X$

2. the composition application is a *homomorphism* (see figure 2.5):
   Let be $X$, $Y$ and $Z$ some *algebraic structures*. Let be $H : X \to Y$ a *homomorphism* such as:

$$\begin{array}{rccc} H & : & X & \to & Y \\ & & f & \mapsto & g \end{array} \tag{2.29}$$

   Let be $F : Y \to Z$ a *homomorphism* such as:

$$\begin{array}{rccc} F & : & Y & \to & Z \\ & & g & \mapsto & h \end{array} \tag{2.30}$$

   The composition application denoted $F \circ H : X \to Z$ such as:

$$\begin{array}{rccc} F \circ H & : & X & \to & Z \\ & & f & \mapsto & h \end{array} \tag{2.31}$$

   is a *homomorphism*.

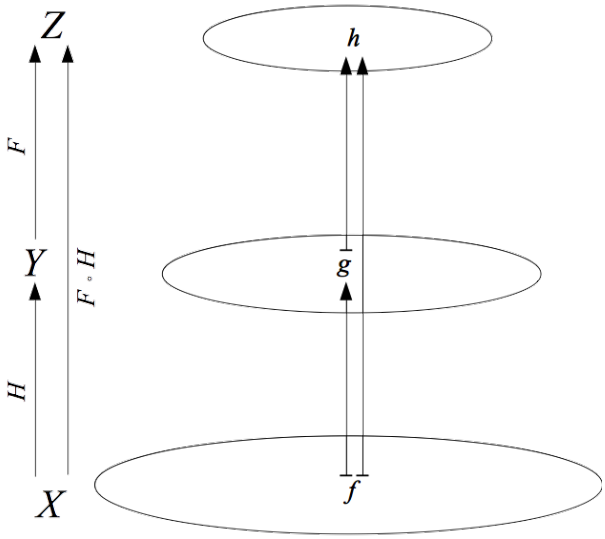


Figure 2.5: *Homomorphism* composition $F \circ H$

3. conditions of isomorphy:
   Let be $H : X \to Y$ a *homomorphism*. If there's exist a *homomorphism* $H' : Y \to X$ such as:

$$\begin{cases} H \circ H' = id_Y \\ H' \circ H = id_X \end{cases} \tag{2.32}$$

then $H$ is an **isomorphism**.

Finally, let us provide a common definition of a category:

**Definition 2.3** *Category*
*A* ***category*** $\mathfrak{C}$ *consists of the following entities:*

1. *a collection, denoted* $ob(\mathfrak{C}) = \{A_i, i \in \mathbb{N}\}$*, of* ***objects*** $A_i$*;*

2. *a collection, denoted* $mor(\mathfrak{C}) = \{f_k, k \in \mathbb{N}\}$*, of* ***morphisms*** $f_k$ *linking two* ***objects*** *of* $ob(\mathfrak{C})$*:*
$\forall f \in mor(\mathfrak{C}), \exists A, B \in ob(\mathfrak{C})$,

$$f \ : \ A \ \rightarrow \ B \tag{2.33}$$

3. *a binary operation, denoted* $\circ$*, called composition of morphisms such that:*
$\forall f : A \rightarrow B, \forall g : B \rightarrow C$,

$$g \circ f \ : \ A \ \rightarrow \ C \tag{2.34}$$

*and satisfying both following axioms:*

   (a) *the binary operation* $\circ$ *is associative:*
   $\forall f : A \rightarrow B, g : B \rightarrow C, h : C \rightarrow D$,

$$h \circ (g \circ f) \ = \ (h \circ g) \circ f \tag{2.35}$$

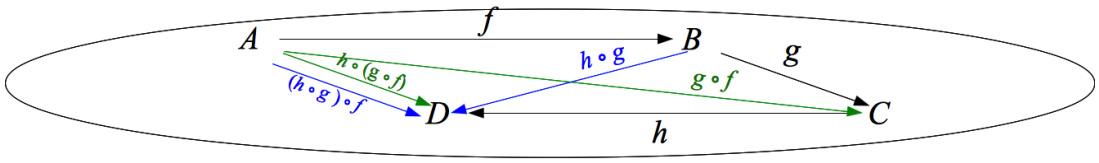


Figure 2.6: Associativity of the composition of morphisms

   (b) *the neutral element of the binary operation* $\circ$ *is the morphism identity :*
   $\exists id_A : A \rightarrow A, \exists id_B : B \rightarrow B, \forall f : A \rightarrow B$,

$$id_B \circ f \ = \ f \circ id_A \ = \ f \tag{2.36}$$

## 2.4 Sum and Product

The precedent section shows that the Category Theory emphasizes the *morphisms*, that is to say the *structure-preserving mappings* between objects. The notion of morphisms allows to reason and learn more about the structure of the objects.

Two operations are of the main interest for this work: the *product* and the *sum* of objects of categories [Hue85, Mél12, Vau08].

To introduce these operations, let us build the *category* $\mathfrak{C}$ (cf. figure 2.7):

- three objects $A$, $B$ and $C$;
- two morphisms $f : A \to B$ and $g : B \to C$;
- the composition of morphisms $g \circ f : A \to C$;
- three identity morphisms $id_A : A \to A$, $id_B : B \to B$ and $id_C : C \to C$.

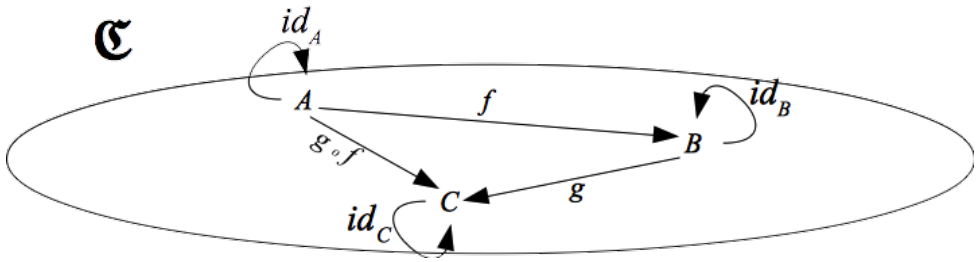


Figure 2.7: Example of a *category*

### 2.4.1 Sum

Let be $A$ and $B$, two objects of $ob(\mathfrak{C})$. The *sum* between object $A$ and object $B$, denoted $\Sigma \equiv A + B$ is an object associated with two morphisms, $f : A \to \Sigma$ and $g : B \to \Sigma$, such that, for all object $X \in ob(\mathfrak{C})$ and for all couple of morphisms $u : A \to X$ and $v : B \to X \in mor(\mathfrak{C})$, there exists a unique morphism $w : \Sigma \to X$ such that (see figure 2.8):

$$\begin{cases} u = w \circ f \\ v = w \circ g \end{cases} \tag{2.37}$$

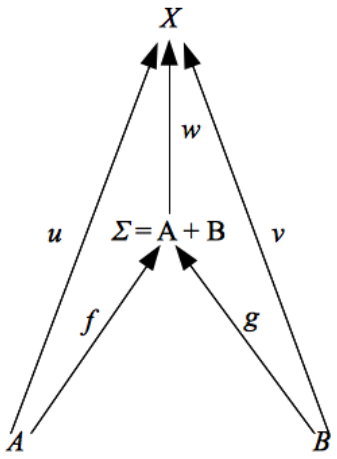


Figure 2.8: Sum of two objects in *category* $\mathfrak{C}$

Intuitively, sum of objects corresponds to the merging of two structures into one in which care is taken to avoid any uncontrolled telescoping. When considering the *category* of sets *Set*, the

sum of two sets is their disjointed union denoted "$\sqcup$". For example, if $A = \{a, b\}$ and $B = \{b, c\}$, the sum $\Sigma$ between $A$ and $B$ is the set $\Sigma = A \sqcup B = \{a, b_1, b_2, c\}$ where the elements $b \in A$ and $b \in B$ has been renamed to differentiate their occurrences.

In his thesis, [Vau08] gives a general formulation for the *sum* between $n, n \in \mathbb{N}$, objects of a *category*.

**Definition 2.4** *Sum $\Sigma$ between objects in a category [Vau08]*
*The sum of $n$ objects $A_i$ in a **category** $\mathfrak{C}$ is an object $\Sigma$ denoted $\Sigma \equiv A_1 + A_2 + ... + A_n$ associated with $n$ morphisms $i_j : A_j \to \Sigma, j \in [1; n]$ such as, for a given object $W$ in **category** $\mathfrak{C}$ and $n$ morphisms $f_i : A_i \to W$, there exists a unique morphism denoted $< f_1; f_2; ...; f_n >: \Sigma \to W$ such that, for all $j \in [1; n]$, $< f_1; f_2; ...; f_n > \circ i_j = f_j$ (diagram of figure 2.9 is commutative).*

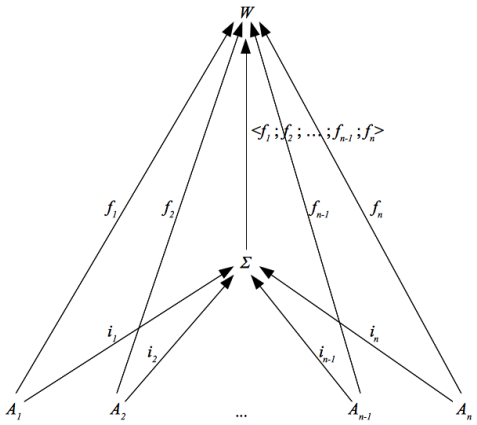


Figure 2.9: Commutative diagram of the sum of $n$ objects in the *category* $\mathfrak{C}$

The diagram of figure 2.9 is called the *sum diagram* of $n$ objects in the *category* $\mathfrak{C}$.

### 2.4.2 Product

Let be $A$ and $B$, two objects of $ob(\mathfrak{C})$. The *product* between object $A$ and object $B$, denoted $\Pi \equiv A \times B$ is an object associated with two morphisms, $f : \Pi \to A$ and $g : \Pi \to B$, such that, for all object $X \in ob(\mathfrak{C})$ and for all couple of morphisms $u : X \to A$ and $v : X \to B \in mor(\mathfrak{C})$, there exists a unique morphism $w : X \to \Pi$ such that (see figure 2.10):

$$\begin{cases} f \circ w = u \\ g \circ w = v \end{cases} \tag{2.38}$$

For example, when the *category* $\mathfrak{C}$ is the *category* of groups *Grp*, where each group is structured with the operation multiplication denoted "$\cdot$", then:

- the *product* $\Pi = A \times B$ is the set of couples $(a, b)$ where $a \in A$ and $b \in B$:

$$A \times B = \{(a, b), a \in A, b \in B\} \tag{2.39}$$

- the morphisms $f : \Pi \to A$ and $g : \Pi \to B$ are the *projection* of respectively $A \times B$ in $A$ and $A \times B$ in $B$:

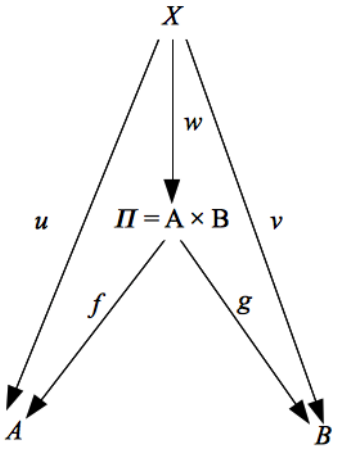


Figure 2.10: Product of two objects in *category* $\mathfrak{C}$

$$f : \begin{array}{ccc} A \times B & \to & A \\ (a,b) & \mapsto & a \end{array} \text{ and } g : \begin{array}{ccc} A \times B & \to & B \\ (a,b) & \mapsto & b \end{array} \tag{2.40}$$

- the multiplication "$\times$" on $A \times B$ is given by the following formula which is the definition of the *cartesian product*:

$$\forall (a,b),(a^{'},b^{'}) \in A \times B, (a,b) \times (a^{'},b^{'}) = (a \cdot a^{'}, b \cdot b^{'}) \tag{2.41}$$

In his thesis, [Vau08] gives a general formulation for the *product* between $n, n \in \mathbb{N}$, objects of a *category*.

**Definition 2.5** *Product* $\Pi$ *between objects in a category [Vau08]*
*The product of* $n$ *objects* $A_i$ *in a* ***category*** $\mathfrak{C}$ *is an object* $\Pi$ *denoted* $\Pi \equiv A_1 \times A_2 \times ... \times A_n$ *associated with* $n$ *morphisms* $p_j : A_j \to \Pi, j \in [1;n]$ *such as, for a given object* $W$ *in* ***category*** $\mathfrak{C}$ *and* $n$ *morphisms* $f_i : A_i \to W$*, there exists a unique morphism denoted* $< f_1, f_2, ..., f_n >: W \to \Pi$ *such that, for all* $j \in [1;n]$*,* $p_j \circ < f_1, f_2, ..., f_n >= f_j$ *(diagram of figure 2.11 is commutative).*

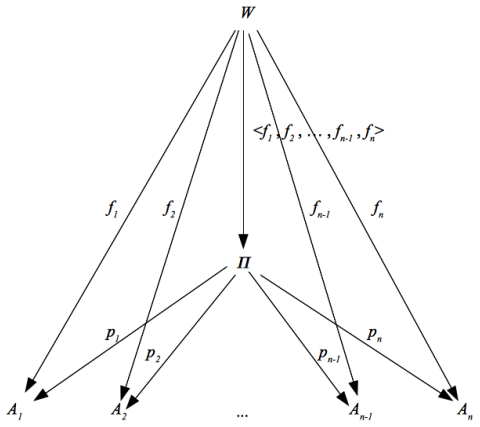


Figure 2.11: Commutative diagram of the product of $n$ objects in the *category* $\mathfrak{C}$

The diagram of figure 2.11 is called the *product diagram* of $n$ objects in the *category* $\mathfrak{C}$.

A category is itself a type of mathematical structure, so we can look for processes which preserve this structure in some sense. Such a process is called a *functor*.

## 2.5 Functors

*Functors* can be seen as morphisms between categories (cf. figure 2.12 for an illustration).

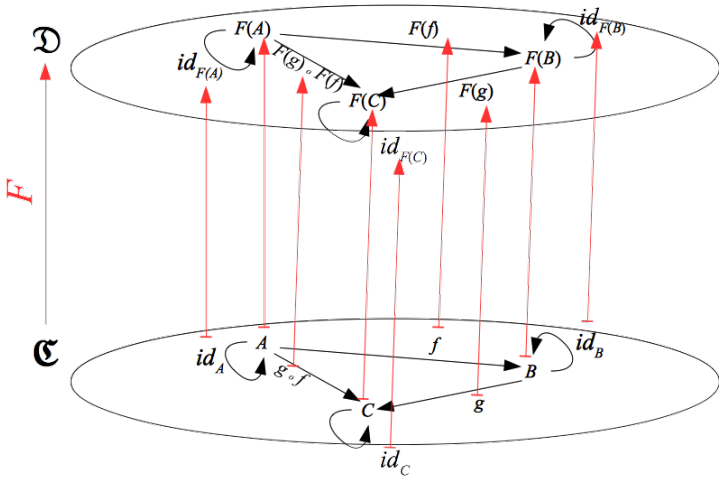


Figure 2.12: Representation of a *functor* between *categories* $\mathfrak{C}$ and $\mathfrak{D}$

A *functor* associates to every object of one category an object of another category, and to every morphism in the first category a morphism in the second.

**Definition 2.6** *Functors*
*Let* $\mathfrak{C}$ *and* $\mathfrak{D}$ *be categories. A* ***functor*** $F$ *from category* $\mathfrak{C}$ *to category* $\mathfrak{D}$*, denoted* $F : \mathfrak{C} \to \mathfrak{D}$*, maps:*

- *each object* $X$ *in the category* $\mathfrak{C}$ *with an object* $F(X)$ *in the category* $\mathfrak{D}$*:*

$$\begin{array}{rccc} F & : & \mathfrak{C} & \to & \mathfrak{D} \\ & & X & \mapsto & F(X) \end{array} \tag{2.42}$$

- *each morphism* $f : X \to Y$ *in the category* $\mathfrak{C}$ *with a morphism* $F(f) : F(X) \to F(Y)$ *in the category* $\mathfrak{D}$*:*

$$\begin{array}{rccc} F & : & \mathfrak{C} & \to & \mathfrak{D} \\ & & f & \mapsto & F(f) \end{array} \tag{2.43}$$

- *and satisfying both following properties:*

  1. *conservation of the composition operation* $\circ$*:*

$$\forall f : X \to Y, \forall g : Y \to Z, F(f \circ g) = F(f) \circ F(g) \tag{2.44}$$

  2. *morphism identity mapping:*

$$\forall id_X : X \to X, F(id_X) = id_{F(X)} \tag{2.45}$$

A concrete utilization of a *functor* can be found in [Mer83] (page 87) where the *Poincaré's functor* is introduced to bring a demonstration to *Brouwer fixed point theorem*.

It is clear that *functors* are powerful tools to analyze the similarities and the differences between models because they allow to *forget* details to emphasis on the structure.

## 2.6 Conclusion

To forget some properties of a concrete situation to be analyzed constitutes a common point of Newell's, Floridi's and Merker's analysis of the role of models in sciences:

1. Newell's ontological view about abstraction and Floridi's epistemological view are *complementary*.
2. Both claim for the *oversight phenomenon* as a price to be paid when *abstracting*.
3. The Category theory is an adequate *mathematical framework* to *control* the oversight phenomenon.

More precisely, to be efficient, any abstraction reasoning must be based on an *adequate choice* of some specific properties that must be considered. The design of an adequate set of functors is then the key point to define an abstraction process that can be used to deal with the interpretation of *data flows*, especially when the flow is big.

According to Floridi, Newell's *Knowledge Level* can be structured in a set of *hierarchical levels of abstraction*, and both agree that changing from a level of abstraction $i$ to a more abstract one $i+1$ relies on the *forgetting* of particular properties described at the level $i$.

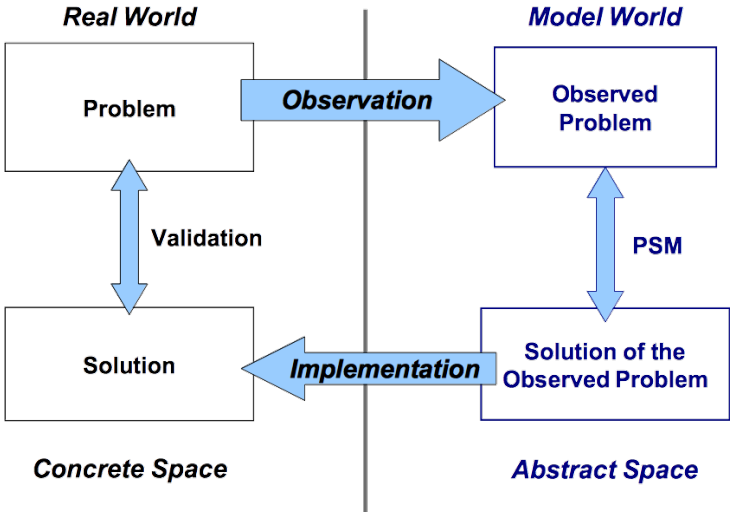


Figure 2.13: Problem Solving Method

Such a sentence could generate numerous and long discussions about the relations between models and reality. Nevertheless, similarly to Merker's praxeological analysis, this debate can be avoided in the filed of Computer Sciences when considering figure 2.13. Computer Sciences are

mainly concerned with the building of computer systems that facilitate the solving of concrete problems. The role of humans is to solve the problems and the role of computer systems is to produce *representations* supposed to help humans in their solving tasks. The key point in Computer Sciences is that the *representations* reside in the *Model World* when the problems reside in the *Real World*. To caricature, Computer Sciences *ignore* what an object of the Real World is, only the objects of the Model World can be *understood* by a Computer Scientists.

So, according to the Computer Sciences, Newell's and Floridi's theories about abstraction have a concrete meaning when applied on *Models*, that is to say an organized set of *Knowledge Representations*. And Merker to clearly explain that the potential debate about what belongs to *reality* and what not must then be moved to *what belongs to a model and what not*.

From this chapter, it is possible to conclude on the possibility to combine the ontological point of view of Newell and the epistemological point of view of Floridi when only considering the world of the *Model* uniquely. In this world, the Category theory provides the conceptual tool of *functor* to formalize the *transformation* of models in an abstraction reasoning. In the field of Knowledge Engineering, such an approach is called a *Problem Solving Method* (PSM).

Such is the aim of our research project: to define a *Problem Solving Method* based on a recursive abstraction reasoning that rationally controls the *oversight phenomenon* by the mean of the design of *adequate functors*.

To this aim, it is also necessary to provide a mathematical consistence to Newell's and Floridi's theories. This is the role of the *Timed Observations Theory* (TOT), introduced in the next chapter, where the TOT's concepts of (i) timed observation and (ii) timed sequential binary relation will be used as paradigm through multiple Levels of Abstraction.

CHAPTER 3

# Introduction to the Timed Observation Theory

## 3.1 Introduction

The Timed Observations Theory ([LeG06]) provides a general mathematical framework for modeling dynamic processes from timed data. This theory combines and extends the Markov Chain Theory, the Poisson Process Theory, the Theory of Communication of Shannon [Sha84], the Logical Theory of Diagnosis [Dag01]. This chapter aims at introducing the concepts of the TOT that are required in order to model a dynamic process and the reasoning with a TOT model.

To this aim, the chapter is made with two main sections. The first one introduces the main mathematical objects of the TOT. The second section makes use of these objects to define the basis the modeling approach of the TOT which are the foundation of the Tom4D method (Timed Observations Modeling for Diagnosis, [Pom12]). The modeling principles of Tom4D constitute the conceptual modeling framework of our contributions.

The original version of the TOT can be found in [LeG06]. Applications in various scientific domains can be found in the PHD of P. Bouché [Bou05] for the Discrete Event System domain, the PHD of N. Benayadi [Ben10] for the Data Mining domain, the PHD of A. Adhab [Ahb10a] for the Bayesian Networks domain, and the PHD of L. Pomponio [Pom12] for the Knowledge Engineering Domain. The Reader will be invited to refer to the main papers providing details about the developments of the TOT.

But this chapter provides the *necessary* and *sufficient* elements of the TOT that are required to understand our contributions. And to allow an intuitive understanding of these elements, the running example of the conversation will be used.

## 3.2 Mathematical Framework of the TOT

The Timed Observations Theory (TOT) defines a *dynamic process* as an arbitrarily constituted set $X(t) = \{x_1(t), ..., x_n(t)\}$ of timed functions $x_i(t)$ of continuous time $t$. The set $X(t)$ of functions implicitly defines a set $X = \{x_1, x_2, ..., x_n\}$ of $n$ variable names $x_i$.

According to the TOT, a dynamic process $X(t)$ is said to be *observed* by a program $\Theta$ when $\Theta$ aims at writing *timed messages* describing the modifications over time of the functions $x_i(t)$ of $X(t)$. Such timed messages can be alarms, warnings, reporting events or simple communication messages (sms for example) that are sent to the environment of the program $\Theta$. A timed message is a sequence of characters that can be recorded in a memory (i.e. a database or a data log).

The TOT considers that the structure of such a message is a pair (*timestamp*, *text*) where *text* is a *constant* denoted $\delta_i$ and *timestamp* is the value of an *index* denoted $t_k$. The timestamp

$t_k$ represents the time of the message that can be the emission time, the reception time or any timestamp associated with the text of the message. Consequently, an extraction of $m$ timed messages from a memory constitutes a set $\Delta = \{\delta_i\}$ of $m_\Delta$ constants $\delta_i$ and a set $\Gamma = \{t_k\}$ of $m$ timestamps $t_k$. Generally speaking, there is less constants than timestamps (i.e. $m_\Delta \leq m$).

As an illustration, let us consider the four observations made by Alice about the engine (see section B.2). Alice observes that:

1. its engine is old;
2. its engine consumed too much;
3. it has a stable market value;
4. its spare parts are expensive.

Let us consider the sentence "*its engine is old*". This sentence implicitly defines a timed function denoted $x_1^A(t)$ representing the evolution over time of the engine's *age*. Such a function admits, as input, an infinity set of values and returns also an infinity set of values. The function $x_1^A(t)$ is a $\mathfrak{R}$-valued function defined on $\mathfrak{R}$:

$$\begin{array}{rcccc} x_1^A & : & \mathfrak{R} & \rightarrow & \mathfrak{R} \\ & & t & \mapsto & x_1^A(t) \end{array} \tag{3.1}$$

Such a sentence shows that there exists, in Alice's mind, a constant describing the *age* of that engine as *old*. Let us denote $\delta_{11}^A \equiv old$ such a constant. This implies that there also exists, in Alice's mind, another constant describing the *age* of that engine as *not old*. Let us denote $\delta_{12}^A \equiv not\_old$ such a constant. The *age* of such an engine can be then described with two constants *old* and *not_ old*. The timed function $x_1^A(t)$, representing the evolution over time of the engine's *age*, can be then described with a variable called *age* and denoted $x_1^A$:

$$x_1^A \equiv age \tag{3.2}$$

Such a variable $x_1^A \equiv age$ takes two discrete values:

$$\begin{cases} \delta_{11}^A \equiv old \\ \delta_{12}^A \equiv not\_old \end{cases} \tag{3.3}$$

Let us denote $\Delta_1^A$, the *countable* set containing these discrete values:

$$\begin{cases} \Delta_1^A = \{\delta_{11}^A, \delta_{12}^A\} \\ x_1^A \in \Delta_1^A \end{cases} \tag{3.4}$$

Let us consider the sentence "*its engine consumed too much*". This sentence implicitly defines a timed function denoted $x_2^A(t)$ representing the evolution over time of the engine's *consumption*. Such a function is also a $\mathfrak{R}$-valued function defined on $\mathfrak{R}$:

$$\begin{array}{rcccc} x_2^A & : & \mathfrak{R} & \rightarrow & \mathfrak{R} \\ & & t & \mapsto & x_2^A(t) \end{array} \tag{3.5}$$

Such a sentence shows that there exists, in Alice's mind, a constant describing the *consumption* of that engine as *too much*. Let us denote $\delta_{21}^A \equiv too\_much$ such a constant. This implies

that there also exists, in Alice's mind, another constant describing the *consumption* of that engine as *not too much*. Let us denote $\delta_{22}^A \equiv not\_too\_much$ such a constant. The *consumption* of such an engine can be then described with two constants *too_much* and *not_too_much*. The timed function $x_2^A(t)$, representing the evolution over time of the engine's *consumption*, can be then described with a variable called *consumption* and denoted $x_2^A$:

$$x_2^A \equiv consumption \tag{3.6}$$

Such a variable $x_2^A \equiv consumption$ takes two discrete values:

$$\begin{cases} \delta_{21}^A \equiv too\_much \\ \delta_{22}^A \equiv not\_too\_much \end{cases} \tag{3.7}$$

Let us denote $\Delta_2^A$, the *countable* set containing these discrete values:

$$\begin{cases} \Delta_2^A = \{\delta_{21}^A, \delta_{22}^A\} \\ x_2^A \in \Delta_2^A \end{cases} \tag{3.8}$$

Let us consider the sentence "*it has a stable market value*". This sentence implicitly defines a timed function denoted $x_3^A(t)$ representing the evolution over time of the engine's *market value*. Such a function is also a $\mathfrak{R}$-valued function defined on $\mathfrak{R}$:

$$\begin{array}{rccc} x_3^A & : & \mathfrak{R} & \rightarrow & \mathfrak{R} \\ & & t & \mapsto & x_3^A(t) \end{array} \tag{3.9}$$

Such a sentence shows that there exists, in Alice's mind, a constant describing the *market value* of that engine as *stable*. Let us denote $\delta_{31}^A \equiv stable$ such a constant. This implies that there also exists, in Alice's mind, another constant describing the *market value* of that engine as *not stable*. Let us denote $\delta_{32}^A \equiv not\_stable$ such a constant. The *market value* of such an engine can be then described with two constants *stable* and *not_stable*. The timed function $x_3^A(t)$, representing the evolution over time of the engine's *market value*, can be then described with a variable called *market_value* and denoted $x_3^A$:

$$x_3^A \equiv market\_value \tag{3.10}$$

Such a variable $x_3^A \equiv market\_value$ takes two discrete values:

$$\begin{cases} \delta_{31}^A \equiv stable \\ \delta_{32}^A \equiv not\_stable \end{cases} \tag{3.11}$$

Let us denote $\Delta_3^A$, the *countable* set containing these discrete values:

$$\begin{cases} \Delta_3^A = \{\delta_{31}^A, \delta_{32}^A\} \\ x_3^A \in \Delta_3^A \end{cases} \tag{3.12}$$

Let us consider the sentence "*its spare parts are expensive*". This sentence implicitly defines a timed function denoted $x_4^A(t)$ representing the evolution over time of the *spare parts' cost*. Such a function is also a $\mathfrak{R}$-valued function defined on $\mathfrak{R}$:

$$\begin{array}{rcccc} x_4^A & : & \mathfrak{R} & \rightarrow & \mathfrak{R} \\ & & t & \mapsto & x_4^A(t) \end{array} \tag{3.13}$$

Such a sentence shows that there exists, in Alice's mind, a constant describing the *spare parts' cost* of that engine as *expensive*. Let us denote $\delta_{41}^A \equiv expensive$ such a constant. This implies that there also exists, in Alice's mind, another constant describing the *spare parts' cost* of that engine as *not expensive*. Let us denote $\delta_{42}^A \equiv not\_expensive$ such a constant. The *spare parts' cost* of such an engine can be then described with two constants *expensive* and *not_expensive*. The timed function $x_4^A(t)$, representing the evolution over time of the engine's *spare parts' cost*, can be then described with a variable called *spare_parts_cost* and denoted $x_4^A$:

$$x_4^A \equiv spare\_parts\_cost \tag{3.14}$$

Such a variable $x_4^A \equiv spare\_parts\_cost$ takes two discrete values:

$$\begin{cases} \delta_{41}^A \equiv expensive \\ \delta_{42}^A \equiv not\_expensive \end{cases} \tag{3.15}$$

Let us denote $\Delta_4^A$, the *countable* set containing these discrete values:

$$\begin{cases} \Delta_4^A = \{\delta_{41}^A, \delta_{42}^A\} \\ x_4^A \in \Delta_4^A \end{cases} \tag{3.16}$$

Program $\Theta$ is said to be :

- *Parametrized* with two sets, the set of constants $\Delta = \{\delta_i\}$ and the set of variable names $X = \{x_i\}$. A parametrized program is denoted $\Theta(X, \Delta)$.
- *Applied* on a set $X(t) = \{x_i(t)\}$ of timed functions. The choice of $X(t)$ is out of the scope of the program.
- An *abstract observer* if the way the program $\Theta$ has been implemented is not known. It can be either a software component or a human. This situation has to be considered as the usual situation.

In the previous example, Alice is the observation program denoted $\Theta^A(X^A, \Delta^A)$ where:

- $X^A = \{x_1^A, x_2^A, x_3^A, x_4^A\}$;
- $\Delta^A = \{\delta_{11}^A, \delta_{12}^A, \delta_{21}^A, \delta_{22}^A, \delta_{31}^A, \delta_{32}^A, \delta_{41}^A, \delta_{42}^A\} \equiv \{old,\ not\_old,\ too\_much,\ not\_too\_much,\ stable,\ not\_stable,\ expensive,\ not\_expensive\}$.

In the suite of the document, the term *program* will be used to denote either the program $\Theta$ or the parametrized program $\Theta(X, \Delta)$.

## 3.3 Timed Observation

The aim of the TOT is to model *observed processes*:

**Definition 3.1** *Observed Process*
*Let $X(t) = \{x_i(t)\}$, i=1...n, be a finite set of time functions $x_i(t)$; let $X = \{x_i\}$, i=1...n, be the corresponding finite set of variable names $x_i$; let $\Delta = \{\delta_j\}$, j=1...m, be a finite set of constant values $\delta_j$; let $\Theta(X, \Delta)$ be a program observing the evolution of the functions of $X(t)$.*

*The couple $(X(t), \Theta(X, \Delta))$ is an observed process.*

In the previous example, the *observed process* is then the couple $(X^A(t), \Theta^A(X^A, \Delta^A))$ where:

$$X^A(t) = \{x_1^A(t), x_2^A(t), x_3^A(t), x_4^A(t)\} \tag{3.17}$$

To this aim, the TOT defines a *timed observation* to provide a *meaning* to a timed message:

**Definition 3.2** *Timed Observation*
*Let $X(t) = \{x_i(t)\}_{i=1...n}$ be a set of timed functions describing the evolution of a process that is observed by a program $\Theta$; let $\Gamma = \{t_k\}_{t_k \in \Re}$ be a set of arbitrary time instants in which $\Theta$ observes the functions; let $\theta(x_\theta, \delta_\theta, t_\theta)$ be a predicate implicitly determined by $\Theta$; and, let $\Delta$ be a set of constant values.*

*A timed observation $(\delta, t_k) \in \Delta \times \Gamma$ made on the time function $x_i(t)$ is the assignation of values $x_i$, $\delta$ and $t_k$ to the predicate $\theta(x_\theta, \delta_\theta, t_\theta)$ such that $\theta(x_i, \delta, t_k)$.*

Technically, $(\delta, t_k)$ (or $O(t_k)$) denotes a *record* of a database. The assigned predicate $\theta(x_i, \delta, t_k)$ represents the *meaning*, the *interpretation* of the record. But, by misuse of language, $(\delta, t_k)$ (or $O(t_k)$) is usually called a *timed observation*.

As an illustration of this definition, let us consider the sentence "*Alice observes that it has a stable market value*". The observation program, Alice, denoted $\Theta^A(X^A, \Delta^A)$, observes the evolution of the function $x_3^A(t)$ representing the temporal evolution of the engine's *market value*. The *timed observation* (*stable*, $t_3^A$) made on time function $x_3^A(t)$ is the assignation of values $x_3^A \equiv market\_value$, $\delta_{31}^A \equiv stable$ and $t_3^A$ to the predicate denoted $\theta^A(x_{\theta^A}, \delta_{\theta^A}, t_{\theta^A})$ implicitly implemented in $\Theta^A(X^A, \Delta^A)$ such that $\theta^A(x_3^A, \delta_{31}^A, t_3^A)$.

One of the most fondamental point of the TOT is to understand that, *without any knowledge about the assigned predicate $\theta(x_i, \delta, t_k)$*, a timed observation $(\delta, t_k)$ has *no meaning*. For example, according to the TOT the text "*Alice observes that it has a stable market value*" has no meaning: it is only a sequence of characters. The section 3.6 illustrates this point.

### 3.3.1 Examples of Sets of Constants and Assignations

Sets of constants $\Delta$ are usually constructed on mathematical sets like $\Re$ or $\mathbb{N}$. But the sets of constants made with respect to an alphabet are even more often used and, this, in very varied fields.

- $\Delta \subset \Re$:
  - A timed observation of the form $O(t_k) \equiv (\mu, t_k)$ corresponding to the assignation $\theta(x, \mu, t_k)$ can be any assertions like $x(t_k) = \mu$, $x(t_k) < \mu$ or $x(t_k) = f(\mu)$ such as:

$$\begin{cases} \exists t_k, & \forall t < t_k, x(t) = f(t) \\ & \forall t \geq t_k, x(t) = f(t) + \mu \end{cases} \tag{3.18}$$

  - In the field of statistics and probabilities, the observations concern propositions relating to a random variable $x$. For example, the observation $O(t_k) \equiv (\delta, t_k)$ can be used to perform the assignation $\theta(x, \delta, t_k)$ meaning that at a time of measurement $t = t_k$, the average of $x$ is $m$ ($\delta = m$) or its standard deviation is $\sigma$ ($\delta = \sigma$).
  - $\Delta = \{]-\infty, \mu_{-2}], ]\mu_{-2}, \mu_{-1}], ]\mu_{-1}, \mu_1], ]\mu_1, \mu_2], ]\mu_2, +\infty]\}, \mu_i \in \Re$. This kind of set is very often used in fields related to Physics: $O(t_k) \equiv (]\mu_1, \mu_2], t_k) \rightarrow \theta(x, ]\mu_1, \mu_2], t_k) : x(t_k) \in ]\mu_1, \mu_2]$.

- $\Delta \subset \mathbb{N}$:
  - $\Delta = [0; 100]$: this kind of set is widely used to normalize values from sensors or statistics.
  - $\Delta = \{-n, ..., 0, ..., +n\}$: this kind of set is used in many fields including monitoring, diagnostics and control of dynamic systems such as Sachem [LeG04]. Indeed, this kind of set can be put into a one-to-one relation with a set of (2n + 1) real number intervals of the form $[\mu_i, \mu_{i+1}]$. The following observations on $x(t_k)$ illustrate this principle with a set $\Delta = \{-2, -1, 0, 1, 2\}$ isomorphic to the set of intervals $I = \{]-\infty, \mu_{-2}], ]\mu_{-2}, \mu_{-1}], ]\mu_{-1}, \mu_1], ]\mu_1, \mu_2], ]\mu_2, +\infty]\}$:
    * $O(t_k) \equiv (+1, t_k) \rightarrow \theta(x, +1, t_k) : x(t_k) \in [\mu_1, \mu_2]$.
    * $O(t_k) \equiv (-2, t_k) \rightarrow \theta(x, -2, t_k) : x(t_k) \in ]-\infty], \mu_{-2}]$.

- $\Delta$ is an alphabet of the form $\{\alpha, \beta, ..., \xi, \zeta\}$:
  - $\Delta = \{do$, ré, mi, fa, sol, la, si $\}$: these constants are used in music to mark notes on a musical scope according to a particular key.
  - $\Delta = \{red, blue, yellow\}$: this set can be retained by a painter as the primary color base of a chromatic circle to design a painting.
  - $\Delta = \{very\_low, low, normal, high, very\_high\}$: this kind of set is widely used to interpret alarms generated by a monitoring or diagnostic system. One of the reasons for the success of this type of set is that it establishes a one-to-one link between a set of (2n+1) real number intervals of the form $I = \{]-\infty, \mu_{-2}], ]\mu_{-2}, \mu_{-1}], ]\mu_{-1}, \mu_1], ]\mu_1, \mu_2], ]\mu_2, +\infty]\}$ and a set of (2n+1) integers of the form $\Delta = \{-n, ..., 0, ..., +n\}$.

### 3.3.2 Observation Function

Let be $\Theta(\{x\}, \Delta)$, a program observing the timed function $x(t)$.

**Definition 3.3** *Observation Function*
*Program $\Theta(\{x\}, \Delta)$ implements an observation function, denoted $\delta_x(k) = O(t_k)$, which is a $\Delta \times \Gamma$-valued function defined on $\mathbb{Z}$:*

$$\begin{array}{rccc} \delta_x & : & \mathbb{Z} & \rightarrow & \Delta \times \Gamma \\ & & k & \mapsto & O(t_k) \equiv (\delta, t_k) \end{array} \tag{3.19}$$

This leads us to the following property:

**Property 1**
*Any program* $\Theta(\{x\}, \Delta)$ *is characterized by an observation function* $\delta_x$.

Such an observation function $\delta_x$ provides the $k^{th}$ timed observation $O(t_k) \equiv (\delta, t_k)$, $\delta \in \Delta$ made on $x(t)$ at timestamp $t = t_k \in \Gamma$. It maps then, in a one to one way, any integer of $\mathbb{Z}$ with a timed observation $O(t_k) \equiv (\delta, t_k)$.

Thus, this allows the rewriting of a timed observation:

$$O(t_k) \equiv (\delta, t_k) \equiv (\delta, k) \equiv O(k) \tag{3.20}$$

When considering the timed observation of the form $O(k) \equiv (\delta, k)$, $k$ designates the index of the timestamp $t_k$ in a set of timestamps $\Gamma$.

## 3.4 Canonical and Safe Program

Let be a program $\Theta(\{x\}, \Delta)$ characterized by its observation function $\delta_x$.

**Definition 3.4** *Canonical Program*
*Program* $\Theta(\{x\}, \Delta)$ *is a canonical program if it aims at writing timed observations applied on the timed function* $x(t)$.

A program $\Theta(\{x\}, \Delta)$ is said to have no simultaneous timed observations iif:

$$\forall O(i) \equiv (\delta_i, i), \forall O(j) \equiv (\delta_j, j), i = j \Rightarrow O(i) = O(j) \Rightarrow \delta_i = \delta_j \wedge t_i = t_j \tag{3.21}$$

An observation function $\delta_x$ is memoryless iff for all $O(i) \equiv (\delta_i, i)$, for all $O(j) \equiv (\delta_j, j)$, $i \neq j$, the choice of the constant $\delta_i$ does not depend on $\delta_j$.

This condition is the rule of independence of observations: program $\Theta(\{x\}, \Delta)$ does not need to memorize the constants assigned in the past to choose the constants to assign in the present.

**Definition 3.5** *Independent Program*
*Any program* $\Theta(\{x\}, \Delta)$ *encoding a memoryless observation function* $\delta_x$ *is an independent program.*

This leads us to the following properties:

**Property 2** Safe Program
*An independent program* $\Theta(\{x\}, \Delta)$ *that does not allow simultaneous observations is a safe program.*

**Property 3** Safe Canonical Program
*A safe program* $\Theta(\{x\}, \Delta)$ *applied on a function* $x(t)$ *is a safe canonical program.*

**Property 4** Non Safe Program
*A program $\Theta(\{x\}, \Delta)$ that does allow some simultaneous observations or that is not independent is a non safe program.*

**Property 5** Non Safe Canonical Program
*A non safe program $\Theta(\{x\}, \Delta)$ applied on a function $x(t)$ is a non safe canonical program.*

In the following, we only consider safe canonical programs.

## 3.5 Spatial Discretization Principle

Let us consider a particular program $\Theta_i(\{x_i\}, \{\delta_i\})$ of an observed process $(x_i(t), \Theta_i(\{x_i\}, \{\delta_i\}))$ made of only one time function $x_i(t)$. Let us suppose that the specification of such a program is based on the generic rule 3.22 which refers to a threshold value $\Psi_j \in \Re$ and two immediately successive values $x_i(t_{k-1}) \in \Re$ and $x_i(t_k) \in \Re$.

$$x_i(t_{k-1}) < \Psi_j \wedge x_i(t_k) \geq \Psi_j \Rightarrow write((\delta_i, t_k)) \tag{3.22}$$

In this rule:

- $x_i(t_{k-1})$ and $x_i(t_k)$ are two immediately successive values of the continuous time function $x_i(t)$,
- $(\delta_i, t_k)$ is a timed message,
- $x_i(t_{k-1}) < \Psi_j \wedge x_i(t_k) \geq \Psi_j$ specifies a particular predicate denoted $\theta_i$,
- $write((\delta_i, t_k))$ denote the action of recording a timed message in a memory.

In other words, a timed observation $(\delta, t_k)$ is the *execution trace* of the program $\Theta_i$. Such a program $\Theta_i$ is called an *Unary Observer* in the framework of the TOT. The equation 3.22 is a current specification for an unary observer in the industrial domain. A more general form is given below.

The use of such a rule is illustrated in figure 3.1. On this figure, the time function $x_3^A(t)$ represents the evolution over time of the engine's *market value*. Let us consider Alice as the program, denoted $\Theta_3^A(\{x_3^A\}, \{stable, not\_stable\})$, observing such a time function $x_3^A(t)$. Let us suppose that the specification of such a program is made with one threshold $\Psi_3^A$ and the two following rules used on the time function $x_3^A(t)$:

$$\text{Rule 1: } x_3^A(t_{3,k-1}^A) < \Psi_3^A \wedge x_3^A(t_{3,k}^A) \geq \Psi_3^A \Rightarrow write((stable, t_{3,k}^A)) \tag{3.23}$$

$$\text{Rule 2: } x_3^A(t_{3,k-1}^A) \geq \Psi_3^A \wedge x_3^A(t_{3,k}^A) < \Psi_3^A \Rightarrow write((not\_stable, t_{3,k}^A)) \tag{3.24}$$

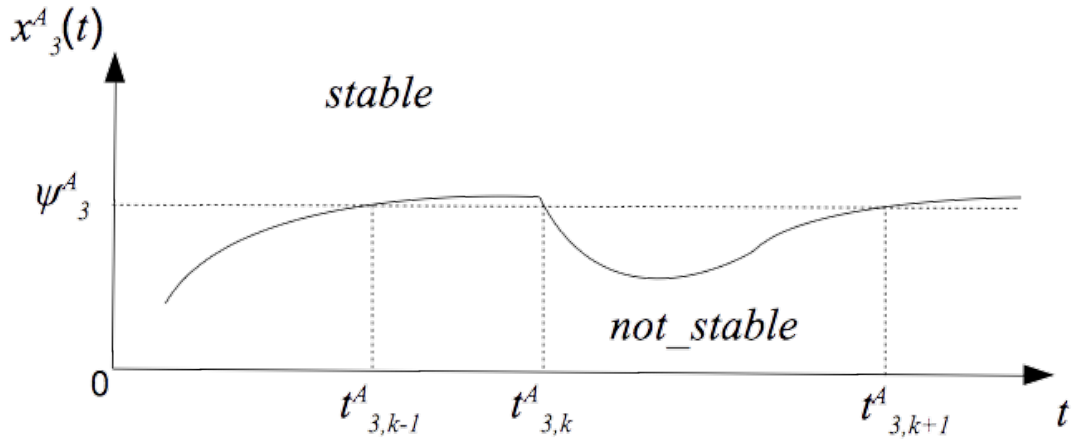


Figure 3.1: Spatial Discretization of the time function $x_3^A(t)$ with one threshold

Let us suppose that these two rules are implemented in two programs, respectively $\Theta_{31}^A(\{x_3^A\}, \{stable\})$ and $\Theta_{32}^A(\{x_3^A\}, \{not\_stable\})$. The program Alice, $\Theta_3^A(\{x_3^A\}, \{stable, not\_stable\})$, can be then considered as a combinaison of these programs:

$$\begin{gathered}\Theta_3^A(\{x_3^A\}, \{stable, not\_stable\}) = \\ \Theta_{31}^A(\{x_3^A\}, \{stable\}) \cup \Theta_{32}^A(\{x_3^A\}, \{not\_stable\})\end{gathered} \tag{3.25}$$

The observed process $(x_3^A(t), \Theta_3^A(\{x_3^A\}, \{stable, not\_stable\}))$ allows the program $\Theta_3^A$ to produce the following sequence $\omega_3^A$ of timed observations with the time function $x_3^A(t)$ of figure 3.1:

- $\omega_3^A = \{(stable, t_{3,k-1}^A), (not\_stable, t_{3,k}^A), (stable, t_{3,k+1}^A)\}$.

The sequence $\omega_3^A$ is the execution trace of the program $\Theta_3^A(\{x_3^A\}, \{not\_stable, stable\})$ when it observes the time function $x_3^A(t)$, representing the evolution over time of the engine's *market value*. According to the definition 3.2, such a sequence defines a set $\Gamma_3^A = \{t_{3,k-1}^A, t_{3,k}^A, t_{3,k+1}^A\}$ containing three timestamps. Because the time function is defined over $\Re$, the duration between two immediately following timestamps is random (i.e. $(t_{3,k+1}^A - t_{3,k}^A) \neq (t_{3,k}^A - t_{3,k-1}^A)$). $\Gamma_3^A$ is then a *stochastic clock*.

Since a predicate $\theta_i$ implemented in a program $\Theta_i(\{x_i\}, \{\delta_i\})$ can have a very complex meaning requiring a lot of computation, the general form of unary observer (i.e. a program implementing the spatial discretization principle) is based on the following rule:

$$\theta_i(x_i, \delta_j, t_k) \Rightarrow write(O(t_k)) \tag{3.26}$$

An important point at this stage is that the definition 3.2 of a timed observation shows that the link between a time function $x_i(t)$, a variable name $x_i$, the value of a constant $\delta_i$ and a timed observation $O(t_k)$ is made by the program $\Theta_i(\{x_i\}, \{\delta_i\})$ that implements an instantiation of the general rule 3.26:

- The relation between a time function $x_i(t)$ and a variable name $x_i$ is made through the application of the program $\Theta_i(\{x_i\}, \{\delta_i\})$ on $x_i(t)$.

- The variable name $x_i$ is defined for the predicate $\theta_i$: the function $x_i(t)$, the variable name $x_i$ and the constant $\delta_i$ are *independent*.

- The constant $\delta_i$ of a timed observation $O(t_k) \equiv (\delta_i, t_k)$ is not directly linked with the predicate $\theta_i$: $\delta_i$ is simply *linked* to $\theta_i$ by the *code*.

A constant $\delta_i$ is then clearly a sequence of characters that has no meaning in it-self: a pair $(\delta_i, t_k)$ is only a record in a database. To provide a meaning to this pair, the reader *must have* an *interpretation model*.

## 3.6 Semantic of a Timed Observation

As an illustration of this important point, let us consider the equations 3.23 and 3.24.

The constants "*stable*" and "*not_stable*" are respectively used in two different predicates $\theta_{31}^A$ and $\theta_{32}^A$ implemented in two programs $\Theta_{31}^A(\{x_3^A\}, \{stable\})$ and $\Theta_{32}^A(\{x_3^A\}, \{not\_stable\})$:

- $\theta_{31}^A$: $x_3^A(t_{3,k-1}^A) < \Psi_3^A \wedge x_3^A(t_{3,k}^A) \geq \Psi_3^A$;

- $\theta_{32}^A$: $x_3^A(t_{3,k-1}^A) \geq \Psi_3^A \wedge x_3^A(t_{3,k}^A) < \Psi_3^A$.

Each constant "*stable*" and "*not_stable*" has then one and only one specific meaning according to the TOT:

- $\theta_{31}^A$: *stable* means that the values of the time function $x_3^A(t)$ left the interval $]-\infty, \Psi_3^A[$ to enter the interval $[\Psi_3^A, +\infty[$;

- $\theta_{32}^A$: *not_stable* means that the values of the time function $x_3^A(t)$ left the interval $[\Psi_3^A, +\infty[$ to enter the interval $]-\infty, \Psi_3^A[$.

If the program $\Theta_{31}^A(\{x_3^A\}, \{stable\})$ has no error, a meaning to the sequence of characters "*stable*" can be provided: the values of the time function $x_3^A(t)$, representing the evolution over time of the engine's *market value*, is in the range *stable* corresponding to the interval $[\Psi_3^A, +\infty[$. In a same way, a meaning to the sequence of characters "*not_stable*" can be provided: the values of the time function $x_3^A(t)$, representing the evolution over time of the engine's *market value*, is in the range *not_stable* corresponding to the interval $]-\infty, \Psi_3^A[$. As a consequence, the two rules 3.23 and 3.24 define two ranges for the values of the time function $x_3^A(t)$:

- *stable*, range: $[\Psi_3^A, +\infty[$;

- *not_stable*, range: $]-\infty, \Psi_3^A[$.

This example shows a fundamental point that is highlighted by the TOT:

- *Without some knowledge about the program $\Theta_i$ that write it, a timed message contained in a database has no meaning in itself.*

So, the timed message "$t^A_{3,k+1}$, `it has a stable market value`" has no meaning because the program that has written this message has not been described. In other words, the sequence of characters *it has a stable market value* is a constant that can be rewritten $\delta^A_{31}$ without changing anything.

But if it is known that Figure 3.1 illustrates the engine's *market value* described by Alice then a particular meaning can be given to the timed message "$t^A_{3,k+1}$, `it has a stable market value`" and so to the corresponding timed observation $O(t^A_{3,k+1}) \equiv (\delta^A_{31}, t^A_{3,k+1}) =$ (`it has a stable market value,` $t^A_{3,k+1}$):

- The time function $x^A_3(t)$ represents the engine's *market value.*
- The sequence of characters *stable* can be associated with the rule 3.23 and then with the predicate $\theta^A_{31}$ (i.e. $x^A_3(t^A_{3,k-1}) < \Psi^A_3 \wedge x^A_3(t^A_{3,k}) \geq \Psi^A_3$);
- The variable name $x^A_3$ denotes the values of $x^A_3(t)$ at the timestamps $t^A_{3,k-1}$, $t^A_{3,k}$ and $t^A_{3,k+1}$.

As a consequence, the sequence of characters *it has a stable market value* means that the predicate $\theta^A_{31}$ has been satisfied at timestamps $t^A_{3,k-1}$ and $t^A_{3,k+1}$. But when no knowledge is available about the way a pair $(\delta, t_k)$ has been recorded in a database, the definition 3.2 allows to infer that:

**Theorem 3.1** *Interpretation*
*Given a timed observation $O(t_k) \equiv (\delta, t_k)$, the following propositions are true:*

1. *$O(t_k)$ has been written by an abstract program $\Theta(\{x\}, \{\delta\})$ that defines a ternary predicate $\theta(x_\theta, \delta_\theta, t_\theta)$.*
   *Proof of proposition 1: directly results from definition 3.2.*
2. *The meaning of $O(t_k) \equiv (\delta, t_k)$ is the assignation $\theta(x, \delta, t_k)$.*
   *Proof of proposition 2: also directly results from definition 3.2.*
3. *At time $t = t_k$, the time function $x(t)$ satisfies the constraints of the predicate $\theta(x_\theta, \delta_\theta, t_\theta)$.*
   *Proof of proposition 3 is an immediate consequence of the property 1 of an observation function: any program $\Theta(\{x\}, \Delta)$ is characterized by an observation function $\delta_x$. As a consequence, such an observation function can be any predicate of the form $\theta(x_\theta, \delta_\theta, t_\theta)$.*
4. *There exists a timed function $x(t)$ which has been observed by the abstract program $\Theta(\{x\}, \{\delta\})$.*
   *Proof of proposition 4: Program $\Theta(\{x\}, \{\delta\})$ is, by hypothesis (see section 3.4), an safe canonical program. According to definition 3.4, there exists then a timed function $x(t)$ which has been observed by such a program.*

When considering the Spatial Discretization Principle (cf. equation 3.22), the assignation $\theta(x, \delta, t_k)$ can have one of the three following interpretations:

- $EQUAL(x, \delta, t_k)$: "At time $t_k$, $x$ **is equal to** $\delta$";
- $IS(x, \delta, t_k)$: "At time $t_k$, $x$ **is** $\delta$";
- $BELONGS(x, \delta, t_k)$: "At time $t_k$, the values of $x(t)$ **belongs to** a range denoted $\delta$".

These three interpretations are clearly misuses of language because $\delta$ is a constant taken from an arbitrary made set $\Delta$ and the definition domain of the function $x(t)$ is the set $\Re$ of the real numbers. For example, according to the theorem 3.1, the timed observation (`it has a stable market value,` $t^A_{3,k+1}$) can be interpreted as :

1. *At time* $t^A_{3,k+1}$*, engine's market value is equal to stable*;
2. *At time* $t^A_{3,k+1}$*, engine's market value is stable*;
3. *At time* $t^A_{3,k+1}$*, engine's market value belongs to the stable range.*

The first interpretation is the most usually used because it corresponds to the abuse of language $x^A_3(t^A_{3,k+1}) = stable$ which has the form of a classical formula:

$$x(t_k) = \delta \tag{3.27}$$

Generally speaking, in practice, a predicate $\theta(x_\theta, \delta_\theta, t_\theta)$ is satisfied when the time function $x(t)$ matches against a behavior model [LeG04] that can be as simple as the switch of an interrupter or, requiring complex techniques, such as signal processing techniques for artificial vision. The precise meaning of such a predicate can be very complex and very difficult to detail. This explains why most experts commonly use the abuse of language of the equation 3.27.

It is noteworthy that a program can have errors: $(\delta, t_k)$ could be written in a database from the assertion $\theta(x, \delta, t_k)$ although the time function $x(t)$ not "really matches" the semantic of this predicate.

## 3.7 Observation Class

When considering a timed observation $O(t_k) \equiv (\delta, t_k)$, the first interpretation (i.e. the equation 3.27) explains the fact that an expert establishes immediately (and often unconsciously) a relation between the constant $\delta$ and a variable name $x$.

This cognitive phenomena being so important and so natural, the TOT defines it with the notion *observation class*:

**Definition 3.6** *Observation Class*
*Let* $X(t) = \{x_i(t)\}_{i=1...n}$ *be a set of time functions that are observed by an abstract program* $\Theta(X, \Delta)$ *where* $\Delta = \{\delta_j\}_{j=1...m}$ *is the set of all the constants the abstract program can use and* $X = \{x_i\}_{i=1...n}$ *is the set of variable names corresponding to* $X(t)$*.*

$\forall i \in [1, n]$, $\forall j \in [1, m]$ *and* $\forall k \in \mathbb{N}$*, an observation class* $O_k = \{..., (x_i, \delta_j), ...\}$ *is a subset of* $X \times \Delta$*.*

An observation class is then any set of pairs $(x_i, \delta_j)$ associating a variable name $x_i$ with a constant $\delta_j$. Such a definition establishes an *explicit* link between a constant and a variable name. Any association can be made, but the simplest way, and the most used, is to associate a variable $x_i$ to each constant $\delta_j$ (i.e. establishing a mapping $\delta_j \mapsto x_i$ for each $\delta_i \in \Delta$) and to define all the observation classes with singletons $O_j = \{(x_i, \delta_j)\}$, that is to say where the pair $(x_i, \delta_j)$ is the unique element the set $O_j$. In that case, the following definition can be applied:

**Definition 3.7** *Class Occurrences*
*Let $\Delta = \{\delta_j\}_{j=1...m}$ be a set of $m$ constants $\delta_i$; let $X = \{x_i\}_{i=1...n}$ be a set of $n$ variable names $x_i$ so that $n \leq m$; let $O = \{O_j\}_{j=1...m}$ be a set of $m$ singletons $O_j = \{(x_i, \delta_j)\}$.*

*Any timed observation $O(t_k) \equiv (\delta_j, t_k)$ written by a program $\Theta(X, \Delta)$ is an occurrence of an observation class $O_j = \{(x_i, \delta_j)\}$.*

This definition and the theorem 3.1 lead to define a mapping from the set of constants $\Delta$ to the set of variable names $X$. This simplifies strongly the situation when the variable names are unknown: it is always possible to map an abstract variable $\phi_j$ to each constant $\delta_j$ that appears in a sequence $\omega$ of timed observations (i.e. in an extraction of a database). This is done with the construction of a set $O = \{O_i\}$ of observation classes $O_i = \{(\phi_i, \delta_i)\}$ where each $O_i$ is a singleton. In that case:

$$O(t_k) \equiv (\delta_i, t_k) \equiv O_i(t_k) \tag{3.28}$$

In other words, when defining observation classes as singletons, a program $\Theta(X, \Delta)$ observing a process $X(t)$ writes occurrences $O_j(t_k)$ of observation classes $O_j$ and the equation 3.26 can then be written in its most abstract form:

$$\theta_i(x_i, \delta_j, t_k) \Rightarrow write(O_j(t_k)) \tag{3.29}$$

For example, the program $\Theta_3^A(\{x_3^A\}, \{stable, not\_stable\})$ associated with the time function $x_3^A(t)$ of figure 3.1 produces the sequence $\omega_3^A$ of timed observations $\omega_3^A = ((stable, t_{3,k-1}^A), (not\_stable, t_{3,k}^A), (stable, t_{3,k+1}^A))$. This sequence allows to define the set $O_3^A$ of observation classes containing the following classes:

- $O_{31}^A = \{(x_3^A, stable)\}$
- $O_{32}^A = \{(x_3^A, not\_stable)\}$

In that case, the preceding sequence can then be written:

- $\omega_3^A = \{O_{31}^A(t_{3,k-1}^A), O_{32}^A(t_{3,k}^A), O_{31}^A(t_{3,k+1}^A)\}$.

This example shows the following important point of the TOT. The definition 3.6 allows to partition the sequence $\omega_3^A$ in two sequences, each of them being associated with an observation class:

- $\omega_{31}^A = \{O_{31}^A(t_{3,k-1}^A), O_{31}^A(t_{3,k+1}^A)\}$, $\omega_{32}^A = \{O_{32}^A(t_{3,k}^A)\}$;
- $\omega_3^A = \omega_{31}^A \cup \omega_{32}^A$;
- $\omega_{31}^A \cap \omega_{32}^A = \emptyset$.

As a consequence, the set $\Gamma_3^A = \{t_{3,k-1}^A, t_{3,k}^A, t_{3,k+1}^A\}$ is also decomposed in two disjoints sets, each of them constituting a stochastic clock:

- $\Gamma_{31}^A = \{t_{3,k-1}^A, t_{3,k+1}^A\}$, $\Gamma_{32}^A = \{t_{3,k}^A\}$;
- $\Gamma_3^A = \Gamma_{31}^A \cup \Gamma_{32}^A$;

- $\Gamma_{31}^A \cap \Gamma_{32}^A = \emptyset$.

This shows that the definition of a set $O$ of observation classes decomposes a given sequence $\omega$ of timed observations $O(t_k)$ in a *superposition* of sequences $\omega_i$, each of them being associated with a particular observation classe $O_i$:

- $O_{31}^A = \{(x_3^A, stable\} \Rightarrow \Theta_{31}^A(O_{31}^A), \omega_{31}^A, \Gamma_{31}^A$;
- $O_{32}^A = \{(x_3^A, not\_stable\} \Rightarrow \Theta_{32}^A(O_{32}^A), \omega_{32}^A, \Gamma_{32}^A$.

The notion of observation class facilitates then the interpretation and the filtering of a given sequence $\omega$ whatever is the program $\Theta(X, \Delta)$. The next section shows that this notion provides also a powerful tool to model an observed process $(X(t), \Theta(X, \Delta))$.

## 3.8 Superposition Theorem

The concept of *abstract binary observer* is the core of the TOT (cf. ([LeG06] for a complete description and [VLGBR16] for a concrete usage). The concept of *abstract binary observer* is directly linked with the following *Superposition Theorem*:

**Theorem 3.2** *Superposition Theorem*
*If a program $\Theta(X, \Delta)$ is independent (that is to say it implements a memoryless observation function so the constants $\delta_i$ of $\Delta$ are independent), then any partition $\cup_{i=1...n}\ \Delta_i$ of $\Delta$ so that $\forall i \neq j,\ \Delta_i \cap \Delta_j = \emptyset$ decomposes the program $\Theta(X, \Delta)$ in a superposition of n independent programs $\Theta_i(X_i, \Delta_i)$ so that:*

1. $(\Delta = \bigcup_{i=1...n} \Delta_i) \Rightarrow X = \bigcup_{i=1...n} X_i$
2. $((\Delta = \bigcup_{i=1...n} \Delta_i) \wedge (X = \bigcup_{i=1...n} X_i)) \Rightarrow \Theta(X, \Delta) = \bigcup_{i=1...n} \Theta_i(X_i, \Delta_i)$
3. $(\Theta(X, \Delta) = \bigcup_{i=1...n} \Theta_i(X_i, \Delta_i)\}) \Rightarrow (X(t), \Theta(X, \Delta)) = \bigcup_{i=1...n} (X_i(t), \Theta_i(X_i, \Delta_i))$

*Proof of theorem 3.2:*
*Let us suppose that there exists a program $\Theta_i(X_i, \Delta_i)$ which is not independent. Such a program generates then timed observations $O(t_i) \equiv (\delta_i, t_i)$ where constants $\delta_i$ are not independent. Let us denote $\Omega_i$, the sequence of timed observations produced by the program $\Theta_i(X_i, \Delta_i)$. Any superposition $\Omega = \Omega_1 \cup ... \cup \Omega_i \cup ... \cup \Omega_n$ contains then timed observations whose constants are not independent. Such a superposition can be considered to have been made by a program denoted $\Theta(\{X_1, ..., X_i, ..., X_n\}, \Delta_1 \cup ...\Delta_i ... \cup ...\Delta_n)$ which superimposes timed observations made by programs $\Theta(X_1, \Delta_1)$, ..., $\Theta(X_i, \Delta_i)$, ..., $\Theta(X_n, \Delta_n)$. Thus, by construction, the observation function $\delta_{X_1...X_i...X_n}$ of the program $\Theta(\{X_1, ..., X_i, ..., X_n\}, \Delta_1 \cup ...\Delta_i ... \cup ...\Delta_n)$ is not memoryless and then such a program is not independent. We have then demonstrated that:*

$$\exists i, \Theta_i(X_i, \Delta_i) \text{ is not independent} \Rightarrow \Theta(\{X_1, ..., X_i, ..., X_n\}, \Delta_1 \cup ...\Delta_i ... \cup ...\Delta_n) \text{ is not independent} \tag{3.30}$$

*The contraposition of 3.30 is:*

$$\Theta(\{X_1, ..., X_i, ..., X_n\}, \Delta_1 \cup ...\Delta_i ... \cup ...\Delta_n) \text{ independent} \Rightarrow \forall i, \Theta_i(X_i, \Delta_i) \text{ are independent} \tag{3.31}$$

*Equation 3.31 demonstrates theorem 3.2.*

In other words, the partitioning of the set $\Delta$ in $n$ disjoint sets $\Delta_i$ transforms a program $\Theta(X, \Delta)$ in a superposition of $n$ independent programs $\Theta_i(X_i, \Delta_i)$. Such a partition is made with the definition of a set $O = \{O_i\}_{i=1...n}$ so that:

$$\Theta(X, \Delta) = \bigcup_{i=1...n} \Theta_i(O_i) \tag{3.32}$$

It is important to note that the theorem 3.2 is only based on an *adequate partition* of the set $\Delta$ of constants and concerns only the program $\Theta$: no hypothesis is made about the dynamics of the process $X(t)$.

The equation 3.32 and the theorem 3.2 mean that, given an adequate set $O = \{O_i\}_{i=1...n}$, any observed process $(X(t), \Theta(X, \Delta))$ can be seen as a *network* of observed processes $\bigcup_i (X_i(t), \Theta_i(O_i))$. The term *network* is used because the partitioning of $\Delta$ does not entail the partition of the process $X(t)$: it does not matter that the subsets $X_i$ are or aren't disjoint. In other words, the observed processes $(X_i(t), \Theta_i(O_i))$ can *share* some time function $x_i(t)$.

The application of these theorems is very simple. For example, considering collectively the theorems 3.7 and 3.2, each observation class $O_i$ can be defined as a singleton so that each $\delta_i$ is associated with one and only one variable $x_i$ (a variable name can be associated with multiple constants). In that case, a program $\Theta(X, \Delta)$ where $\Delta$ contains $n$ constants $\delta_i$ can be considered as a superposition of $n$ independent memoryless programs $\Theta_i(O_i)$ (cf. equation 3.32). This way of defining the set $O$ is very current in practice. And when the variable $x_i$ is not known, an abstract variable $\phi_i$ can be used.

More generally, the importance of the Superposition Theorem comes from the fact it allows to describe recursively any complex observed process $(X(t), \Theta(X, \Delta))$ as a network of observed processes $\bigcup_i (X_i, \Theta_i(X_i, \Delta_i))$. This property is very important to diagnose a complex observed process $(X(t), \Theta(X, \Delta))$.

## 3.9 Temporal Binary Relation

This section aims at defining the concepts of Temporal Binary Relation and Observed Relation.

**Definition 3.8** *Temporal Binary Relation*
*A temporal binary relation $r(O_i, O_j, [\tau_{ij}^-, \tau_{ij}^+])$, $\tau_{ij}^- \in \Re$, $\tau_{ij}^+ \in \Re$, is an oriented (sequential) relation between two observation classes $O_i$ and $O_j$ that are timed constrained with the $[\tau_{ij}^-, \tau_{ij}^+]$ interval.*

The temporal constraint $[\tau_{ij}^-, \tau_{ij}^+]$ of a temporal binary relation $r(O_i, O_j, [\tau_{ij}^-, \tau_{ij}^+])$ is the time interval for observing an occurrence $O_j(t_{k_j})$ of the "output" observation class $O_j$ after the observation of an occurrence $O_i(t_{k_i})$ of the "input" observation class $O_i$:

**Definition 3.9** *Observed Relation*
*Let the couple $(X(t), \Theta(X, \Delta))$ be an observed process defining a particular set $O = \{O_i\}$ of $m$ observation classes containing two classes $O_i$ and $O_j$; let $\omega = \{..., O_l(t_k), ...\}$, $t_k \in \Gamma \subseteq \Re$, $k = 0...n-1$, $l = 0...m-1$, be a sequence of $n$ timed observations $O_l(t_k)$ provided by $(X(t), \Theta(X, \Delta))$.*

*A temporal binary relation* $r(O_i, O_j, [\tau_{ij}^-, \tau_{ij}^+])$ *between two classes* $O_i$ *and* $O_j$ *is said to be observed in* $\omega$ *if there is at least two timed observations* $O_i(t_{k_i})$ *and* $O_j(t_{k_j})$ *so that* $t_{k_j} - t_{k_i}$ *satisfies the timed constraint* $[\tau_{ij}^-, \tau_{ij}^+]$ *of* $r(O_i, O_j, [\tau_{ij}^-, \tau_{ij}^+])$.

Formally, the relation $r(O_i, O_j, [\tau_{ij}^-, \tau_{ij}^+])$ is observed if and only if:

$$r(O_i, O_j, [\tau_{ij}^-, \tau_{ij}^+]) \Leftrightarrow \exists O_i(t_{k_i}) \in \omega, \exists O_j(t_{k_j}) \in \omega,\ t_{k_j} - t_{k_i} \in [\tau_{ij}^-, \tau_{ij}^+] \tag{3.33}$$

A temporal binary relation of the form $r(O_i, O_j, ]0, +\infty[)$ is a purely sequential binary relation: to be observed, the occurrence $O_j(t_{k_j})$ must succeed the occurrence $O_i(t_{k_i})$ (i.e. $t_{k_j} > t_{k_i}$). For simplicity reasons, such a sequential binary relation is denoted $r(O_i, O_j)$:

$$r(O_i, O_j, ]0, +\infty[) \equiv r(O_i, O_j) \tag{3.34}$$

## 3.10 Abstract Chronicle Model

Nevertheless, the definition of a temporal binary relation $r(O_i, O_j, [\tau_{ij}^-, \tau_{ij}^+])$ (cf. definition 3.8) is the basis of the notion of *Abstract Chronicle Model*:

**Definition 3.10** *Abstract Chronicle Model*
*Any arbitrarily made set* $M = \{r_k(O_i, O_j, [\tau_{ij}^-, \tau_{ij}^+])\}_{k=1...n}$ *of* $n$ *temporal binary relations* $r_k(\ O_i,\ O_j,\ [\tau_{ij}^-, \tau_{ij}^+])$ *is an abstract chronicle model.*

The abstract chronicle models of the TOT framework are represented with a graphical knowledge representation language called "ELP" for "Event Language for Process behavior modeling" [BLGC08].

A particular sequence $\omega_i = \{O(t_k)\}_{k=0...n-1}$ of $n$ timed observations that is consistent with the logical and the timed constraints of a given abstract chronicle model $M$ is called an *instance* of $M$. For example, let us consider the following abstract chronicle model $M_{123} = \{\ r_{12}(O_1, O_2, [0, 5]), r_{23}(O_2, O_3, [3, 8])\}$. The sequence $\omega_i = \{O_1(1), O_4(3), O_2(4), O_1(8), O_3(10)\}$ is an instance of $M_{123}$ because $\omega_i$ contains the occurrences $O_1(1)$, $O_2(4)$ and $O_3(10)$ satisfying the logical and the timed constraints of $M_{123}$:

- $O_1(1)$ and $O_2(4)$ satisfy the logical condition of the relation $r_{12}(O_1, O_2, [0, 5])$ (i.e. the observation class of $O_1(1)$ is $O_1$ (resp. $O_2$ for $O_2(4)$).
- $O_1(1)$ and $O_2(4)$ satisfy the temporal condition of the relation $r_{12}(O_1, O_2, [0, 5])$ (i.e. $4-1 = 3,\ 3 \in [0, 5]$).
- $O_2(4)$ and $O_3(10)$ satisfy the logical condition of the relation $r_{23}(O_2, O_3, [3, 8])$ (i.e. the observation class of $O_2(4)$ is $O_2$ (resp. $O_3$ for $O_3(10)$).
- $O_2(4)$ and $O_3(10)$ satisfy the temporal condition of the relation $r_{23}(O_2, O_3, [3, 8])$ (i.e. $10 - 4 = 6,\ 6 \in [3, 8]$).

The notions of *abstract chronicle model* and *instance of model* are of the most interest for the diagnosis of an observed process $(X(t), \Theta(X, \Delta))$. Specifically, a particular type of abstract chronicle models, called a *path*, plays an important role:

**Definition 3.11** *Path*
*An abstract chronicle model $M$ made with a suite of $n-1$ timed binary relations $M = \{ r_1(\ O_i,\ O_{i+1},\ [\tau_1^-, \tau_1^+]),\ r_2(\ O_{i+1},\ O_{i+2},\ [\tau_2^-, \tau_2^+])\ ,...,\ r_n(\ O_{i+n-1},\ O_{i+n},\ [\tau_n^-, \tau_n^+])\}$ is a path.*

The $M_{123}$ abstract chronicle model, for example, is a path. More generally, a set $P = \{..., p_i, ...\}$ of $n_P$ pathes $p_i = \{..., r^i_{ni}(O_{k-1}, O_k, [\tau_k^-, \tau_k^+]), ...\}$ where the last relation $r^i_{ni}(\ O_{k-1},\ O_k,\ [\tau_k^-, \tau_k^+])$ of each path $p_i$, except one, is a relation contained in another path $p_j$ of $P$ is also called a path because in that case, $P$ constitutes a kind of *path of paths*. Graphically represented, $P$ is a chained list of observation classes (cf. figure 3.2 for example, where the classes are the ellipses denoted with a number).

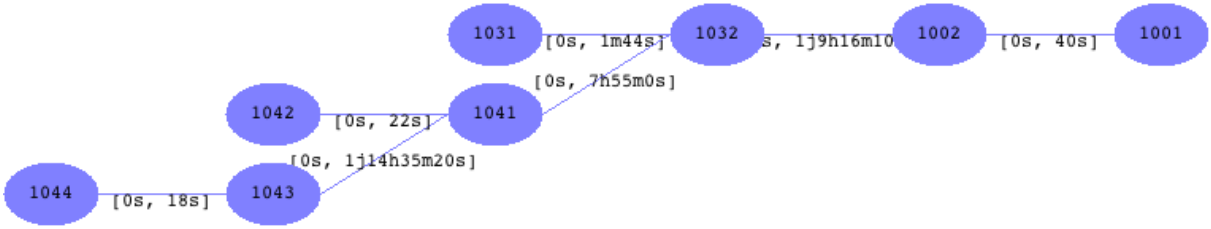


Figure 3.2: Example of a path made with 3 paths

A particular (set of) path $P$ constitutes a specific structure that allows the *reading* of a given sequence $\omega_i$ of timed observations $O(t_{k_i})$: if $\omega_i$ satisfies the logical and the temporal constraints of the suite of relations $r_j(O_{j-1}, O_j, [\tau_j^-, \tau_j^+])$, $\forall j$, of a given path $p_i$ ($p_i \in P$), then $\omega_i$ is an instance of the path $p_i$. In that case, the path $p_i$ constitutes an *interpretation structure* of the sequence $\omega_i$. This interpretation structure is like a *synopsis* of the *narrative structure* that is the sequence $\omega_i$. Inversely, $\omega_i$ is like a *scenario* (or the *story*) that *must* satisfy the logical and the temporal constraints of the path $p_i$.

## 3.11 Modeling with the TOT

Technically, the timed observations $O(t_{k_i})$ of a sequence $\omega_i$ constitute an *information flow*. A path $p_i$ is then a *representation* of the knowledge that is necessary to *interpret* the timed observations $O(t_{k_i})$ of $\omega_i$.

Generally speaking, knowledge results of the interaction between an information flow and an arbitrary purpose. This interaction is assumed by humans which define their purpose according to their own expectations [Non94], [Non91] and [AL01]. Information comes from all the possible sources: believes, observations, experimentation, scientific axioms, sensors, etc [Pol66], [NK98] and [SBF98]. The interaction is basically an interpretation of the information flow that traverses a thinking human [Dam05] and [Dam99].

To define the modeling principles of the TOT, the following operational definition of *knowledge* will be used:

**Definition 3.12** *Operational Notion of Knowledge*
*Knowledge results from an intentional interpretation of a flow of information.*

This definition establishes a relation between knowledge, information and a purpose (an intention). The purpose is always defined by humans: in the framework of the TOT, the purpose

is implemented in an observer program $\Theta(X, \Delta)$ which can be either executed by a human or a computer. Considering the diagnosis task of a dynamic process, the purpose is typically the assessment of a fault linked with the occurrence of an undesired behavior.

## 3.12 Model according to the TOT

The fundamental role of a model is the sharing of knowledge between humans. This sharing is facilitated through the mediation of signs belonging to a particular set (often called alphabet). These signs have no meaning in themselves but are necessary to represent knowledge in order to share a common understanding of an observed set of phenomena.

As a consequence, a model is made with a particular arrangement of signs: the meaning results precisely of the specific arrangement the modeler choose to share its knowledge. The representation of a knowledge corpus requires then a set of rules that defines the authorized arrangements (i.e. a grammar).

This leads to define the notion of model according to the TOT:

**Definition 3.13** *Model according to the TOT*
*A model is an organized set of knowledge representations.*

It is clear that the organization of the knowledge representations within a model is of the main importance. The TOT being concerned with the evolutions of an observed process over time, the knowledge under consideration is linked with the relations between functions of time $x_i(t)$, the constants $\delta_i$ and the stochastic clocks $\Gamma_i$=$\{t_{k_i}\}$, $t_{k_i} \in R$, $k_i \in N$. The TOT organizes these relations around the notion of variable $x_i$ (cf. figure 3.3).

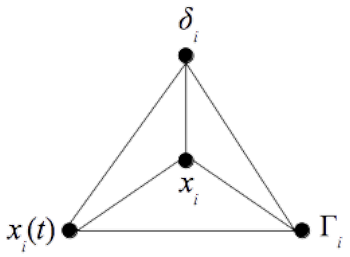


Figure 3.3: Relations between the Basic Objects of the TOT

A piece of knowledge belongs then to three fundamental categories:

1. Relations between the functions $x_i(t)$ of a process $X(t)$. This category of knowledge is called the "structural knowledge" because, in the framework of the TOT, the functions $x_i(t)$ are the constituents (i.e. the components) of a process $X(t)$.

2. Relations between the constants $\delta_i$ of the set $\Delta$ used by a program $\Theta$ to describe the evolutions of the functions $x_i(t)$. This category is called the "functional knowledge" because the relations between the constants $\delta_i$ can be represented with logical rules linking together subsets $\Delta_i$ of constants of $\Delta$ and so, specifies abstract mathematical functions under the form of "tables of values".

3. Relations between the stochastic clocks $\Gamma_i$. This category is called the "behavioral knowledge" because these relations describe the links between the evolutions of the functions $x_i(t)$ of a process $X(t)$. This type of knowledge is directly linked with the set $O = \{O_i\}_{i=1..n_c}$ of observation classes.

As figure 3.3 aims at showing, these three categories of relations are linked together: a specific set $\Delta$ of constant $\delta_i$ will lead an observer program $\Theta$ to describe the evolution of a process $X(t)$ with a particular set $\Gamma$ of timestamps. The role of the concept of *variable* in the TOT framework is to provide the mean to analyze the consistency of these three categories of knowledge about a process $X(t)$.

So, the concept of variable defines a supplementary category of knowledge, which is a kind of *meta-knowledge* (i.e. the modeling point of view), that fundamentally defines the way a dynamic process $X(t)$ is perceived by humans (i.e. the *perception knowledge*). In practice, this leads to the following definition of the aim of the modeling activity:

**Definition 3.14** *Modeling activity according to the TOT*
*The modeling activity of a dynamic process $X(t)$ aims at representing the elicited knowledge according to a formalism and at distributing the knowledge representations over three models, the structural, the functional and the behavioral model, according to a definition of a particular set of variables $X$.*

By construction, a particular set of variables $X$ is a subset of all the variables that can be defined about a dynamic process. The only rational way to specify $X$ is precisely the modeling purpose: are only required the variables that play a role in the modeling purpose (i.e. the diagnosis task for example). The other variables can be forgotten (at least in a first step). This set of variables $X$ defines then the process according to modeling purpose, and so fixes the abstraction level of the model.

## 3.13 The TOT Modeling Principles

This chapter introduces the *necessary* and *sufficient* mathematical tools of the TOT that are required to define the modeling framework of our contributions.

These tools can be sum up through the five modeling principles of the TOT:

1. **Variable localization.**
   A time function $x_i(t)$ is a *signal* provided by a *sensor* located at a particular place defined as a *component*. So, a function $x_i(t)$ specifies a variable $x_i$, a component $c_i$ and a binary relation that associate $x_i$ to $c_i$. As a consequence, a variable $x_i$ is always associated with a sensor that is physically located on a component $c_i$. In other words, any variable $x_i$ of X must be associated with one and only one component $c_i$.

2. **Multi-valued variable.**
   A variable $x_i$ is necessarily defined over a set $\Delta_{x_i}$ of possible values containing at least two elements. This means that when the experts' knowledge defines only one value $\delta_i$ for a variable $x_i(t)$ , the knowledge engineer must introduce in $\Delta_{x_i}$ another constant, denoted

$\delta_j$ for example, meaning "not $\delta_i$" (i.e. $\Delta_{x_i} = \{\delta_i, \delta_j\}$ and $\delta_j \equiv \neg\ \delta_i$). This principle is a direct consequence of the spatial segmentation of the TOT (cf. Figure 3.3).

3. **Discernible state.**
   According to the interpretation 3.27, an occurrence $O_i(t_k)$ of an observation class $O_i$ corresponds to the assignment of a value $\delta_i$ to a variable $x_i$. Such an assignation results necessarily of an *observable* modification in the dynamic process $X(t)$. So two occurrences $O_i(t_k)$ and $O_j(t_{k+1})$ mark a *discernible state transition* in an observed process $(X(t), \Theta(X, \Delta))$. This means that a temporal binary relation $r(O_i, O_j, [\tau_{ij}^-, \tau_{ij}^+])$ defines a particular *discernible state*.

4. **Separation of knowledge of different nature.**
   Since the TOT defines four categories of knowledge (structural, functional, behavioral and perception), four models will contain a specific category of knowledge representations: a *Structural Model SM* will contain all the structural knowledge, a *Functional Model FM* will contain all the functional knowledge, a *Behavioral Model BM* will contain all the behavioral knowledge, and a *Perception Model PM* will contain the perception knowledge. This constitutes the multi-modeling framework of the TOT [CGTT93], [CR99] and [ZGF06a].

5. **Symbol driven modeling.**
   The knowledge interpretation aims at identifying the minimal set of symbols denoting a time function $x_i(t)$, a variable $x_i$ or a constant $\delta_i$ and the minimal set of relations between them (cf. figure 3.3). The logical properties coming from these minimal sets are necessary and sufficient to complete the model. Among other meanings, this principle means that the introduction of a symbol that is not associated with an element of the domain knowledge is prohibited.

The *Discernible state* principle is particularly important according to the notion of *Behavioral Model BM*. A *discernible state* is a property of the *model* of an observed process $(X(t), \Theta(X, \Delta))$:

**Definition 3.15** *Discernible State*
*An arbitrary made set* $\{r_k(O_{i_k}, O_{j_k})\}_{k=1\ldots n_k}$, $\forall k, i_k \neq j_k$, *of* $n_k$ *observed relations of a temporal binary relation* $r(O_i, O_j, [\tau_{ij}^-, \tau_{ij}^+])$ *defines a discernible state.*

Clearly, the Tom4D notion of *discernible state* is directly linked with the TOT notion of *variable*: if a *constant* $\delta_i$ is assigned to a *variable* $x_i$, then the *only reason* is that something happens in the observed process. In other words, the observed process moves from a *discernible state* to another so that the *variable* $x_i$ has a new value, the *constant* $\delta_i$. It is important to note that *what happens during the discernible state transition is not known* but it is *certain that something happens*: this is role of the Tom4D notion of *discernible state* to denote this *certainty.*

This notion is then conceptually different to the classical *state* notion of the Discrete Event System (DES) domain where *a state represents a property of the process* $X(t)$ itself.

## 3.14 Conclusion: about Abstraction Level

The aim of a modeling framework is to provide the tools allowing the building of a model that:

1. resides at a particular level of abstraction,

2. is logically coherent (i.e. contains no contradiction), and

3. is as complete as possible.

These goals are given in the order of their importance: clearly, providing a coherent model at the right abstraction level is the main modeling law to apply the five TOT modeling principles, its completeness being desired but does not constitute a primary condition ([New81]).

Within the mathematical framework of the TOT, an abstraction level is defined by the observed process $(X(t), \Theta(X, \Delta))$ itself: no constraint is made about the way the sets $X(t)$ and $\Delta$ are made (the TOT imposes only two constraints on the *program* $\Theta$). In other words, the definition of a particular set $X$ of *variables* and a set $\Delta$ of *constants* constitutes the *core* of the modeling process.

The definition of a recursive abstraction reasoning must then rely on the TOT's notions of *variables* and *constants*. To this aim, the next step is to define a sampling device that is coherent with the *Spatial Discretization Principle* of the TOT, the principle that assigns a *constant* to a *variable* when the *observed process* moves from a *discernible state* to another.

CHAPTER 4

# Composition of Observers

## 4.1 Introduction

This chapter aims at introducing the concept of *composition of observers* and of *Abstract Binary Observers*. Both concepts are based on the notion of addition under temporal constraints of timed observation whose a complete description can be found in [LeG06].

This reasoning chosen in this chapter has the following steps:

1. introduction of the neutral observation and timestamps and constants observations which provide the basic elements to define the addition under temporal constraints of timed observations;
2. deduction of an assignation from assignations which is the basic process of the abstraction in the TOT framework;
3. addition of timed observations which is built from the deduction of an assignation from two assignations;
4. composition of observers which is built from the addition of two timed observations and allows to design observers producing observations from other observations;
5. abstract unary and binary observers which allow to provide an abstract structure to any sequence of timed observations without having any knowledge about such sequences or about concrete observers that have produced them.

## 4.2 Neutral Observation, Observation of a Timestamp, Observation of a Constant

Let us consider the observed process $(X(t), \Theta(X, \Delta)$ where $X(t) = \{x_i(t)\}_{i=1...n}$ is a set of timed functions. Let us recall here (see definition 3.2) that a timed observation $O(t_k) \equiv (\delta, t_k) \in \Delta \times \Gamma$ made on a timed function $x_i(t)$ is the assignation of values $x_\theta = x_i$, $\delta_\theta = \delta$ and $t_\theta = t_k$ to the predicate $\theta(x_\theta, \delta_\theta, t_\theta)$, implemented in program $\Theta(X, \Delta)$ such as $\theta(x_i, \delta, t_k)$. Technically, timed observation $O(t_k) \equiv (\delta, t_k)$ denotes a record in the database whereas the assigned predicate $\theta(x_i, \delta, t_k)$ represents the meaning or the interpretation of such a timed observation.

**Definition 4.1** *Neutral Observation*

*A neutral observation denoted* $O_\phi(t_\phi) \equiv (\phi, t_\phi)$ *is the assignation of values* $x_\Phi = x_\phi$, $\delta_\Phi = \phi$ *and* $t_\Phi = t_\phi$ *to the predicate* $\Phi(x_\Phi, \delta_\Phi, t_\Phi)$ *such as* $\Phi(x_\phi, \phi, t_\phi)$ *where:*

- $x_\phi$ *is a variable name of an undefined timed function* $x_\phi(t)$ *which does not belong to the set* $X(t)$ *of timed functions observed by the program* $\Theta(X, \Delta)$*;*
- $\phi$ *is a symbolic constant which does not belong to the set* $\Delta$*;*
- $t_\phi$ *is a symbolic timestamps which respects any temporal constraints* $\Delta\tau_{ij} \subset \Re$*:*

$$\forall t \in \Re, \forall \Delta\tau_{ij} \equiv [\tau_{ij}^-; \tau_{ij}^+] \subset \Re, t - t_\phi \in \Delta\tau_{ij} \tag{4.1}$$

- $\Phi(x_\Phi, \delta_\Phi, t_\Phi)$ *is the neutral predicate such as:*

$$\begin{gathered} \exists \Phi(x_\phi, \phi, t_\phi), \forall \theta(x_i, \delta, t_k), \\ \Phi(x_\phi, \phi, t_\phi) \wedge \theta(x_i, \delta, t_k) \Leftrightarrow \\ \theta(x_i, \delta, t_k) \wedge \Phi(x_\phi, \phi, t_\phi) \Leftrightarrow \\ \theta(x_i, \delta, t_k) \end{gathered} \tag{4.2}$$

Elements are now in place to give the definitions of the observation of a timestamp and the observation of a constant.

**Definition 4.2** *Observation of a Timestamp*
*The observation of a timestamp, denoted* $d(k) \equiv (\phi, k)$*, is the assignation of values* $x_{\theta_d} = x_\phi$, $\delta_{\theta_d} = \phi$ *and* $t_{\theta_d} = t_k$ *to the predicate* $\theta_d(x_{\theta_d}, \delta_{\theta_d}, t_{\theta_d})$ *such as* $\theta_d(x_\phi, \phi, t_k)$*.*

The assignation $\theta_d(x_\phi, \phi, t_k)$ is the interpretation of the observation $d(k)$ designating the index $k$ of the timestamp $t_k$ in a set of timestamps $\Gamma$.

**Definition 4.3** *Observation of a Constant*
*The observation of a constant, denoted* $\delta(k) \equiv (\delta_k, t_\phi)$*, is the assignation of values* $x_{\theta_\delta} = x_\phi$, $\delta_{\theta_\delta} = \delta_k$ *and* $t_{\theta_\delta} = t_\phi$ *to the predicate* $\theta_\delta(x_{\theta_\delta}, \delta_{\theta_\delta}, t_{\theta_\delta})$ *such as* $\theta_\delta(x_\phi, \delta_k, t_\phi)$*.*

The assignation $\theta_\delta(x_\phi, \delta_k, t_\phi)$ is the interpretation of the observation $\delta(k)$ designating the index $k$ of the constant $\delta_k$ in a set of constants $\Delta$.

## 4.3 Deduction of an Assignation from Two Assignations

Let be two assignations $\theta_1(x_1, \delta_i, t_i)$ and $\theta_2(x_2, \delta_j, t_j)$ respectively associated to two timed observations $O(t_i) \equiv (\delta_i, t_i)$ and $O(t_j) \equiv (\delta_j, t_j)$ made by respectively two programs $\Theta_1(\{x_1\}, \Delta_1)$ and $\Theta_2(\{x_2\}, \Delta_2)$. The assignation $\theta_3(x_3, \delta_k, t_k), t_k \in [t_i, t_j]$ associated to the timed observation $O(t_k) \equiv (\delta_k, t_k)$ is the assignation deducted from both assignations $\theta_1(x_1, \delta_i, t_i)$ and $\theta_2(x_2, \delta_j, t_j)$ according to the following implication:

$$\theta_1(x_1, \delta_i, t_i) \wedge \theta_2(x_2, \delta_j, t_j) \wedge |t_j - t_i| \in \Delta\tau_{12} \Rightarrow \theta_3(x_3, \delta_k, t_k) \wedge t_k \in [t_i, t_j] \tag{4.3}$$

Where:

- $x_3$ is a variable that may be $x_1$, $x_2$ or any abstract variable;
- $\delta_k$ is a constant arbitrarily chosen in a set $\Delta_3$;
- $t_k \in \Gamma_k$ is a timestamp arbitrarily chosen in the interval $[t_i, t_j] \subset \Re$;
- $\Delta\tau_{12}$ is a temporal constraint of the form $\Delta\tau_{12} \equiv [\tau_{12}^-; \tau_{12}^+] \subset \Re$;
- $\theta_3(x_{\theta_3}, \delta_{\theta_3}, t_{\theta_3})$ is any ternary predicate.

The assignation $\theta_3(x_3, \delta_k, t_k)$ can then be deducted by the application of the Modus Ponens from the knowledge of both assignations $\theta_1(x_1, \delta_i, t_i)$ and $\theta_2(x_2, \delta_j, t_j)$ and if the temporal constraint $|t_j - t_i| \in \Delta\tau_{12}$ is satisfied. Assignations $\theta_1(x_1, \delta_i, t_i)$ and $\theta_2(x_2, \delta_j, t_j)$ are then linked to the assignation $\theta_3(x_3, \delta_k, t_k)$ by Modus Ponens and by temporal constraint $|t_j - t_i| \in \Delta\tau_{12}$.

The implication 4.3 thus defines a process to build assignations from other assignations by Modus Ponens. This process can be encoded in the program $\Theta_3(\{x_3\}, \Delta_3)$. Such a program aims then at writing timed observations of the form $O(t_k) \equiv (\delta_k, t_k)$ corresponding to the assignation $\theta_3(x_3, \delta_k, t_k)$. The way to choose the variable $x_3$, the constant $\delta_k$ and the timestamp $t_k$ depends on the knowledge encoded in the program $\Theta_3(\{x_3\}, \Delta_3)$. If the program $\Theta_3(\{x_3\}, \Delta_3)$ has no simultaneous observations and if it is memoryless then it is a safe program.

Such a process in an *abstraction process*: the timed observation $O(t_k) \equiv (\delta_k, t_k)$ is *not produced by a canonical program* observing a timed function $x_3(t)$. The timed observation $O(t_k) \equiv (\delta_k, t_k)$ has been *deducted* from timed observations produced by other programs, $\Theta_1(\{x_1\}, \Delta_1)$ and $\Theta_2(\{x_2\}, \Delta_2)$ in this particular case. The assignation $\theta_3(x_3, \delta_k, t_k)$ can also be used in an implication of the form 4.3: such an abstraction process can be realized recursively. The notion of timed observation plays then the role of a *paradigm* because its binary structure $(\delta_k, t_k)$ is invariant whatever the level of abstraction is.

## 4.4 Addition of Two Timed Observations

The deduction of an abstract assignation from a binary deduction rule corresponds to a time-constrained addition of observations.

Let be two timed observations $O(t_i) \equiv (\delta_i, t_i)$ and $O(t_j) \equiv (\delta_j, t_j)$ respectively corresponding to both assignations $\theta_i(x_i, \delta_i, t_i)$ and $\theta_j(x_j, \delta_j, t_j)$ in order to apply Modus Ponens with the following rule:

$$\theta_i(x_i, \delta_i, t_i) \wedge \theta_j(x_j, \delta_j, t_j) \wedge |t_j - t_i| \in \Delta\tau_{ij} \Rightarrow \theta_k(x_k, \delta_k, t_k) \wedge t_k \in [min(t_i, t_j), max(t_i, t_j)] \quad (4.4)$$

In the following, we denote $t_m = min(t_i, t_j)$ and $t_M = max(t_i, t_j)$.

Let be $O(t_k) \equiv (\delta_k, t_k)$, the timed observation corresponding to the assignation $\theta_k(x_k, \delta_k, t_k)$ deducted by application of the Modus Ponens with the rule 4.4. Such an application of the Modus Ponens with the rule 4.4 corresponds to the following operation of addition under temporal constraints $\Delta\tau_{ij}$ of timed observations $O(t_i)$ and $O(t_j)$:

$$O(t_i) \overset{\Delta\tau_{ij}}{+} O(t_j) = O(t_k), t_k \in [t_m, t_M] \quad (4.5)$$

Such an addition is then only possible iff the following temporal conditions are respected:

$$t_M \in t_m + \Delta\tau_{ij} \tag{4.6}$$

This means that if such temporal conditions are not respected, the application of the Modus Ponens with the rule 4.4 fails. This leads to the following definition:

**Definition 4.4** *Addition under Temporal Constraints of Timed Observations*
*The operation of addition, denoted* $\overset{\Delta\tau_{ij}}{+}$*, of timed observations* $O(t_i)$ *and* $O(t_j)$ *under the temporal constraints* $\Delta\tau_{ij}$ *corresponds to the application of the Modus Ponens with the rule 4.4.*

Now, let us focus on some properties of such an addition.

**Property 6** Neutral Element
*The neutral observation is the neutral element for the operation of addition under temporal constraints.*

According to definition 4.1 and equation 4.1, we can compose the equation 4.2 with the temporal constraints $|t_\phi - t_i| \in \Delta\tau_{ij}$ without changing anything:

$$\begin{gathered}\exists\Phi(x_\phi,\phi,t_\phi),\forall\theta_i(x_i,\delta_i,t_i),\\ \theta_i(x_i,\delta_i,t_i)\wedge\Phi(x_\phi,\phi,t_\phi)\wedge|t_\phi-t_i|\in\Delta\tau_{ij}\Leftrightarrow\\ \Phi(x_\phi,\phi,t_\phi)\wedge\theta_i(x_i,\delta_i,t_i)\wedge|t_\phi-t_i|\in\Delta\tau_{ij}\Leftrightarrow\\ \theta_i(x_i,\delta_i,t_i)\end{gathered} \tag{4.7}$$

According to definition 4.4, this corresponds to the following addition:

$$\exists O_\phi(t_\phi),\forall O(t_i), O(t_i)\overset{\Delta\tau_{ij}}{+}O_\phi(t_\phi)=O_\phi(t_\phi)\overset{\Delta\tau_{ij}}{+}O(t_i)=O(t_i) \tag{4.8}$$

This demonstrates the existence of a neutral element, $O_\phi(t_\phi)$, for the operation $\overset{\Delta\tau_{ij}}{+}$.

**Property 7** Commutativity
*The operation of addition under temporal constraints is commutative.*

The logical operator $\wedge$ being commutative, it is clear that the rule 4.4 is equivalent to the following equation:

$$\theta_j(x_j,\delta_j,t_j)\wedge\theta_i(x_i,\delta_i,t_i)\wedge|t_j-t_i|\in\Delta\tau_{ij}\Rightarrow\theta_k(x_k,\delta_k,t_k)\wedge t_k\in[t_m,t_M] \tag{4.9}$$

It Corresponds then to the following addition under temporal constraints $\Delta\tau_{ij}$ of timed observations $O(t_j)$ and $O(t_i)$:

$$O(t_j)\overset{\Delta\tau_{ij}}{+}O(t_i)=O(t_k),t_k\in[t_m,t_M] \tag{4.10}$$

The operation $\overset{\Delta\tau_{ij}}{+}$ is then a commutative one:

$$\forall O(t_i),O(t_j),O(t_i)\overset{\Delta\tau_{ij}}{+}O(t_j)=O(t_j)\overset{\Delta\tau_{ij}}{+}O(t_i) \tag{4.11}$$

**Property 8** Associativity
*The operation of addition under temporal constraints is associative.*

Let be three assignations $\theta_1(x_1, \delta_1, t_1)$, $\theta_2(x_2, \delta_2, t_2)$ and $\theta_3(x_3, \delta_3, t_3)$ respectively associated to three timed observations $O(t_1) \equiv (\delta_1, t_1)$, $O(t_2) \equiv (\delta_2, t_2)$ and $O(t_3) \equiv (\delta_3, t_3)$ made by respectively three programs $\Theta_1(\{x_1\}, \Delta_1)$, $\Theta_2(\{x_2\}, \Delta_2)$ and $\Theta_3(\{x_3\}, \Delta_3)$. Let be $\theta_4(x_4, \delta_k, t_k), t_k \in \{t_1, t_2, t_3\}$, the assignation associated to the timed observation $O(t_k) \equiv (\delta_k, t_k)$, deducted from the three assignations $\theta_1(x_1, \delta_1, t_1)$, $\theta_2(x_2, \delta_2, t_2)$ and $\theta_3(x_3, \delta_3, t_3)$ by application of the Modus Ponens with the following rule:

$$\begin{array}{c}(\theta_1(x_1, \delta_1, t_1) \wedge \theta_2(x_2, \delta_2, t_2) \wedge |t_2 - t_1| \in \Delta\tau_{12}) \\ \wedge(\theta_2(x_2, \delta_2, t_2) \wedge \theta_3(x_3, \delta_3, t_3) \wedge |t_3 - t_2| \in \Delta\tau_{23}) \\ \wedge(\theta_1(x_1, \delta_1, t_1) \wedge \theta_3(x_3, \delta_3, t_3) \wedge |t_3 - t_1| \in \Delta\tau_{13}) \\ \Rightarrow \theta_4(x_4, \delta_k, t_k) \wedge t_k \in \{t_1, t_2, t_3\}, \\ \text{where } \Delta\tau_{13} = [\tau_{12}^- + \tau_{23}^-, \tau_{12}^+ + \tau_{23}^+].\end{array} \tag{4.12}$$

The application of the Modus Ponens with the rule 4.12 corresponds to the three following additions under temporal constraints $\Delta\tau_{12}$, $\Delta\tau_{23}$ and $\Delta\tau_{13}$:

$$\begin{array}{c}O(t_1) \overset{\Delta\tau_{12}}{+} O(t_2) = O_{12}(t_{12}), t_{12} \in [t_1, t_2] \text{ if } max(t_1, t_2) \in min(t_1, t_2) + \Delta\tau_{12} \\ O_{12}(t_{12}) \overset{\Delta\tau_{23}}{+} O(t_3) = O_{123}(t_{123}), t_{123} \in [t_{12}, t_3] \text{ if } max(t_{12}, t_3) \in min(t_{12}, t_3) + \Delta\tau_{23} \\ O(t_1) \overset{\Delta\tau_{13}}{+} O_{123}(t_{123}) = O(t_k), t_k \in [t_1, t_{123}] \text{ if } max(t_1, t_{123}) \in min(t_1, t_{123}) + \Delta\tau_{13}\end{array} \tag{4.13}$$

Since $t_k \in [t_1, t_{123}]$, $t_{123} \in [t_{12}, t_3]$ and $t_{12} \in [t_1, t_2]$, we have then $t_k \in [t_1, t_{123}] \Rightarrow t_k \in \{t_1, t_2, t_3\}$.

The application of the Modus Ponens with the rule 4.12 corresponds also to the three following additions:

$$\begin{array}{c}O(t_2) \overset{\Delta\tau_{23}}{+} O(t_3) = O_{23}(t_{23}), t_{23} \in [t_2, t_3] \text{ if } max(t_2, t_3) \in min(t_2, t_3) + \Delta\tau_{23} \\ O(t_1) \overset{\Delta\tau_{13}}{+} O_{23}(t_{23}) = O_{123}(t_{123}), t_{123} \in [t_1, t_{23}] \text{ if } max(t_1, t_{23}) \in min(t_1, t_{23}) + \Delta\tau_{13} \\ O_{123}(t_{123}) \overset{\Delta\tau_{12}}{+} O(t_2) = O(t_k), t_k \in [t_{123}, t_2] \text{ if } max(t_{123}, t_2) \in min(t_{123}, t_2) + \Delta\tau_{12}\end{array} \tag{4.14}$$

Since $t_k \in [t_{123}, t_2]$, $t_{123} \in [t_1, t_{23}]$ and $t_{23} \in [t_2, t_3]$, we have then $t_k \in [t_{123}, t_2] \Rightarrow t_k \in \{t_1, t_2, t_3\}$.

Since $\Delta\tau_{13} = [\tau_{12}^- + \tau_{23}^-, \tau_{12}^+ + \tau_{23}^+]$, additions of 4.13 and 4.14 can be written according to the unique following form:

$$\begin{array}{c}O(t_1) \overset{\Delta\tau_{12}}{+} O(t_2) \overset{\Delta\tau_{23}}{+} O(t_3) = O(t_k), t_k \in \{t_1, t_2, t_3\} \\ \text{if } max(t_1, t_2) \in min(t_1, t_2) + \Delta\tau_{12} \text{ and } max(t_2, t_3) \in min(t_2, t_3) + \Delta\tau_{23}\end{array} \tag{4.15}$$

This leads to conclude that the addition under temporal constraints is associative:

$$O(t_1) \overset{\Delta\tau_{12}}{+} O(t_2) \overset{\Delta\tau_{23}}{+} O(t_3) = \left(O(t_1) \overset{\Delta\tau_{12}}{+} O(t_2)\right) \overset{\Delta\tau_{23}}{+} O(t_3) = O(t_1) \overset{\Delta\tau_{12}}{+} \left(O(t_2) \overset{\Delta\tau_{23}}{+} O(t_3)\right) \quad (4.16)$$

This method extends in the same way to the additions of $n$ observations. The advantage of this definition of addition of observations is that it applies to the sequences of timed observations as shown in the following section.

## 4.5 Composition of Observers

Let be two programs $\Theta_1(\{x_1\}, \Delta_1)$ and $\Theta_2(\{x_2\}, \Delta_2)$ defining respectively sets of timestamps $\Gamma_1$ and $\Gamma_2$ and implementing respectively the predicates $\theta_1(x_{\theta_1}, \delta_{\theta_1}, t_{\theta_1})$ and $\theta_2(x_{\theta_2}, \delta_{\theta_2}, t_{\theta_2})$. Let be $\omega_1(t_n) = \{O(t_i) \equiv (\delta_i, t_i), \delta_i \in \Delta_1, t_i \in \Gamma_1\}$, the sequence of $n$ timed observations produced by the program $\Theta(\{x_1\}, \Delta_1)$. Let be $\omega_2(t_m) = \{O(t_j) \equiv (\delta_j, t_j), \delta_j \in \Delta_2, t_j \in \Gamma_2\}$, the sequence of $m$ timed observations produced by the program $\Theta(\{x_2\}, \Delta_2)$.

The operation of addition of sequences of timed observations $\omega_1(t_n)$ and $\omega_2(t_m)$ is built from the operation of addition under temporal constraints $\Delta\tau_{ij}$:

$$\forall t_i \in \Gamma_1, \forall t_j \in \Gamma_2, O(t_i) \overset{\Delta\tau_{ij}}{+} O(t_j) = O(t_k) \quad (4.17)$$

Such an addition defines a sequence $\omega_3(t_p) = \{O(t_k) \equiv (\delta_k, t_k), \delta_k \in \Delta_3, t_k \in \Gamma_3\}$ of $p$ timed observations. Each timed observation $O(t_k)$ corresponds to the application of the Modus Ponens with the following rule:

$$\begin{array}{c} \forall t_i \in \Gamma_1, \forall t_j \in \Gamma_2, \theta_1(x_1, \delta_i, t_i) \wedge \theta_2(x_2, \delta_j, t_j) \wedge |t_j - t_i| \in \Delta\tau_{ij} \\ \Rightarrow \exists t_k \in \Gamma_3, \theta_3(x_3, \delta_k, t_k) \wedge t_k \in [t_i, t_j] \end{array} \quad (4.18)$$

Each timed observation $O(t_k)$ is then the assignation $\theta_3(x_3, \delta_k, t_k)$ of a predicate $\theta_3(x_{\theta_3}, \delta_{\theta_3}, t_{\theta_3})$ which is implemented in an abstract program $\Theta_3(\{x_3\}, \Delta_3)$.

This leads us to the definition of the composition of observers:

**Definition 4.5** *Composition of Observers*
*Given two programs $\Theta_1(\{x_1\}, \Delta_1)$ and $\Theta_2(\{x_2\}, \Delta_2)$, any program $\Theta_3(\{x_3\}, \Delta_3)$ implementing the rule 4.18 defines an operation of addition of all the timed observations produced by programs $\Theta_1(\{x_1\}, \Delta_1)$ and $\Theta_2(\{x_2\}, \Delta_2)$. Such an operation is called the composition of observers and is denoted $\oplus$:*

$$\Theta_3(\{x_3\}, \Delta_3) = \Theta_1(\{x_1\}, \Delta_1) \oplus \Theta_2(\{x_2\}, \Delta_2) \quad (4.19)$$

The composition of observers $\oplus$ being built from the addition under temporal constraints of timed observations, it inherits of its properties:

- existence of a neutral element, *the neutral observer*, denoted $\Theta_\Phi(\{x_\phi\}, \{\phi\})$ implementing the neutral predicate defined in 4.2:

$$\begin{gathered}\exists\Theta_{\Phi}(\{x_{\phi}\},\{\phi\}),\forall\Theta_1(\{x_1\},\Delta_1),\\ \Theta_1(\{x_1\},\Delta_1)\oplus\Theta_{\Phi}(\{x_{\phi}\},\{\phi\})=\Theta_{\Phi}(\{x_{\phi}\},\{\phi\})\oplus\Theta_1(\{x_1\},\Delta_1)=\Theta_1(\{x_1\},\Delta_1)\end{gathered} \tag{4.20}$$

- commutativity:

$$\begin{gathered}\forall\Theta_1(\{x_1\},\Delta_1),\forall\Theta_2(\{x_2\},\Delta_2),\\ \Theta_1(\{x_1\},\Delta_1)\oplus\Theta_2(\{x_2\},\Delta_2)=\Theta_2(\{x_2\},\Delta_2)\oplus\Theta_1(\{x_1\},\Delta_1)\end{gathered} \tag{4.21}$$

- associativity:

$$\begin{gathered}\forall\Theta_1(\{x_1\},\Delta_1),\forall\Theta_2(\{x_2\},\Delta_2),\forall\Theta_3(\{x_3\},\Delta_3),\\ (\Theta_1(\{x_1\},\Delta_1)\oplus\Theta_2(\{x_2\},\Delta_2))\oplus\Theta_3(\{x_3\},\Delta_3)=\\ \Theta_1(\{x_1\},\Delta_1)\oplus(\Theta_2(\{x_2\},\Delta_2)\oplus\Theta_3(\{x_3\},\Delta_3))\end{gathered} \tag{4.22}$$

## 4.6 Abstract Unary Observer

Let us now go back to the definition of the addition under temporal constraints of two timed observations introduced in 4.5:

$$O(t_i)\overset{\Delta\tau_{ij}}{+}O(t_j)=O(t_k),t_k\in[t_m,t_M] \tag{4.23}$$

Such an operation being true for all timed observations $O(t_i)$ and $O(t_j)$, we can use it in the particular case where $O(t_i)$ is the observation of a constant (cf definition 4.3), $\delta(k)\equiv(\delta_k,t_{\phi})$, and where $O(t_j)$ is the observation of a timestamps (cf definition 4.2), $d(k)\equiv(\phi,t_k)$. Since $O(t_k)\equiv(\delta_k,t_k)$, equation 4.23 gives then:

$$(\delta_k,t_{\phi})\overset{\Re}{+}(\phi,t_k)=(\delta_k,t_k),\forall t_k\in\Re \tag{4.24}$$

Such an addition has no temporal constraints that is to say: $\Delta\tau_{ij}=\Re$.

This demonstrates the following property:

**Property 9**
*Any timed observation is always the result of the addition of the observation of a constant and of the observation of a timestamp.*

Let us now consider a sequence $\omega(t_n)=\{O(t_k)\equiv(\delta_k,t_k),\delta_k\in\Delta,t_k\in\Gamma\}$ of $n$ timed observations. Such a sequence defines a set $K=\{k,k\in[1,n]\}$ of index $k$. Each index $k$ of $K$ maps the constant $\delta_k$ in $\Delta$ and the timestamp $t_k$ in $\Gamma$. Since, $(\delta_k,t_k)=(\delta_k,t_{\phi})\overset{\Re}{+}(\phi,t_k)$, timed observations of the sequence $\omega(t_n)$ are made by the composition of both following observers:

- an observer of constants, denoted $\Theta_{\delta}(\{x_{\phi}\},\Delta)$, implementing the predicate $\theta_{\delta}(x_{\theta_{\delta}},\delta_{\theta_{\delta}},t_{\theta_{\delta}})$ defined in 4.3. The assignation of such a predicate is the interpretation of the observation $\delta(k)$ designating the index $k$ of the constant $\delta_k$ in a set of constants $\Delta$.

- an observer of timestamps, denoted $\Theta_d(\Gamma, \{\phi\})$, implementing the predicate $\theta_d(x_{\theta_d}, \delta_{\theta_d}, t_{\theta_d})$ defined in 4.2. The assignation of such a predicate is the interpretation of the observation $\delta(k)$ designating the index $k$ of the timestamp $t_k$ in a set of timestamps $\Gamma$.

The sequence of timed observations $\omega(t_n)$ is then obtained by the application of the Modus Ponens with the following rule:

$$\begin{aligned} \forall k \in K, \theta_\delta(x_\phi, \delta_k, t_\phi) \wedge \theta_d(x_\phi, \phi, t_k) \wedge |t_k - t_\phi| \in \Re \\ \Rightarrow \theta(x_\phi, \delta_k, t_k) \end{aligned} \tag{4.25}$$

In the assigned predicate $\theta(x_\phi, \delta_k, t_k)$, the variable name $x_\phi$ is used to indicate the absence of knowledge about the way the sequence $\omega(t_n)$ has been produced. Such a predicate can then be implemented in an abstract program, called an *Abstract Unary Observer* and denoted $\Theta(\{x_\phi\}, \Delta)$, observing an unknown timed function $x_\phi(t)$. According to definition 4.5, we have the following composition of observers:

$$\Theta(\{x_\phi\}, \Delta) = \Theta_\delta(\{x_\phi\}, \Delta) \oplus \Theta_d(\Gamma, \{\phi\}) \tag{4.26}$$

Figure 4.1 illustrates the internal structure of the Abstract Unary Observer $\Theta(\{x_\phi\}, \Delta)$.

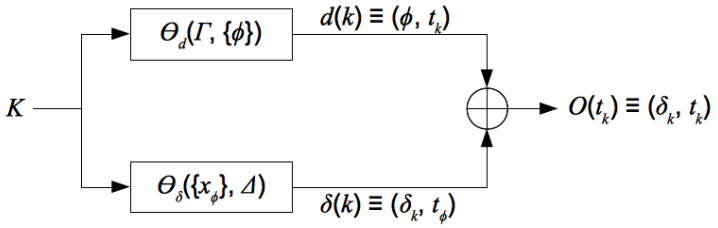


Figure 4.1: Internal structure of the Abstract Unary Observer $\Theta(\{x_\phi\}, \Delta)$

In this section, both following properties have then been demonstrated:

**Property 10**
*Any sequence of timed observations* $\omega(t_n) = \{O(t_k) \equiv (\delta_k, t_k), \delta_k \in \Delta, t_k \in \Gamma\}$ *is characterized by a set* $K$ *of index* $k$ *designating the constants* $\delta_k$ *contained in the set* $\Delta$ *and the timestamps* $t_k$ *contained in the set* $\Gamma$.

**Property 11**
*Any sequence of timed observations* $\omega(t_n) = \{O(t_k) \equiv (\delta_k, t_k), \delta_k \in \Delta, t_k \in \Gamma\}$ *can be represented by an Abstract Unary Observer* $\Theta(\{x_\phi\}, \Delta)$*, decomposable into an observer of constants,* $\Theta(\{x_\phi\}, \Delta)$*, and an observer of timestamps,* $\Theta_d(\Gamma, \{\phi\})$*:* $\Theta(\{x_\phi\}, \Delta) = \Theta_\delta(\{x_\phi\}, \Delta) \oplus \Theta_d(\Gamma, \{\phi\})$.

An Abstract Unary Observer is an abstract structural model of a sequence of timed observations. This model is applicable to any sequence of timed observations, whether produced by a canonical or composed observer. This structure does not say anything about the embedded knowledge in the observer who has produced the sequence. It designates neither the observed variable nor the program, neither the structure of the clock associated with the set of timestamps,

nor the rules of choice of the constants. This knowledge is meta regarding to the sequence of timed observations.

## 4.7 Abstract Binary Observer

Let us consider again the sequence $\omega(t_n) = \{O(t_k) \equiv (\delta_k, t_k), \delta_k \in \Delta, t_k \in \Gamma\}$ of $n$ timed observations. Such a sequence defines a set $K = \{k, k \in [1, n]\}$ of index $k$. Each index $k$ of $K$ maps the constant $\delta_k$ in $\Delta$ and the timestamp $t_k$ in $\Gamma$.

Let us arbitrarily split the set $\Delta$ into two disjoint sets $\Delta_i$ and $\Delta_j$ such as $\Delta = \Delta_i \cup \Delta_j$ and $\Delta_i \cap \Delta_j = \emptyset$. The sequence $\omega(t_n)$ is then the superposition of two sequences $\omega_i(t_{n_i})$ and $\omega_j(t_{n_j})$ of respectively $n_i$ and $n_j$ timed observations.

The sequence $\omega_i(t_{n_i}) = \{O(t_{k_i}) \equiv (\delta_{k_i}, t_{k_i}), \delta_{k_i} \in \Delta_i, t_{k_i} \in \Gamma_i\}$ defines a set $K_i = \{k_i, k_i \in [1, n_i]\}$ of index $k_i$. Each index $k_i$ of $K_i$ maps the constant $\delta_{k_i}$ in $\Delta_i$ and the timestamp $t_{k_i}$ in $\Gamma_i$.

The sequence $\omega_j(t_{n_j}) = \{O(t_{k_j}) \equiv (\delta_{k_j}, t_{k_j}), \delta_{k_j} \in \Delta_j, t_{k_j} \in \Gamma_j\}$ defines a set $K_j = \{k_j, k_j \in [1, n_j]\}$ of index $k_j$. Each index $k_j$ of $K_j$ maps the constant $\delta_{k_j}$ in $\Delta_j$ and the timestamp $t_{k_j}$ in $\Gamma_j$.

In the following, a timed observation $O(t_k)$ is then denoted under the following form: $O(t_k) \equiv O(k) \equiv (\delta_k, t_k)$.

According to property 11, sequences $\omega_i(t_{n_i})$ and $\omega_j(t_{n_j})$ can be respectively represented by two Abstract Unary Observers denoted $\Theta_i(\{x_{\phi_i}\}, \Delta_i)$ and $\Theta_j(\{x_{\phi_j}\}, \Delta_j)$. Abstract variables $x_{\phi_i}$ and $x_{\phi_j}$ are unknown but take some values in sets $\Delta_i$ and $\Delta_j$. We have then:

$$\begin{gathered} \forall O(k) \in \omega(t_n), \delta(k) \in \Delta_i \Rightarrow O(k) \in \omega_i(t_{n_i}) \\ \forall O(k) \in \omega(t_n), \delta(k) \in \Delta_j \Rightarrow O(k) \in \omega_j(t_{n_j}) \\ K = K_i \cup K_j \Rightarrow \Delta = \Delta_i \cup \Delta_j \wedge \omega(t_n) = \omega_i(t_{n_i}) \cup \omega_j(t_{n_j}) \wedge \Gamma = \Gamma_i \cup \Gamma_j \end{gathered} \tag{4.27}$$

Let us now consider the partitions $K_{ij}^1 \subseteq K_i$ and $K_{ij}^2 \subseteq K_j$ of the sets $K_i$ and $K_j$ such as:

$$\forall (O(k), O(k+1)) \subseteq \omega(t_n), \delta(k) \in \Delta_i \wedge \delta(k+1) \in \Delta_j \Rightarrow k \in K_{ij}^1 \wedge k+1 \in K_{ij}^2 \tag{4.28}$$

Partitions $K_{ij}^1$ and $K_{ij}^2$ define partitions $\Omega_{ij}^1 \subseteq \omega_i(t_{n_i})$ and $\Omega_{ij}^2 \subseteq \omega_j(t_{n_j})$ of sequences $\omega_i(t_{n_i})$ and $\omega_j(t_{n_j})$ such as:

$$\begin{gathered} \forall O(k) \in \omega(t_n), k \in K_{ij}^1 \Rightarrow \delta(k) \in \Delta_i \wedge O(t_k) \in \Omega_{ij}^1 \\ \forall O(k) \in \omega(t_n), k \in K_{ij}^1 \Rightarrow \exists k+1 \in K_{ij}^2, \delta(k+1) \in \Delta_j \wedge O(k+1) \in \Omega_{ij}^2 \end{gathered} \tag{4.29}$$

And:

$$\begin{gathered} \forall O(k+1) \in \omega(t_n), k+1 \in K_{ij}^2 \Rightarrow \delta(k+1) \in \Delta_j \wedge O(k+1) \in \Omega_{ij}^2 \\ \forall O(k+1) \in \omega(t_n), k+1 \in K_{ij}^2 \Rightarrow \exists k \in K_{ij}^1, \delta(k) \in \Delta_i \wedge O(k) \in \Omega_{ij}^1 \end{gathered} \tag{4.30}$$

The timestamps $t_k, k \in K_{ij}^1$ and $t_{k+1}, k+1 \in K_{ij}^2$ are linked by a temporal interval $\Delta\tau_{ij}$ of the form $\Delta\tau_{ij} = [\tau_{ij}^-, \tau_{ij}^+]$ such as:

$$\begin{array}{c} \tau_{ij}^{-} = min(t_k, t_{k+1}) \\ \tau_{ij}^{+} = max(t_k, t_{k+1}) \end{array} \tag{4.31}$$

Thus, timed observations $O(k), k \in K_{ij}^1$ and $O(k+1), k+1 \in K_{ij}^2$ respect the following equation:

$$\begin{array}{c} \forall (k, k+1) \in K_{ij}^1 \times K_{ij}^2, \exists (O(k), O(k+1)) \subseteq \omega(t_n), \\ O(k) \in \Omega_{ij}^1 \wedge O(k+1) \in \Omega_{ij}^2 \wedge t_{k+1} - t_k \in \Delta\tau_{ij} = [\tau_{ij}^{-}, \tau_{ij}^{+}] \end{array} \tag{4.32}$$

It is then possible to link timed observations $O(k) \in \Omega_{ij}^1$ and $O(k+1) \in \Omega_{ij}^2$ to a binary observation $O_{ij}(k+1)$ defined by the addition under temporal constraint $\Delta\tau_{ij}$:

$$\forall (k, k+1) \in K_{ij}^1 \times K_{ij}^2, O_{ij}(k+1) = O(k) \overset{\Delta\tau_{ij}}{+} O(k+1) \tag{4.33}$$

Where:

- $O(k) \equiv (\delta_k, t_k)$ is the $k^{th}$ timed observation of the sequence $\omega(t_n)$ built from the index $k$ of the set $K_{ij}^1$,
- $O(k+1) \equiv (\delta_{k+1}, t_{k+1})$ is the $(k+1)^{th}$ timed observation of the sequence $\omega(t_n)$ built from the index $k+1$ of the set $K_{ij}^2$
- $O_{ij}(k+1) \equiv (\delta_{k+1}^{ij}, t_{k+1}^{ij})$ is a timed observation such as $t_{k+1}^{ij} \in \{t_k, t_{k+1}\}$.

The couple of index $(k, k+1) \in K_{ij}^1 \times K_{ij}^2$ linking the timestamps of the timed observations $O(k)$ and $O(k+1)$, the temporal constraint $\Delta\tau_{ij}$ is always verified by construction of the sets $K_{ij}^1$ and $K_{ij}^2$. It is then useless and the addition 4.33 is a pure sequential one:

$$\forall (k, k+1) \in K_{ij}^1 \times K_{ij}^2, O_{ij}(k+1) = O(k) + O(k+1) \tag{4.34}$$

The timed observation $O_{ij}(k+1)$ corresponds then to the observation of a binary sequence of observations $(O(k), O(k+1)) \subset \omega(t_n)$ such as:

$$\exists (k, k+1) \subset K, \delta(k) \in \Delta_i \wedge \delta(k+1) \in \Delta_j \tag{4.35}$$

Thus, the timed observation $O_{ij}(k+1)$ is the observation of a couple of constants $(\delta_k, \delta_{k+1})$ and a couple of timestamps $(t_k, t_{k+1})$ linked to the couple of index $(k, k+1)$:

$$\begin{array}{c} \delta_{k+1}^{ij} \Leftrightarrow (\delta_k, \delta_{k+1}) \\ t_{k+1}^{ij} \in \{t_k, t_{k+1}\} \end{array} \tag{4.36}$$

The index $k+1$ referencing a unique timestamp $t_{k+1}$ in the set $\Gamma$ of the sequence $\omega(t_n)$, there is no ambiguity regarding the timed observation $O_{ij}(k+1)$. Since (cf 9):

$$O_{ij}(k+1) = \delta(k+1) + d(k+1) \equiv (\delta_{k+1}^{ij}, t_\phi) + (\phi, t_{k+1}^{ij}) \tag{4.37}$$

We have then:

$$d(k+1) = t_{k+1} \equiv (\phi, t_{k+1}^{ij}) \Rightarrow t_{k+1}^{ij} = t_{k+1} \tag{4.38}$$

The choice of a constant in a timed observation is arbitrary. It is thus always possible to define a constant $\delta^{ij}$ designating a couple $(\delta_k, \delta_{k+1})$ such as $\forall(\delta_k, \delta_{k+1}) \in \Delta_i \times \Delta_j, \delta_k \in \Delta_i \wedge \delta_{k+1} \in \Delta_j$. This constant is sufficient since, on one hand, $\forall k+1 \in K_{ij}^2, \exists(\delta_k, \delta_{k+1}) \in \Delta_i \times \Delta_j, \delta_k \in \Delta_i \wedge \delta_{k+1} \in \Delta_j$ and, on the other hand, the reference to the index $k$ and $k+1$ is given by the index $k+1$ of the timed observation $O_{ij}(k+1)$ itself. Thus:

$$\forall k+1, \delta_{k+1}^{ij} = \delta^{ij} \tag{4.39}$$

Equations 4.38 and 4.39 lead then to:

$$O_{ij}(k+1) \equiv (\delta^{ij}, t_{k+1}) \tag{4.40}$$

The partitions $K_{ij}^1$ and $K_{ij}^2$ implicitly define an abstract function $\phi_{ij}(t)$ and a variable name $\phi_{ij}$ which is equal to the constant $\delta^{ij}$ when a couple $(\delta(k) \in \Delta_i, \delta(k+1) \in \Delta_j)$ is observed. The addition 4.33 corresponds then to an *Abstract Binary Observer*, denoted $\Theta_{ij}(\{\phi_{ij}\}, \{\delta^{ij}\})$. Such an Abstract Binary Observer produces a sequence $\omega_{ij}(t_{n_{ij}}) = \{O_{ij}(k+1) \equiv (\delta^{ij}, t_{k+1})\}$ of $n_{ij} = card(K_{ij}^1) = card(K_{ij}^2)$ abstract timed observations $O_{ij}(k+1)$. Such an observer is a binary one because it observes couples of successive constants $(\delta(k) \in \Delta_i, \delta(k+1) \in \Delta_j)$ linked to a binary sequence $(O(k), O(k+1))$ in a sequence of timed observations $\omega(t_n)$.

The Abstract Binary Observer $\Theta_{ij}(\{\phi_{ij}\}, \{\delta^{ij}\})$ is composed of two observers of timestamps, $\Theta_d(\Gamma_{ij}^1, \{\phi\})$ and $\Theta_d(\Gamma_{ij}^2, \{\phi\})$, and two observers of constants, $\Theta_\delta(\{x_\phi\}, \Delta_i)$ and $\Theta_\delta(\{x_\phi\}, \Delta_j)$, such as:

$$\Theta_{ij}(\{\phi_{ij}\}, \{\delta^{ij}\}) = \Theta_\delta(\{x_\phi\}, \Delta_i) \oplus \Theta_d(\Gamma_{ij}^1, \{\phi\}) \oplus \Theta_\delta(\{x_\phi\}, \Delta_j) \oplus \Theta_d(\Gamma_{ij}^2, \{\phi\}) \tag{4.41}$$

Figure 4.2 illustrates the internal structure of the Abstract Binary Observer $\Theta_{ij}(\{\phi_{ij}\}, \{\delta^{ij}\})$.

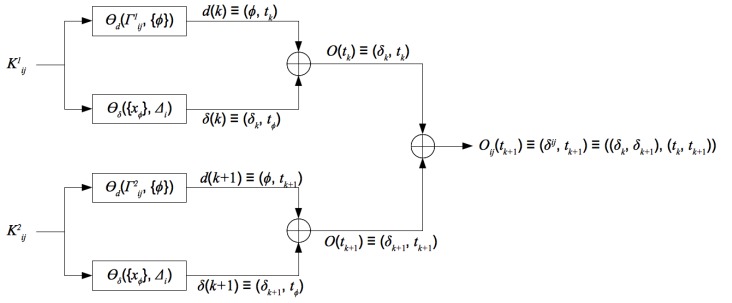


Figure 4.2: Internal structure of the Abstract Binary Observer $\Theta_{ij}(\{\phi_{ij}\}, \{\delta^{ij}\})$

An Abstract Binary Observer $\Theta_{ij}(\{\phi_{ij}\}, \{\delta^{ij}\})$ characterizes thus an oriented sequential binary relation satisfying a temporal constraint $\Delta\tau_{ij}$ from timed observations of the sequence $\omega_i(t_{n_i})$ to timed observations of the sequence $\omega_j(t_{n_j})$ (cf figure 4.3).

Such a relation corresponds to the addition 4.34 which is both a sequential and a temporal constrained addition.

Let us denoted $\theta_i(x_{\theta_i}, \delta_{\theta_i}, t_{\theta_i})$ and $\theta_j(x_{\theta_j}, \delta_{\theta_j}, t_{\theta_j})$, the predicate respectively implemented in the programs $\Theta_i(\{x_{\phi_i}\}, \Delta_i)$ and $\Theta_j(\{x_{\phi_j}\}, \Delta_j)$. Timed observations $O_{ij}(k+1)$ of the sequence $\Omega_{ij}$ produced by the Abstract Binary Observer $\Theta_{ij}(\{\phi_{ij}\}, \{\delta^{ij}\})$ are then obtained by the application

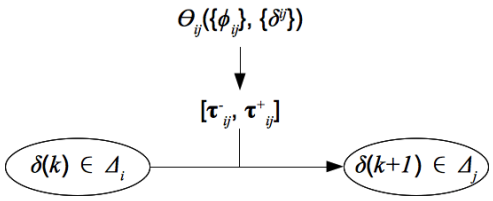


Figure 4.3: Sequential relation temporally constrained

of the Modus Ponens with the following rule:

$$\begin{array}{c} \forall k+1 \in K_{ij}, \theta_i(x_{\phi_i}, \delta_k, t_k) \wedge \theta_j(x_{\phi_j}, \delta_{k+1}, t_{k+1}) \wedge |t_{k+1} - t_k| \in \Delta\tau_{ij} \\ \Rightarrow \theta_{ij}(\phi_{ij}, \delta^{ij}, t_{k+1}) \end{array} \tag{4.42}$$

Where:

- $\theta_i(x_{\phi_i}, \delta_k, t_k)$ is the meaning of the timed observation $O(k)$ such as $\delta(k) \in \Delta_i$ made by the Abstract Unary Observer $\Theta_i(\{x_{\phi_i}\}, \Delta_i)$;
- $\theta_j(x_{\phi_j}, \delta_{k+1}, t_{k+1})$ is the meaning of the timed observation $O(k+1)$ such as $\delta(k+1) \in \Delta_j$ made by the Abstract Unary Observer $\Theta_j(\{x_{\phi_j}\}, \Delta_j)$;
- $\theta_{ij}(\phi_{ij}, \delta^{ij}, t_{k+1})$ is the meaning of the timed observation $O_{ij}(k+1)$ of the sequence $\Omega_{ij}$ made by the Abstract Binary Observer $\Theta_{ij}(\{\phi_{ij}\}, \{\delta^{ij}\})$.

Sets of constants $\Delta_i$ and $\Delta_j$ have been built in an arbitrary way. Thus, these results can be applied to any binary superposition $\Omega = \Omega_i \cup \Omega_j$ whatever the way sequences $\Omega_i$ and $\Omega_j$ have been produced. These results are, in particular, true when such sequences have been produced by two safe and canonical programs $\Theta_i(\{x_i\}, \Delta_i)$ and $\Theta_j(\{x_j\}, \Delta_j)$. The only condition is the obligation for the sets $\Delta_i$ and $\Delta_j$ to be disjoined: $\Delta_i \cap \Delta_j = \emptyset$.

Thus, to sum up, the following properties have been demonstrated in this section:

**Property 12**
*To any superposition $\Omega = \Omega_i \cup \Omega_j$ of sequences $\Omega_i$ and $\Omega_j$ can be associated an Abstract Binary Observer $\Theta_{ij}(\{\phi_{ij}\}, \{\delta^{ij}\})$ observing binary sequences of the form $(O(k) \in \Omega_i, O(k+1) \in \Omega_j)$.*

**Property 13**
*An Abstract Binary Observer $\Theta_{ij}(\{\phi_{ij}\}, \{\delta^{ij}\})$ generates a sequence of timed observation $\Omega_{ij}$ defining a set of index $K_{ij}$ mapping timestamps contained in a set $\Gamma_{ij}$ such as:*

$$\forall k+1 \in K_{ij}, O(k+1) \in \Omega_{ij}, \delta(k+1) = \delta^{ij} \wedge k \in K_i \wedge k+1 \in K_j \tag{4.43}$$

**Property 14**
*An Abstract Binary Observer $\Theta_{ij}(\{\phi_{ij}\}, \{\delta^{ij}\})$ represents a sequential binary relation $(O(k), O(k+1))$ satisfying a temporal constraint $\Delta\tau_{ij}$, oriented from a partition $\Omega^1_{ij}$ of a sequence of timed observations $\Omega_i$, defined on a set of index $K^1_{ij}$, to a partition $\Omega^2_{ij}$ of a sequence of timed observations $\Omega_j$, defined on a set of index $K^2_{ij}$, such as:*

$$\forall (k, k+1) \in K_{ij}^1 \times K_{ij}^2, O_{ij}(k+1) = O(k) \overset{\Delta\tau_{ij}}{+} O(k+1) \tag{4.44}$$

**Property 15**
*The assignation $\theta_{ij}(\phi_{ij}, \delta^{ij}, t_{k+1})$ and the timed observation $O_{ij}(t_{k+1})$ have the same meaning: a binary sequence $(O(k) \in \Omega_i, O(k+1) \in \Omega_j)$ have been observed at the date $t_{k+1}$ in a binary superposition of timed observations $\Omega = \Omega_i \cup \Omega_j$.*

## 4.8 Conclusion

It has been demonstrated in this chapter that the addition under temporal constraints of timed observations is the basic operation to define the composition of observers. Such a composition inherits of the properties of the addition of timed observations: existence of a neutral element, commutativity and associativity. These properties are used to demonstrate in chapter 6 that an safe and canonical program implementing the Spacial Discretization Principle (see 3.5), called a *Unary Observer*, plays the role of a sampling device in the TOT framework.

It has also been demonstrated that such an addition induces an abstraction process that allows to define observers at different level of abstractions. These observers, may they be concrete or abstract, allow to model any sequences of timed observations that is to say any system generating discrete events or alarms. The existence of such an abstraction process and of such observers are the key point to formalize the TOT abstraction process.

This formalization is introduced, in a first step, in chapter 5 and, in a second step, in chapter 7 using the Category Theory as a mathematical tool.

CHAPTER 5

# Process of Abstraction in the TOT Framework

## 5.1 Introduction

This chapter aims at introducing the mechanism of the abstraction process in the TOT framework.

To this aim, we demonstrate that a m-ary superposition of sequences of timed observations can be modeled thanks to a collection of Abstract Binary Observers. This provides us the tools to build the algebraic structure and observable space of the observed process. This allows us to introduce the concepts of abstract chronicle model and behaviour model of such an observed process. These elements altogether allow to reveal the process of abstraction in the TOT framework.

## 5.2 Modelisation of a Superposition of Sequences of Timed Observations

Let us consider the dynamic process $X(t) = \{x_1(t), ..., x_m(t)\}$, $m \in \mathbb{N}^\star$, composed of $m$ timed functions $x_i(t), i \in [1; m]$. Such a dynamic process defines the set $X = \{x_1, ..., x_m\}$ of $m$ variable names $x_i, i \in [1; m]$. Let us consider the independent program of observation $\Theta(X, \Delta)$ observing the dynamic process $X(t)$. Let us consider the observed process $(X(t), \Theta(X, \Delta))$. Let us partition the set $\Delta$ of constants into $m$ sets $\Delta_i$ such as :

- $\Delta_i = \{\delta_{i_k}, i_k \in [1; n_{\Delta_i}]\}$ is the set of the $n_{\Delta_i} \in \mathbb{N}^\star$ constants $\delta_{i_k}$ that the variable $x_i$ can take;
- $\Delta = \bigcup_{i \in [1;m]} \Delta_i$;
- $\forall (i, j) \in [1; m]^2, i \neq j, \Delta_i \bigcap \Delta_j = \emptyset$.

According to the Superposition Theorem 3.2, the program $\Theta(X, \Delta)$ can be decomposed in a superposition of $m$ independent programs $\Theta_i(X_i, \Delta_i)$ where $X_i = \{x_i\}$.

Any program $\Theta_i(X_i, \Delta_i)$ implements a predicate denoted $\theta_i(x_{\theta_i}, \delta_{\theta_i}, t_{\theta_i})$. The program $\Theta_i(X_i, \Delta_i)$ writes a timed observation denoted $O(t_{i_k}) \equiv (\delta_{i_k}, t_{i_k})$ for each assignation $\theta_i(x_i, \delta_{i_k}, t_{i_k})$ of the predicate $\theta_i(x_{\theta_i}, \delta_{\theta_i}, t_{\theta_i})$:

$$\theta_i(x_i, \delta_{i_k}, t_{i_k}) \Rightarrow write(O(t_{i_k}) \equiv (\delta_{i_k}, t_{i_k})) \tag{5.1}$$

Let us denote $O_{i_k} = \{(x_i, \delta_{i_k})\}$, the observation class linking variable name $x_i$ to constant $\delta_{i_k}$. Let us denote $\omega_i(t_{n_i}) = \{O(t_{i_k}) \equiv (\delta_{i_k}, t_{i_k}), \delta_{i_k} \in \Delta_i, t_{i_k} \in \Gamma_i\}$, the sequence of $n_i \in \mathbb{N}$ timed observations $O(t_{i_k})$. Such a sequence defines the stochastic clock $\Gamma_i = \{t_{i_k}, i_k \in [1; n_i]\}$ of timestamps $t_{i_k}$. The sequence $\omega_i(t_{n_i})$ may be denoted $\omega_i$ in order to lighten the writing.

Let us denote $\Omega = \bigcup_{i \in [1;m]} \omega_i$, the superposition of the $m$ sequences of timed observations $\omega_i$. This aim of this section is to demonstrate that such a superposition $\Omega$ can be modeled with a structure composed of Abstract Binary Observers. Let us then first focus on the case of a binary superposition.

### 5.2.1 Superposition of Two Sequences

Let us consider here the superposition $\Omega = \omega_p \cup \omega_q$, $p \neq q$, composed of two sequences of timed observations $\omega_p$ and $\omega_q$.

Let us build the eight sequences denoted $\omega^1_{xy}$ and $\omega^2_{xy}$ where $(x, y) \in \{p, q\}^2$ from $\omega_p$ and $\omega_q$ such that:

- $O(t_{p_k}) \in \omega_p \wedge O(t_{p_{k+1}}) \in \omega_p \Rightarrow O(t_{p_k}) \in \omega^1_{pp} \wedge O(t_{p_{k+1}}) \in \omega^2_{pp}$;
- $O(t_{p_k}) \in \omega_p \wedge O(t_{q_{k+1}}) \in \omega_q \Rightarrow O(t_{p_k}) \in \omega^1_{pq} \wedge O(t_{q_{k+1}}) \in \omega^2_{pq}$;
- $O(t_{q_k}) \in \omega_q \wedge O(t_{p_{k+1}}) \in \omega_p \Rightarrow O(t_{q_k}) \in \omega^1_{qp} \wedge O(t_{p_{k+1}}) \in \omega^2_{qp}$;
- $O(t_{q_k}) \in \omega_q \wedge O(t_{q_{k+1}}) \in \omega_q \Rightarrow O(t_{q_k}) \in \omega^1_{qq} \wedge O(t_{q_{k+1}}) \in \omega^2_{qq}$.

According to property 12, let us consider the four Abstract Binary Observers, $\Theta_{pp}(\{\phi_{pp}\}, \{\delta^{pp}\})$, $\Theta_{pq}(\{\phi_{pq}\}, \{\delta^{pq}\})$, $\Theta_{qp}(\{\phi_{qp}\}, \{\delta^{qp}\})$ and $\Theta_{qq}(\{\phi_{qq}\}, \{\delta^{qq}\})$ such that:

- $\Theta_{pp}(\{\phi_{pp}\}, \{\delta^{pp}\})$ observes any binary sequence of successive timed observations of the form $(O(t_{p_k}) \in \omega^1_{pp}, O(t_{p_{k+1}}) \in \omega^2_{pp})$;
- $\Theta_{pq}(\{\phi_{pq}\}, \{\delta^{pq}\})$ observes any binary sequence of successive timed observations of the form $(O(t_{p_k}) \in \omega^1_{pq}, O(t_{q_{k+1}}) \in \omega^2_{pq})$;
- $\Theta_{qp}(\{\phi_{qp}\}, \{\delta^{qp}\})$ observes any binary sequence of successive timed observations of the form $(O(t_{q_k}) \in \omega^1_{qp}, O(t_{p_{k+1}}) \in \omega^2_{qp})$;
- $\Theta_{qq}(\{\phi_{qq}\}, \{\delta^{qq}\})$ observes any binary sequence of successive timed observations of the form $(O(t_{q_k}) \in \omega^1_{qq}, O(t_{q_{k+1}}) \in \omega^2_{qq})$.

Let us now denote $\Theta^S = [\Theta_{xy}]_{(x,y) \in \{p,q\}^2}$, the matrix containing these four Abstract Binary Observers. This means that all couples of successive timed observations contained in a superpositions $\Omega = \omega_p \cup \omega_q$ can be observed by such a matrix of observers. This lead us to the following property:

**Property 16**
*Considering a binary superposition $\Omega = \omega_p \cup \omega_q$ of sequences of timed observations $\omega_p$ and $\omega_q$, produced by two independent programs $\Theta_p(X_p, \Delta_p)$ and $\Theta_q(X_q, \Delta_q)$ such that $\Delta_p \cap \Delta_q = \emptyset$, all binary sequences of successive timed observations of the superposition $\Omega$ can be observed by a*

*matrix $\Theta^S$ of four Abstract Binary Observers.*
*Such a matrix $\Theta^S$ models then a binary superposition $\Omega = \omega_p \cup \omega_q$ of sequences of timed observations.*

Let us then now focus on the general case of a $m$, $(m > 2)$, superpositions.

### 5.2.2 Superposition of m, m > 2, Sequences

Let us consider here the superposition $\Omega = \bigcup_{i\in[1;m]} \omega_i$, of $m$ sequences of timed observations $\omega_i$.

Such a superposition contains $\frac{m\times(m-1)}{2}$ binary superpositions of the form $\omega_p \cup \omega_q$, $(p,q) \in [1;m]^2$. Thus, according to property 16, all binary sequences of successive timed observations can be observed in these $\frac{m\times(m-1)}{2}$ binary superpositions by $\frac{m\times(m-1)}{2}$ matrices $\Theta^S_k$, $k \in [1; \frac{m\times(m-1)}{2}]$, of four Abstract Binary Observers of the form $\Theta_{xy}(\{\phi_{xy}\}, \{\delta^{xy}\}), (x,y) \in \{p,q\}^2$. Nevertheless, these matrices are not suffisent to observe all binary sequences of successives timed observations of the superposition $\Omega$. Indeed, the way the sequences are interlaced with each other is not described: such an interweaving can be described thanks to another matrix, denoted $\Theta^E$ of $m^2$ Abstract Binary Observers of the form $\Theta_{pq}(\{\phi_{pq}\}, \{\delta^{pq}\})$:

$$\Theta^E = [\Theta_{pq}(\{\phi_{pq}\}, \{\delta^{pq}\})]_{(p,q)\in[1;m]^2} \tag{5.2}$$

This leads us to the following property:

**Property 17**
*Considering a m-ary superposition $\Omega = \bigcup_{i\in[1;m]} \omega_i$ of $m$ sequences of timed observations $\omega_i$, produced by $m$ independent programs $\Theta_i(X_i, \Delta_i)$ such that $\bigcap_{i\in[1;m]} \Delta_i = \emptyset$, all binary sequences of successive timed observations of the superposition $\Omega$ can be observed by a collection of matrices $\Theta^S_k$, $k \in [1; \frac{m\times(m-1)}{2}]$ of four Abstract Binary Observers of the form $\Theta_{xy}(\{\phi_{xy}\}, \{\delta^{xy}\}), (x,y) \in \{p,q\}^2$ and by a matrix $\Theta^E$ of $m^2$ Abstract Binary Observers of the form $\Theta_{pq}(\{\phi_{pq}\}, \{\delta^{pq}\}), (p,q) \in [1;m]^2$.*
*The collection of such matrices models then a m-ary superposition $\Omega = \bigcup_{i\in[1;m]} \omega_i$ of $m$ sequences of timed observations.*

Now, all elements are in places to build the algebraic structure and the observable space of an observed process containing such a m-ary superposition of sequences of timed observations.

## 5.3 Algebraic Structure of the Observed Process

Let us consider the m-ary superposition $\Omega = \bigcup_{i\in[1;m]} \omega_i$, of $m$ sequences of timed observations $\omega_i$. Let us denote $\Gamma = \bigcup_{i\in[1;m]} \Gamma_i$, the stochastic clock containing all timestamps $t_{i_k}$ of $\Gamma_i$ for all $i \in [1;m]$.

The superposition $\Omega = \{O(t_k) \equiv (\delta_j, t_k), \delta_j \in \Delta, t_k \in \Gamma\}$ contains then timed observation of the form $O(t_k) \equiv (\delta_j, t_k)$ such as $\delta_j \in \Delta$ and $t_k \in \Gamma$. The superposition $\Omega$ is then clearly a subset

of $\Delta \times \Gamma$.

$$\Omega \subseteq \Delta \times \Gamma \tag{5.3}$$

Figure 5.1 illustrates the superposition $\Omega = \{\gamma_1, \alpha_2, \beta_3, \alpha_4, \gamma_5, \alpha_6, \beta_7, \alpha_8, \gamma_9\}$ of three sequences $\omega_1$, $\omega_2$ and $\omega_3$ of timed observations such as:

- $\omega_1 = \{\alpha_2, \alpha_4, \alpha_6, \alpha_8\}$ where $\alpha_i \equiv \alpha_i(t_i) \equiv (\alpha, t_i), i \in \{2, 4, 6, 8\}$,
- $\omega_2 = \{\beta_3, \beta_7\}$ where $\beta_i \equiv \beta_i(t_i) \equiv (\beta, t_i), i \in \{3, 7\}$,
- $\omega_3 = \{\gamma_1, \gamma_5, \gamma_9\}$ where $\gamma_i \equiv \gamma_i(t_i) \equiv (\gamma, t_i), i \in \{1, 5, 9\}$.

Such a superposition defines a stochastic clock $\Gamma = \{t_1, t_2, t_3, t_4, t_5, t_6, t_7, t_8, t_9\}$.

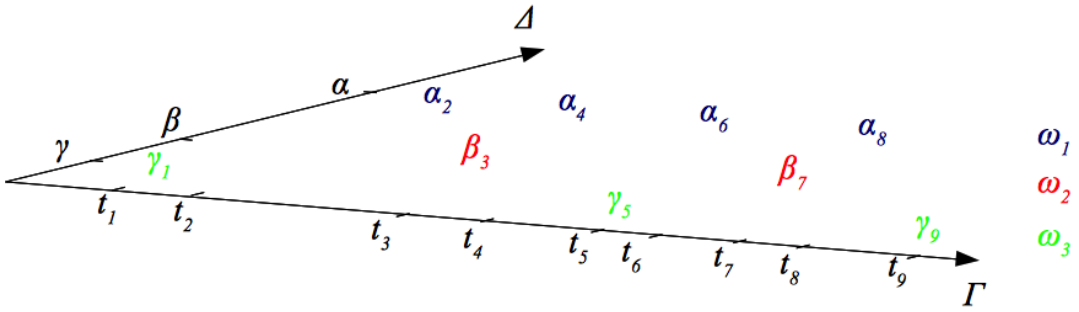


Figure 5.1: Example of a ternary superposition $\Omega = \omega_1 \cup \omega_2 \cup \omega_3$

Let us now provide the superposition $\Omega$ with the operation of addition under timed constrains $\overset{\Delta\tau_{ij}}{+}$ introduced in 4.4. Thus, the couple denoted $\mathcal{S} = (\Omega, \overset{\Delta\tau_{ij}}{+})$ is the algebraic structure of the observed process $(X(t), \Theta(X, \Delta))$.

This leads us to the following property:

**Property 18**

*Considering a m-ary superposition* $\Omega = \bigcup_{i \in [1;m]} \omega_i$ *of m sequences of timed observations* $\omega_i$*, produced by m independent programs* $\Theta_i(X_i, \Delta_i)$*, the algebraic structure* $\mathcal{S} = (\Omega, \overset{\Delta\tau_{ij}}{+})$ *of the observed process* $(X(t), \Theta(X, \Delta))$ *can always be built.*

## 5.4 Observable Space of the Observed Process

In section 5.2.2, it has been seen (property 17) that a m-ary superposition $\Omega = \bigcup_{i \in [1;m]} \omega_i$ can be modeled thanks to a collection of matrices $\Theta_s^S$, $s \in [1; \frac{m \times (m-1)}{2}]$, and a matrix $\Theta^E$. A matrix $\Theta_s^S$ is built from eight sequences of the form $\omega_{xy}^1$ and $\omega_{xy}^2$ with $(x, y) \in \{p, q\}^2$, $(p, q) \in [1; m]^2$. Such sequences are necessary and sufficient to build also the matrix $\Theta^E$.

Let us denote $\mathcal{I}_s$, the set containing the eight sequences denoted $\omega_{pp}^{1,s}$, $\omega_{pp}^{2,s}$, $\omega_{pq}^{1,s}$, $\omega_{pq}^{2,s}$, $\omega_{qp}^{1,s}$, $\omega_{qp}^{2,s}$, $\omega_{qq}^{1,s}$ and $\omega_{qq}^{2,s}$, corresponding to the $s^{th}$ binary superposition denoted $\omega_{pq}^s = \omega_p \cup \omega_q$:

$$\forall s \in [1; \frac{m \times (m-1)}{2}], \mathcal{I}_s = \{\omega_{pp}^{1,s}, \omega_{pp}^{2,s}, \omega_{pq}^{1,s}, \omega_{pq}^{2,s}, \omega_{qp}^{1,s}, \omega_{qp}^{2,s}, \omega_{qq}^{1,s}, \omega_{qq}^{2,s}\} \tag{5.4}$$

There are three superpositions of the form $\omega_p \cup \omega_q$ that can be built:

- $\omega_{12} = \omega_1 \cup \omega_2$,
- $\omega_{13} = \omega_1 \cup \omega_3$,
- $\omega_{23} = \omega_2 \cup \omega_3$.

From $\omega_{12}$, we can build the set $\mathcal{I}_{12} = \{\omega_{11}^1, \omega_{11}^2, \omega_{12}^1, \omega_{12}^2, \omega_{21}^1, \omega_{21}^2, \omega_{22}^1, \omega_{22}^2\}$ where:

$$\begin{array}{ll} \omega_{11}^1 = \{\alpha_2, \alpha_4, \alpha_6\} & \omega_{11}^2 = \{\alpha_4, \alpha_6, \alpha_8\} \\ \omega_{12}^1 = \{\alpha_2, \alpha_4, \alpha_6\} & \omega_{12}^2 = \{\beta_3, \beta_7, \beta_7\} \\ \omega_{21}^1 = \{\beta_3, \beta_7\} & \omega_{21}^2 = \{\alpha_4, \alpha_8\} \\ \omega_{22}^1 = \{\beta_3\} & \omega_{22}^2 = \{\beta_7\} \end{array} \tag{5.5}$$

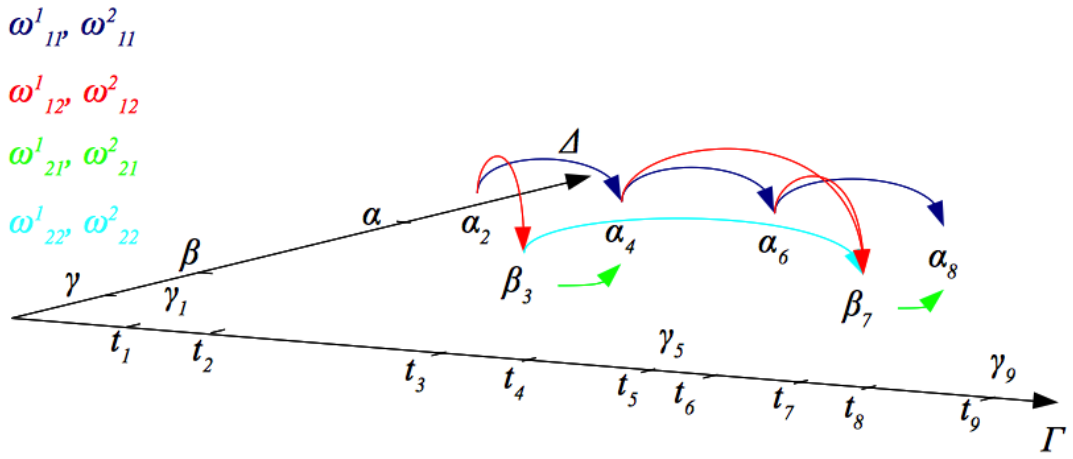


Figure 5.2: Sequences $\omega_{11}^1, \omega_{11}^2, \omega_{12}^1, \omega_{12}^2, \omega_{21}^1, \omega_{21}^2, \omega_{22}^1, \omega_{22}^2$ built from $\omega_{12} = \omega_1 \cup \omega_2$

From $\omega_{13}$, we can build the set $\mathcal{I}_{13} = \{\omega_{11}^1, \omega_{11}^2, \omega_{13}^1, \omega_{13}^2, \omega_{31}^1, \omega_{31}^2, \omega_{33}^1, \omega_{33}^2\}$ where:

$$\begin{array}{ll} \omega_{11}^1 = \{\alpha_2, \alpha_4, \alpha_6\} & \omega_{11}^2 = \{\alpha_4, \alpha_6, \alpha_8\} \\ \omega_{13}^1 = \{\alpha_2, \alpha_4, \alpha_6\} & \omega_{13}^2 = \{\gamma_5, \gamma_5, \gamma_9\} \\ \omega_{31}^1 = \{\gamma_1, \gamma_5\} & \omega_{31}^2 = \{\alpha_2, \alpha_6\} \\ \omega_{33}^1 = \{\gamma_1, \gamma_5\} & \omega_{33}^2 = \{\gamma_5, \gamma_9\} \end{array} \tag{5.6}$$

From $\omega_{23}$, we can build the set $\mathcal{I}_{23} = \{\omega_{22}^1, \omega_{22}^2, \omega_{23}^1, \omega_{23}^2, \omega_{32}^1, \omega_{32}^2, \omega_{33}^1, \omega_{33}^2\}$ where :

$$\begin{array}{ll} \omega_{22}^1 = \{\beta_3\} & \omega_{22}^2 = \{\beta_7\} \\ \omega_{23}^1 = \{\beta_3, \beta_7\} & \omega_{23}^2 = \{\gamma_5, \gamma_9\} \\ \omega_{32}^1 = \{\gamma_1, \gamma_5\} & \omega_{32}^2 = \{\beta_3, \beta_7\} \\ \omega_{33}^1 = \{\gamma_1, \gamma_5\} & \omega_{33}^2 = \{\gamma_5, \gamma_9\} \end{array} \tag{5.7}$$

Let us denote $\mathcal{T} = (\Omega, \mathcal{I})$, the couple composed of the superposition $\Omega$ and of the collection $\mathcal{I}$ such as:

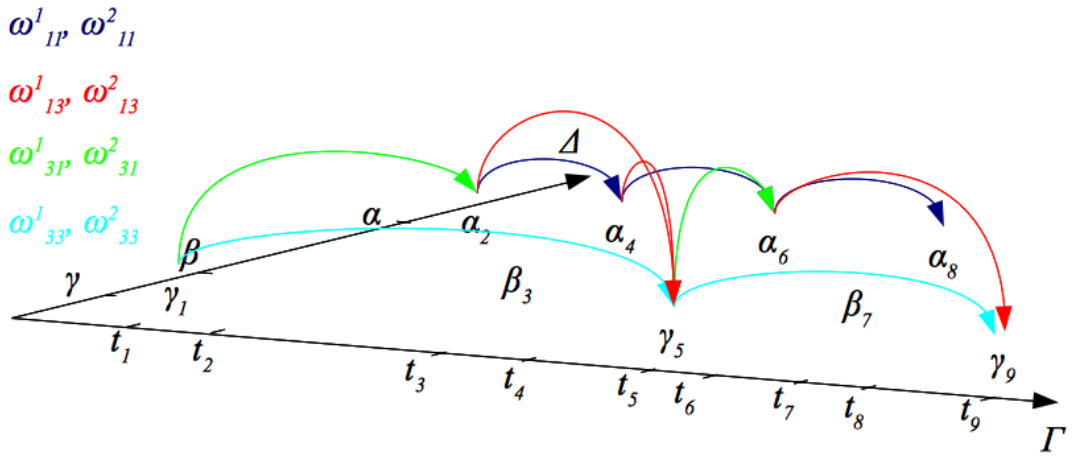


Figure 5.3: Sequences $\omega_{11}^1, \omega_{11}^2, \omega_{13}^1, \omega_{13}^2, \omega_{31}^1, \omega_{31}^2, \omega_{33}^1, \omega_{33}^2$ built from $\omega_{13} = \omega_1 \cup \omega_3$

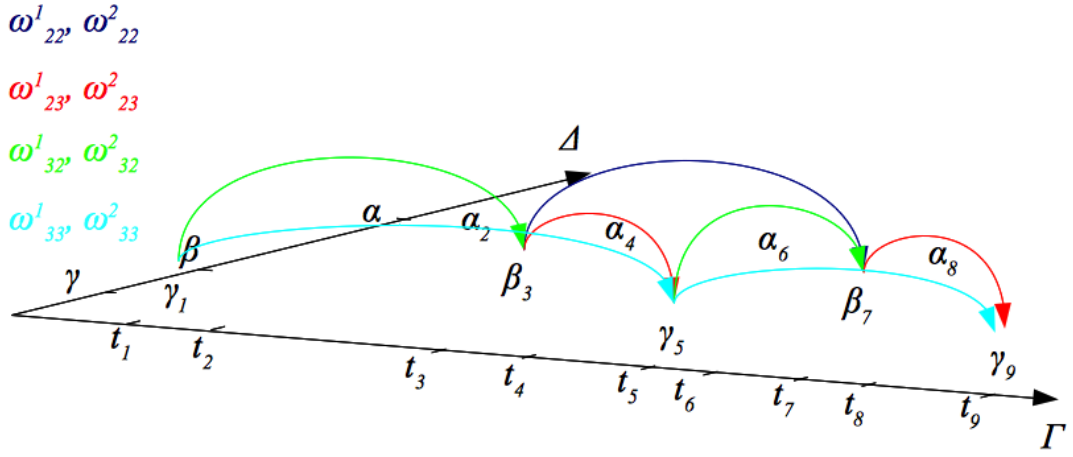


Figure 5.4: Sequences $\omega_{22}^1, \omega_{22}^2, \omega_{23}^1, \omega_{23}^2, \omega_{32}^1, \omega_{32}^2, \omega_{33}^1, \omega_{33}^2$ built from $\omega_{23} = \omega_2 \cup \omega_3$

$$\mathcal{I} = \left\{\emptyset, \Omega, (\mathcal{I}_s)_{s\in[1;\frac{m\times(m-1)}{2}]}\right\} \tag{5.8}$$

Each set $\mathcal{I}_s$ contains the eight sequences $\omega_{xy}^{1,s}$ and $\omega_{xy}^{2,s}$, $(x,y) \in \{p,q\}^2$ of the $s^{th}$ binary superposition $\omega_{pq}^s = \omega_p \cup \omega_q$, $(p,q) \in [1;m]^2$, $p \neq q$.

This leads us to the following definition:

**Definition 5.1** *Observable Space*
*Considering a m-ary superposition* $\Omega = \bigcup_{i\in[1;m]} \omega_i$ *of m sequences of timed observations* $\omega_i$*, the structure* $\mathcal{T} = (\Omega, \mathcal{I})$ *can be built such as* $\mathcal{I} = \left\{\emptyset, \Omega, (\mathcal{I}_s)_{s\in[1;\frac{m\times(m-1)}{2}]}\right\}$ *where* $\mathcal{I}_s$ *are the sets containing the eight sequences* $\omega_{xy}^{1,s}$ *and* $\omega_{xy}^{2,s}$, $(x,y) \in \{p,q\}^2$*, built from the* $s^{th}$ *binary superposition* $\omega_{pq}^s = \omega_p \cup \omega_q$ *between the sequences* $w_p$ *and* $w_q$ *for* $(p,q) \in [1;m]^2$.
*Such a structure* $\mathcal{T} = (\Omega, \mathcal{I})$ *is called the Observable Space of the observed process* $(X(t), \Theta(X,\Delta))$.

An *Observable Space* aims at distributing timed observations contained in the m-ary superposition $\Omega$ into sequences of the form $\omega_{xy}^{1,s}$ and $\omega_{xy}^{2,s}$, $(x,y) \in \{p,q\}^2$ for $(p,q) \in [1;m]^2$. Thus, Abstract Binary Observers contained in matrices $\Theta_k^S$, $k \in [1; \frac{m\times(m-1)}{2}]$ and $\Theta^E$ are able to observe binary sequences of such timed observations.

## 5.5 Abstract Chronicle Model of the Observed Process

Let us consider the observable space $\mathcal{T} = (\Omega, \mathcal{I})$ and let us have a closer look on the sets $(\mathcal{I}_s)_{s\in[1;\frac{m\times(m-1)}{2}]}$ contained in the collection $\mathcal{I}$ :

$$\mathcal{I}_s = \{\omega_{pp}^{1,s}, \omega_{pp}^{2,s}, \omega_{pq}^{1,s}, \omega_{pq}^{2,s}, \omega_{qp}^{1,s}, \omega_{qp}^{2,s}, \omega_{qq}^{1,s}, \omega_{qq}^{2,s}\} \tag{5.9}$$

The sequence $\omega_{pp}^{1,s}$ of $\omega_{pq}^{s} = \omega_p \cup \omega_q$ contains all timed observation of the form $O(t_{p_k}) \equiv (\delta_{p_k}, t_{p_k})$ such as $\delta_{p_k} \in \Delta_p$ and $t_{p_k} \in \Gamma_p$ for $p \in [1; m]$. The sequence $\omega_{pp}^{2,s}$ of $\omega_{pq}^{s} = \omega_p \cup \omega_q$ contains all timed observation of the form $O(t_{p_{k+1}}) \equiv (\delta_{p_{k+1}}, t_{p_{k+1}})$ such as $\delta_{p_{k+1}} \in \Delta_p$ and $t_{p_{k+1}} \in \Gamma_p$ for $p \in [1; m]$.

So, from $\omega_{pp}^{1,s}$ and $\omega_{pp}^{2,s}$, we can get all the binary sequences of timed observations of the form $(O(t_{p_k}), O(t_{p_{k+1}}))$ such as $O(t_{p_k}) \in \omega_p$ and $O(t_{p_{k+1}}) \in \omega_p$. The binary sequence of timed observations $(O(t_{p_k}), O(t_{p_{k+1}}))$ can be observed by the Abstract Binary Observer $\Theta_{pp}(\{\phi_{pp}\}, \{\delta^{pp}\})$ of the collection of matrices introduced in property 17.

Let us denote $O_{p_k} \equiv (x_p, \delta_{p_k})$, the observation class linking variable name $x_p \in X$ with constant $\delta_{p_k} \in \Delta_p$. Let us denote $O_{p_{k+1}} \equiv (x_p, \delta_{p_{k+1}})$, the observation class linking variable name $x_p \in X$ with constant $\delta_{p_{k+1}} \in \Delta_p$. So, the Abstract Binary Observers $\Theta_{pp}(\{\phi_{pp}\}, \{\delta^{pp}\})$ represents the temporal binary relation $r_u(O_{p_k}, O_{p_{k+1}}, \Delta\tau_{p_k p_{k+1}} \equiv [\tau^-_{p_k p_{k+1}}, \tau^+_{p_k p_{k+1}}])$ (see definition 3.8).

Thus, all binary sequences of timed observations of the form $(O(t_{p_k}), O(t_{p_{k+1}}))$ can be represented in a set $m_{pp}^s$ of $n_{pp}^s \in \mathbb{N}$ temporal binary relations $r_u(O_{p_k}, O_{p_{k+1}}, \Delta\tau_{p_k p_{k+1}})$:

$$m_{pp}^s = \{r_u(O_{p_k}, O_{p_{k+1}}, \Delta\tau_{p_k p_{k+1}})_{u\in[1;n_{pp}^s]}\} \tag{5.10}$$

According to definition 3.10, the set $m_{pp}^s$ represents the abstract chronicle model built from the sequences $\omega_{pp}^{1,s}$ and $\omega_{pp}^{2,s}$ of the $s^{th}$ binary superposition $\omega_{pq}^{s} = \omega_p \cup \omega_q$.

With the same reasoning, we can build the sets $m_{pq}^s$, $m_{qp}^s$ and $m_{qq}^s$ containing respectively $n_{pq}^s$, $n_{qp}^s$ and $n_{qq}^s$ temporal binary relations of the form $r_u(O_{p_k}, O_{q_{k+1}}, \Delta\tau_{p_k q_{k+1}})$, $r_u(O_{q_k}, O_{p_{k+1}}, \Delta\tau_{q_k p_{k+1}})$ and $r_u(O_{q_k}, O_{q_{k+1}}, \Delta\tau_{q_k q_{k+1}})$:

- $m_{pq}^s = \{r_u(O_{p_k}, O_{q_{k+1}}, \Delta\tau_{p_k q_{k+1}})_{u\in[1;n_{pq}^s]}\}$;
- $m_{qp}^s = \{r_u(O_{q_k}, O_{p_{k+1}}, \Delta\tau_{q_k p_{k+1}})_{u\in[1;n_{pq}^s]}\}$;
- $m_{qq}^s = \{r_u(O_{q_k}, O_{q_{k+1}}, \Delta\tau_{q_k q_{k+1}})_{u\in[1;n_{qq}^s]}\}$.

Where:

- $O_{q_k} \equiv (x_q, \delta_{q_k})$ is the observation class linking variable name $x_q \in X$ with constant $\delta_{q_k} \in \Delta_q$;
- $O_{q_{k+1}} \equiv (x_q, \delta_{q_{k+1}})$ is the observation class linking variable name $x_q \in X$ with constant $\delta_{q_{k+1}} \in \Delta_q$.

Thus, the abstract chronicle model built from the sets $\mathcal{I}_s$ is:

$$m^s = \bigcup_{(x,y)\in\{p,q\}^2} m^s_{xy} \tag{5.11}$$

And the abstract chronicle model of the observed process $(X(t), \Theta(X, \Delta))$ is:

$$\mathcal{M} = \bigcup_{s\in[1;\frac{m\times(m-1)}{2}]} m^s \tag{5.12}$$

The abstract chronicle model $\mathcal{M}$ contains all the temporal binary relations that could be observed in the m-ary superposition $\Omega$.

This leads us to the following property:

**Property 19**
*Given an observable space $\mathcal{T} = (\Omega, \mathcal{I})$ built from a m-ary superposition $\Omega$ of sequences of timed observations, we can build the abstract chronicle model $\mathcal{M}$ of the observed process $(X(t), \Theta(X, \Delta))$ containing all the temporal binary relations that are observable in $\Omega$.*

Let us consider the m-ary superposition $\Omega = \bigcup_{i\in[1;m]} \omega_i$ of $m$ sequences of timed observations $\omega_i$. Let us consider the $s^{th}$ binary superposition $\omega^s_{pq} = \omega_p \cup \omega_q, s \in [1; \frac{m\times(m-1)}{2}]$. Let be $N_p \in \mathbb{N}^\star$, the number of observation classes $O_{p_k}$, $k \in [1; N_p]$ associated to the sequence $\omega_p$. Let be $N_q \in \mathbb{N}^\star$, the number of observation classes $O_{q_k}$, $k \in [1; N_q]$ associated to the sequence $\omega_q$.

Such a binary superposition can be observed thanks to 4 Abstract Binary Observers:

- $\Theta_{pp}(\{\phi_{pp}\}, \{\delta^{pp}\})$ representing temporal binary relations of the form:

$$\forall k \in [1; N_p], \forall (k+1) \in [1; N_p], r_u(O_{p_k}, O_{p_{k+1}}, \Delta\tau_{p_k p_{k+1}}) \tag{5.13}$$

  Thus the number of temporal binary relations contained in the set $m^s_{pp}$ is:

$$Card(m^s_{pp}) = N_p \times N_p = N_p^2 \Rightarrow n^s_{pp} = N_p^2 \tag{5.14}$$

- $\Theta_{pq}(\{\phi_{pq}\}, \{\delta^{pq}\})$ representing temporal binary relations of the form:

$$\forall k \in [1; N_p], \forall (k+1) \in [1; N_q], r_u(O_{p_k}, O_{q_{k+1}}, \Delta\tau_{p_k q_{k+1}}) \tag{5.15}$$

  Thus the number of temporal binary relations contained in the set $m^s_{pq}$ is:

$$Card(m^s_{pq}) = N_p \times N_q \Rightarrow n^s_{pq} = N_p \times N_q \tag{5.16}$$

- $\Theta_{qp}(\{\phi_{qp}\}, \{\delta^{qp}\})$ representing temporal binary relations of the form:

$$\forall k \in [1; N_q], \forall (k+1) \in [1; N_p], r_u(O_{q_k}, O_{p_{k+1}}, \Delta\tau_{q_k p_{k+1}}) \tag{5.17}$$

  Thus the number of temporal binary relations contained in the set $m^s_{qp}$ is:

$$Card(m^s_{qp}) = N_q \times N_p \Rightarrow n^s_{qp} = N_q \times N_p \tag{5.18}$$

- $\Theta_{qq}(\{\phi_{qq}\},\{\delta^{qq}\})$ representing temporal binary relations of the form:

$$\forall k \in [1;N_q], \forall (k+1) \in [1;N_q], r_u(O_{q_k}, O_{q_{k+1}}, \Delta\tau_{q_k q_{k+1}}) \tag{5.19}$$

Thus the number of temporal binary relations contained in the set $m^s_{qq}$ is:

$$Card(m^s_{qq}) = N_q \times N_q = N_q^2 \Rightarrow n^s_{qq} = N_q^2 \tag{5.20}$$

Thus, the number of temporal binary relations contained in the set $m^s$ is:

$$Card(m^s) = N_p^2 + N_p \times N_q + N_q \times N_p + N_q^2 = (N_p + N_q)^2 \tag{5.21}$$

And then the number of temporal binary relations contained in the abstract chronicle model $\mathcal{M}$ is:

$$Card(\mathcal{M}) = \sum_{\substack{(p,q)\in[1;m]^2 \\ p\neq q}} (N_p + N_q)^2 \tag{5.22}$$

This leads to the following property:

**Property 20**
*Given a m-ary superposition* $\Omega = \bigcup_{i\in[1;m]} \omega_i$ *of m sequences of timed observations* $\omega_i$ *and denoting* $N_i \in \mathbb{N}^\star$*, the number of observation classes* $O_{i_k}$*,* $k \in [1;N_i]$ *associated to the sequence* $\omega_i$*, the number of temporal binary relations in the abstract chronicle model* $\mathcal{M}$ *is given by* $Card(\mathcal{M}) = \sum_{\substack{(p,q)\in[1;m]^2 \\ p\neq q}} (N_p + N_q)^2$.

A numerical application of this property is given in chapter 8.

## 5.6 Behaviour Model of the Observed Process

According to property 14, the Abstract Binary Observer $\Theta_{pq}(\{\phi_{pq}\},\{\delta^{pq}\})$ represents the sequential binary relation $(O(t_{p_k}), O(t_{q_{k+1}}))$ satisfying a temporal constraint $\Delta\tau_{p_k p_{k+1}}$, oriented from the sequence $\omega^{1,s}_{pq}$ built from a sequence of timed observations $\omega_p$ to a sequence $\omega^{2,s}_{pq}$ built from a sequence of timed observations $\omega_q$ implementing the following operation of addition under temporal constraints:

$$O_{pq}(t_{q_{k+1}}) = O(t_{p_k}) \overset{\Delta\tau_{p_k p_{k+1}}}{+} O(t_{q_{k+1}}) \tag{5.23}$$

Such an operation corresponds to application of the Modus Ponens with the following rule:

$$\begin{gathered}\theta_p(x_p, \delta_{p_k}, t_{p_k}) \wedge \theta_q(x_q, \delta_{q_{k+1}}, t_{q_{k+1}}) \wedge |t_{q_{k+1}} - t_{p_k}| \in \Delta\tau_{p_k q_{k+1}} \\ \Rightarrow \theta_{pq}(\phi_{pq}, \delta^{pq}, t_{q_{k+1}})\end{gathered} \tag{5.24}$$

The application of the Modus Ponens with the rule 5.24 fails iff:

$$(\theta_p(x_p, \delta_{p_k}, t_{p_k}) = false) \vee (\theta_q(x_q, \delta_{q_{k+1}}, t_{q_{k+1}}) = false) \vee (|t_{q_{k+1}} - t_{p_k}| \notin \Delta\tau_{p_k q_{k+1}}) \tag{5.25}$$

That is to say iff:

- the predicate $\theta_p(x_{\theta_p}, \delta_{\theta_p}, t_{\theta_p})$ is not assigned or
- the predicate $\theta_q(x_{\theta_q}, \delta_{\theta_q}, t_{\theta_q})$ is not assigned or
- the temporal constraints $\Delta\tau_{p_k q_{k+1}}$ are not satisfied.

In such cases, no sequential binary relation of the $(O(t_{p_k}), O(t_{q_{k+1}}))$ is observed in $\Omega$ by the Abstract Binary Observer $\Theta_{pq}(\{\phi_{pq}\}, \{\delta^{pq}\})$. The corresponding temporal binary relation $r_u(O_{p_k}, O_{q_{k+1}}, \Delta\tau_{p_k q_{k+1}})$ of the abstract chronicle model $\mathcal{M}$ is then also not observed.

In other cases, the Modus Ponens can be applied with the rule 5.24 and the Abstract Binary Observer $\Theta_{pq}(\{\phi_{pq}\}, \{\delta^{pq}\})$ writes the timed observation $O_{pq}(t_{q_{k+1}}) \equiv (\delta^{pq}, t_{q_{k+1}})$ corresponding, in an equivalent way, to:

- the assignation $\theta_{pq}(\phi_{pq}, \delta^{pq}, t_{q_{k+1}})$;
- the observation of the sequential binary relation $(O(t_{p_k}), O(t_{q_{k+1}}))$ in $\Omega$;
- the observation of the temporal binary relation $r_u(O_{p_k}, O_{q_{k+1}}, \Delta\tau_{p_k q_{k+1}})$ which is said to be an *observed relation* in $\Omega$ (see 3.9). Such an observed relation, denoted $r_u(O_{p_k}(t_{p_k}), O_{q_{k+1}}(t_{q_{k+1}}))$, is called an instance of the temporal binary relation $r_u(O_{p_k}, O_{q_{k+1}}, \Delta\tau_{p_k q_{k+1}})$.

Let us denote $\mathcal{B}$, the set containing the $n_{\mathcal{B}} \in \mathbb{N}$ observed relations $r_u(O_{p_k}(t_{p_k}), O_{q_{k+1}}(t_{q_{k+1}}))$ in $\Omega$:

$$\mathcal{B} = \{r_u(O_{p_k}(t_{p_k}), O_{q_{k+1}}(t_{q_{k+1}}))_{u\in[1;n_{\mathcal{B}}]}\} \tag{5.26}$$

The set $\mathcal{B}$ is called the *behaviour model* of the observed process $(X(t), \Theta(X, \Delta))$. Such a behaviour model $\mathcal{B}$ is called an *instance* of the abstract chronicle model $\mathcal{M}$. This leads us to the following property:

**Property 21**
*Any instance $\mathcal{B}$ of an abstract chronicle model $\mathcal{M}$ represents the behaviour model of the observed process $(X(t), \Theta(X, \Delta))$.*

## 5.7 Abstraction Process

Let us sum up elements we have for the given observed process $(X(t), \Theta(X, \Delta))$:

- a m-ary superposition $\Omega = \bigcup_{i\in[1;m]} \omega_i$, of $m$ sequences of timed observations $\omega_i$;
- an algebraic structure $\mathcal{S} = (\Omega, \overset{\Delta\tau_{ij}}{+})$;
- an observable space $\mathcal{T} = (\Omega, \mathcal{I})$;
- an abstract chronicle model $\mathcal{M}$ composed of $n_{\mathcal{M}}$ temporal binary relations;

- a behaviour model $\mathcal{B}$ composed of $n_{\mathcal{B}} \leq n_{\mathcal{M}}$ observed relations.

Let us now provide the definition of a *Level of Abstraction* of such an observed process $(X(t), \Theta(X, \Delta))$:

**Definition 5.2** *Level of Abstraction of an Observed Process*
*Given an observed process $(X(t), \Theta(X, \Delta))$, the Level of Abstraction (LoA) of such an observed process is the structure $\mathcal{L}$ composed of the algebraic structure $\mathcal{S} = (\Omega, \overset{\Delta\tau_{ij}}{+})$ and of the observable space $\mathcal{T} = (\Omega, \mathcal{I})$ of that observed process:*

$$\mathcal{L} = \langle \mathcal{S}, \mathcal{T} \rangle \tag{5.27}$$

Such a definition is consistent with the definition of a moderated LoA given by Floridi: *a moderated LoA is defined to consist of a LoA together with a behaviour at that LoA.*

From the algebraic structure $\mathcal{S}$, we have access to the superposition $\Omega$ that is to say to the timed observations $(\delta_{i_k}, t_{i_k})$. From timed observations $(\delta_{i_k}, t_{i_k})$, we have access to the constants $\delta_{i_k}$. From constants $\delta_{i_k}$, we have access to their variable names $x_i$ thanks to observations classes $O_{i_k} = \{x_i, \delta_{i_k}\}$. Such variable names $x_i$ correspond to Floridi's notion of observables (see notion 3) that he uses to give a definition of a LoA (see notion 4).

From the observable space $\mathcal{T} = (\Omega, \mathcal{I})$, we have access to the abstract chronicle model $\mathcal{M}$ and to its instance, the behaviour model $\mathcal{B}$. Such a behaviour model $\mathcal{B}$ corresponds to the notion of Floridi's behaviour of a system (see notion 5) that he uses to define a moderated LoA.

Let us now consider again the operation of addition under temporal constraints of 5.23 corresponding to the application of the Modus Ponens with the rule 5.24.

According to section 4.3, the implication 5.24 defines an *abstract process* to build assignations from other assignations by Modus Ponens. The timed observation $O_{pq}(t_{q_{k+1}}) \equiv (\delta^{pq}, t_{q_{k+1}})$ is *not produced by a canonical program* observing a timed function $\phi_{pq}(t)$ but by an Abstract Binary Observer $\Theta_{pq}(\{\phi_{pq}\}, \{\delta^{pq}\})$. The timed observation $O_{pq}(t_{q_{k+1}}) \equiv (\delta^{pq}, t_{q_{k+1}})$ has been *deducted* from timed observations produced by other programs, $\Theta_p(\{x_p\}, \Delta_p)$ and $\Theta_q(\{x_q\}, \Delta_q)$. The assignation $\theta_{pq}(\phi_{pq}, \delta^{pq}, t_{q_{k+1}})$ can also be used in an implication of the form 5.24: such an abstraction process can be realized recursively.

Let us then consider the LoA $\mathcal{L} = \langle \mathcal{S}, \mathcal{T} \rangle$ and let us have a closer look to the behaviour model $\mathcal{B}$. As seen in section 5.6, such a behaviour model $\mathcal{B}$ has been obtained by the application of the Modus Ponens with the rule 5.24 and is composed of $n_{\mathcal{B}}$ observed relations $r_u(O_{p_k}(t_{p_k}), O_{q_{k+1}}(t_{q_{k+1}}))$ or, in an equivalent way, of $n_{\mathcal{B}}$ timed observations of the form $O_{pq}(t_{q_{k+1}}) \equiv (\delta^{pq}, t_{q_{k+1}})$ written by $m^1 \leq n_{\mathcal{B}}$ Abstract Binary Observers $\Theta_{pq}(\{\phi_{pq}\}, \{\delta^{pq}\})$.

Let us denote $\Omega^1$, the sequence containing the $n_{\mathcal{B}}$ timed observations:

$$\Omega^1 = \{O_{pq}(t_{q_{k+1}}) \equiv (\delta^{pq}, t_{q_{k+1}}) \equiv O_u(t_{q_{k+1}}), u \in [1; n_{\mathcal{B}}]\} \tag{5.28}$$

The sequence $\Omega^1$ being produced by $m^1$ Abstract Binary Observers $\Theta_{pq}(\{\phi_{pq}\}, \{\delta^{pq}\})$, it can be partitioned, according to the Superposition Theorem 3.2, into $m^1$ sequences $\omega_i^1$. The sequence $\Omega^1$ is then the $m^1$-ary superposition:

$$\Omega^1 = \bigcup_{i \in [1; m^1]} \omega_i^1 \tag{5.29}$$

Where $\omega_i^1 = \{O_{pq}(t_{q_{k+1}}) \equiv (\delta^{pq}, t_{q_{k+1}}) \equiv O_i(t_{q_{k+1}}), i \in [1; n_i^1]\}$ is the sequence of $n_i^1$ timed observations $O_i(t_{q_{k+1}})$ produced by the Abstract Binary Observers $\Theta_i(\{\phi_{pq}\}, \{\delta^{pq}\})$.

Let us consider the observed process $(X^1(t), \Theta^1(X^1, \Delta^1))$ such as:

- $X^1(t) = \bigcup_{i \in [1; m^1]} X_i^1(t)$ where $X_i^1(t) = \phi_{pq}(t)$ for $(p, q) \in [1; m]^2$;
- $\Theta^1(X^1, \Delta^1)) = \bigcup_{i \in [1; m^1]} \Theta_i^1(X_i^1, \Delta_i^1))$ where:
  - $X_i^1 = \{\phi_{pq}\}$ for $(p, q) \in [1; m]^2$;
  - $\Delta_i^1 = \{\delta^{pq}\}$ for $(p, q) \in [1; m]^2$.

According to properties 18 and 5.1, we can build the algebraic structure $\mathcal{S}^1 = (\Omega^1, \overset{\Delta\tau_{ij}}{+})$ and the observable space $\mathcal{T}^1 = (\Omega^1, \mathcal{I}^1)$. According to definition 5.27, let us consider the LoA $\mathcal{L}^1 = \langle \mathcal{S}^1, \mathcal{T}^1 \rangle$. As seen in equation 4.36, a constant $\delta^{pq}$ of a timed observation $O_{pq}(t_{q_{k+1}})$ of the superposition $\Omega^1$ is equivalent, for all $k \in \mathbb{Z}$, (see equation 4.39) to a couple of constants $(\delta_{p_k}, \delta_{q_{k+1}})$ belonging respectively to timed observations $O(t_{p_k})$ and $O(t_{q_{k+1}})$ of the superposition $\Omega$:

$$\forall k \in \mathbb{Z}, \delta^{pq} \Leftrightarrow (\delta_{p_k}, \delta_{q_{k+1}}) \tag{5.30}$$

This means that the knowledge worn by the constant $\delta^{pq}$ is *semantically richer* than the knowledge worn by constants $\delta_{p_k}$ and $\delta_{q_{k+1}}$ for all $k \in \mathbb{Z}$. This thus means that the knowledge at the LoA $\mathcal{L}^1$ is also *semantically richer* than the knowledge at the LoA $\mathcal{L}$.

Such an equivalence also demonstrates that less constants are needed at the LoA $\mathcal{L}^1$ than at the LoA $\mathcal{L}$ in order to model an observed process. This means that the knowledge at the LoA $\mathcal{L}^1$ is *syntacticly poorer* than the knowledge contained at the LoA $\mathcal{L}$.

This is consistent with Floridi's point of view [Flo08]: *the quantity of information in a model varies with the LoA: a lower LoA, of greater resolution of finer granularity, produces a model that contains more information than a model produced at a higher, or more abstract, LoA.*

This allows us to affirm that the LoA $\mathcal{L}^1$ is higher, or more abstract, than the LoA $\mathcal{L}$ and leads us to the following property:

**Property 22**
*The existence of a behaviour model $\mathcal{B}$ at a LoA $\mathcal{L} = \langle \mathcal{S}, \mathcal{T} \rangle$ induces an abstraction process which allows to define a higher (or more abstract) LoA $\mathcal{L}^1 = \langle \mathcal{S}^1, \mathcal{T}^1 \rangle$. The knowledge at the LoA $\mathcal{L}^1 = \langle \mathcal{S}^1, \mathcal{T}^1 \rangle$ is semantically richer and syntacticly poorer than the knowledge at the LoA $\mathcal{L}$.*

Now all elements are in place to give the definition of a *Gradient of Abstraction*:

**Definition 5.3** *Gradient of Abstraction*
*Let us consider two integers $(i, j) \in \mathbb{N}^2$ such as $i < j$.*
*Let us consider the LoA $\mathcal{L}^i = \langle \mathcal{S}^i, \mathcal{T}^i \rangle$ of the observed process $(X^i(t), \Theta^i(X^i, \Delta^i))$.*
*Let be $\mathcal{B}^i$ the behaviour model of the observed process $(X^i(t), \Theta^i(X^i, \Delta^i))$ at the LoA $\mathcal{L}^i$.*
*Let us consider the LoA $\mathcal{L}^j = \langle \mathcal{S}^j, \mathcal{T}^j \rangle$ of the observed process $(X^j(t), \Theta^j(X^j, \Delta^j))$.*

*Let be $\mathcal{B}^j$ the behaviour model of the observed process $(X^j(t), \Theta^j(X^j, \Delta^j))$ at the LoA $\mathcal{L}^j$.*

*A Gradient of Abstraction (GoA), $\mathcal{G}$ consists of a finite collection $\mathcal{G} = \{\mathcal{L}^i, i \in \mathbb{N}\}$ of LoAs $\mathcal{L}^i$ such as:*

- *there exists a relation $\Delta^i \to \Delta^j$ linking sets of constants $\Delta^i$ and $\Delta^j$ for all $(i, j) \in \mathbb{N}^2$;*
- *there exists an inverse relation $\Delta^j \to \Delta^i$ linking sets of constants $\Delta^j$ and $\Delta^i$ for all $(i, j) \in \mathbb{N}^2$;*
- *the behaviour $\mathcal{B}^j$ at the LoA $\mathcal{L}^j$ is stronger than the behaviour $\mathcal{B}^i$ at the LoA $\mathcal{L}^i$ that is to say, $\mathcal{B}^j \Rightarrow \mathcal{B}^i$, for all $(i, j) \in \mathbb{N}^2$. In other words, knowing the behaviour at the LoA $j$ is knowing the behaviour at the LoA $i$.*

The existence of a relation $\Delta^i \to \Delta^j$ makes the link between sets of constants $\Delta^i$ and $\Delta^j$ during the abstraction process from $\mathcal{L}^i$ to $\mathcal{L}^j$. Reciprocally, the reverse relation $\Delta^j \to \Delta^i$ is useful when travelling from an abstract level $\mathcal{L}^j$ to a less abstract level $\mathcal{L}^i$: such a process is called a *reification process*. An example of a reification process is given in chapter 8.

## 5.8 Conclusion

Figure 5.5 illustrates the characteristic elements and the abstraction process in the TOT framework introduced in this chapter.

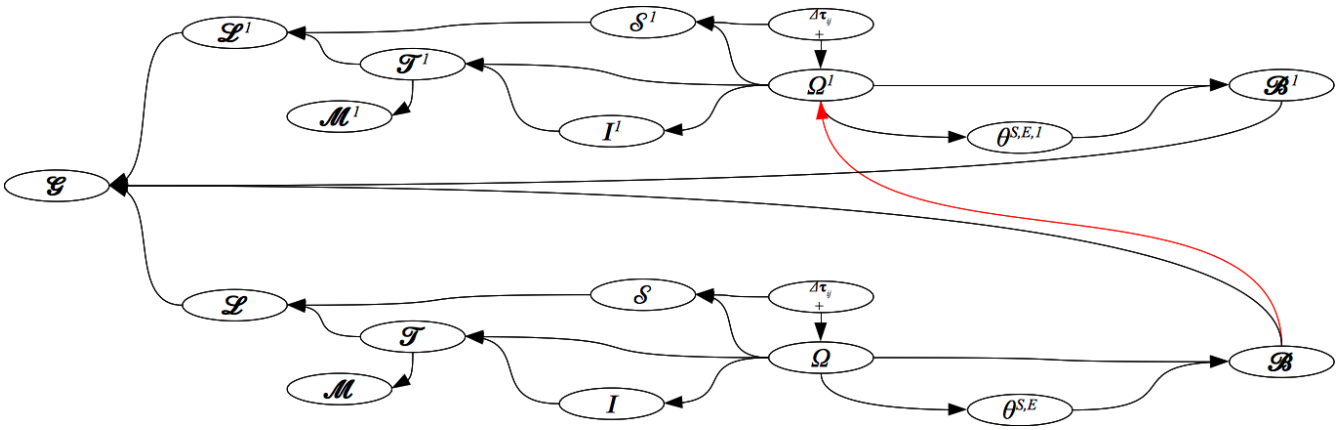


Figure 5.5: Characteristic elements and abstraction process in the TOT framework

From a m-ary superposition $\Omega$, we can build:

- the collection $\mathcal{I}$ related to such a superposition;
- the matrices $\Theta^S$ and $\Theta^E$ of Abstract Binary Observers, denoted $\Theta^{S,E}$ on this figure, modeling such a superposition.

The superposition $\Omega$ provided with the operation of addition under temporal constraints $\overset{\Delta\tau_{ij}}{+}$ builds the algebraic structure $\mathcal{S}$. The superposition $\Omega$ provided with the collection $\mathcal{I}$ builds the observable space $\mathcal{T}$ from which the abstract chronicle model $\mathcal{M}$ is built. The association of the observable space $\mathcal{T}$ and the algebraic structure $\mathcal{S}$ build the Level of Abstraction $\mathcal{L}$.

The application of the operation of addition under temporal constraints $\overset{\Delta\tau_{ij}}{+}$ on timed observations of the superposition $\Omega$ associated with the Abstract Binary Observers of the matrices $\Theta^S$ and $\Theta^E$ builds the behaviour model $\mathcal{B}$ and induces a process of abstraction (red arrow on this figure) generating a superposition $\Omega^1$ containing timed observations of a higher level of abstraction.

Given this superposition $\Omega^1$, characteristic elements $\mathcal{I}^1, \Theta^{S,E,1}, \mathcal{S}^1, \mathcal{T}^1, \mathcal{M}^1, \mathcal{L}^1, \mathcal{B}^1$ can be built again and another abstraction process can be performed. The associations of $\mathcal{L}$, $\mathcal{B}$, $\mathcal{L}^1$, $\mathcal{B}^1$ builds the gradient of abstraction $\mathcal{G}$.

Such operations can be repeated in a recursive way: the important point is that, whatever the level of abstraction is, the binary structure of a timed observation $(\delta_k, t_k)$ is invariant. The notion of timed observation plays the role of a *paradigm* like natural numbers $\mathbb{N}$ or real numbers $\mathbb{R}$.

Now that such an abstraction process has been introduced, we need to build a sampler in the framework of the TOT: this is the aim of the following chapter.

CHAPTER 6

# The TOT Sampler

## 6.1 Introduction

This chapter aims at demonstrating that an safe and canonical program $\Theta_i(\{x_i\}, \Delta_i)$ of an observed process $(x_i(t), \Theta_i(\{x_i\}, \Delta_i))$ observing an only one timed function $x_i(t)$ and implementing the Spacial Discretization Principle (section 3.5, equation 3.22), called a *Unary Observer*, plays the role of sampler device in the framework of Theory of Timed Observations (TOT).

To this goal, we first consider the usual Dirac's sampler in order to emphasize the algebraic structure of its mathematical framework. In a second step, we consider the TOT's Unary Observer in order to also emphasize the algebraic structure of its mathematical framework. In a last step, we demonstrate that there exists a homomorphism linking both these structures that allows us to conclude that such algebraic structures are of the same species and then that their respective elements play the same role.

## 6.2 Dirac's Sampler

Let us consider the Dirac distribution $\delta$ defined by its action on any test function $\varphi$ as:

$$\langle \delta, \varphi \rangle = \int_{-\infty}^{+\infty} \delta(t)\varphi(t)\mathrm{d}t = \varphi(0) \tag{6.1}$$

Let us recall that the convolution product $*$ between the Dirac distribution $\delta$ and a timed function $x_i$ allows to know any particular value $x_i(t_k) \equiv x_{i_k}$ of such a function:

$$\forall t_k \in \mathfrak{R}, (x_i * \delta)(t_k) = \int_{-\infty}^{+\infty} x_i(t)\delta(t_k - t)\,\mathrm{d}t = x_i(t_k) \tag{6.2}$$

Let us now consider a program denoted $\text{Ш}_T(x_i)$ implementing such a convolution product at regularly distributed timestamps $t_k = k.T, k \in \mathbb{Z}$. At each step $k \in \mathbb{Z}$, such a program generates the value $x_{i_k}$ which represents the value of the function $x_i(t)$ at the timestamp $t_k$:

$$x_i(t_k) \equiv x_{i_k} \tag{6.3}$$

Thus, such a program operates the following mapping:

$$\begin{array}{rcccc} \text{Ш}_T(x_i) & : & \mathbb{Z} & \rightarrow & \mathfrak{R} \\ & & k & \mapsto & x_{i_k} \end{array} \tag{6.4}$$

In fine, the $(N_i + 1) \in \mathbb{N}$ values $x_{i_k}, x_{i_{k+1}}, ..., x_{i_{k+N_i}}$ generated by such a program can be stored in set $X_i = \{x_{i_k}, x_{i_{k+1}}, ..., x_{i_{k+N_i}}\}$ ordered according to the step $k \in \mathbb{Z}$. Such a set is clearly a subset of $\mathfrak{R}$:

$$X_i \subseteq \mathfrak{R} \tag{6.5}$$

Figure 6.1 provides a representation of the sampling of a timed function $x(t)$ such as $t_k = k.T, k \in [-3; 3], T \in \mathfrak{R}$ and the storing set $X = \{x_{-3}, x_{-2}, x_{-1}, x_0, x_1, x_2, x_3\}$.

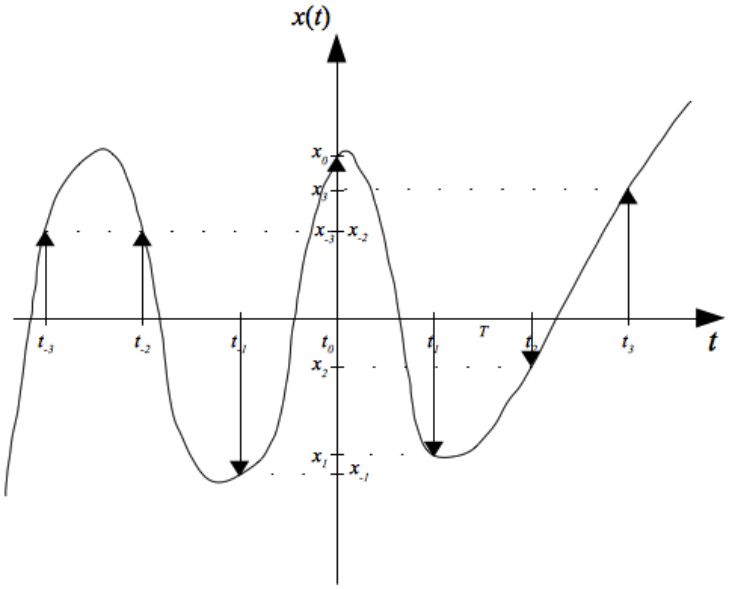


Figure 6.1: Dirac's sampler $\text{Ш}_T(x)$ applied on the timed function $x(t)$

The program $\text{Ш}_T(x_i)$ links then the set $\mathbb{Z}$ of integers with set $\mathfrak{R}$ of real numbers. The program $\text{Ш}_T(x_i)$ defines then a relation between elements of $\mathbb{Z}$ and elements of $\mathfrak{R}$ and is then a subset $\mathcal{S}_{\text{Ш}} \subseteq \mathbb{Z} \times \mathfrak{R}$:

$$\mathcal{S}_{\text{Ш}} = \{(k, x_{i_k}), k \in \mathbb{Z}, x_{i_k} \in \mathfrak{R}\} \subseteq \mathbb{Z} \times \mathfrak{R} \tag{6.6}$$

The set $\mathcal{S}_{\text{Ш}}$ defined by Dirac's sampler of figure 6.1 is then:

$$\mathcal{S}_{\text{Ш}} = \{(-3, x_{-3}), (-2, x_{-2}), (-1, x_{-1}), (0, x_0), (1, x_1), (2, x_2), (3, x_3)\} \subseteq \mathbb{Z} \times \mathfrak{R} \tag{6.7}$$

## 6.3 Unary Observer

Let us consider the dynamic process $X(t) = \{x_i(t)\}$ composed of only one timed function $x_i(t)$. Such a dynamic process implicitly defines a set $X = \{x_i\}$ of the variable name $x_i$. Let us consider an safe program $\Theta(X, \Delta)$ observing the dynamic process $X(t)$ and implementing the predicate $\theta(x_\theta, \delta_\theta, t_\theta)$ according to the Spacial Discretization Principle (equation 6.8) for a given threshold $\Psi_i \in \mathfrak{R}$:

$$\theta(x_\theta, \delta_\theta, t_\theta) \equiv x_\theta(t_{\theta-1}) < \Psi_i \wedge x_\theta(t_\theta) \geq \Psi_i \tag{6.8}$$

For each assignation $k \in \mathbb{Z}$ of the predicate $\theta(x_\theta, \delta_\theta, t_\theta)$ such as $\theta(x_i, \delta_i, t_k)$, Unary Observer $\Theta_i(\{x_i\}, \Delta_i)$ writes a timed observation of the form $(\delta_i, t_k)$, $\delta_i \in \Delta_i$ (equation 6.9):

$$x_i(t_{k-1}) < \Psi_i \wedge x_i(t_k) \geq \Psi_i \Rightarrow write((\delta_i, t_k)) \tag{6.9}$$

In such conditions, the program $\Theta(X, \Delta)$ is called a Unary Observer an is denoted $\Theta_i(\{x_i\}, \Delta_i)$, $\Theta_i(\{x_i\}, \{\delta_i\})$ or $\Theta_i$.

$$\Theta_i(\{x_i\}, \Delta_i) \equiv \Theta_i(\{x_i\}, \{\delta_i\}) \equiv \Theta_i \tag{6.10}$$

Such a Unary Observer operates then a mapping between the integer $k$ and the couple $(\delta_i, t_k)$.

$$\begin{array}{rccc} \Theta_i(\{x_i\}, \Delta_i) & : & \mathbb{Z} & \rightarrow & \Delta_i \times \Re \\ & & k & \mapsto & (\delta_i, t_k) \end{array} \tag{6.11}$$

In fine, the $(n_i + 1) \in \mathbb{N}$ couples $(\delta_i, t_k)$ are stored in a set denoted $\omega(t_{k+n_i}) = \{(\delta_i, t_k), \delta_i \in \Delta_i, t_k \in \Re\}$ ordered according to the step $k \in \mathbb{Z}$ and called a sequence of timed observations. Such a set $\omega(t_{k+n_i})$ is clearly a subset of $\Delta_i \times \Re$:

$$\omega(k + t_{n_i}) \subseteq \Delta_i \times \Re \tag{6.12}$$

Figure 6.2 provides a representation of a Unary Observer applied on a timed function $x(t)$ implementing the Spatial Discretization Principle with the threshold $\Psi \in \Re$. The sequence of timed observations is composed of three timed observations: $\omega(t_2) = \{(\delta, t_0), (\delta, t_1), (\delta, t_2)\}$.

The Unary Observer $\Theta_i(\{x_i\}, \Delta_i)$ is then a relation between elements of $\mathbb{Z}$ and elements of $\Delta_i \times \Re$. The Unary Observer $\Theta_i(\{x_i\}, \Delta_i)$ defines then a subset $\mathcal{S}_{\Theta_i} \subseteq \mathbb{Z} \times (\Delta_i \times \Re)$:

$$\mathcal{S}_{\Theta_i} = \{(k, (\delta_i, t_k)), k \in \mathbb{Z}, (\delta_i, t_k) \in \Delta_i \times \Re\} \subseteq \mathbb{Z} \times (\Delta_i \times \Re) \tag{6.13}$$

The set $\mathcal{S}_\Theta$ defined by Unary Observer of figure 6.2 is then:

$$\mathcal{S}_\Theta = \{(0, (\delta, t_0)), (1, (\delta, t_1)), (2, (\delta, t_2))\} \subseteq \mathbb{Z} \times (\Delta \times \Re) \tag{6.14}$$

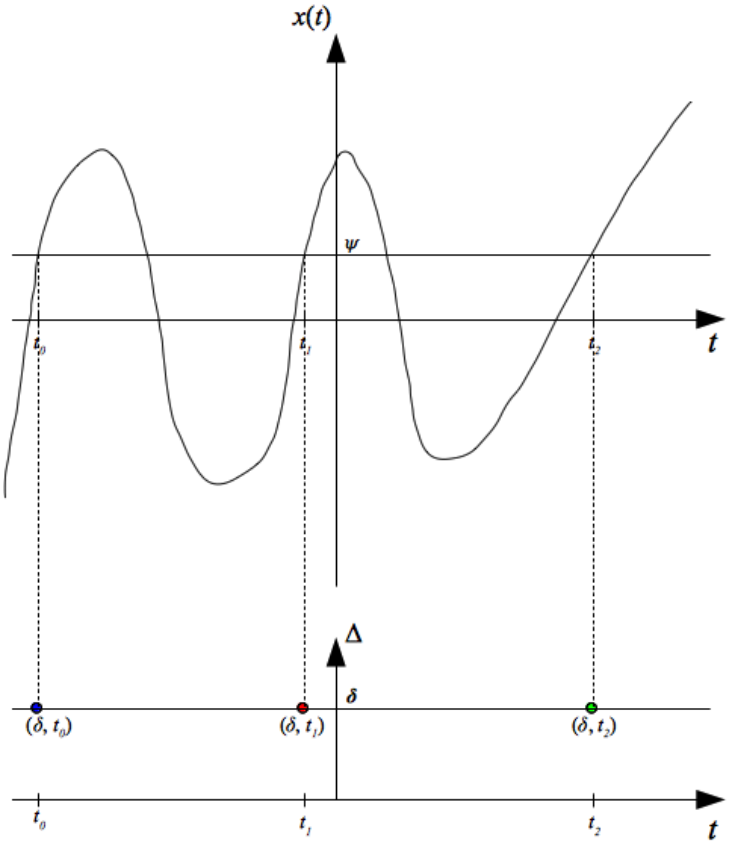


Figure 6.2: Unary Observer $\Theta(\{x\},\{\delta\})$ applied on the timed function $x(t)$ implementing the Spatial Discretization Principle with the threshold $\Psi$.

## 6.4 Algebraic Structure in the Dirac's Sampler Framework

Let us consider the timed function $x_i(t)$. As seen in section 6.2, sampled value $x_{i_k}$ of the function $x_i(t)$, produced by Dirac's sampler $\text{Ш}_T(x_i)$, is an element of $\mathfrak{R}$. Let us also consider Dirac's samplers $\text{Ш}_T(x_j)$ and $\text{Ш}_T(x_l)$, respectively sampling timed functions $x_j(t)$ and $x_l(t)$.

Let us now consider the algebraic structure $(\mathfrak{R},+)$ where $+$ is the usual addition on $\mathfrak{R}$. Such an algebraic structure allows to formulate the following properties about Dirac's sampler addition operation:

1. the addition operation of Dirac's samplers presents a neutral element.
   Let us consider the timed function $\forall t \in \mathfrak{R}, x_\phi(t) = 0$. The Dirac's sampler denoted $\text{Ш}_T(x_\phi)$ sampling such a function generates a sampling values equals to zero for each step $k \in \mathbb{Z}, t_k = k.T$. So such a sampler is the neutral element of Dirac's samplers addition operation:

$$\begin{gathered}\exists\ \text{Ш}_T(x_\phi)(k) \in \mathfrak{R}, \forall\ \text{Ш}_T(x_i)(k) \in \mathfrak{R}, \\ \text{Ш}_T(x_\phi)(k) + \text{Ш}_T(x_i)(k) = \text{Ш}_T(x_i)(k) + \text{Ш}_T(x_\phi)(k) = \text{Ш}_T(x_i)(k)\end{gathered} \tag{6.15}$$

2. the addition operation of Dirac's samplers is commutative:

$$\begin{array}{c} \forall\ \text{Ш}_T(x_i)(k) \in \Re, \forall\ \text{Ш}_T(x_j)(k) \in \Re, \\ \text{Ш}_T(x_i)(k) + \text{Ш}_T(x_j)(k) = \text{Ш}_T(x_j)(k) + \text{Ш}_T(x_i)(k) \end{array} \tag{6.16}$$

3. the addition operation of Dirac's samplers is associative:

$$\begin{array}{c} \forall\ \text{Ш}_T(x_i)(k) \in \Re, \forall\ \text{Ш}_T(x_j)(k) \in \Re, \forall\ \text{Ш}_T(x_l)(k) \in \Re, \\ (\text{Ш}_T(x_i)(k) + \text{Ш}_T(x_j)(k)) + \text{Ш}_T(x_l)(k) = \text{Ш}_T(x_j)(k) + (\text{Ш}_T(x_i)(k) + \text{Ш}_T(x_l)(k)) \end{array} \tag{6.17}$$

## 6.5 Algebraic Structure in the Unary Observer Framework

Let us consider again the timed function $x_i(t)$. As seen in section 6.3, the couple of values $(\delta_i, t_{k_i})$, produced by Unary Observer $\Theta_i(\{x_i\}, \Delta_i)$, is an element of $\Delta_i \times \Re$. Let us consider the Unary Observer $\Theta_j(\{x_j\}, \Delta_j)$, $\Delta_j = \{\delta_j\}$, observing the timed function $x_j(t)$. A couple of values $(\delta_j, t_{k_j})$, produced by such a Unary Observer, is an element of $\Delta_j \times \Re$. Let us consider the Unary Observer $\Theta_l(\{x_l\}, \Delta_l)$, $\Delta_l = \{\delta_l\}$, observing the timed function $x_l(t)$. A couple of values $(\delta_l, t_{k_l})$, produced by such a Unary Observer, is an element of $\Delta_l \times \Re$.

Let us consider the neutral Unary Observer $\Theta(\{x_\phi\}, \{\phi\})$ observing an unknown timed function $x_\phi(t)$. A couple of values $(\phi, t_\phi)$, produced by such a Unary Observer, is the neutral observation introduced in 4.2. Such a neutral Unary Observer maps any integer $k \in \mathbb{Z}$ with the neutral timed observation $(\phi, t_\phi)$:

$$\begin{array}{rccc} \Theta(\{x_\phi\}, \phi) & : & \mathbb{Z} & \rightarrow & \{\phi\} \times \Re \\ & & k & \mapsto & (\phi, t_\phi) \end{array} \tag{6.18}$$

It produces a neutral timed observation $(\phi, t_\phi)$ for each assignation $k \in \mathbb{Z}$ of the predicate $\theta(x_\theta, \delta_\theta, t_\theta)$ such as $\theta(x_i, \delta_i, t_k)$ (equation 6.19):

$$x_i(t_{k-1}) < \Psi_i \wedge x_i(t_k) \geq \Psi_i \Rightarrow write((\phi, t_\phi)) \tag{6.19}$$

Let us now consider the operation of addition of timed observations under temporal constraints $\overset{\Delta\tau_{ij}}{+}$ defined in 4.4. Let us consider the algebraic structure $(\Delta \times \Re, \overset{\Delta\tau_{ij}}{+})$ where $\Delta$ is the set containing all the constants associated with Unary Observers $\Theta_i(\{x_i\}, \Delta_i)$, $\Theta_j(\{x_j\}, \Delta_j)$, $\Delta_j = \{\delta_j\}$ and $\Theta(\{x_\phi\}, \{\phi\})$, that is to say, $\Delta = \Delta_i \cup \Delta_j \cup \Delta_l \cup \{\phi\}$.

In section 4.4 (see properties 6, 7, 8), it has been demonstrated that such an operation of addition has the following properties:

1. the neutral observation $(\phi, t_\phi)$ is the neutral element for the operation of addition under temporal constraints.

   Thus, for each assignation $k \in \mathbb{Z}$, we have:

$$\begin{array}{c} \exists\ \Theta(\{x_\phi\}, \phi)(k) \in \Delta \times \Re, \forall\ \Theta_i(\{x_i\}, \Delta_i)(k) \in \Delta \times \Re, \\ \Theta(\{x_\phi\}, \phi)(k) \overset{\Delta\tau_{ij}}{+} \Theta_i(\{x_i\}, \Delta_i)(k) = \Theta_i(\{x_i\}, \Delta_i)(k) \overset{\Delta\tau_{ij}}{+} \Theta(\{x_\phi\}, \phi)(k) \end{array} \tag{6.20}$$

2. the operation of addition under temporal constraints is commutative: Thus, for each assignation $k \in \mathbb{Z}$, we have:

$$\begin{array}{c} \forall\ \Theta_i(\{x_i\},\Delta_i)(k) \in \Delta \times \Re, \forall\ \Theta_j(\{x_j\},\Delta_j)(k) \in \Delta \times \Re, \\ \Theta_i(\{x_i\},\Delta_i)(k) + \Theta_j(\{x_j\},\Delta_j)(k) = \Theta_j(\{x_j\},\Delta_j)(k) + \Theta_i(\{x_i\},\Delta_i)(k) \end{array} \tag{6.21}$$

3. the operation of addition under temporal constraints is associative:

$$\begin{array}{c} \forall\ \Theta_i(\{x_i\},\Delta_i)(k) \in \Delta \times \Re, \forall\ \Theta_j(\{x_j\},\Delta_j)(k) \in \Delta \times \Re, \forall\ \Theta_l(\{x_l\},\Delta_l)(k) \in \Delta \times \Re, \\ (\Theta_i(\{x_i\},\Delta_i)(k) + \Theta_j(\{x_j\},\Delta_j)(k)) + \Theta_l(\{x_l\},\Delta_l)(k) = \\ \Theta_i(\{x_i\},\Delta_i)(k) + (\Theta_j(\{x_j\},\Delta_j)(k) + \Theta_l(\{x_l\},\Delta_l)(k)) \end{array} \tag{6.22}$$

## 6.6 Homomorphism between Algebraic Structures $(\Re, +)$ and $(\Delta \times \Re, \overset{\Delta\tau_{ij}}{+})$

Let us sum up the situation. On one hand, we have an algebraic structure denoted $(\Re, +)$ whose elements are real numbers generated by Dirac's sampler devices. The operation $+$ presents a neutral element, is commutative and is associative. On the other hand, we have another algebraic structure denoted $(\Delta \times \Re, \overset{\Delta\tau_{ij}}{+})$ whose elements are timed observations generated by Unary Observers. The operation $\overset{\Delta\tau_{ij}}{+}$ presents also a neutral element, is also commutative and is also associative.

We have then two algebraic structures of the same species, of the same shape. It is then natural to build a homomorphism between them in order to compare their respective elements.

Let us denote $\Phi$, the morphism mapping real numbers $x_{i_k}$ produces by Dirac's samplers $\text{Ш}_T(x_i)$, and timed observations $(\delta_i, t_{i_k})$, produced by Unary Observers $\Theta_i(\{x_i\},\Delta_i)$:

$$\begin{array}{rccc} \Phi & : & \Re & \rightarrow & \Delta \times \Re \\ & & x_{i_k} & \mapsto & (\delta_i, t_{i_k}) \end{array} \tag{6.23}$$

Figure 6.3 provides a representation of such a homomorphism:

On this example, the homomorphism $\Phi$ does the following mapping:

$$\begin{array}{rccc} \Phi & : & \Re & \rightarrow & \Delta \times \Re \\ & & x_{-3} & \mapsto & (\delta, t_0) \\ & & x_{-2} & \mapsto & (\delta, t_0) \\ & & x_{-1} & \mapsto & (\delta, t_0) \\ & & x_0 & \mapsto & (\delta, t_1) \\ & & x_1 & \mapsto & (\delta, t_1) \\ & & x_2 & \mapsto & (\delta, t_1) \\ & & x_3 & \mapsto & (\delta, t_2) \end{array} \tag{6.24}$$

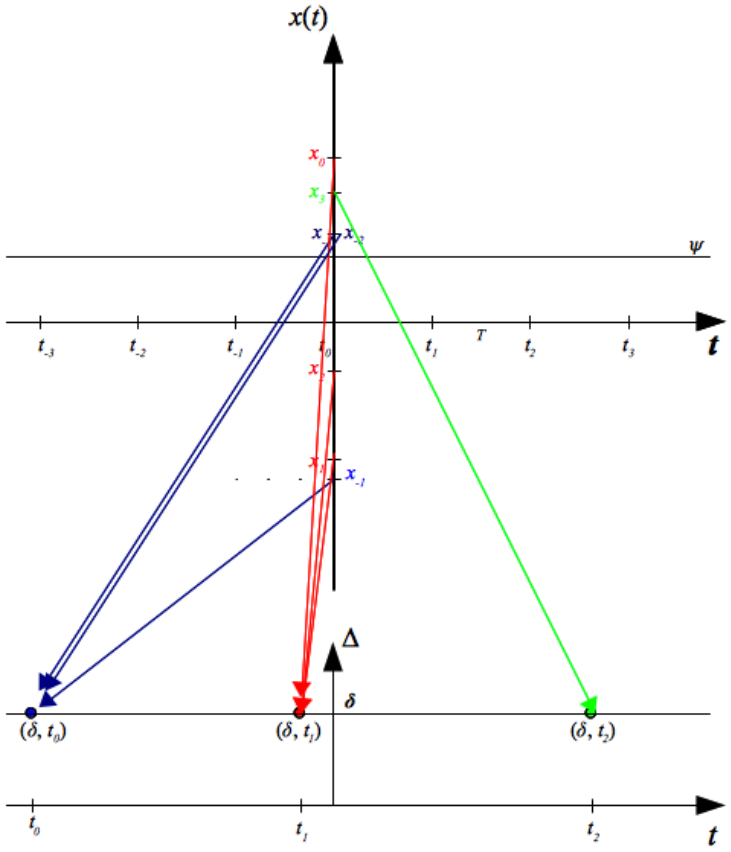


Figure 6.3: Representation of the homomorphism $\Phi : \Re \rightarrow \Delta \times \Re$ mapping sampled values of $x(t)$ produced by Dirac's sampler with timed observations produced by a Unary Observer observing the same function.

Such a homomorphism maps neutral elements of each structure:

$$\begin{array}{rcccc} \Phi & : & \Re & \rightarrow & \Delta \times \Re \\ & & 0 & \mapsto & (\phi, t_\phi) \end{array} \tag{6.25}$$

Such a homomorphism preserves operations of each structure:

$$\begin{array}{c} \forall x_{i_k} \in \Re, \forall x_{j_k} \in \Re, \exists (\delta_i, t_{i_k}) \in \Delta \times \Re, \exists (\delta_j, t_{j_k}) \in \Delta \times \Re, \\ \Phi(x_{i_k} + x_{j_k}) = \Phi(x_{i_k}) \overset{\Delta\tau_{ij}}{+} \Phi(x_{j_k}) = (\delta_i, t_{i_k}) \overset{\Delta\tau_{ij}}{+} (\delta_j, t_{j_k}) \end{array} \tag{6.26}$$

The important point here is that the existence of such a homomorphism demonstrates that programs producing real numbers in the algebraic structure $(\Re, +)$ and timed observations in the algebraic structure $(\Delta \times \Re, \overset{\Delta\tau_{ij}}{+})$ play the same role: programs of the algebraic structure $(\Re, +)$ being samplers, we can affirm then that Unary Observers play also the role of samplers in the algebraic structure $(\Delta \times \Re, \overset{\Delta\tau_{ij}}{+})$. An application can be found in [Ahb10b].

## 6.7 Conclusion

This chapter demonstrates that a Unary Observer $\Theta_i(\{x_i\}, \{\delta_i\})$ plays the role of a *sampler device* of the framework of the TOT.

The general form of a Unary Observer has been given with the equation 3.26 of the section 3.5:

$$\theta_i(x_i, \delta_j, t_k) \Rightarrow write(O(t_k)) \tag{6.27}$$

This form is important because a Unary Observer can be very simple or very complex. For example, the sampler used to solve the bank problem of chapter 8 is the following:

$$x_i(t_{k-1}) \neq x_i(t_k) \Rightarrow write((\delta_i, t_k)) \tag{6.28}$$

Another example can be found in appendix A.

Maybe this TOT sampler is the simplest one. In industry, most of the samplers implement the equation 3.22 of section 3.5:

$$x_i(t_{k-1}) < \Psi_j \wedge x_i(t_k) \geq \Psi_j \Rightarrow write((\delta_i, t_k)) \tag{6.29}$$

Such industrial samplers are also simple. Nevertheless, for example, numerous of very complex TOT samplers have been designed and implemented in the Sachem system (cf. [LeG04] for a description of some of them). The papers [LT98, LTT98] describe one of the most complex Unary Observer of the Sachem system, which is made of an architecture of $2 \times 9$ neural networks of various kinds (mainly multilayers perceptron and Kohonen Cards) to solve a particularly difficult problem of perception.

Now all elements are in place to build the TOT Category. Using the Category Theory is of a main importance in order to keep on formalizing the concept of abstraction introduced in chapter 5.

CHAPTER 7

# The TOT Category

## 7.1 Introduction

Let us go back to Merker's lecture notes of 1983 [Mer83]. Merker recalls that in science a *model* $\mathcal{M}$ is used to represent a *concrete situation* $\mathcal{C}$ so that the *transformation* $\mathfrak{F}$ linking the situation $\mathcal{C}$ to its model $\mathcal{M}$ expresses an *analogy report*. Merker then claims that such an analogy report is like a *functor*. As a consequence, according to Merker, only the model $\mathcal{M}$ is well defined. Generally speaking, the situation $\mathcal{C}$ and consequently the transformation $\mathfrak{F}$ is less well defined. Merker's point of view is clearly a *praxeological* one inspired from Halbwachs' book [Hal74]. The concrete situation $\mathcal{C}$ is substituted with a *praxeological* situation, that is to say, an *experimental* situation followed by a *production* situation of an object: *in each case, the concrete situation* $\mathcal{C}$ *consists in an operating mode defining materially the succession and the sequence of experimental or production transformations*. To illustrate this position, Merker uses two examples, the first belonging to Physics, the second to Human Sciences: (i) the example of an optical system provided by Halbwacks' book and (ii) Lévi-Strauss elementary structures of kinship. With these two examples, Merker illustrates what is, for us, a key point when considering the usage of the Category Theory: generally speaking, the model $\mathcal{M}$ used to represent a situation is a set of models $\mathcal{M}_i$ with the adequate homomorphisms $\mathfrak{F}_{ij}$ linking a model $\mathcal{M}_i$ to another $\mathcal{M}_j$. And Merker to say that *the model* $\mathcal{M}$ *is a category made of simple algebraic models* $\mathcal{M}_i$ *with the associated homomorphisms* $\mathfrak{F}_{ij}$. To sum up, Merker claims that the concrete situation $\mathcal{C}$ to study acts like a *category* $\mathcal{C}$ *of effective physical transformations* and that *in front of this category* $\mathcal{C}$*, can be presented a mathematical category* $\mathcal{M}$. The Timed Observation Theory (chapters 3, 4) represents then a natural mathematical framework to be such a mathematical category $\mathcal{M}$. The algebraic properties, the existence of an observable space, the existence of an abstraction process inherent to the concept of timed observations as a paradigm (chapter 5), the existence of a sampler (Unary Observer, chapter 6) in the TOT framework, provide the tools to formalize the category of model $\mathcal{M}$ described by Merker. Such a mathematical category of model $\mathcal{M}$ is called the *TOT Category*.

## 7.2 Characteristic Elements of the TOT Category

Let us consider the timed function $x_i(t)$ being observed by the Unary Observer $\Theta_i(\{x_i\}, \Delta_i)$. The set $\Delta_i = \{\delta_{i_k}, k \in \mathbb{Z}\}$ is the set of constants $\delta_{i_k}$ that variable name $x_i$ can take. Let us recall that the Unary Observer $\Theta_i(\{x_i\}, \Delta_i)$ implements a predicate $\theta_i(x_{\theta_i}, \delta_{\theta_i}, t_{\theta_i})$ according to the Spacial Discretization Principle 6.8. Let us recall that for each assignation $k \in \mathbb{Z}$ of

the predicate $\theta_i(x_{\theta_i}, \delta_{\theta_i}, t_{\theta_i})$ such as $\theta_i(x_i, \delta_{i_k}, t_{i_k})$, Unary Observer $\Theta_i(\{x_i\}, \Delta_i)$ writes a timed observation denoted $O_{i_k}(t_{i_k}) \equiv (\delta_{i_k}, t_{i_k})$. Let be $O_{i_k} = \{(x_i, \delta_{i_k})\}$, the observation class linking variable name $x_i$ with constant $\delta_{i_k}$. Let us denote $\Gamma_i = \{t_{i_k}\}$, the stochastic clock containing timestamps $t_{i_k} \in \Re$. Let be $\omega_i(t_{n_i}) = \{O_{i_k}(t_{i_k}) \equiv (\delta_{i_k}, t_{i_k}), \delta_{i_k} \in \Delta_i, t_{i_k} \in \Gamma_i\}$, the sequence of $n_i \in \mathbb{N}$ timed observations produces by the Unary Observer $\Theta_i(\{x_i\}, \Delta_i)$. Such a sequence $\omega_i(t_{n_i})$ is a subset of $\Delta_i \times \Gamma_i$. In the following, a Unary Observer $\Theta_i(\{x_i\}, \Delta_i)$ and a sequence of timed observations $\omega_i(t_{n_i})$ may be rewritten $\Theta_i(\{x_i\}, \Delta_i) \equiv \Theta_i(x_i, \Delta_i) \equiv \Theta_i$ and $\omega_i(t_{n_i}) \equiv \omega_i$ in order to lighten their writings.

The TOT Category consists of the following entities:

- a collection, denoted $ob(TOT) = \{\Theta_i(x_i, \Delta_i), i \in [1; n_{TOT}]\}$, of objects made of $n_{TOT} \in \mathbb{N}$ Unary Observers $\Theta_i(x_i, \Delta_i)$:

$$\begin{array}{rcccc} \Theta_i(x_i, \Delta_i) & : & k \in \mathbb{Z} & \rightarrow & \Delta_i \times \Gamma_i \\ & & k & \mapsto & O_{i_k}(t_{i_k}) \equiv (\delta_{i_k}, t_{i_k}) \end{array} \tag{7.1}$$

- a collection, denoted $mor(TOT) = \{f_{ij}, (i,j) \in [1; n_{TOT}]^2\}$, of morphism $f_{ij}$ linking two objects, that is to say, two Unary Observers $\Theta_i(x_i, \Delta_i)$ and $\Theta_j(x_j, \Delta_j)$ of $ob(TOT)$:

$$\begin{array}{c} \forall \Theta_i(x_i, \Delta_i) \in ob(TOT), \forall \Theta_j(x_j, \Delta_j) \in ob(TOT), \exists f_{ij} \in mor(TOT), \\ \begin{array}{rcccc} f_{ij} & : & \Theta_i(x_i, \Delta_i) & \rightarrow & \Theta_j(x_j, \Delta_j) \\ & & O_{i_k}(t_{i_k}) \equiv (\delta_{i_k}, t_{i_k}) & \mapsto & O_{j_l}(t_{j_l}) \equiv (\delta_{j_l}, t_{j_l}) \end{array} \end{array} \tag{7.2}$$

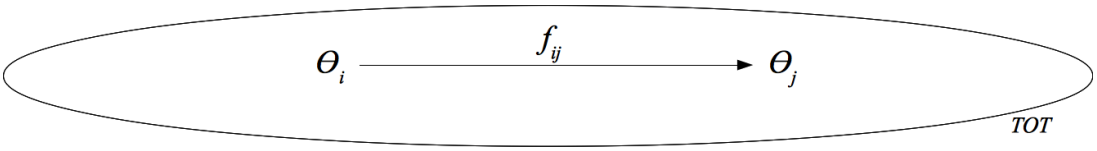


Figure 7.1: Representation of objects $\Theta_i(x_i, \Delta_i)$, $\Theta_j(x_j, \Delta_j)$ and morphism $f_{ij}$ in the TOT Category

- an identity morphism for each object:

$$\begin{array}{c} \forall \Theta_i(x_i, \Delta_i) \in ob(TOT), \exists f_{ii} \in mor(TOT), \\ \begin{array}{rcccc} f_{ii} & : & \Theta_i(x_i, \Delta_i) & \rightarrow & \Theta_i(x_i, \Delta_i) \\ & & O_{i_k}(t_{i_k}) \equiv (\delta_{i_k}, t_{i_k}) & \mapsto & O_{i_k}(t_{i_k}) \equiv (\delta_{i_k}, t_{i_k}) \end{array} \end{array} \tag{7.3}$$

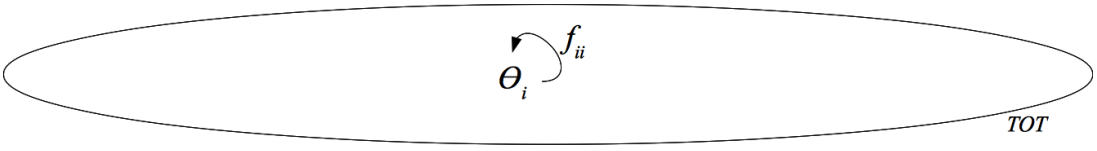


Figure 7.2: Representation of the object $\Theta_i(x_i, \Delta_i)$ and its identity morphism $f_{ii}$ in the TOT Category

- a binary operation, denoted $\circ$, called composition of morphisms such as:

$$\begin{array}{c} \forall f_{ij} : \Theta_i(x_i, \Delta_i) \rightarrow \Theta_j(x_j, \Delta_j), \forall f_{jk} : \Theta_j(x_j, \Delta_j) \rightarrow \Theta_k(x_k, \Delta_k) \\ f_{jk} \circ f_{ij} : \Theta_i(x_i, \Delta_i) \rightarrow \Theta_k(x_k, \Delta_k) \end{array} \tag{7.4}$$

Such a composition of morphisms satisfies the following axioms:

1. the composition of morphisms is associative:

$$\begin{gathered}\forall f_{ij} : \Theta_i(x_i, \Delta_i) \rightarrow \Theta_j(x_j, \Delta_j), \\ \forall f_{jk} : \Theta_j(x_j, \Delta_j) \rightarrow \Theta_k(x_k, \Delta_k), \\ \forall f_{kl} : \Theta_k(x_k, \Delta_k) \rightarrow \Theta_l(x_l, \Delta_l), \\ f_{kl} \circ (f_{jk} \circ f_{ij}) = (f_{kl} \circ f_{jk}) \circ f_{ij}\end{gathered} \tag{7.5}$$

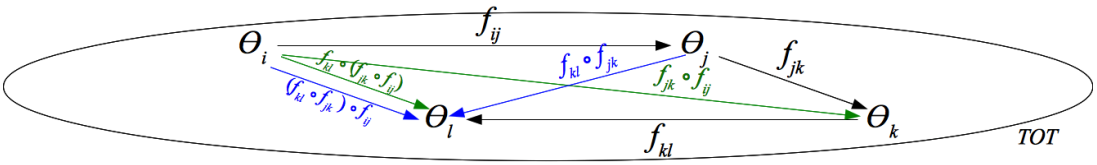


Figure 7.3: Representation of the composition of morphisms in the TOT Category

2. the neutral element of the composition of morphisms is the identity morphism:

$$\begin{gathered}\exists f_{ii} : \Theta_i(x_i, \Delta_i) \rightarrow \Theta_i(x_i, \Delta_i), \exists f_{jj} : \Theta_j(x_j, \Delta_j) \rightarrow \Theta_j(x_j, \Delta_j), \\ \forall f_{ij} : \Theta_i(x_i, \Delta_i) \rightarrow \Theta_j(x_j, \Delta_j), \\ f_{jj} \circ f_{ij} = f_{ij} \circ f_{ii}\end{gathered} \tag{7.6}$$

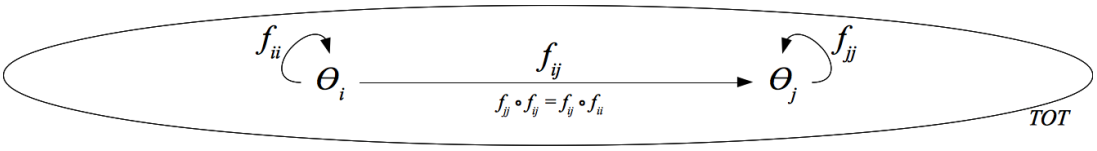


Figure 7.4: Representation of the neutral element for the composition of morphisms in the TOT Category

- a binary operation, denoted $\bigoplus$, called composition of objects, that is to say, composition of observers defined in 4.5 such as:

$$\begin{gathered}\forall \Theta_i(x_i, \Delta_i) \in ob(TOT), \forall \Theta_j(x_j, \Delta_j) \in ob(TOT), \\ \exists \Theta_k(x_k, \Delta_k) \in ob(TOT), \\ \Theta_k(x_k, \Delta_k) = \Theta_i(x_i, \Delta_i) \bigoplus \Theta_j(x_j, \Delta_j)\end{gathered} \tag{7.7}$$

Let us recall that such an operation defines an abstract observer $\Theta_k(x_k, \Delta_k)$ implementing a predicate $\theta_k(x_{\theta_k}, \delta_{\theta_k}, t_{\theta_k})$ producing abstract timed observation $O_{k_l} \equiv (\delta_{k_l}, t_{k_l})$, $\delta_{k_l} \in \Delta_k$, $t_{k_l} \in \Gamma_k$. Such a timed observation $O_{k_l}$ is the assignation $\theta_k(x_k, \delta_{k_l}, t_{k_l})$ of the predicate $\theta_k(x_{\theta_k}, \delta_{\theta_k}, t_{\theta_k})$ produced by the application of the Modus Ponens with the following rule:

$$\begin{gathered}\forall t_{i_m} \in \Gamma_i, \forall t_{j_n} \in \Gamma_j, \\ \theta_i(x_i, \delta_{i_m}, t_{i_m}) \wedge \theta_j(x_j, \delta_{j_n}, t_{j_n}) \wedge |t_{j_n} - t_{i_m}| \in \Delta\tau_{i_m j_n} \\ \Rightarrow \exists \delta_{k_l} \in \Delta_k, \theta_k(x_k, \delta_{k_l}, t_{k_l}) \wedge t_{k_l} \in \{t_{i_m}, t_{j_n}\}\end{gathered} \tag{7.8}$$

Such a composition of objects satisfies the following axioms:

1. the composition of objects is associative (see 4.22):

$$\begin{gathered}\forall\Theta_i(x_i,\Delta_i)\in ob(TOT),\\ \forall\Theta_j(x_j,\Delta_j)\in ob(TOT),\\ \forall\Theta_k(x_k,\Delta_k)\in ob(TOT),\\ \Theta_i(x_i,\Delta_i)\bigoplus(\Theta_j(x_j,\Delta_j)\bigoplus\Theta_k(x_k,\Delta_k))=(\Theta_i(x_i,\Delta_i)\bigoplus\Theta_j(x_j,\Delta_j))\bigoplus\Theta_k(x_k,\Delta_k)\end{gathered}\tag{7.9}$$

2. the neutral element of the composition of objects is the *the neutral observer* $\Theta_\Phi(\{x_\phi\},\{\phi\})$ (see 4.20):

$$\begin{gathered}\exists\Theta_\Phi(\{x_\phi\},\{\phi\})\in ob(TOT),\\ \forall\Theta_i(x_i,\Delta_i)\in ob(TOT),\\ \Theta_\Phi(\{x_\phi\},\{\phi\})\bigoplus\Theta_i(x_i,\Delta_i)=\Theta_i(x_i,\Delta_i)\bigoplus\Theta_\Phi(\{x_\phi\},\{\phi\})=\Theta_i(x_i,\Delta_i)\end{gathered}\tag{7.10}$$

The operation of composition of observers $\bigoplus$ is based on the addition under temporal constraints of timed observations which is the key element in the process of abstraction introduced in chapter 5. The operation of composition of observers $\bigoplus$ is then the key element in the process of abstraction in the TOT Category.

## 7.3 The Categories of the TOT

Let us consider the TOT Category whose objects are two Unary Observers $\Theta_i(x_i,\Delta_i)$ and $\Theta_j(x_j,\Delta_j)$. Let us consider the binary superposition $\Omega$ such as $\Omega=\omega_i\cup\omega_j$. As seen in chapter 5, property 18, we can build the algebraic structure $\mathcal{S}=(\Omega,\overset{\Delta\tau_{ij}}{+})$ such as $\Omega\subseteq\Delta\times\Gamma$, $\Delta=\Delta_i\cup\Delta_j$, $\Gamma=\Gamma_i\cup\Gamma_j$.

Let us now suppose that we decide to partition the sequence $\omega_i$ into two sequences $\omega_{i_1}$ and $\omega_{i_2}$, such as $\omega_i=\omega_{i_1}\cup\omega_{i_2}$, partitioning then the set of constants $\Delta_i$ into two disjoint sets $\Delta_{i_1}$ and $\Delta_{i_2}$ such as $\Delta_i=\Delta_{i_1}\cup\Delta_{i_2}$ and $\Delta_{i_1}\cap\Delta_{i_2}=\emptyset$. From this binary superposition $\omega_i=\omega_{i_1}\cup\omega_{i_2}$, we can also, according to property 18, build another algebraic structure $\mathcal{S}_i=(\omega_i,\overset{\Delta\tau_{ij}}{+})$ such as $\omega_i\subseteq\Delta_i\times\Gamma_i$, $\Delta_i=\Delta_{i_1}\cup\Delta_{i_2}$, $\Gamma_i=\Gamma_{i_1}\cup\Gamma_{i_2}$.

The same reasoning can be done with the sequence $\omega_j$ that would build the algebraic structure $\mathcal{S}_j=(\omega_j,\overset{\Delta\tau_{ij}}{+})$, such as $\omega_j\subseteq\Delta_j\times\Gamma_j$, $\Delta_j=\Delta_{j_1}\cup\Delta_{j_2}$, $\Gamma_j=\Gamma_{j_1}\cup\Gamma_{j_2}$.

The choice of an adequate algebraic structure is important because this choice has consequences on the observable space, that is to say, on the modeling and on the behaviour of the observed process. An algebraic structure has also its own level of abstraction and thus its own syntax and its own semantic that describe the observed process.

In order to specify which algebraic structure we consider, we denote $TOT(\Delta_i)$, the TOT Category whose object is the Unary Observer $\Theta_i(x_i,\Delta_i)$. This leads us to the definition of a $TOT(\Delta_i)$ *Category*:

**Definition 7.1** $TOT(\Delta_i)$ *Category*
*A* $TOT(\Delta_i)$ *Category is the TOT Category whose object is the Unary Observer* $\Theta_i(x_i,\Delta_i)$. *Such*

*a $TOT(\Delta_i)$ Category is algebraically structured with $\mathcal{S}_i = (\omega_i, \overset{\Delta\tau_{ij}}{+})$, $\omega_i \subseteq \Delta_i \times \Gamma_i$.*

Any set $\Delta_i$ of constants $\delta_{i_k}$ is a *discrete* and *finite* set. It is then possible to map such a set to a subset of $\mathbb{Z}$ according to an adequate mapping function. The section 7.4 introduces such a mapping thanks to the Gödel numbering function. Thus, the most general TOT Category is the category $TOT(\mathbb{Z})$ whatever the integer $z_i \in \mathbb{Z}$ represents (i.e. a real number, a string of characters, etc.). As a consequence, the building of the category $TOT(\Re)$ is not possible.

### 7.3.1 Modeling Functors

Let us consider the $TOT(\Delta_i)$ Category whose object is Unary Observer $\Theta_i(x_i, \Delta_i)$ defining the algebraic structure $\mathcal{S}_i = (\omega_i, \overset{\Delta\tau_{ij}}{+})$, $\omega_i \subseteq \Delta_i \times \Gamma_i$. Let us consider the $TOT(\Delta_j)$ Category whose object is Unary Observer $\Theta_j(x_j, \Delta_j)$ defining the algebraic structure $\mathcal{S}_j = (\omega_j, \overset{\Delta\tau_{ij}}{+})$, $\omega_j \subseteq \Delta_j \times \Gamma_j$. Let us consider the morphism $f_{ij} \in mor(TOT)$ linking Unary Observer $\Theta_i(x_i, \Delta_i)$ with Unary Observer $\Theta_j(x_j, \Delta_j)$.

Let us recall that such a morphism maps some timed observations $(\delta_{i_k}, t_{i_k})$ of sequence $\omega_i(t_{n_i})$ with some timed observations $(\delta_{j_l}, t_{j_l})$ of sequence $\omega_j(t_{n_j})$. A priori, such a mapping is not known. Let us suppose that there exists a well known relation, denoted $\mathcal{R}_{\Delta_i \to \Delta_j} : \Delta_i \to \Delta_j$, linking set of constants $\Delta_i$ with set of constants $\Delta_j$. This means that the mapping of constants $\delta_{i_k}$ of $\Delta_i$ with constants $\delta_{j_l}$ of $\Delta_j$ is well known. So, according to the relation $\mathcal{R}_{\Delta_i \to \Delta_j}$, we are able to know which timed observations $(\delta_{i_k}, t_{i_k})$ of sequence $\omega_i(t_{n_i})$ maps which timed observations $(\delta_{j_l}, t_{j_l})$ of sequence $\omega_j(t_{n_j})$:

$$\delta_{i_k} \mathcal{R}_{\Delta_i \to \Delta_j} \delta_{j_l} \Rightarrow (\delta_{i_k}, t_{i_k}) f_{ij} (\delta_{j_l}, t_{j_l}) \tag{7.11}$$

Thus, the morphism $f_{ij}$ can be built and known according to the relation $\mathcal{R}_{\Delta_i \to \Delta_j}$. This leads us to the defintion of a *Modeling functor*:

**Definition 7.2** *Modeling Functor*
*Let us consider two TOT Categories $TOT(\Delta_i)$ and $TOT(\Delta_i)$ whose objects are respectively Unary Observers $\Theta_i(x_i, \Delta_i)$ and $\Theta_j(x_j, \Delta_j)$. If there exists a relation $\mathcal{R}_{\Delta_i \to \Delta_j}$ linking sets of constants $\Delta_i$ and $\Delta_j$ then the morphism $f_{ij}$ linking $\Theta_i(x_i, \Delta_i)$ and $\Theta_j(x_j, \Delta_j)$ is called a Modeling functor. Such a Modeling functor links categories $TOT(\Delta_i)$ and $TOT(\Delta_i)$ and is denoted $TOT(\Delta_i) \to TOT(\Delta_i)$.*

The relation $\mathcal{R}_{\Delta_i \to \Delta_j}$ can be anything: injective, surjective or bijective. A Modeling functor is then a morphism allowing to go from one particular TOT Category to another particular TOT Category. Each particular TOT Category has its own level of abstraction, that is to say, its own syntax and its own semantic to describe the observed process. In other words, each particular TOT Category has its own point of view to describe and model the system under observation. A Modeling functor is then a tool allowing to change the point of view under which the system is observed.

### 7.3.2 Level of Abstraction of a $TOT(\Delta_m)$ Category

Let us consider the TOT Category composed of $n \in \mathbb{N}^\star$ independent Unary Observers $\Theta_i(x_i, \Delta_i)$, $i \in [1; n]$ such as $\bigcap_{i\in[1;n]} \Delta_i = \emptyset$. According to the superposition theorem 3.2, let us consider the observed process $(X(t), \Theta(X, \Delta))$ such as:

- $X(t) = \bigcup_{i\in[1;n]} \{x_i(t)\}$;
- $X = \bigcup_{i\in[1;n]} \{x_i\}$;
- $\Delta = \bigcup_{i\in[1;n]} \Delta_i$;
- $\Theta(X, \Delta) = \bigcup_{i\in[1;n]} \Theta_i(x_i, \Delta_i)$.

Let us consider the observed process $(X_m(t), \Theta(X_m, \Delta_m))$, $m \in \mathbb{N}^\star$, $m \leq n$, such as:

- $X_m(t) = \bigcup_{i\in[1;m]} \{x_i(t)\} \subseteq X(t)$;
- $X_m = \bigcup_{i\in[1;m]} \{x_i\} \subseteq X$;
- $\Delta_m = \bigcup_{i\in[1;m]} \Delta_i \subseteq \Delta$;
- $\Theta_m(X_m, \Delta_m) = \bigcup_{i\in[1;m]} \Theta_i(x_i, \Delta_i)$.

Let be $\Gamma_m$ the stochastic clock such as $\Gamma_m = \bigcup_{i\in[1;m]} \Gamma_i$. Let be $\Omega_m$ the m-ary superposition such as $\Omega_m = \bigcup_{i\in[1;m]} \omega_i$. Let us consider the $TOT(\Delta_m)$ Category defining the algebraic structure $\mathcal{S}_m = (\Omega_m, \overset{\Delta\tau_{ij}}{+})$, $\Omega_m \subseteq \Delta_m \times \Gamma_m$. Let be $\mathcal{T}_m = (\Omega_m, \mathcal{I}_m)$, the observable space of the observed process $(X_m(t), \Theta_m(X_m, \Delta_m))$ where $\mathcal{I}_m$ is the collection related to $\Omega_m$ (see property 5.1).

**Definition 7.3** *Level of Abstraction of a $TOT(\Delta_m)$ Category*
*The structure $\mathcal{L}_m = \langle \mathcal{S}_m, \mathcal{T}_m \rangle$ is the level of abstraction of the $TOT(\Delta_m)$ Category.*

The LoA $\mathcal{L}_m = \langle \mathcal{S}_m, \mathcal{T}_m \rangle$ of a $TOT(\Delta_m)$ Category is then composed of:

- an algebraic structure $\mathcal{S}_m = (\Omega_m, \overset{\Delta\tau_{ij}}{+})$, $\Omega_m \subseteq \Delta_m \times \Gamma_m$, from which we can get all the constants $\delta_{m_k} \in \Delta_m$ and all the variable names $x_i, i \in [1; m]$: these constants together with these variable names represent the syntax and the semantic of the LoA $\mathcal{L}_m$, that is to say, provides a particular point of view of the observed process $(X(t), \Theta(X, \Delta))$;
- an observable space $\mathcal{T}_m = (\Omega_m, \mathcal{I}_m)$ from which we can build the abstract chronicle model $\mathcal{M}_m$ whose instance is the behaviour $\mathcal{B}_m$ of the observed process $(X_m(t), \Theta_m(X_m, \Delta_m))$.

### 7.3.3 Abstraction Functors

Let us consider the $TOT(\Delta^0)$ Category composed of two Unary Observers $\Theta_i(x_i, \Delta_i)$ and $\Theta_j(x_j, \Delta_j)$ such as $\Delta^0 = \Delta_i \cup \Delta_j$ and $\Delta_i \cap \Delta_j = \emptyset$. Let be $\mathcal{L}^0 = \langle \mathcal{S}^0, \mathcal{T}^0 \rangle$, the level of abstraction of the $TOT(\Delta^0)$ Category where $\mathcal{S}^0$ and $\mathcal{T}^0$ are respectively the algebraic structure and observable space of the $TOT(\Delta^0)$ Category.

Let us consider the operation of composition of observers, denoted $\bigoplus$, defined in 4.5:

$$\Theta_k(x_k, \Delta^1) = \Theta_i(x_i, \Delta_i) \bigoplus \Theta_j(x_j, \Delta_j) \tag{7.12}$$

This operation is based on the application of the Modus Ponens with the following rule:

$$\begin{array}{c} \forall t_{i_m} \in \Gamma_i, \forall t_{j_n} \in \Gamma_j, \\ \theta_i(x_i, \delta_{i_m}, t_{i_m}) \wedge \theta_j(x_j, \delta_{j_n}, t_{j_n}) \wedge |t_{j_n} - t_{i_m}| \in \Delta\tau_{i_m j_n} \\ \Rightarrow \exists \delta_{k_l} \in \Delta^1, \theta_k(x_k, \delta_{k_l}, t_{k_l}) \wedge t_{k_l} \in \{t_{i_m}, t_{j_n}\} \end{array} \tag{7.13}$$

Let us consider the $TOT(\Delta^1)$ Category whose object is the Unary Observer $\Theta_k(x_k, \Delta^1)$. Let us denote $\Omega^1$ the sequence of timed observations $O_{k_l}(t_{k_l}) \equiv (\delta_{k_l}, t_{k_l})$ written by $\Theta_k(x_k, \Delta^1)$. According to definition 7.3, the LoA of $TOT(\Delta^1)$ Category is the structure $\mathcal{L}^1 = \langle \mathcal{S}^1, \mathcal{T}^1 \rangle$ where $\mathcal{S}^1 = (\Omega^1, \overset{\Delta\tau_{ij}}{+})$ and $\mathcal{T}^1 = (\Omega^1, \mathcal{I}^1)$ are respectively the algebraic structure and observable space of the $TOT(\Delta^1)$ Category.

By construction, each couple of constants $(\delta_{i_m}, \delta_{j_n}) \in \Delta_i \times \Delta_j \subset \Delta^0 \times \Delta^0$ is mapped to the constant $\delta_{k_l} \in \Delta^1$. The composition of observers induces then a surjective relation denoted $\Delta^0 \times \Delta^0 \rightarrow \Delta^1$ linking sets of constants $\Delta^0 \times \Delta^0$ to set of constants $\Delta^1$. Such a relation induces the existence of a morphism, denoted $TOT(\Delta^0) \rightarrow TOT(\Delta^1)$, linking categories $TOT(\Delta^0)$ and $TOT(\Delta^1)$. Such a morphism maps couples of timed observations $((\delta_{i_m}, t_{i_m}), (\delta_{j_n}, t_{j_n}))$ produced respectively by Unary Observers $\Theta_i(x_i, \Delta_i)$ and $\Theta_j(x_j, \Delta_j)$ of $TOT(\Delta^0)$ Category to timed observation $(\delta_{k_l}, t_{k_l})$ produced by Unary Observer $\Theta_k(x_k, \Delta^1)$ of $TOT(\Delta^1)$ Category:

$$\begin{array}{rcccc} TOT(\Delta^0) \rightarrow TOT(\Delta^1) & : & TOT(\Delta^0) & \rightarrow & TOT(\Delta^1) \\ & & ((\delta_{i_m}, t_{i_m}), (\delta_{j_n}, t_{j_n})) & \mapsto & (\delta_{k_l}, t_{k_l}) \end{array} \tag{7.14}$$

This leads us to the defintion of an *Abstraction functor*:

**Definition 7.4** *Abstraction Functor*
*Let us consider two categories, $TOT(\Delta^0)$ and $TOT(\Delta^1)$. Let us consider a surjective relation $\Delta^0 \times \Delta^0 \rightarrow \Delta^1$ built from the application of the Modus Ponens with the rule 7.13. Any morphism, denoted $TOT(\Delta^0) \rightarrow TOT(\Delta^1)$, linking categories $TOT(\Delta^0)$ and $TOT(\Delta^1)$ according to the relation $\Delta^0 \times \Delta^0 \rightarrow \Delta^1$ is an Abstraction functor. Such an Abstraction functor is said to implement the composition of observers.*

From this definition, we can deduce the following properties:

**Property 23**
*According to property 22, the LoA $\mathcal{L}^1$ is higher or more abstract than the LoA $\mathcal{L}^0$.*

**Property 24**
*If the rule 7.13 is applied in the particular case where $t_{j_n} = t_{j_{m+1}}$ then the Unary Observer*

*$\Theta_k(x_k, \Delta^1)$ of the $TOT(\Delta^1)$ Category can be decomposed into the four Abstract Binary Observers $\Theta_{ii}(\{\phi_{ii}\}, \{\delta^{ii}\})$, $\Theta_{ij}(\{\phi_{ij}\}, \{\delta^{ij}\})$, $\Theta_{ji}(\{\phi_{ji}\}, \{\delta^{ji}\})$, $\Theta_{jj}(\{\phi_{jj}\}, \{\delta^{jj}\})$, defined in 4.7. Timed observations of the $TOT(\Delta^1)$ Category represent the observation of binary sequences $((\delta_{i_m}, t_{i_m}), (\delta_{j_{m+1}}, t_{j_{m+1}}))$ from consecutive timed observations produced by Unary Observers $\Theta_i(x_i, \Delta_i)$ and $\Theta_j(x_j, \Delta_j)$ of the $TOT(\Delta^0)$ Category (see chapter 8, section 8.6.4, for a concrete utilisation).*

**Property 25**
*If the rule 7.13 is applied in the particular case where only the Unary Observer $\Theta_i(x_i, \Delta_i)$ is concerned (i.e. there is no Unary Observer $\Theta_j(x_j, \Delta_j)$ and no timed constraints):*

$$\begin{array}{c} \forall t_{i_m} \in \Gamma_i, \\ \theta_i(x_i, \delta_{i_m}, t_{i_m}) \Rightarrow \exists \delta_{k_l} \in \Delta^1, \theta_k(x_k, \delta_{k_l}, t_{i_m}) \end{array} \tag{7.15}$$

*Then the Abstraction functor is built according to a surjective relation of the form:*

$$\begin{array}{rccc} \Delta^0 \rightarrow \Delta^1 & : & \Delta^0 & \rightarrow & \Delta^1 \\ & & \delta_{i_m} & \mapsto & \delta_{k_l} \end{array} \tag{7.16}$$

*Such a relation can be a usual application $\mu$ such as:*

$$\begin{array}{rccc} \mu & : & \Delta^0 & \rightarrow & \Delta^1 \\ & & \delta_{i_m} & \mapsto & \mu(\delta_{i_m}) \end{array} \tag{7.17}$$

A concrete example of such an application, called the *classification function*, is given in chapter 8.

**Property 26**
*Given two sets of constants $\Delta^0$ and $\Delta^1$:*

- *if the relation $\Delta^0 \rightarrow \Delta^1$ is not surjective then the morphism $TOT(\Delta^0) \rightarrow TOT(\Delta^1)$ is a Modeling functor,*
- *if the relation $\Delta^0 \rightarrow \Delta^1$ is surjective then:*
  - *if the relation $\Delta^0 \rightarrow \Delta^1$ is induced by the application of the Modus Ponens with the rule 7.13 then the morphism $TOT(\Delta^0) \rightarrow TOT(\Delta^1)$ is an Abstraction functor,*
  - *else the morphism $TOT(\Delta^0) \rightarrow TOT(\Delta^1)$ is a Modeling functor.*

Modeling functors can be injective, surjective or bijective morphisms. Their goal is to change the point of view under which the system is observed. They link TOT categories whose LoAs are as abstract as each other, that is to say, the syntax and the semantic in each LoA is equivalent.

Abstraction functors are necessarily surjective morphisms that generate, on one hand, a loss of informations between the input category and the output category but, on the other hand, allows to make easier reasonings in the output category in a way to get detailed informations on

the input one. There exists famous functors such as the "forgetting" functor or the Poincaré's functor (see [Mer83] for details).

This leads us to the following definition of a *gradient of abstraction of the TOT Category*:

**Definition 7.5** *Gradient of Abstraction of the TOT Category*
*Let us consider the* $n \in \mathbb{N}^\star$ *categories* $TOT(\Delta_i)$, $i \in [1; n]$.
*Let be* $\mathcal{L}_i = \langle \mathcal{S}_i, \mathcal{T}_i \rangle$, *the LoA of the* $TOT(\Delta_i)$ *Category.*
*Let be* $\mathcal{B}_i$, *the behaviour of the observed process of the* $TOT(\Delta_i)$ *Category.*
*A gradient of abstraction (GoA) of the TOT Category is a collection* $\mathcal{G} = \{\mathcal{L}_i, i \in [0; n]\}$ *of* $n \in \mathbb{N}^\star$ *LoAs* $\mathcal{L}_i$ *such as:*

- *there exists an Abstraction functor* $TOT(\Delta_i) \rightarrow TOT(\Delta_{i+1})$ *linking two consecutive categories* $TOT(\Delta_i)$ *and* $TOT(\Delta_{i+1})$ *for all* $i \in [0; n-1]$*;*
- *the behaviour at the LoA* $\mathcal{L}_{i+1}$ *is stronger than the behaviour at the LoA* $\mathcal{L}_i$*:*

$$\forall i \in [0; n-1], \mathcal{B}_{i+1} \Rightarrow \mathcal{B}_i \tag{7.18}$$

The consequence of the first point is that the LoA $\mathcal{L}_{i+1}$ is higher or more abstract than the LoA $\mathcal{L}_i$ for all $i \in [0; n-1]$.

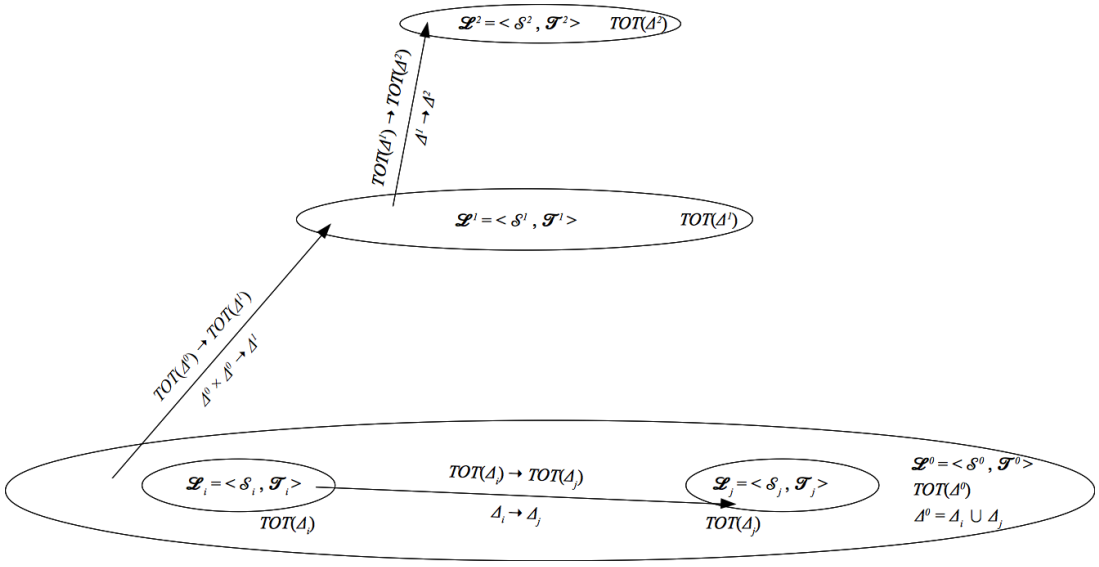


Figure 7.5: Gradient of abstraction in the TOT Category

Figure 7.5 represents a gradient of abstraction containing three levels of abstraction:

- The level $\mathcal{L}^0 = \langle \mathcal{S}^0, \mathcal{T}^0 \rangle$ is composed of two TOT Categories $TOT(\Delta_i)$ and $TOT(\Delta_j)$ defining two LoAs $\mathcal{L}_i = \langle \mathcal{S}_i, \mathcal{T}_i \rangle$ and $\mathcal{L}_j = \langle \mathcal{S}_j, \mathcal{T}_j \rangle$: categories $TOT(\Delta_i)$ and $TOT(\Delta_j)$ are linked by a Modeling functor $TOT(\Delta_i) \rightarrow TOT(\Delta_j)$ built according to a relation $\Delta_i \rightarrow \Delta_j$.
- The level $\mathcal{L}^1 = \langle \mathcal{S}^1, \mathcal{T}^1 \rangle$ is composed of the $TOT(\Delta^1)$ Category: an Abstraction functor $TOT(\Delta^0) \rightarrow TOT(\Delta^1)$ links categories $TOT(\Delta^0)$ and $TOT(\Delta^1)$ according to a surjective relation $\Delta^0 \times \Delta^0 \rightarrow \Delta^1$ as described in definition 7.4 or property 24.

- The level $\mathcal{L}^2 = \langle \mathcal{S}^2, \mathcal{T}^2 \rangle$ is composed of the $TOT(\Delta^2)$ Category: an Abstraction functor $TOT(\Delta^1) \rightarrow TOT(\Delta^2)$ links categories $TOT(\Delta^1)$ and $TOT(\Delta^2)$ according to a surjective relation $\Delta^1 \rightarrow \Delta^2$ as described in property 25.

## 7.4 Syntactic Arithmetization

As said in 7.3, any set $\Delta_i$ of constants $\delta_{i_k}$ is finite and discrete. It is then possible to link such a set with a subset $\mathbb{P}_i$ of $\mathbb{Z}$ according to an adequate numbering function $g : \Delta_i \rightarrow \mathbb{P}_i \subseteq \mathbb{Z}$. In his thesis [Göd31], Kurt Gödel provides a general and adequate numbering function, called the Gödel numbering function, mapping any primitive symbol, expression and finite sequence of expressions $e$ belonging to a language $\mathfrak{L}_K$ of a theory $K$ with a positive integer of $\mathbb{N}^\star$:

$$\begin{array}{rccc} g & : & \mathfrak{L}_K & \rightarrow & \mathbb{N}^\star \\ & & e & \mapsto & g(e) \end{array} \tag{7.19}$$

The language $\mathfrak{L}_{TOT}$ of the Timed Observation Theory is composed of the following symbols:

- connectors: $\wedge$, $\Rightarrow$;
- quantifiers: $\forall$, $\exists$, $\in$;
- ponctuation: (, ), ",", {, }, [, ];
- variable names: $x_i$;
- constants: $\delta_{i_j}$;
- timestamps: $t_{i_j}$;
- observation classes: $O_{i_j}$;
- time constraints: $\tau_{ij}^-$, $\tau_{ij}^+$;
- predicates: $\theta_i$;
- abstract predicates: $\theta_{ij}$;
- abstract constants: $\delta^{ij}$.

In a first step, the function $g$ maps any symbol $e$ of $\mathfrak{L}_{TOT}$ with its Gödel number $g(e)$ such as, for all $i \geq 1$ and $j \geq 1$ :

$$\begin{array}{ll} g(\wedge) = 3 & g(x_i) = 3 + 8 * i \\ g(\Rightarrow) = 5 & g(\theta_i) = 5 + 8 * i \\ g(\forall) = 7 & g(\delta_{i_j}) = 3 + 8 * 2^i * 3^j \\ g(\exists) = 9 & g(t_{i_j}) = 5 + 8 * 2^i * 3^j \\ g(\in) = 11 & g(O_{i_j}) = 7 + 8 * 2^i * 3^j \\ g(() = 13 & g(\tau_{ij}^-) = 9 + 8 * 2^i * 3^j \\ g()) = 15 & g(\tau_{ij}^+) = 11 + 8 * 2^i * 3^j \\ g(,) = 17 & g(\theta_{ij}) = 13 + 8 * 2^i * 3^j \\ g(\{) = 19 & g(\delta^{ij}) = 15 + 8 * 2^i * 3^j \\ g(\}) = 21 & \\ g([) = 23 & \\ g(]) = 25 & \end{array} \tag{7.20}$$

In a second step, any expression $u_0 u_1 ... u_r$ made of symbols $u_0, u_1, ..., u_r$ is mapped with its Gödel number according the following way:

$$g(u_0 u_1 ... u_r) = 2^{g(u_0)} 3^{g(u_1)} ... p_r^{g(u_r)} \tag{7.21}$$

where $p_r$ is the $r^{th}$ prime number.

In last step, any sequence of expressions $e_0, e_1, ..., e_r$ is mapped with its Gödel number according the following way:

$$g(e_0, e_1, ..., e_r) = 2^{g(e_0)} 3^{g(e_1)} ... p_r^{g(e_r)} \tag{7.22}$$

where $p_r$ is the $r^{th}$ prime number.

The key point is, given an element $e$ of the language $\mathfrak{L}_{TOT}$, its Gödel number is unique:

$$\forall e \in \mathfrak{L}_{TOT}, \exists! g(e) \in \mathbb{N}^\star \tag{7.23}$$

Thus, the Gödel numbering function is a bijective homomorphism between any algebraic structure $\mathcal{S}_i = (\omega_i, \overset{\Delta\tau_{ij}}{+})$, $\omega_i \subseteq \Delta_i \times \Gamma_i$ of the $TOT(\Delta_i)$ Category and algebraic structure $\mathcal{S}_i^{'} = (\mathbb{P}_i, +)$, mapping any timed observation $O_{i_j}(t_{i_j})$ of $\omega_i$ with a unique positive integer $p_{ij}$ of $\mathbb{P}_i$:

$$\begin{array}{rccc} g & : & \omega_i & \rightarrow & \mathbb{P}_i \\ & & O_{i_j}(t_{i_j}) & \mapsto & p_{ij} \end{array} \tag{7.24}$$

such that:

$$\begin{array}{c} \forall O_{i_j}(t_{i_j}) \in \omega_i, \exists! p_{ij} \in \mathbb{P}_i, g(O_{i_j}(t_{i_j})) = p_{ij} \\ \forall O_{i_j}(t_{i_j}) \in \omega_i, \forall O_{i_k}(t_{i_k}) \in \omega_i, g(O_{i_j}(t_{i_j}) \overset{\Delta\tau_{ik}}{+} O_{i_k}(t_{i_k})) = p_{ij} + p_{ik} \end{array} \tag{7.25}$$

In this case, the $TOT(\Delta_i)$ Category is isomorphic with the $TOT(\mathbb{P}_i)$ Category and then the following properties can be provided:

**Property 27** Modeling Functor

*Let us consider $\mathbb{P}$ and $\mathbb{Q}$, two subsets of $\mathbb{N}^\star$.*

*If there exists a relation $\mathbb{P} \rightarrow \mathbb{Q}$ between $\mathbb{P}$ and $\mathbb{Q}$ then there exists a Modeling functor $TOT(\mathbb{P}) \rightarrow$*

*$TOT(\mathbb{Q})$ between categories $TOT(\mathbb{P})$ and $TOT(\mathbb{Q})$.*

**Property 28** Abstraction Functor
*If the relation $\mathbb{P} \to \mathbb{Q}$ is surjective and induced by the application of the Modus Ponens with the rule 7.13 then the morphism $TOT(\mathbb{P}) \to TOT(\mathbb{Q})$ is an Abstraction functor between categories $TOT(\mathbb{P})$ and $TOT(\mathbb{Q})$.*

Working with a $TOT(\mathbb{P}_i)$ Category, $\mathbb{P}_i \subseteq \mathbb{Z}$, allows the renaming of elements composing the $TOT(\Delta_i)$ Category into natural numbers. Such a renaming is, in practice, very useful when implemented in computer programs.

## 7.5 Sum and Product in the TOT Category

Let us consider the TOT Category and two objects, that is to say, two Unary Observers $\Theta_i(x_i, \Delta_i) \equiv \Theta_i$ and $\Theta_j(x_j, \Delta_j) \equiv \Theta_j$. This aim of this section is to give a definition and an interpretation of what a sum and a product of objects in the TOT Category represent. The general definitions of sum and product can be found in 2.4.

### 7.5.1 Sum in the TOT Category

The sum between the object $\Theta_i$ and the object $\Theta_j$, denoted $\Theta_\Sigma \equiv \Theta_i + \Theta_j$, is an object of the TOT Category associated with two morphisms, $f_{i\Sigma} : \Theta_i \to \Theta_\Sigma$ and $f_{j\Sigma} : \Theta_j \to \Theta_\Sigma$, such that, for all object $\Theta_k \in ob(TOT)$ and for all couple of morphisms $f_{ik} : \Theta_i \to \Theta_k$ and $f_{jk} : \Theta_j \to \Theta_k$ in $mor(TOT)$, there exists a unique morphism $f_{\Sigma k} : \Theta_\Sigma \to \Theta_k$ such as :

$$\begin{cases} f_{ik} = f_{\Sigma k} \circ f_{i\Sigma} \\ f_{jk} = f_{\Sigma k} \circ f_{j\Sigma} \end{cases} \tag{7.26}$$

The morphism $f_{ik}$ maps timed observations $(\delta_i, t_i)$ written by $\Theta_i$ with timed observations $(\delta_k, t_k)$ written by $\Theta_k$:

$$\begin{array}{rccc} f_{ik} & : & \Theta_i & \to & \Theta_k \\ & & (\delta_i, t_i) & \mapsto & (\delta_k, t_k) \end{array} \tag{7.27}$$

The morphism $f_{jk}$ maps timed observations $(\delta_j, t_j)$ written by $\Theta_j$ with timed observations $(\delta_k, t_k)$ written by $\Theta_k$:

$$\begin{array}{rccc} f_{jk} & : & \Theta_j & \to & \Theta_k \\ & & (\delta_j, t_j) & \mapsto & (\delta_k, t_k) \end{array} \tag{7.28}$$

Let us consider that the Unary Observer $\Theta_\Sigma$ is the result of the operation of composition of observers of equation 7.7:

$$\Theta_\Sigma = \Theta_i \bigoplus \Theta_j \tag{7.29}$$

In that case, timed observations written by $\Theta_\Sigma$ are of the form $(\delta_\Sigma, t_\Sigma)$ where:

$$\begin{array}{l} \delta_\Sigma = (\delta_i, \delta_j) \\ t_\Sigma \in \{t_i, t_j\} \end{array} \tag{7.30}$$

Thus, morphism $f_{i\Sigma}$ maps timed observations $(\delta_i, t_i)$ written by $\Theta_i$ with timed observations $(\delta_\Sigma, t_\Sigma) \equiv ((\delta_i, \delta_j), t_\Sigma \in \{t_i, t_j\})$ written by $\Theta_\Sigma$ such as:

$$\begin{array}{rccc} f_{i\Sigma} & : & \Theta_i & \rightarrow & \Theta_\Sigma \\ & & (\delta_i, t_i) & \mapsto & (\delta_\Sigma, t_\Sigma) \equiv ((\delta_i, \delta_j), t_\Sigma \in \{t_i, t_j\}) \end{array} \tag{7.31}$$

And morphism $f_{j\Sigma}$ maps timed observations $(\delta_j, t_j)$ written by $\Theta_j$ with timed observations $(\delta_\Sigma, t_\Sigma) \equiv ((\delta_i, \delta_j), t_\Sigma \in \{t_i, t_j\})$ written by $\Theta_\Sigma$ such as:

$$\begin{array}{rccc} f_{j\Sigma} & : & \Theta_j & \rightarrow & \Theta_\Sigma \\ & & (\delta_j, t_j) & \mapsto & (\delta_\Sigma, t_\Sigma) \equiv ((\delta_i, \delta_j), t_\Sigma \in \{t_i, t_j\}) \end{array} \tag{7.32}$$

Let us have a look on the morphism $f_{\Sigma k}$. Such a morphism maps timed observations $(\delta_\Sigma, t_\Sigma) \equiv ((\delta_i, \delta_j), t_\Sigma \in \{t_i, t_j\})$ written by $\Theta_\Sigma$ with timed observations $(\delta_k, t_k)$ written by $\Theta_k$ such as:

$$\begin{array}{rccc} f_{\Sigma k} & : & \Theta_\Sigma & \rightarrow & \Theta_k \\ & & (\delta_\Sigma, t_\Sigma) \equiv ((\delta_i, \delta_j), t_\Sigma \in \{t_i, t_j\}) & \mapsto & (\delta_k, t_k) \end{array} \tag{7.33}$$

Let us now suppose that there exists another morphism denoted $f'_{\Sigma k}$ linking observers $\Theta_\Sigma$ and $\Theta_k$. If such a morphism exists, this means that the observer $\Theta_\Sigma$ would be able to write a timed observation of the form $(\delta'_\Sigma, t'_\Sigma)$ whose constant $\delta'_\Sigma$ and timestamp $t'_\Sigma$ would be different than $\delta_\Sigma$ and timestamp $t_\Sigma$:

- timestamps $t'_\Sigma$ and $t_\Sigma$ are arbitrarily chosen in $\{t_i, t_j\}$, we can then impose $t'_\Sigma = t_\Sigma$;
- by hypothesis (see 3.4), the observer $\Theta_\Sigma$ is a safe program, that is to say, it cannot write simultaneous timed observations. Thus, at the date $t'_\Sigma = t_\Sigma$, we have:

$$(\delta'_\Sigma, t_\Sigma) = (\delta_\Sigma, t_\Sigma) \Rightarrow \delta'_\Sigma = \delta_\Sigma \tag{7.34}$$

This means that the morphism $f_{\Sigma k}$ is unique.

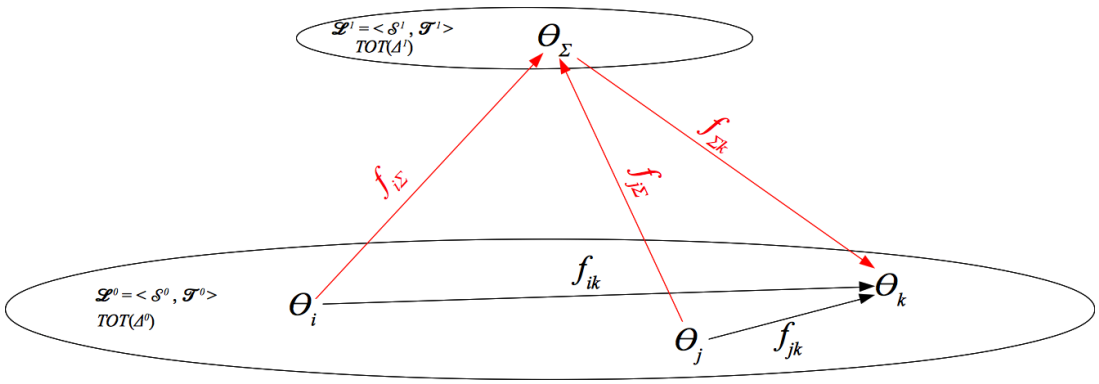


Figure 7.6: Sum of objects in the TOT Category

This leads us to the following properties:

**Property 29**

*The sum of two objects* $\Theta_i(x_i, \Delta_i)$ *and* $\Theta_j(x_j, \Delta_j)$ *in the TOT Category is an observer* $\Theta_\Sigma(x_\Sigma, \Delta_\Sigma)$ *writing timed observation of the form* $(\delta_\Sigma, t_\Sigma)$ *such as:*

$$\begin{array}{c} \delta_\Sigma = (\delta_i, \delta_j), \delta_i \in \Delta_i, \delta_j \in \Delta_j \\ t_\Sigma \in \{t_i, t_j\}, t_i \in \Gamma_i, t_j \in \Gamma_j \end{array} \tag{7.35}$$

*Morphism* $f_{i\Sigma}$ *allows to map the constant* $\delta_i$ *to the constant* $\delta_\Sigma = (\delta_i, \delta_j)$.
*Morphism* $f_{j\Sigma}$ *allows to map the constant* $\delta_j$ *to the constant* $\delta_\Sigma = (\delta_i, \delta_j)$.

**Property 30**
*The sum of two objects in the TOT Category corresponds to the operation of composition of observers.*

**Property 31**
*The sum of two objects in the TOT Category induces an abstraction process.*

**Property 32**
*An Abstraction functor implements the sum of objects in the TOT Category.*

### 7.5.2 Product in the TOT Category

The product between the object $\Theta_i$ and the object $\Theta_j$, denoted $\Theta_\Pi \equiv \Theta_i \times \Theta_j$ is an object associated with two morphisms, $f_{\Pi i} : \Theta_\Pi \to \Theta_i$ and $f_{\Pi j} : \Theta_\Pi \to \Theta_j$, such that, for all object $\Theta_k \in ob(TOT)$ and for all couple of morphisms $f_{ki} : \Theta_k \to \Theta_i$ and $f_{kj} : \Theta_k \to \Theta_j \in mor(TOT)$, there exists a unique morphism $f_{k\Pi} : \Theta_k \to \Theta_\Pi$ such as:

$$\begin{cases} f_{\Pi i} \circ f_{k\Pi} = f_{ki} \\ f_{\Pi j} \circ f_{k\Pi} = f_{kj} \end{cases} \tag{7.36}$$

The morphism $f_{ki}$ maps timed observations $(\delta_k, t_k)$ written by $\Theta_k$ with timed observations $(\delta_i, t_i)$ written by $\Theta_i$:

$$\begin{array}{rccc} f_{ki} & : & \Theta_k & \to & \Theta_i \\ & & (\delta_k, t_k) & \mapsto & (\delta_i, t_i) \end{array} \tag{7.37}$$

The morphism $f_{kj}$ maps timed observations $(\delta_k, t_k)$ written by $\Theta_k$ with timed observations $(\delta_j, t_j)$ written by $\Theta_j$:

$$\begin{array}{rccc} f_{kj} & : & \Theta_k & \to & \Theta_j \\ & & (\delta_k, t_k) & \mapsto & (\delta_j, t_j) \end{array} \tag{7.38}$$

Let us consider that the Unary Observer $\Theta_\Pi$ writes timed observations of the form $(\delta_\Pi, t_\Pi)$ where:

$$\begin{array}{c} \delta_\Pi = (\delta_i, \delta_j) \\ t_\Pi \in \{t_i, t_j\} \end{array} \tag{7.39}$$

where $\delta_i \in \Delta_i$, $\delta_j \in \Delta_j$, $t_i \in \Gamma_i$ and $t_j \in \Gamma_j$.

Thus, morphism $f_{\Pi i}$ maps timed observations $(\delta_\Pi, t_\Pi) \equiv ((\delta_i, \delta_j), t_\Sigma \in \{t_i, t_j\})$ written by $\Theta_\Pi$ with timed observations $(\delta_i, t_i)$ written by $\Theta_i$ such as:

$$\begin{array}{rccc} f_{\Pi i} & : & \Theta_\Pi & \rightarrow & \Theta_i \\ & & (\delta_\Pi, t_\Pi) \equiv ((\delta_i, \delta_j), t_\Sigma \in \{t_i, t_j\}) & \mapsto & (\delta_i, t_i) \end{array} \tag{7.40}$$

The set $\Delta_\Pi$ containing constants $\delta_\Pi$ being a subset of $\Delta_i \times \Delta_j$, morphism $f_{\Pi i}$ is the projection of $\Delta_i \times \Delta_j$ in $\Delta_i$.

Morphism $f_{\Pi j}$ maps timed observations $(\delta_\Pi, t_\Pi) \equiv ((\delta_i, \delta_j), t_\Sigma \in \{t_i, t_j\})$ written by $\Theta_\Pi$ with timed observations $(\delta_j, t_j)$ written by $\Theta_j$ such as:

$$\begin{array}{rccc} f_{\Pi j} & : & \Theta_\Pi & \rightarrow & \Theta_j \\ & & (\delta_\Pi, t_\Pi) \equiv ((\delta_i, \delta_j), t_\Sigma \in \{t_i, t_j\}) & \mapsto & (\delta_j, t_j) \end{array} \tag{7.41}$$

The set $\Delta_\Pi$ containing constants $\delta_\Pi$ being a subset of $\Delta_i \times \Delta_j$, morphism $f_{\Pi j}$ is the projection of $\Delta_i \times \Delta_j$ in $\Delta_j$.

Let us have a look on the morphism $f_{k\Pi}$. Such a morphism maps timed observations $(\delta_k, t_k)$ written by $\Theta_k$ with timed observations $(\delta_\Pi, t_\Pi) \equiv ((\delta_i, \delta_j), t_\Pi \in \{t_i, t_j\})$ written by $\Theta_\Pi$ such as:

$$\begin{array}{rccc} f_{k\Pi} & : & \Theta_k & \rightarrow & \Theta_\Pi \\ & & (\delta_k, t_k) & \mapsto & (\delta_\Pi, t_\Pi) \equiv ((\delta_i, \delta_j), t_\Pi \in \{t_i, t_j\}) \end{array} \tag{7.42}$$

With the same reasoning than the one made for the sum, we demonstrate that the morphism $f_{k\Pi}$ is unique.

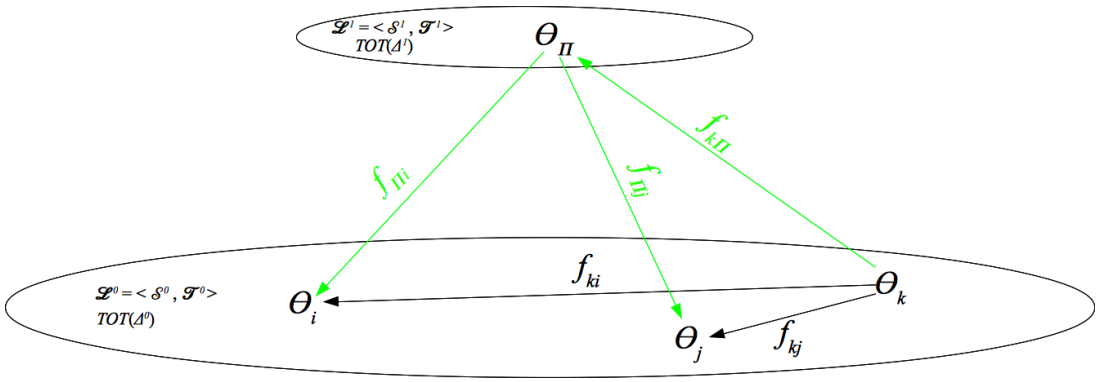


Figure 7.7: Product of objects in the TOT Category

This leads us to the following properties:

**Property 33**

*The product of two objects* $\Theta_i(x_i, \Delta_i)$ *and* $\Theta_j(x_j, \Delta_j)$ *in the TOT Category is an observer* $\Theta_\Pi(x_\Pi, \Delta_\Pi)$*, where* $\Delta_\Pi \subseteq \Delta_i \times \Delta_j$*, writing timed observation of the form* $(\delta_\Pi, t_\Pi)$ *such as:*

$$\begin{array}{c} \delta_\Pi = (\delta_i, \delta_j), \delta_i \in \Delta_i, \delta_j \in \Delta_j \\ t_\Pi \in \{t_i, t_j\}, t_i \in \Gamma_i, t_j \in \Gamma_j \end{array} \tag{7.43}$$

*Morphism* $f_{\Pi i}$ *is the projection of* $\Delta_i \times \Delta_j$ *in* $\Delta_i$*.*

*Morphism $f_{\Pi j}$ is the projection of $\Delta_i \times \Delta_j$ in $\Delta_j$.*

**Property 34**
*The product of two objects $\Theta_i(x_i, \Delta_i) \times \Theta_j(x_j, \Delta_j)$ in the TOT Category allows to get Unary Observers $\Theta_i(x_i, \Delta_i)$ and $\Theta_j(x_j, \Delta_j)$.*

**Property 35**
*The product of two objects in the TOT Category induces a reification process.*

**Property 36**
*A morphism implementing the product of objects in the TOT Category is called a reification functor.*

## 7.6 Conclusion

This chapter provides the basis of *abstraction* and *reification* reasonings according to Category Theory. Characteristic elements composing the TOT Category were given in order to define the concept of *Modeling* and *Abstraction* functors. Such functors are tools aiming at linking particular $TOT(\Delta_i)$ Categories which are isomorphic to $TOT(\mathbb{P}_i)$ Categories, $\mathbb{P}_i \in \mathbb{Z}$, whose elements are represented with natural numbers. These functors are of the main importance to choose the adequate level of abstraction in order to model the observed process.

Modeling the observed process at the right level of abstraction provides a powerful solution to solve a given problem. The next chapter proposes an organisation of these reasonings under the form of a the TOM4A methodology (Timed Observations Methodology for Abstraction) implementing a Recursive Abstraction Reification Based Problem Solving Method (AR-PSM). An application of such an AR-PSM is implemented on a concrete problem concerning the discovering and the modeling of internal fraudulent transactions in a famous French bank.

CHAPTER 8

# The TOM4A Methodology

## 8.1 Introduction

This section is dedicated to the definition of the principles of a problem solving method (PSM) based on recursive abstraction reification reasoning process, called AR-PSM (Recursive Abstraction Reification Based Problem Solving Method). Such a process is an application of abstraction functors of the TOT($\mathbb{Z}$) Category presented in the preceding section. This section illustrates the AR-PSM principles with an application which aims at discovering potential frauds in a stream of banking transactions. This example comes from a famous French bank for which the principles of the proposed AR-PSM have been implemented in an algorithm called TOM4FFS (Timed Observations Mining for Fraud Fighting System).

Nevertheless, the annexe B describes another application of the AR-PSM principles. This application is the paper [LGV17] published in the proceeding of the ICAART international congres in 2017. This application is our running example of the conversation between Alice, Bob and Carol. It is to note that this paper has been cited by Floridi himself in [Flo17]. If less concrete than the fraud detection problem, this application aims at showing that the principles of the proposed AR-PSM are useful in current life activities.

Next section recalls some important aspects of any PSM in order to introduce TOM4A basic cognitive operations in the TOT framework in section 8.3.

The following sections are all dedicated to the application of the TOM4 Methodology to the concrete problem of the internal fraud detection in the banking industry.

## 8.2 Problem Solving Method

Let us introduce the basis about problem solving (cf. figure 8.1). A problem $P$ exists in a concrete space generally called the *real world*. The aim of a PSM is to allow the implementation $S$ of a solution such that $P$ disappears: $S \equiv P^{-1}$. In practice, any implementation using specific technologies, a concrete solution is such that $S \approx P^{-1}$ (cf. *technologies are approximation*, in [New81]).

To this aim, a PSM is always based on two basic operations: the *observation* of the problem $O(P)$ and the *implementation* of an abstract solution. The *observation* aims at building a *model* of the problem $M(O(P))$, which is used by a PSM to build a model $M(S)$ of all the solutions $S$ that can be implemented given the available technologies. In other words, $M(S)$ contains all the constraints that any concrete solution $S$ must satisfy in order to be a concrete solution: $S \approx P^{-1}$.

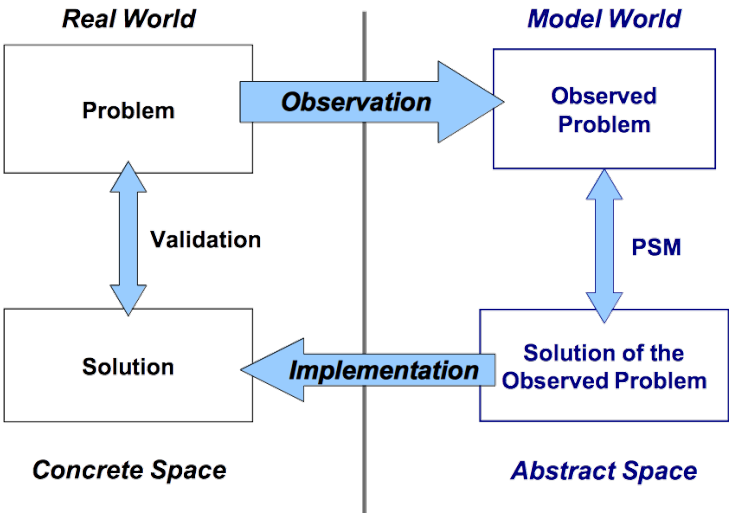


Figure 8.1: Problem Solving Method

The important point here is that *model* of the observed problem $M(O(P))$ and the *model* $M(S)$ of the solutions $S$ belong to an abstract world, the *model's world*. This world is dual: it is made of a *semantic space* and a *syntactic space* (figure 8.2). The role of the concept of formal system is precisely to establish and to clarify the relations between these two spaces: this is the fundamental role of the knowledge *representation* and *interpretation* dual operations (figure 8.3).

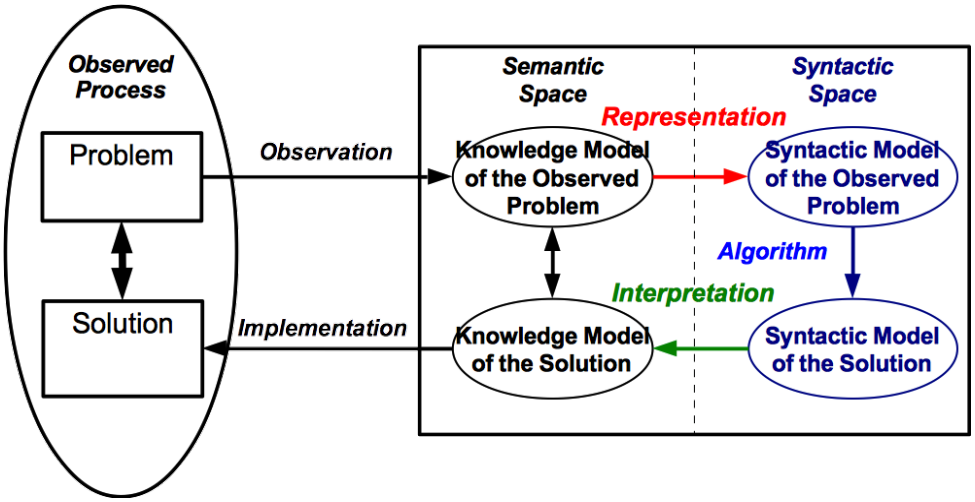


Figure 8.2: Semantic and Syntactic Spaces

As a consequence, solving a problem consists in an adequate suite of cognitive processes belonging to one of the four basic cognitive operations (*Observation*, *Representation*, *Interpretation* and *Implementation*) as illustrated by figure 8.3. Such a suite can be any length, chaining in an arbitrary number and in an arbitrary order of the four basic cognitive operations.

In this figure, the concept of problem has been generalized with the one of *Observed Process*, the *Knowledge Model* contains all the knowledge in the mind of a *Human Observer* (i.e. the knowledge about the problem and the solutions) and the *Representations' Model* contains all the syntactic formulas built over a given formal system.

Figure 8.3 allows to formulate the hypothesis of the TOM4A Methodology:

**Definition 8.1** *TOM4A Hypothesis*

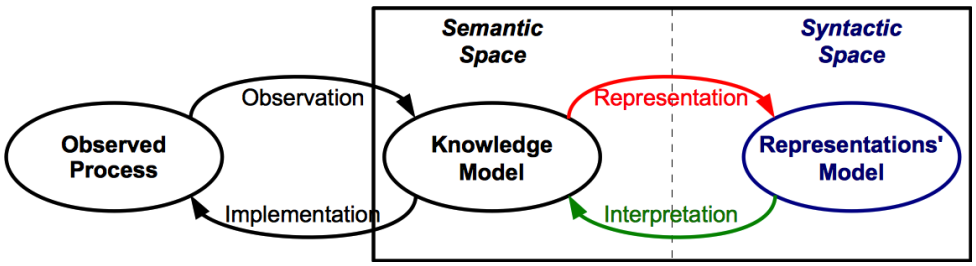


Figure 8.3: Knowledge Model and the Model of the Knowledge Representation's

*Representation and Interpretation cognitive processes correspond respectively to Abstraction and Reification reasonings.*

Such an hypothesis seems to be quite obvious when considering that the knowledge model of the observed process (i.e. the one leaving in the Semantic Space of figure 8.3) resides at the *same* level of abstraction than the observed process itself: both exist in real worlds even if the real world of the problem may differ from the real world of the model (i.e. the semantic space is in the mind of the observers). But, according to our analysis, there is no evidence to place these two worlds in different levels of abstractions and it is simpler to consider that these two worlds are two parts of a unique world, *the* real world.

On the contrary, it seems quite obvious that the *Representations' Model* leaves at another level of abstraction than the *Knowledge Model* or *the Observed Process*. So the hypothesis of the TOM4A Methodology only asserts that the LoA of the *Representations' Model* is more abstract than the one of the *Knowledge Model* and the *Observed Process*. With this hypothesis in mind, a simple analogy allows to deduce that, according to the Category Theory, the *Representation* cognitive process is a kind of *Abstraction* operation and that the *Interpretation* cognitive process is a kind of *Reification* operation. Figure 8.4 aims at illustrating this idea. In this figure, both the *Observed Process* and the *Knowledge Model* reside at the same LoA, those of the *concrete world* (i.e. *the* real world) and the *Representations' Model* resides at a higher abstraction level, denoted consequently the *LoA 1*.

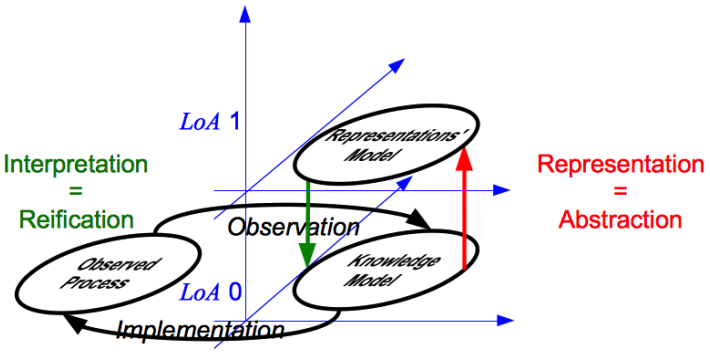


Figure 8.4: TOM4A Hypothesis

The major interest of such an hypothesis is that it is possible to build a nested GoA composed of as many LoAs as necessary (cf. figure 8.5):

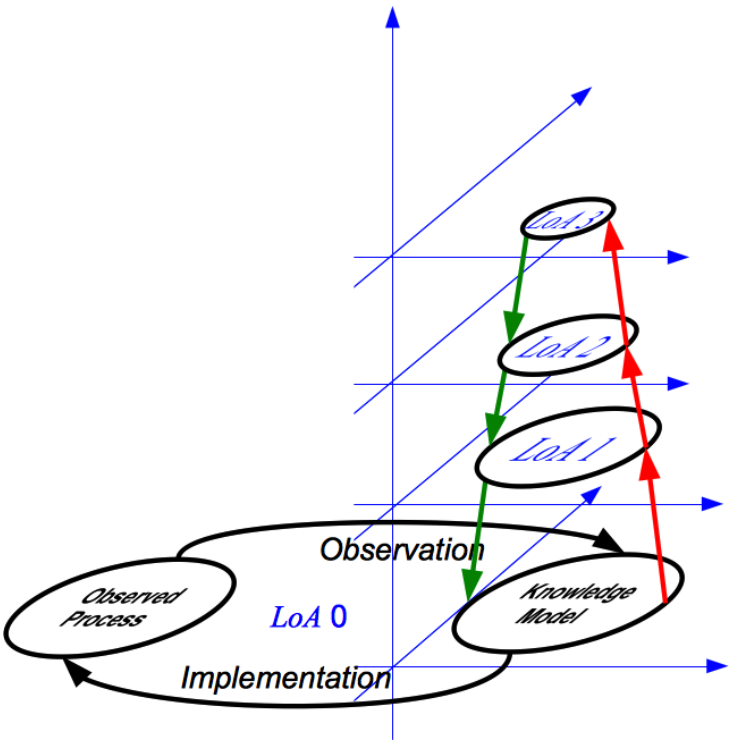


Figure 8.5: Nested GoA composed of several LoAs

## 8.3 TOM4A, an AR-PSM based on $TOT(\mathbb{Z})$ Category

Let us consider the four basic cognitive operations of any PSM (cf. figure 8.2):

- Observation.
  The instrumentation (sensors and actuators) of the *observed process* (i.e. from which a problem occurs) produces a *flow of information* (typically a flow of sequence of characters) that associates *messages* and *timestamps* (i.e. timed messages). The human analysis of such a flow of timed messages allows the construction of the *Knowledge Model*. Such a *Knowledge Model* belongs to the semantic space (see figure 8.2) and may be composed of a collection of timed functions producing the *flow of information*, a set of discrete values that timed functions can take, a set of relations between these discrete values, etc. All elements of the semantic space can be given a semantic interpretation by humans. This *Knowledge Model* represents the most concrete level of abstraction of the observed process (see figure 8.4). Such a LoA is denoted $\mathcal{L}^0$ and is said to be at the level 0 of abstraction.

- Representation.
  From the *Observation step*, basic elements of the TOT framework can be defined:

  - timed functions $x_i(t)$,
  - variable names $x_i$,
  - constants $\delta_{i_k}$ and sets of constants $\Delta_i$,
  - stochastic clocks $\Gamma_i$,
  - relations $\Delta_i \to \Delta_j$ between sets of constants $\Delta_i$ and $\Delta_j$,

– observed process $(X(t), \Theta(X, \Delta))$.

From these basic elements, observations classes $O_{i_k} = \{(x_i, \delta_{i_k})\}$ are built, linking variables names $x_i$ with constant $\delta_{i_k}$.

Timed messages are represented as *timed observations* $(\delta_{i_k}, t_{i_k})$, that is to say, occurrences of observation classes $O_{i_k}(t_{i_k}) \equiv (\delta_{i_k}, t_{i_k})$: these timed observations are written by a Unary Observer $\Theta_i(x_i, \Delta_i)$ whose meaning is the assignation $\theta_i(x_i, \delta_{i_k}, t_{i_k})$ of the predicate $\theta_{\Theta_i}(x_{\Theta_i}, \delta_{\Theta_i}, t_{\Theta_i})$. The observed process is represented by the superposition $\bigcup_{i\in[1;m]} (x_i(t), \Theta_i(x_i, \Delta_i))$ of $m$ safe Unary Observers $\Theta_i(x_i, \Delta_i)$ observing timed function $x_i(t)$. The observed *flow of information* is represented with a *m-ary sequences* $\Omega^1 = \bigcup_{i\in[1;m]} \omega_i(t_{n_i})$, $m \in \mathbb{N}^\star$. Each sequence $\omega_i(t_{n_i}) = \{O_{i_k}(t_{i_k}) \equiv (\delta_{i_k}, t_{i_k}), \delta_{i_k} \in \Delta_i, t_{i_k} \in \Gamma_i\}$ contains $n_i \in \mathbb{N}^\star$ timed observations. As seen in chapter 7, the $TOT(\Delta^1)$ Category can be built where $\Delta^1 = \bigcup_{i\in[1;m]} \Delta_i$.

According to chapter 5, the LoA $\mathcal{L}^1 = \langle \mathcal{S}^1, \mathcal{T}^1 \rangle$ can be defined where:

– $\mathcal{S}^1 = (\Omega^1, \overset{\Delta\tau_{ij}}{+})$ is the algebraic structure of the observed process;

– $\mathcal{T}^1 = (\Omega^1, \mathcal{I}^1)$ is the observable space of the observed process.

From observable space $\mathcal{T}^1$, the abstract chronicle model $\mathcal{M}^1$ can be built. This means that the representation cognitive process called the *Representations' Model* is represented with the abstract chronicle model $\mathcal{M}^1$ in the TOT mathematical framework. From relations between sets of constants $\Delta_i$ and $\Delta_j$, Modeling functors $TOT(\Delta_i) \rightarrow TOT(\Delta_j)$ between categories $TOT(\Delta_i)$ and $TOT(\Delta_j)$ can be built at this level of abstraction (see chapter 7). Abstraction functors implementing the composition of observers are also built. They link a category at a given LoA to a category at a more abstract LoA, syntacticly poorer and semantically richer. Abstraction functors allows to change the LoA of the observed process in order to solve the observed problem at the adequate LoA. These categories and its elements may be arithmetized thanks to the Gödel numbering function seen in 7.4 in oder to work only with categories of the form $TOT(\mathbb{P}_i)$ with $\mathbb{P}_i \subseteq \mathbb{Z}$.

- Interpretation.
  From the *Representation step*, a *reification process* is operated in order to provide a solution of the observed problem at a more concrete level of abstraction. This step *specifies* then an *interpretable* solution by humans of the observed problem.

- Implementation.
  Using the adequate (and available) technologies, the specified solution allows the design of technical artifacts that can be implemented in the real world to improve the observability of the process. An improved observability of the process *makes possible* or simplifies *all the operations* that can be made on the process (monitoring, diagnosis, prognosis or control for instance).

These four basic cognitive operations are part of the TOM4A Methodology: next sections give a concrete application of these steps concerning the *discovering of potential internal fraud models in a stream of banking transactions.*

## 8.4 Internal Fraud Detection in the Banking Industry

For the last three decades, the exponential development of the information systems of banks and financial companies has allowed the usage of Data Mining and Machine Learning techniques to define new services. In particular, these techniques have been used to tackle the problem of fraud detection in the banking industry where two types of fraud are distinguished: external and internal frauds. External fraud is the usual notion of fraud: a swindler uses a bank customer's payment means to get money illegaly. Internal frauds are, fortunately, less common and less frequent: the swindler is an employee of the bank having the access rights to execute transactions. This peculiarity of internal frauds has a significant impact on the fraud detection problem. Because the swindler is an employee of the bank, the fraud is based on a set of *pairs of transactions*, the first being a debit to the account of a customer and the second being a credit to the account of a bank employee. In other words, the internal fraud detection problem is based on the discovering of pairs of potentially fraudulent transactions establishing a binary relation between a customer and a bank employee. By contrast, detecting external frauds is based on the discovering of a set of *single* potentially fraudulent transactions. But in both cases, the problem of fraud detection is particularly difficult to solve because since, by construction, swindlers are very imaginative, the models of fraud evolve continuously so that the fraud detection systems must learn from the new fraud techniques.

This section proposes a concrete application of the TOM4A Methodology to detect internal frauds and model the fraud technique with a simple graph describing the financial movements characterizing the fraud technique. The major benefits of the proposed approach are (i) the reduction of the complexity of the problem from $O(n^2)$ to $O(n)$ and (ii) the ability of representing the fraud technique at three levels of abstraction simultaneously (customer, account and transaction type levels). This approach is implemented in a Java program called TOM4FFS (Timed Observations Mining for Fraud Fighting System), that is able to detect and model internal frauds online, in real time, using a mere professional personal computer. This program can handle more than 4 billions transactions a day.

In order to solve such a problem, basic cognitive operations of the TOM4A Methodology are concretely implemented. Next section is concerned with the *Observation* step followed by a section dealing with the *Representation* step for finally ending with a section concerning the *Interpretation* step.

## 8.5 Observation Step

### 8.5.1 Problem of the Observed Process

An internal fraud is a particular sequence of non-compliant transactions whose aim is to move money from customer accounts to some account of a tactless bank employee. The type of fraud sequences studied in this paper is made of a set of pairs $(a_{k_i}(t_{k_i}), a_{k_j}(t_{k_j}))$ of transactions where

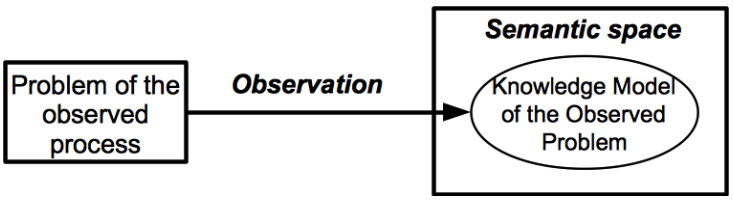


Figure 8.6: Observation step in the TOM4 Methodology

the first transaction $a_{k_i}(t_{k_i})$ of each pair *debits* an amount $a_{k_i}$ of money from an account of a customer *before* the second transaction $a_{k_j}(t_{k_j})$ *credits the same amount* to one of the accounts of the tactless employee of the bank (i.e. $t_{k_i} < t_{k_j}$).

The problem of the internal fraud detection is then to find the *minimal* set of transaction pairs $(a_{k_i}(t_{k_i}), a_{k_j}(t_{k_j}))$ satisfying the following constraints:

**Definition 8.2** *Internal Fraud Constraints*

1. *The debited account belongs to a bank customer.*
2. *The credited account belongs to a bank employee.*
3. *The debited and credited amounts are the same:* $a_{k_i} = -a_{k_j}$.
4. *The debiting transaction precedes the crediting transaction:* $t_{k_i} \leq t_{k_j}$.

Given a database of $n$ transactions, the brute-force solution of this problem consists in building a $n \times n$ matrix to check the constraints of definition 8.2 for each pair of transactions. The complexity of this problem is thus $O(n^2)$: for example, when $n$ is evaluated in hundreds of thousands of transactions (i.e. 100 000 or $10^5$), which is very frequent in the banking domain, the number of pairs to analyze is evaluated in dozens of billion (i.e. $10^{10}$), making the problem hard for humans as much as for computers. As an illustration, the internal fraud studied in this section has been detected by one of the victims (i.e. a customer) some time after the fraud. The bank's expert took around 6 months to demonstrate *non-compliance* of the fraud with the following method: 1°) getting an extraction of the agency database containing the potentially fraudulent transaction, 2°) reducing (in time and space) this extraction to a minimal set of transactions, 3°) finding the pairs of fraudulent transactions in this minimal set, and 4°) build the model of the fraud technique used by the tactless employee. With such a proof, the bank can refund the customer and take action toward the tactless employee. Anyway, such a theoretical complexity of $O(n^2)$ is hardly compatible with a program aiming to detect and model, online and in real time, the *potentially* fraudulent transactions from a continuous flow (note that a transaction remains *potentially* fraudulent until its *non-compliance* has been demonstrated). This example shows that internal fraud bank's experts need tools to assist the steps 2 (database reduction), 3 (detection of potentially fraudulent transactions) and 4 (fraud modeling). It is appropriate to mention that the bank's expert considers that the example under study is a particularly complicated fraud case. In this section, we present the TOM4FFS program designed for online and real time internal fraud detection in the banking domain. TOM4FFS supports the steps 2 (database reduction), 3 (potentially fraudulent transaction detection) and 4 (fraud modeling) of the general problem solving method of internal fraud detection.

### 8.5.2 Knowledge Model of the Observed Problem

Figure 8.7 represents the first rows of the transaction database from which we have to find the set of potential fraudulent transactions. This datababe contains a total of 1492 rows and six columns:

- column "ID_CLI" represents the identification number of a client. There are four clients denoted 1001 to 1004 defining a set $\Delta_{ID\_CLI}$ such as:

$$ID\_CLI \in \{1001; 1002; 1003; 1004\} \equiv \Delta_{ID\_CLI} \tag{8.1}$$

  Moreover, we know that the client 1003 is the manager of the bank.

- column "ID_CPTE" represents the identification number of a banking account. There are 30 banking accounts denoted 2001 to 2024 and 2026 to 2031 defining a set $\Delta_{ID\_CPTE}$ such as:

$$ID\_CPTE \in \{2001; ...; 2024\} \cup \{2026; ...; 2031\} \equiv \Delta_{ID\_CPTE} \tag{8.2}$$

  We know that:

  - client 1001 owns 8 banking accounts: 2006, 2008, 2009, 2012, 2013, 2014, 2015 and 2022.

$$ID\_CLI = 1001 \Rightarrow ID\_CPTE \in \{2006; 2008; 2009; 2012; 2013; 2014; 2015; 2022\} \tag{8.3}$$

  - client 1002 owns 4 banking accounts: 2005, 2011, 2019 and 2023.

$$ID\_CLI = 1002 \Rightarrow ID\_CPTE \in \{2005; 2011; 2019; 2023\} \tag{8.4}$$

  - client 1003 (i.e. the manager) owns 11 banking accounts: 2001, 2002, 2003, 2004, 2024, 2026, 2027, 2028, 2029, 2030 and 2031.

$$ID\_CLI = 1003 \Rightarrow ID\_CPTE \in \{2001; 2002; 2003; 2004; 2024; 2026; 2027; 2028; 2029; 2030; 2031\} \tag{8.5}$$

  - client 1004 owns 7 banking accounts: 2007, 2010, 2016, 2017, 2018, 2020 and 2021.

$$ID\_CLI = 1004 \Rightarrow ID\_CPTE \in \{2007; 2010; 2016; 2017; 2018; 2020; 2021\} \tag{8.6}$$

- column "ID_TYP_EVT" represents the identification number of a transaction type. There are 40 transaction types denoted 3001 to 3040 defining a set $\Delta_{ID\_TYP\_EVT}$ such as:

$$ID\_TYP\_EVT \in \{3001; ...; 3040\} \equiv \Delta_{ID\_TYP\_EVT} \tag{8.7}$$

- column "ID_MVT" represents the identification number of a transaction: each transaction has a unique identification number (between 8 and 2920) defining a set $\Delta_{ID\_MVT}$. Such a number is not used in our analyses.

- column "DAT_EVT" represents the date and time a transaction type. The first transaction occurs on 2009-01-02 at 01:19:18 and the last transaction on 2010-03-12 at 01:12:09 defining a set $\Delta_{DAT\_EVT}$ such as:

$$DAT\_EVT \in \{2009-01-02\ 01:19:18; ...; 2010-03-12\ 01:12:09\} \equiv \Delta_{DAT\_EVT} \tag{8.8}$$

  This database represents then transactions spread over 1 year and 2 months.

- column "MT_EVT" represents the amount of money (in €) of a transaction. This involves amounts of money from $-445200.00$€ to $460614.36$€ defining an interval $\Delta_{MT\_EVT}$ such as:

$$MT\_EVT \in [-445200.00; ...; 460614.36] \equiv \Delta_{MT\_EVT} \tag{8.9}$$

  Amount of money are real numbers of $\Re$ with a two digits precision. Multiplying them by 100 allows us to consider these numbers as relative integers of $\mathbb{Z}$ and, in this case, these numbers represent cents of €.

| ID_CLI | ID_CPTE | ID_TYP_EVT | ID_MVT | DAT_EVT | MT_EVT |
|---|---|---|---|---|---|
| 1001 | 2008 | 3019 | 978 | 2009-01-02 01:19:18 | -6,10 |
| 1003 | 2001 | 3024 | 1400 | 2009-01-02 02:28:09 | -100,00 |
| 1001 | 2008 | 3024 | 979 | 2009-01-05 01:47:26 | -100,00 |
| 1001 | 2008 | 3006 | 987 | 2009-01-05 02:35:48 | -16,85 |
| 1001 | 2008 | 3039 | 986 | 2009-01-05 03:56:08 | 394,16 |
| 1002 | 2005 | 3035 | 1257 | 2009-01-05 04:02:20 | 100,00 |
| 1002 | 2005 | 3006 | 1256 | 2009-01-05 05:32:30 | -7,99 |
| 1002 | 2005 | 3006 | 1255 | 2009-01-05 06:03:11 | -15,50 |
| 1003 | 2001 | 3019 | 1403 | 2009-01-05 07:30:31 | -99,50 |
| 1003 | 2001 | 3019 | 1405 | 2009-01-05 08:27:51 | -223,53 |
| 1003 | 2001 | 3019 | 1406 | 2009-01-05 09:02:51 | -173,17 |
| 1003 | 2001 | 3019 | 1404 | 2009-01-05 10:11:40 | -43,11 |
| 1003 | 2001 | 3006 | 1408 | 2009-01-05 11:35:49 | -9,59 |
| 1003 | 2001 | 3024 | 1407 | 2009-01-05 12:43:57 | -100,00 |
| 1004 | 2007 | 3019 | 9 | 2009-01-05 13:53:07 | -19,80 |
| 1004 | 2007 | 3024 | 11 | 2009-01-05 14:26:35 | -60,00 |
| 1004 | 2007 | 3006 | 12 | 2009-01-05 15:07:38 | -3,00 |
| 1004 | 2007 | 3024 | 10 | 2009-01-06 01:15:47 | -60,00 |
| 1004 | 2007 | 3004 | 8 | 2009-01-06 02:06:39 | -724,08 |
| 1001 | 2008 | 3004 | 988 | 2009-01-07 01:17:24 | -1 423,00 |
| 1003 | 2001 | 3019 | 1414 | 2009-01-07 02:37:02 | -370,00 |
| 1004 | 2007 | 3019 | 16 | 2009-01-07 03:22:14 | -164,59 |
| 1003 | 2001 | 3036 | 1424 | 2009-01-08 01:25:39 | -10,00 |
| 1003 | 2004 | 3026 | 1423 | 2009-01-08 02:35:10 | -53,55 |
| 1003 | 2001 | 3024 | 1421 | 2009-01-08 03:36:45 | -100,00 |
| 1003 | 2001 | 3019 | 1420 | 2009-01-08 04:11:14 | -2,10 |
| 1003 | 2001 | 3030 | 1427 | 2009-01-09 01:41:34 | -15,94 |
| 1003 | 2001 | 3031 | 1432 | 2009-01-09 02:28:34 | -1 037,80 |

Figure 8.7: First rows of the transaction database

Let us denote:

- $x_i(t)$, $i \in \Delta_{ID\_CLI}$, the temporal evolution of the transaction amount of *client id* $i$;
- $x_j(t)$, $j \in \Delta_{ID\_CPTE}$, the temporal evolution of the transaction amount of *account id* $j$;
- $x_l(t)$, $l \in \Delta_{ID\_TYP\_EVT}$, the temporal evolution of the transaction amount of *transaction type id* $l$;

All these temporal functions are $\mathbb{Z}$-valued piecewise functions defined on $\mathfrak{R}$:

$$\forall \alpha \in \{i,j,l\}, \forall a_{\alpha_k} \in \mathbb{Z}, \forall t_k \in \mathfrak{R}, \quad \begin{array}{rccc} x_\alpha & : & \mathfrak{R} & \rightarrow & \mathbb{Z} \\ & & t & \mapsto & \sum_{k=-\infty}^{+\infty} a_{\alpha_k}.H(t-t_k) \end{array} \tag{8.10}$$

The function $H(t)$ is the Heaviside step function:

$$H(t) = \begin{cases} 0 & \text{if } t < 0 \\ 1 & \text{if } t \geq 0 \end{cases} \tag{8.11}$$

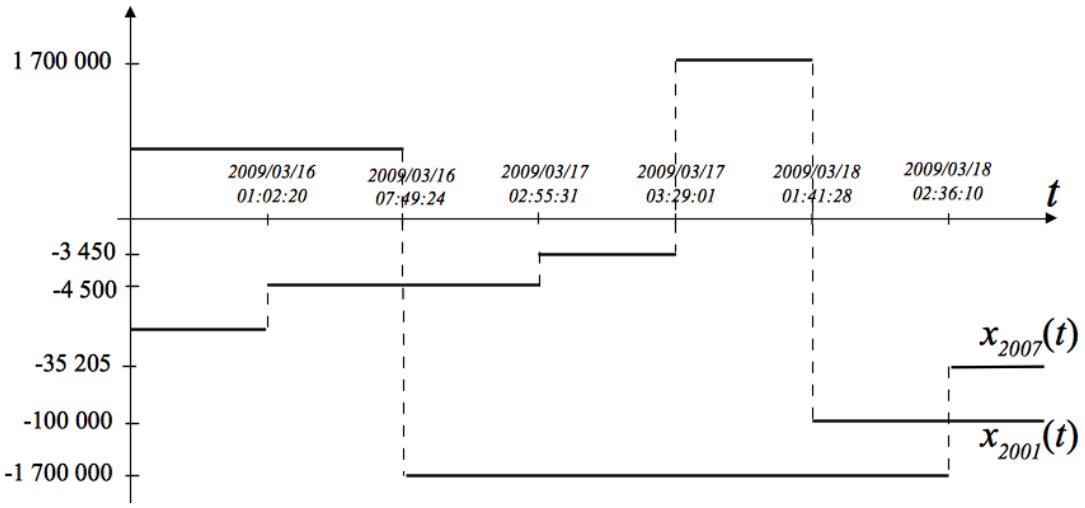


Figure 8.8: Piecewise functions for accounts 2001 and 2007

Figure 8.8 illustrates the piecewise functions representing the evolution over time of the transaction amount expressed in cents for banking account ids 2001 and 2007.

From such a database, some relations between sets $\Delta_{ID\_CLI}$, $\Delta_{ID\_CPTE}$ and $\Delta_{ID\_TYP\_EVT}$ can be revealed. Claiming that client id 1001 owns 8 banking accounts subsumes the existence of a relation denoted $\mathcal{R}^{1001}_{ID\_CLI \rightarrow ID\_CPTE}$ linking sets $\Delta_{ID\_CLI}$ and $\Delta_{ID\_CPTE}$ such as:

$$\begin{array}{rccc} \mathcal{R}^{1001}_{ID\_CLI \rightarrow ID\_CPTE} & : & \Delta_{ID\_CLI} & \rightarrow & \Delta_{ID\_CPTE} \\ & & 1001 & \mapsto & 2006 \\ & & 1001 & \mapsto & 2008 \\ & & 1001 & \mapsto & 2009 \\ & & 1001 & \mapsto & 2012 \\ & & 1001 & \mapsto & 2013 \\ & & 1001 & \mapsto & 2014 \\ & & 1001 & \mapsto & 2015 \\ & & 1001 & \mapsto & 2022 \end{array} \tag{8.12}$$

The same reasoning can be done concerning clients 1002, 1003 and 1004 revealing relations respectively denoted $\mathcal{R}^{1002}_{ID\_CLI \to ID\_CPTE}$, $\mathcal{R}^{1003}_{ID\_CLI \to ID\_CPTE}$ and $\mathcal{R}^{1004}_{ID\_CLI \to ID\_CPTE}$. From this relations, the relation denoted $\mathcal{R}_{ID\_CLI \to ID\_CPTE}$ mapping client ids with account ids can be built:

$$\begin{array}{l} \mathcal{R}_{ID\_CLI \to ID\_CPTE} = \\ \mathcal{R}^{1001}_{ID\_CLI \to ID\_CPTE} \bigcup \\ \mathcal{R}^{1002}_{ID\_CLI \to ID\_CPTE} \bigcup \\ \mathcal{R}^{1003}_{ID\_CLI \to ID\_CPTE} \bigcup \\ \mathcal{R}^{1004}_{ID\_CLI \to ID\_CPTE} \end{array} \tag{8.13}$$

From this relation, we can build the inverse relation denoted $\mathcal{R}_{ID\_CPTE \to ID\_CLI}$ linking sets $\Delta_{ID\_CPTE}$ to $\Delta_{ID\_CLI}$.

With the same method, relation $\mathcal{R}_{ID\_CPTE \to ID\_TYPE\_EVT}$ between sets $\Delta_{ID\_CPTE}$ and $\Delta_{ID\_TYP\_EVT}$ can be built. For example, the first row of the database of figure 8.7 shows that the account id 2008 is related to the transaction type id 3019. Doing so for the whole database allows to build $\mathcal{R}_{ID\_CPTE \to ID\_TYPE\_EVT}$. Once built, the reverse relation $\mathcal{R}_{ID\_TYPE\_EVT \to ID\_CPTE}$ can also be built. And again with the same method, relations $\mathcal{R}_{ID\_CLI \to ID\_TYPE\_EVT}$ and $\mathcal{R}_{ID\_TYPE\_EVT \to ID\_CLI}$ between sets $\Delta_{ID\_CLI}$ and $\Delta_{ID\_TYP\_EVT}$ can also be built. These relation are anything: injective, surjective, etc.

To sum up the situation, at this stage of the methodology, we have then:

- a collection of piecewise functions representing the evolution over time of the transaction amounts according to the client ids, banking account ids and transaction type ids;
- relations between sets of client ids, banking account ids and transaction type ids.

These elements represent the semantic space of the observed problem which is the most concrete description of the situation. The Knowledge Model represents the most concrete level of abstraction of the observed problem. Such a LoA is denoted $\mathcal{L}^0$ and is said to be at the level 0 of abstraction.

Let us now deal with the Representation step of the TOM4A Methodology.

## 8.6 Representation Step

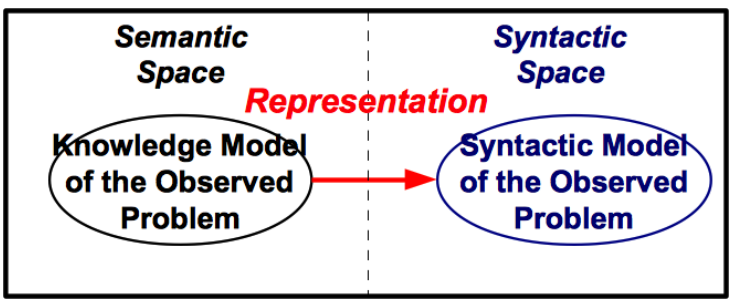


Figure 8.9: Representation step in the TOM4 Methodology

### 8.6.1 Representation of a transaction

Let us consider the first transaction of the database, *"1001|2008|3019|978|2009-01-02 01:19:18|-6,10"*. Such a transaction is represented with a timed observation $O(t_1) \equiv (\delta_1, t_1) \equiv (\text{"}1001|2008|3019|978|-6,10\text{"}, 2009-01-02\ 01:19:18)$ where constant $\delta_1 = \text{"}1001|2008|3019|978|-6,10\text{"}$ and timestamp $t_1 = 2009-01-02\ 01:19:18$.

Thus, the constant $\delta_1$ has a particular structure containing the symbol "|" breaking this latter in five different items:

1. the client id `1001`,
2. the account id `2008`,
3. the transaction type id `3019`,
4. the index of the transaction `978` and
5. the amount of the transaction `-6,10`.

In a more general way, a constant $\delta_k$ is an instance of the following general structure:

$$\delta_k = \text{"}p_1(k)|p_2(k)|p_3(k)|p_4(k)|p_5(k)\text{"} \tag{8.14}$$

Where:

1. $p_1(k)$ is an element of the set $\Delta_{ID\_CLI}$,
2. $p_2(k)$ is an element of the set $\Delta_{ID\_CPTE}$,
3. $p_3(k)$ is an element of the set $\Delta_{ID\_TYPE\_EVT}$,
4. $p_4(k)$ is an element of the set $\Delta_{ID\_MVT}$,
5. $p_5(k)$ is an element of the set $\Delta_{MT\_EVT}$.

In order to solve the internal fraud problem introduced in section 8.5.1 with respect of the internal fraud constraints of definition 8.2, the analyse of the transactions contained in the database of figure 8.7 can be done according to three different categories:

- the category of the clients: to this aim, the set $\Delta_{ID\_CLI}$ of client ids is considered;
- the category of the banking accounts: to this aim, the set $\Delta_{ID\_CPTE}$ of account ids is considered;
- the category of the transaction types: to this aim, the set $\Delta_{ID\_TYPE\_EVT}$ of transaction type ids is considered.

The next section details these points.

### 8.6.2 Syntactic Model of the Observed Problem at the First LoA

Let us first deal with the category of the clients. To this aim let us consider that the transaction database has been provided by the observed process $(X_{CLI}(t), \Theta_{CLI}(X_{CLI}, \Delta^1_{CLI}))$ where the dynamic process $X_{CLI}(t)$ is composed of the four piecewise functions $x_i(t)$ representing the evolution over time of transaction amounts for client ids $i \in \Delta_{ID\_CLI}$:

$$X_{CLI}(t) = \{x_i(t), i \in \Delta_{ID\_CLI}\} = \{x_{1001}(t), x_{1002}(t), x_{1003}(t), x_{1004}(t)\} \tag{8.15}$$

This allows to define the set $X_{CLI}$ of variable names such as:

$$X_{CLI} = \{x_i, i \in \Delta_{ID\_CLI}\} = \{x_{1001}, x_{1002}, x_{1003}, x_{1004}\} \tag{8.16}$$

Each variable name $x_i$ takes discrete values $\delta_{i_k}$ of the form:

$$\delta_{i_k} = "i|p_2(k)|p_3(k)|p_4(k)|p_5(k)" \tag{8.17}$$

Let be $\Delta^1_i$ the set of $n_i \in \mathbb{N}$ constants $\delta_{i_k}$ for the client id $i \in \Delta_{ID\_CLI}$:

$$\forall i \in \Delta_{ID\_CLI}, \Delta^1_i = \{\delta_{i_k}, k \in [1; n_i], n_i \in \mathbb{N}\} \tag{8.18}$$

The set $\Delta^1_{CLI}$ containing constants $\delta_k$ of the form 8.14 from the transaction database, it can be partitioned into four sets $\Delta^1_i$ such as:

$$\begin{aligned} \Delta^1_{CLI} = \bigcup_{i \in \Delta_{ID\_CLI}} \Delta^1_i = \Delta^1_{1001} \bigcup \Delta^1_{1002} \bigcup \Delta^1_{1003} \bigcup \Delta^1_{1004} \\ \forall (i,j) \in \Delta^2_{ID\_CLI}, i \neq j, \Delta^1_i \bigcap \Delta^1_j = \emptyset \end{aligned} \tag{8.19}$$

According to the superposition theorem 3.2, the program $\Theta_{CLI}(X_{CLI}, \Delta^1_{CLI})$ can be decomposed into a superposition of four Unary Observers such as:

$$\begin{aligned} \Theta_{CLI}(X_{CLI}, \Delta^1_{CLI}) = \bigcup_{i \in \Delta_{ID\_CLI}} \Theta_i(x_i, \Delta^1_i) \\ = \Theta_{1001}(x_{1001}, \Delta^1_{1001}) \bigcup \Theta_{1002}(x_{1002}, \Delta^1_{1002}) \bigcup \Theta_{1003}(x_{1003}, \Delta^1_{1003}) \bigcup \Theta_{1004}(x_{1004}, \Delta^1_{1004}) \end{aligned} \tag{8.20}$$

Each Unary Observer $\Theta_i(x_i, \Delta^1_i)$ implements a predicate denoted $\theta_i(x_{\theta_i}, \delta_{\theta_i}, t_{\theta_i})$ such as:

$$\theta_i(x_{\theta_i}, \delta_{\theta_i}, t_{\theta_i}) \equiv p_1(\theta_i) = i \wedge t_{\theta_i - 1} < t_{\theta_i} \wedge x_{\theta_i}(t_{\theta_i - 1}) \neq x_{\theta_i}(t_{\theta_i}) \tag{8.21}$$

Each Unary Observer $\Theta_i(x_i, \Delta^1_i)$ writes a timed observation denoted $O(t_{i_k}) \equiv (\delta_{i_k}, t_{i_k})$ for each assignation $\theta_i(x_i, \delta_{i_k}, t_{i_k})$ of the predicate $\theta_i(x_{\theta_i}, \delta_{\theta_i}, t_{\theta_i})$:

$$\theta_i(x_i, \delta_{i_k}, t_{i_k}) \Rightarrow write(O(t_{i_k}) \equiv (\delta_{i_k}, t_{i_k})) \tag{8.22}$$

According to the predicate 8.21, the assignation 8.22 means that the Unary Observer $\Theta_i(x_i, \Delta^1_i)$ writes a timed observation $O(t_{i_k}) \equiv (\delta_{i_k}, t_{i_k})$ each time the value of the function $x_i(t)$, for the client id $i$, changes.

Let us denote $O_{i_k} = \{(x_i, \delta_{i_k})\}$, the observation class linking variable name $x_i$ to constant

$\delta_{i_k}$. Let us denote $\omega_i^1(t_{n_i}) = \{O(t_{i_k}) \equiv (\delta_{i_k}, t_{i_k}), \delta_{i_k} \in \Delta_i^1, t_{i_k} \in \Gamma_i\}$, the sequence of $n_i \in \mathbb{N}$ timed observations $O(t_{i_k})$. Such a sequence defines the stochastic clock $\Gamma_i = \{t_{i_k}, i_k \in [1; n_i]\}$ of timestamps $t_{i_k}$.

Let us denote $\Omega_{CLI}^1 = \bigcup_{i \in \Delta_{ID\_CLI}} \omega_i^1(t_{n_i})$, the superposition of the four sequences of timed observations $\omega_i^1(t_{n_i})$.

| **Sequence** | $n_i$ |
|---|---|
| $\omega_{1001}^1(t_{n_{1001}})$ | 180 |
| $\omega_{1002}^1(t_{n_{1002}})$ | 103 |
| $\omega_{1003}^1(t_{n_{1003}})$ | 709 |
| $\omega_{1004}^1(t_{n_{1004}})$ | 407 |
| **Total** | **1399** |

Table 8.1: Number $n_i$ of timed observations for sequences $\omega_i^1(t_{n_i})$, $i \in \Delta_{ID\_CLI}$

Table 8.1 sums up the number of timed observations for each client id. It is notable that the most active client is the manager of the bank (client 1003).

As seen in chapter 7, we can build the $TOT(\Delta_{CLI}^1)$ Category where:

- objects are Unary Observers $\Theta_i(x_i, \Delta_i^1)$, $i \in \Delta_{ID\_CLI}$;
- morphisms are relations $f_{ij} : \Theta_i \to \Theta_j$, $(i, j) \in \Delta_{ID\_CLI} \times \Delta_{ID\_CLI}$.

According to definition 7.3, let us consider the level of abstraction of the $TOT(\Delta_{CLI}^1)$ Category, denoted $\mathcal{L}_{CLI}^1 = \langle \mathcal{S}_{CLI}^1, \mathcal{T}_{CLI}^1 \rangle$, where:

- $\mathcal{S}_{CLI}^1 = (\Omega_{CLI}^1, \overset{\Delta\tau_{ij}}{+})$ is the algebraic structure of the observed process $(X_{CLI}(t), \Theta_{CLI}(X_{CLI}, \Delta_{CLI}^1))$;
- $\mathcal{T}_{CLI}^1 = (\Omega_{CLI}^1, \mathcal{I}_{CLI}^1)$ is the observable space of the observed process $(X_{CLI}(t), \Theta_{CLI}(X_{CLI}, \Delta_{CLI}^1))$.

Let us recall that the algebraic structure $\mathcal{S}_{CLI}^1$ provides a particular description of the transaction database according to the point of view of client identification numbers. From observable space $\mathcal{T}_{CLI}^1$, we can build the abstract chronicle model $\mathcal{M}_{CLI}^1$ containing all temporal binary relations $r_{i_k j_{k+1}}(O_{i_k}, O_{j_{k+1}}, \Delta\tau_{i_k j_{k+1}})$ between observation classes $O_{i_k}$ and $O_{j_{k+1}}$ (see section 5.5 and property 19):

$$\mathcal{M}_{CLI}^1 = \{r_{i_k j_{k+1}}(O_{i_k}, O_{j_{k+1}}, \Delta\tau_{i_k j_{k+1}}), (i, j) \in \Delta_{ID\_CLI} \times \Delta_{ID\_CLI}\} \tag{8.23}$$

The abstract chronicle model $\mathcal{M}_{CLI}^1$ represents all binary sequences of successive transactions from a client $i \in \Delta_{ID\_CLI}$ to a client $j \in \Delta_{ID\_CLI}$ under the temporal constraint $\Delta\tau_{i_k j_{k+1}}$ that may be observable in the transaction database. The size of such an abstract chronicle model $\mathcal{M}_{CLI}^1$ is, according to property 20:

$$Card(\mathcal{M}_{CLI}^1) = \sum_{\substack{(i,j) \in \Delta_{ID\_CLI}^2 \\ i \neq j}} (N_i + N_j)^2 \tag{8.24}$$

The number $N_i$ represents the number of observations classes associated to the sequence $\omega_i^1(t_{n_i})$. In this case, this number is equal to the number of timed observations contained in this sequence. Thus, equation 8.24 leads to:

$$\begin{aligned} Card(\mathcal{M}^1_{CLI}) = \; & (N_{1001} + N_{1002})^2 + (N_{1001} + N_{1003})^2 + (N_{1001} + N_{1004})^2 + \\ & (N_{1002} + N_{1003})^2 + (N_{1002} + N_{1004})^2 + \\ & (N_{1003} + N_{1004})^2 \\ Card(\mathcal{M}^1_{CLI}) = \; & (180 + 103)^2 + (180 + 709)^2 + (180 + 407)^2 + \\ & (103 + 709)^2 + (103 + 407)^2 + \\ & (709 + 407)^2 \\ Card(\mathcal{M}^1_{CLI}) = \; & 3\ 379\ 879 \end{aligned} \tag{8.25}$$

Thus, the abstract chronicle model $\mathcal{M}^1_{CLI}$ contains 3 379 879 temporal binary relations modeling all the successive transactions between a client $i \in \Delta_{ID\_CLI}$ to a client $j \in \Delta_{ID\_CLI}$. There exists then 3 379 879 binary sequences of transactions to analyse in order to find the ones which are potentially fraudulent. Let us remark that transactions from a client $i$ to himself are contained in this model. Such transactions have no real interrest in our particular problem. At this stage of the TOM4A Methodology, the complexity of this problem needs to be reduced. In the next sections, we demonstrate that operating an abstraction on the raw data of the database simplifies widely the problem.

But before this, let us deal with the category of the banking accounts. The reasoning is the same than the one operated for the clients.

Let us consider that the transaction database has been provided by the observed process $(X_{CPTE}(t), \Theta_{CPTE}(X_{CPTE}, \Delta^1_{CPTE}))$ where the dynamic process $X_{CPTE}(t)$ is composed of the 30 piecewise functions $x_j(t)$ representing the evolution over time of transaction amounts for account ids $j \in \Delta_{ID\_CPTE}$:

$$X_{CPTE}(t) = \{x_j(t), j \in \Delta_{ID\_CPTE}\} = \{x_{2001}(t), x_{2002}(t), ..., x_{2024}(t)\} \cup \{x_{2026}(t), ..., x_{2031}(t)\} \tag{8.26}$$

This allows to define the set $X_{CPTE}$ of variable names such as:

$$X_{CPTE} = \{x_j, j \in \Delta_{ID\_CPTE}\} = \{x_{2001}, x_{2002}, ..., x_{2024}\} \cup \{x_{2026}, ..., x_{2031}\} \tag{8.27}$$

Each variable name $x_j$ takes discrete values $\delta_{j_k}$ of the form:

$$\delta_{j_k} = "p_1(k)|j|p_3(k)|p_4(k)|p_5(k)" \tag{8.28}$$

Let be $\Delta^1_j$ the set of $n_j \in \mathbb{N}$ constants $\delta_{j_k}$ for the account id $j \in \Delta_{ID\_CPTE}$:

$$\forall j \in \Delta_{ID\_CPTE}, \Delta^1_j = \{\delta_{j_k}, k \in [1; n_j], n_j \in \mathbb{N}\} \tag{8.29}$$

The set $\Delta^1_{CPTE}$ containing constants $\delta_k$ of the form 8.14 from the transaction database, it can be partitioned into 30 sets $\Delta^1_j$ such as:

$$\begin{gathered}\Delta^1_{CPTE} = \bigcup_{j \in \Delta_{ID\_CPTE}} \Delta^1_j \\ \forall (i,j) \in \Delta^2_{ID\_CPTE}, i \neq j, \Delta^1_i \bigcap \Delta^1_j = \emptyset\end{gathered} \tag{8.30}$$

According to the superposition theorem 3.2, the program $\Theta_{CPTE}(X_{CPTE}, \Delta^1_{CPTE})$ can be decomposed into a superposition of 30 Unary Observers such as:

$$\Theta_{CPTE}(X_{CPTE}, \Delta^1_{CPTE}) = \bigcup_{j \in \Delta_{ID\_CPTE}} \Theta_j(x_j, \Delta^1_j) \tag{8.31}$$

Each Unary Observer $\Theta_j(x_j, \Delta^1_j)$ implements a predicate denoted $\theta_j(x_{\theta_j}, \delta_{\theta_j}, t_{\theta_j})$ such as:

$$\theta_j(x_{\theta_j}, \delta_{\theta_j}, t_{\theta_j}) \equiv p_2(\theta_j) = j \wedge t_{\theta_j - 1} < t_{\theta_j} \wedge x_{\theta_j}(t_{\theta_j - 1}) \neq x_{\theta_j}(t_{\theta_j}) \tag{8.32}$$

Each Unary Observer $\Theta_j(x_j, \Delta^1_j)$ writes a timed observation denoted $O(t_{j_k}) \equiv (\delta_{j_k}, t_{j_k})$ for each assignation $\theta_j(x_j, \delta_{j_k}, t_{j_k})$ of the predicate $\theta_j(x_{\theta_j}, \delta_{\theta_j}, t_{\theta_j})$:

$$\theta_j(x_j, \delta_{j_k}, t_{j_k}) \Rightarrow write(O(t_{j_k}) \equiv (\delta_{j_k}, t_{j_k})) \tag{8.33}$$

According to the predicate 8.32, the assignation 8.33 means that the Unary Observer $\Theta_j(x_j, \Delta^1_j)$ writes a timed observation $O(t_{j_k}) \equiv (\delta_{j_k}, t_{j_k})$ each time the value of the function $x_j(t)$, for the account id $j$, changes.

Let us denote $O_{j_k} = \{(x_j, \delta_{j_k})\}$, the observation class linking variable name $x_j$ to constant $\delta_{j_k}$. Let us denote $\omega^1_j(t_{n_j}) = \{O(t_{j_k}) \equiv (\delta_{j_k}, t_{j_k}), \delta_{j_k} \in \Delta^1_j, t_{j_k} \in \Gamma_j\}$, the sequence of $n_j \in \mathbb{N}$ timed observations $O(t_{j_k})$. Such a sequence defines the stochastic clock $\Gamma_j = \{t_{j_k}, i_k \in [1; n_j]\}$ of timestamps $t_{j_k}$.

Let us denote $\Omega^1_{CPTE} = \bigcup_{j \in \Delta_{ID\_CPTE}} \omega^1_j(t_{n_j})$, the superposition of the 30 sequences of timed observations $\omega^1_j(t_{n_j})$.

| **Sequence** | $n_j$ | **Sequence** | $n_j$ | **Sequence** | $n_j$ |
|---|---|---|---|---|---|
| $\omega^1_{2001}(t_{n_{2001}})$ | 486 | $\omega^1_{2011}(t_{n_{2011}})$ | 3 | $\omega^1_{2021}(t_{n_{2021}})$ | 1 |
| $\omega^1_{2002}(t_{n_{2002}})$ | 6 | $\omega^1_{2012}(t_{n_{2012}})$ | 5 | $\omega^1_{2022}(t_{n_{2022}})$ | 2 |
| $\omega^1_{2003}(t_{n_{2003}})$ | 4 | $\omega^1_{2013}(t_{n_{2013}})$ | 1 | $\omega^1_{2023}(t_{n_{2023}})$ | 1 |
| $\omega^1_{2004}(t_{n_{2004}})$ | 11 | $\omega^1_{2014}(t_{n_{2014}})$ | 2 | $\omega^1_{2024}(t_{n_{2024}})$ | 198 |
| $\omega^1_{2005}(t_{n_{2005}})$ | 95 | $\omega^1_{2015}(t_{n_{2015}})$ | 1 | $\omega^1_{2026}(t_{n_{2026}})$ | 2 |
| $\omega^1_{2006}(t_{n_{2006}})$ | 14 | $\omega^1_{2016}(t_{n_{2016}})$ | 2 | $\omega^1_{2027}(t_{n_{2027}})$ | 2 |
| $\omega^1_{2007}(t_{n_{2007}})$ | 381 | $\omega^1_{2017}(t_{n_{2017}})$ | 10 | $\omega^1_{2028}(t_{n_{2028}})$ | 2 |
| $\omega^1_{2008}(t_{n_{2008}})$ | 149 | $\omega^1_{2018}(t_{n_{2018}})$ | 3 | $\omega^1_{2029}(t_{n_{2029}})$ | 2 |
| $\omega^1_{2009}(t_{n_{2009}})$ | 2 | $\omega^1_{2019}(t_{n_{2019}})$ | 4 | $\omega^1_{2030}(t_{n_{2030}})$ | 2 |
| $\omega^1_{2010}(t_{n_{2010}})$ | 3 | $\omega^1_{2020}(t_{n_{2020}})$ | 3 | $\omega^1_{2031}(t_{n_{2031}})$ | 6 |
| | | | | **Total** | **1403** |

Table 8.2: Number $n_j$ of timed observations for sequences $\omega^1_j(t_{n_j})$, $j \in \Delta_{ID\_CPTE}$

Table 8.2 sums up the number of timed observations for each account id. We can notice here that accounts 2001, 2007, 2008 and 2024 are the ones on which most transactions occur. Accounts 2001 and 2024 belong to the manager (client id 1003). Account 2007 belongs to client 1004. Account 2008 belongs to client 1001.

As seen in chapter 7, we can build the $TOT(\Delta^1_{CPTE})$ Category where:

- objects are Unary Observers $\Theta_j(x_j, \Delta^1_j)$, $j \in \Delta_{ID\_CPTE}$;
- morphisms are relations $f_{ij} : \Theta_i \to \Theta_j$, $(i,j) \in \Delta_{ID\_CPTE} \times \Delta_{ID\_CPTE}$.

According to definition 7.3, let us consider the level of abstraction of the $TOT(\Delta^1_{CPTE})$ Category, denoted $\mathcal{L}^1_{CPTE} = \langle \mathcal{S}^1_{CPTE}, \mathcal{T}^1_{CPTE} \rangle$, where:

- $\mathcal{S}^1_{CPTE} = (\Omega^1_{CPTE}, \overset{\Delta\tau_{ij}}{+})$ is the algebraic structure of the observed process $(X_{CPTE}(t), \Theta_{CPTE}(X_{CPTE}, \Delta^1_{CPTE}))$;
- $\mathcal{T}^1_{CPTE} = (\Omega^1_{CPTE}, \mathcal{I}^1_{CPTE})$ is the observable space of the observed process $(X_{CPTE}(t), \Theta_{CPTE}(X_{CPTE}, \Delta^1_{CPTE}))$.

The algebraic structure $\mathcal{S}^1_{CPTE}$ provides a particular description of the transaction database according to the point of view of account identification numbers. From observable space $\mathcal{T}^1_{CPTE}$, we can build the abstract chronicle model $\mathcal{M}^1_{CPTE}$ containing all temporal binary relations $r_{i_k j_{k+1}}(O_{i_k}, O_{j_{k+1}}, \Delta\tau_{i_k j_{k+1}})$ between observation classes $O_{i_k}$ and $O_{j_{k+1}}$ :

$$\mathcal{M}^1_{CPTE} = \{r_{i_k j_{k+1}}(O_{i_k}, O_{j_{k+1}}, \Delta\tau_{i_k j_{k+1}}), (i,j) \in \Delta_{ID\_CPTE} \times \Delta_{ID\_CPTE}\} \tag{8.34}$$

The abstract chronicle model $\mathcal{M}^1_{CPTE}$ represents all binary sequences of successive transactions from a account $i \in \Delta_{ID\_CPTE}$ to a account $j \in \Delta_{ID\_CPTE}$ under the temporal constraint $\Delta\tau_{i_k j_{k+1}}$ that may be observable in the transaction database. The size of such an abstract chronicle model $\mathcal{M}^1_{CPTE}$ is, according to property 20:

$$Card(\mathcal{M}^1_{CPTE}) = \sum_{\substack{(i,j) \in \Delta^2_{ID\_CPTE} \\ i \neq j}} (N_i + N_j)^2 \tag{8.35}$$

In this case, equation 8.35 leads to:

$$Card(\mathcal{M}^1_{CPTE}) = 14\ 635\ 861 \tag{8.36}$$

Thus, the abstract chronicle model $\mathcal{M}^1_{CPTE}$ contains 14 635 861 temporal binary relations modeling all the successive transactions between an account $i \in \Delta_{ID\_CLI}$ to an account $j \in \Delta_{ID\_CLI}$. So, when considering the category of the account ids, there exists 14 635 861 binary sequences of transactions to analyse in order to find the ones which are potentially fraudulent. This means that if we choose to analyse the database according to the point of view of the banking account ids, it would take four times more operations than if we had chosen the point of view of the client ids. This demonstrates, in a pratical way, that choosing the right category is important to model a given problem in order to bring a solution. This confirms also the need of an abstraction of the raw data in order to reduce the syntax and improve the semantic of the model of such a problem.

Let us consider that the transaction database has been provided by the observed process $(X_{TYP}(t), \Theta_{TYP}(X_{TYP}, \Delta^1_{TYP}))$ where the dynamic process $X_{TYP}(t)$ is composed of the 40 piecewise functions $x_l(t)$ representing the evolution over time of transaction amounts for transaction type ids $l \in \Delta_{ID\_TYP\_EVT\_EVT}$:

$$X_{TYP}(t) = \{x_l(t), l \in \Delta_{ID\_TYP\_EVT}\} = \{x_{3001}(t), x_{3002}(t), ..., x_{3040}(t)\} \tag{8.37}$$

This allows to define the set $X_{TYP}$ of variable names such as:

$$X_{TYP} = \{x_l, l \in \Delta_{ID\_TYP\_EVT}\} = \{x_{3001}, x_{3002}, ..., x_{3040}\} \tag{8.38}$$

Each variable name $x_l$ takes discrete values $\delta_{l_k}$ of the form:

$$\delta_{l_k} = "p_1(k)|p_2(k)|l|p_4(k)|p_5(k)" \tag{8.39}$$

Let be $\Delta^1_l$ the set of $n_l \in \mathbb{N}$ constants $\delta_{l_k}$ for the transaction type id $l \in \Delta_{ID\_TYP\_EVT}$:

$$\forall l \in \Delta_{ID\_TYP\_EVT}, \Delta^1_l = \{\delta_{l_k}, k \in [1; n_l], n_l \in \mathbb{N}\} \tag{8.40}$$

The set $\Delta^1_{TYP}$ containing constants $\delta_k$ of the form 8.14 from the transaction database, it can be partitioned into 40 sets $\Delta^1_l$ such as:

$$\begin{gathered} \Delta^1_{TYP} = \bigcup_{l \in \Delta_{ID\_TYP\_EVT}} \Delta^1_l \\ \forall (i,j) \in \Delta^2_{ID\_TYP\_EVT}, i \neq j, \Delta^1_i \bigcap \Delta^1_j = \emptyset \end{gathered} \tag{8.41}$$

According to the superposition theorem 3.2, the program $\Theta_{TYP}(X_{TYP}, \Delta^1_{TYP})$ can be decomposed into a superposition of 40 Unary Observers such as:

$$\Theta_{TYP}(X_{TYP}, \Delta^1_{TYP}) = \bigcup_{l \in \Delta_{ID\_TYP\_EVT}} \Theta_l(x_l, \Delta^1_l) \tag{8.42}$$

Each Unary Observer $\Theta_l(x_l, \Delta^1_l)$ implements a predicate denoted $\theta_l(x_{\theta_l}, \delta_{\theta_l}, t_{\theta_l})$ such as:

$$\theta_l(x_{\theta_l}, \delta_{\theta_l}, t_{\theta_l}) \equiv p_3(\theta_l) = l \wedge t_{\theta_l - 1} < t_{\theta_l} \wedge x_{\theta_l}(t_{\theta_l - 1}) \neq x_{\theta_l}(t_{\theta_l}) \tag{8.43}$$

Each Unary Observer $\Theta_l(x_l, \Delta^1_l)$ writes a timed observation denoted $O(t_{l_k}) \equiv (\delta_{l_k}, t_{l_k})$ for each assignation $\theta_l(x_l, \delta_{l_k}, t_{l_k})$ of the predicate $\theta_l(x_{\theta_l}, \delta_{\theta_l}, t_{\theta_l})$:

$$\theta_l(x_l, \delta_{l_k}, t_{l_k}) \Rightarrow write(O(t_{l_k}) \equiv (\delta_{l_k}, t_{l_k})) \tag{8.44}$$

According to the predicate 8.43, the assignation 8.44 means that the Unary Observer $\Theta_l(x_l, \Delta^1_l)$ writes a timed observation $O(t_{l_k}) \equiv (\delta_{l_k}, t_{l_k})$ each time the value of the function $x_l(t)$, for the transaction type id $l$, changes.

Let us denote $O_{l_k} = \{(x_l, \delta_{l_k})\}$, the observation class linking variable name $x_l$ to constant $\delta_{l_k}$. Let us denote $\omega^1_l(t_{n_l}) = \{O(t_{l_k}) \equiv (\delta_{l_k}, t_{l_k}), \delta_{l_k} \in \Delta^1_l, t_{l_k} \in \Gamma_l\}$, the sequence of $n_l \in \mathbb{N}$ timed observations $O(t_{l_k})$. Such a sequence defines the stochastic clock $\Gamma_l = \{t_{l_k}, i_k \in [1; n_l]\}$ of timestamps $t_{l_k}$.

Let us denote $\Omega^1_{TYP} = \bigcup_{l \in \Delta_{ID\_TYP\_EVT}} \omega^1_l(t_{n_l})$, the superposition of the 40 sequences of timed observations $\omega^1_l(t_{n_l})$.

| **Sequence** | $n_l$ | **Sequence** | $n_l$ | **Sequence** | $n_l$ | **Sequence** | $n_l$ |
|---|---|---|---|---|---|---|---|
| $\omega^1_{3001}(t_{n_{3001}})$ | 3 | $\omega^1_{3011}(t_{n_{3011}})$ | 1 | $\omega^1_{3021}(t_{n_{3021}})$ | 0 | $\omega^1_{3031}(t_{n_{3031}})$ | 57 |
| $\omega^1_{3002}(t_{n_{3002}})$ | 3 | $\omega^1_{3012}(t_{n_{3012}})$ | 0 | $\omega^1_{3022}(t_{n_{3022}})$ | 9 | $\omega^1_{3032}(t_{n_{3032}})$ | 2 |
| $\omega^1_{3003}(t_{n_{3003}})$ | 20 | $\omega^1_{3013}(t_{n_{3013}})$ | 1 | $\omega^1_{3023}(t_{n_{3023}})$ | 24 | $\omega^1_{3033}(t_{n_{3033}})$ | 1 |
| $\omega^1_{3004}(t_{n_{3004}})$ | 146 | $\omega^1_{3014}(t_{n_{3014}})$ | 0 | $\omega^1_{3024}(t_{n_{3024}})$ | 238 | $\omega^1_{3034}(t_{n_{3034}})$ | 37 |
| $\omega^1_{3005}(t_{n_{3005}})$ | 1 | $\omega^1_{3015}(t_{n_{3015}})$ | 0 | $\omega^1_{3025}(t_{n_{3025}})$ | 5 | $\omega^1_{3035}(t_{n_{3035}})$ | 27 |
| $\omega^1_{3006}(t_{n_{3006}})$ | 65 | $\omega^1_{3016}(t_{n_{3016}})$ | 0 | $\omega^1_{3026}(t_{n_{3026}})$ | 17 | $\omega^1_{3036}(t_{n_{3036}})$ | 23 |
| $\omega^1_{3007}(t_{n_{3007}})$ | 2 | $\omega^1_{3017}(t_{n_{3017}})$ | 66 | $\omega^1_{3027}(t_{n_{3027}})$ | 3 | $\omega^1_{3037}(t_{n_{3037}})$ | 10 |
| $\omega^1_{3008}(t_{n_{3008}})$ | 3 | $\omega^1_{3018}(t_{n_{3018}})$ | 21 | $\omega^1_{3028}(t_{n_{3028}})$ | 3 | $\omega^1_{3038}(t_{n_{3038}})$ | 33 |
| $\omega^1_{3009}(t_{n_{3009}})$ | 1 | $\omega^1_{3019}(t_{n_{3019}})$ | 422 | $\omega^1_{3029}(t_{n_{3029}})$ | 3 | $\omega^1_{3039}(t_{n_{3039}})$ | 63 |
| $\omega^1_{3010}(t_{n_{3010}})$ | 1 | $\omega^1_{3020}(t_{n_{3020}})$ | 7 | $\omega^1_{3030}(t_{n_{3030}})$ | 59 | $\omega^1_{3040}(t_{n_{3040}})$ | 16 |
| | | | | | | **Total** | **1409** |

Table 8.3: Number $n_l$ of timed observations for sequences $\omega^1_l(t_{n_l})$, $l \in \Delta_{ID\_TYP\_EVT}$

Table 8.3 sums up the number of timed observations for each transaction type id. We can notice here that most of the transactions are of types 3004, 3019 and 3024.

As seen in chapter 7, we can build the $TOT(\Delta^1_{TYP})$ Category where:

- objects are Unary Observers $\Theta_l(x_l, \Delta^1_l)$, $l \in \Delta_{ID\_TYP\_EVT}$;
- morphisms are relations $f_{ij} : \Theta_i \to \Theta_l$, $(i,j) \in \Delta_{ID\_TYP\_EVT} \times \Delta_{ID\_TYP\_EVT}$.

According to definition 7.3, let us consider the level of abstraction of the $TOT(\Delta^1_{TYP})$ Category, denoted $\mathcal{L}^1_{TYP} = \langle \mathcal{S}^1_{TYP}, \mathcal{T}^1_{TYP} \rangle$, where:

- $\mathcal{S}^1_{TYP} = (\Omega^1_{TYP}, \overset{\Delta\tau_{ij}}{+})$ is the algebraic structure of the observed process $(X_{TYP}(t), \Theta_{TYP}(X_{TYP}, \Delta^1_{TYP}))$;
- $\mathcal{T}^1_{TYP} = (\Omega^1_{TYP}, \mathcal{I}^1_{TYP})$ is the observable space of the observed process $(X_{TYP}(t), \Theta_{TYP}(X_{TYP}, \Delta^1_{TYP}))$.

The algebraic structure $\mathcal{S}^1_{TYP}$ provides a particular description of the transaction database according to the point of view of transaction type identification numbers. From observable space $\mathcal{T}^1_{TYP}$, we can build the abstract chronicle model $\mathcal{M}^1_{TYP}$ containing all temporal binary relations $r_{i_k j_{k+1}}(O_{i_k}, O_{j_{k+1}}, \Delta\tau_{i_k j_{k+1}})$ between observation classes $O_{i_k}$ and $O_{j_{k+1}}$ :

$$\mathcal{M}^1_{TYP} = \{r_{i_k j_{k+1}}(O_{i_k}, O_{j_{k+1}}, \Delta\tau_{i_k j_{k+1}}), (i,j) \in \Delta_{ID\_TYP\_EVT} \times \Delta_{ID\_TYP\_EVT}\} \quad (8.45)$$

The abstract chronicle model $\mathcal{M}^1_{TYP}$ represents all binary sequences of successive transactions from a transaction type $i \in \Delta_{ID\_TYP\_EVT}$ to a transaction type $j \in \Delta_{ID\_TYP\_EVT}$ under the temporal constraint $\Delta\tau_{i_k j_{k+1}}$ that may be observable in the transaction database. The size of such an abstract chronicle model $\mathcal{M}^1_{CPTE}$ is, according to property 20:

$$Card(\mathcal{M}^1_{TYP}) = \sum_{\substack{(i,j)\in\Delta^2_{ID\_TYP\_EVT} \\ i\neq j}} (N_i + N_j)^2 \tag{8.46}$$

In this case, equation 8.35 leads to:

$$Card(\mathcal{M}^1_{TYP}) = 12\ 630\ 799 \tag{8.47}$$

Thus, the abstract chronicle model $\mathcal{M}^1_{TYP}$ contains 12 630 799 temporal binary relations modeling all the successive transactions between a transaction type $i \in \Delta_{ID\_TYP\_EVT}$ to a transaction type $j \in \Delta_{ID\_TYP\_EVT}$. So, when considering the category of the transaction type ids, there exists 12 630 799 binary sequences of transactions to analyse in order to find the ones which are potentially fraudulent which is equivalent of the size of accounts abstract chronicle model.

These three categories represent then three ways to describe and to model the observed problem. In other words, the observed problem can be seen under three different points of view or three levels of abstraction. Each level of abstraction has its own syntax and semantic. As seen in section 7.3.1, Modeling functors are tools to change the point of view of a given observed problem. Let us now see how to implement such Modeling functors in the current case of study.

We have seen in section 8.5.2 that there exists a relation $\mathcal{R}_{ID\_CLI\to ID\_CPTE}$ mapping client ids of the set $\Delta_{ID\_CLI}$ and banking account ids of the set $\Delta_{ID\_CPTE}$:

$$\begin{array}{rccc} \mathcal{R}_{ID\_CLI\to ID\_CPTE} & : & \Delta_{ID\_CLI} & \to & \Delta_{ID\_CPTE} \\ & & i & \mapsto & j \end{array} \tag{8.48}$$

Such a relation implies the existence of another relation denoted $\Delta^1_{CLI} \to \Delta^1_{CPTE}$ between sets of constants $\Delta^1_{CLI}$ and $\Delta^1_{CPTE}$:

$$i\ \mathcal{R}_{ID\_CLI\to ID\_CPTE}\ j \Rightarrow \exists \Delta^1_{CLI} \to \Delta^1_{CPTE}, \delta_{i_k}\ (\Delta^1_{CLI} \to \Delta^1_{CPTE})\ \delta_{j_k} \tag{8.49}$$

And:

$$\delta_{i_k}\ (\Delta^1_{CLI} \to \Delta^1_{CPTE})\ \delta_{j_k} \Rightarrow (\delta_{i_k}, t_{i_k}) f_{ij}(\delta_{j_k}, t_{j_k}) \tag{8.50}$$

So, from relation $\Delta^1_{CLI} \to \Delta^1_{CPTE}$, we know all the morphisms $f_{ij}$ linking Unary Observers $\Theta_i(x_i, \Delta^1_i)$ of the $TOT(\Delta^1_{CLI})$ Category with Unary Observers $\Theta_j(x_j, \Delta^1_j)$ of the $TOT(\Delta^1_{CPTE})$ Category.

For example, let us focus on client 1002 owning accounts 2005, 2011, 2019 and 2023:

$$\begin{array}{rccc} \mathcal{R}^{1002}_{ID\_CLI\to ID\_CPTE} & : & \Delta_{ID\_CLI} & \to & \Delta_{ID\_CPTE} \\ & & 1002 & \mapsto & 2005 \\ & & 1002 & \mapsto & 2011 \\ & & 1002 & \mapsto & 2019 \\ & & 1002 & \mapsto & 2023 \end{array} \tag{8.51}$$

Relation 8.51 implies the existence of the relation $\Delta^1_{CLI} \to \Delta^1_{CPTE}$ such as:

$$\begin{array}{rcccc} \Delta^1_{CLI} \rightarrow \Delta^1_{CPTE} & : & \Delta_{ID\_CLI} & \rightarrow & \Delta_{ID\_CPTE} \\ & & \delta_{1002_k} & \mapsto & \delta_{2005_k} \\ & & \delta_{1002_k} & \mapsto & \delta_{2011_k} \\ & & \delta_{1002_k} & \mapsto & \delta_{2019_k} \\ & & \delta_{1002_k} & \mapsto & \delta_{2023_k} \end{array} \tag{8.52}$$

The concerned morphisms are then:

$$\begin{array}{rcccc} f_{1002,2005} & : & \Theta_{1002} & \rightarrow & \Theta_{2005} \\ & & (\delta_{1002_k}, t_{1002_k}) & \mapsto & (\delta_{2005_k}, t_{2005_k}) \\ f_{1002,2011} & : & \Theta_{1002} & \rightarrow & \Theta_{2011} \\ & & (\delta_{1002_k}, t_{1002_k}) & \mapsto & (\delta_{2011_k}, t_{2011_k}) \\ f_{1002,2019} & : & \Theta_{1002} & \rightarrow & \Theta_{2019} \\ & & (\delta_{1002_k}, t_{1002_k}) & \mapsto & (\delta_{2019_k}, t_{2019_k}) \\ f_{1002,2023} & : & \Theta_{1002} & \rightarrow & \Theta_{2023} \\ & & (\delta_{1002_k}, t_{1002_k}) & \mapsto & (\delta_{2023_k}, t_{2023_k}) \end{array} \tag{8.53}$$

Morphism $f_{1002,2005}$ maps timed observations of the sequence $\omega^1_{1002}(t_{n_{1002}})$ with timed observations of the sequence $\omega^1_{2005}(t_{n_{2005}})$. Morphism $f_{1002,2011}$ maps timed observations of the sequence $\omega^1_{1002}(t_{n_{1002}})$ with timed observations of the sequence $\omega^1_{2011}(t_{n_{2011}})$. Morphism $f_{1002,2019}$ maps timed observations of the sequence $\omega^1_{1002}(t_{n_{1002}})$ with timed observations of the sequence $\omega^1_{2019}(t_{n_{2019}})$. Morphism $f_{1002,2023}$ maps timed observations of the sequence $\omega^1_{1002}(t_{n_{1002}})$ with timed observations of the sequence $\omega^1_{2023}(t_{n_{2023}})$. In this particular case, the Modeling functor $TOT(\Delta^1_{CLI}) \rightarrow TOT(\Delta^1_{CPTE})$ would implement all the morphisms $f_{1002,2005}$, $f_{1002,2011}$, $f_{1002,2019}$ and $f_{1002,2023}$.

In a more general way, the Modeling functor $TOT(\Delta^1_{CLI}) \rightarrow TOT(\Delta^1_{CPTE})$ can be built according to the relation $\Delta^1_{CLI} \rightarrow \Delta^1_{CPTE}$. Such a building is consistent with the definition 7.2 of a Modeling functor. Such a Modeling functor implements then all concerned morphisms $f_{ij} : \Theta_i \rightarrow \Theta_j$ where $\Theta_i$ is an object of the $TOT(\Delta^1_{CLI})$ Category and $\Theta_j$ is an object of the $TOT(\Delta^1_{CPTE})$ Category. Morphism $f_{ij}$ maps timed observations $(\delta_{i_k}, t_{i_k})$, representing a transaction for a client $i$, with a timed observation $(\delta_{j_k}, t_{j_k})$, representing a transaction on a banking account $j$.

This Modeling functor aims at changing the way to model the observed problem: from the model of client to the model of accounts.

Program TOM4FFS, given the relation $\mathcal{R}_{ID\_CLI \rightarrow ID\_CPTE}$, builds and implements the Modeling functor $TOT(\Delta^1_{CLI}) \rightarrow TOT(\Delta^1_{CPTE})$. It does the same to build the other Modeling functors of the syntactic model of the first level of abstraction:

- from relation $\mathcal{R}_{ID\_CPTE \rightarrow ID\_CLI}$, it builds relation $\Delta^1_{CPTE} \rightarrow \Delta^1_{CLI}$ then Modeling functor $TOT(\Delta^1_{CPTE}) \rightarrow TOT(\Delta^1_{CLI})$;
- from relation $\mathcal{R}_{ID\_CPTE \rightarrow ID\_TYP\_EVT}$, it builds relation $\Delta^1_{CPTE} \rightarrow \Delta^1_{TYP}$ then Modeling functor $TOT(\Delta^1_{CPTE}) \rightarrow TOT(\Delta^1_{TYP})$;
- from relation $\mathcal{R}_{ID\_TYP\_EVT \rightarrow ID\_CPTE}$, it builds relation $\Delta^1_{TYP} \rightarrow \Delta^1_{CPTE}$ then Modeling functor $TOT(\Delta^1_{TYP}) \rightarrow TOT(\Delta^1_{CPTE})$;

- from relation $\mathcal{R}_{ID\_TYP\_EVT \rightarrow ID\_CLI}$, it builds relation $\Delta^1_{TYP} \rightarrow \Delta^1_{CLI}$ then Modeling functor $TOT(\Delta^1_{TYP}) \rightarrow TOT(\Delta^1_{CLI})$;
- from relation $\mathcal{R}_{ID\_CLI \rightarrow ID\_TYP\_EVT}$, it builds relation $\Delta^1_{CLI} \rightarrow \Delta^1_{TYP}$ then Modeling functor $TOT(\Delta^1_{CLI}) \rightarrow TOT(\Delta^1_{TYP})$;

Such a LoA, denoted $\mathcal{L}^1$, is composed of:

- three TOT Categories providing different ways to model the observed problem;
- six Modeling functors to change the way to model the observed problem.

### 8.6.3 Syntactic Model of the Observed Problem at the Second LoA

This step consists of a spacial reduction of the observed process using the property 25 introduced in chapter 7. Let us recall that the structure of a constant $\delta_{i_k}$ contains the attribute $p_5(k)$ corresponding to the amount of the $k^{th}$ transaction of the database. This amount in an integer of $\mathbb{Z}$ representing cents of €. Thus, the number of observation classes $O_{i_k} = (x_i, \delta_{i_k})$ is, a priori, infinite. In order to make this number finite, the idea is to define an *abstract representation* of the transaction amounts, inspired from the Benford's Law [Ben38], also called the *First-Digit Law*, to define a *compact* set of constants.

An amount $z = p_5(k) = x_i(t_{k_i})$ of a transaction is a signed sum of powers of 10 (see 8.54) where (i) $s(z)$ is the sign function of $z$ (i.e. $s : \mathbb{Z} \rightarrow \{-1, 1\}, z \mapsto -1$ if $z < 0$, 1 otherwise), (ii) $n$ is the highest power of 10 of $z$ ($n \geq 0$) and (iii) $a_j \in D = \{1, ..., 9\}$ is the digit defining the value of the coefficient of the $j^{th}$ power of $z$:

$$\forall z \in \mathbb{Z}, z = s(z) \cdot \sum_{j=0}^{n} a_j \cdot 10^j. \tag{8.54}$$

Let us then define the following abstraction function $\mu$, called *classification function*:

$$\begin{array}{rcccc} \mu & : & \mathbb{Z} & \rightarrow & \mathbb{M} \\ & & z & \mapsto & \mu(z) = s(z).(10.n + a_n) \end{array} \tag{8.55}$$

The classification function $\mu(z)$ associates any element $z$ of $\mathbb{Z}$ with an element of the subset $M$ of $\mathbb{Z}$: $\mathbb{M}$ = {..., -21, -19, ..., -11, -9, -8, ..., -2, -1, 1, 2, ..., 8, 9, 11, ..., 19, 21, ... }. Each element $\mu(z)$ of $\mathbb{M}$ is made with the sign of $z$ followed by two digits, the first representing the *highest power* of 10 of $z$ and the second representing its *first digit*. For example, $z$ = - 7 451 214 is mapped to $\mu(z)$ = - 67. Clearly, this classification function is a surjective application. Classification function $\mu$ provides a mean to define a finite set $O$ of observation classes $O_i = (\phi_i, \delta_i)$: each constant $\delta_i$ is an element of a finite subset $\Delta^2$ of $\mathbb{M}$ (the abstract variable $\phi_i$ has no concrete meaning).

Let us consider the dynamic process $\Phi_{CLI}(t) = \{\phi_i(t), i \in \Delta_{ID\_CLI}\}$ composed of four undefined abstract functions $\phi_i(t), i \in \Delta_{ID\_CLI}$. Such a dynamic process defines the set $\Phi_{CLI} = \{\phi_i, i \in \Delta_{ID\_CLI}\}$ of four variable names $\phi_i, i \in \Delta_{ID\_CLI}$. Let us consider the independent program of observation $\Theta^2_{CLI}(\Phi_{CLI}, \Delta^2_{CLI})$ observing the dynamic process $\Phi_{CLI}(t)$.

Let us consider the observed process $(\Phi_{CLI}(t), \Theta^2_{CLI}(\Phi_{CLI}, \Delta^2_{CLI}))$. According to the Superposition Theorem 3.2, the program $\Theta^2_{CLI}(\Phi_{CLI}, \Delta^2_{CLI})$ can be decomposed in a superposition of four independent Unary Observers $\Theta^2_i(\phi_i, \Delta^2_i), i \in \Delta_{ID\_CLI}$. Any Unary Observer $\Theta^2_i(\phi_i, \Delta^2_i)$ implements a predicate denoted $\theta^2_i(x_{\theta^2_i}, \delta_{\theta^2_i}, t_{\theta^2_i})$ based on the application of the Modus Ponens with the rule 7.8 in the particular case described in property 25:

$$\theta_i(x_i, \delta_{i_k}, t_{i_k}) \Rightarrow \exists m_k \in \Delta^2_i, \theta^2_i(\phi_i, m_k, t_{i_k}) \tag{8.56}$$

Unary Observer $\Theta^2_i(\phi_i, \Delta^2_i)$ writes then a timed observation denoted $O^2_{m_k}(t_{i_k}) \equiv (m_k, t_{i_k})$ for each assignation $\theta^2_i(\phi_i, m_k, t_{i_k})$ of the predicate $\theta^2_i(x_{\theta^2_i}, \delta_{\theta^2_i}, t_{\theta^2_i})$:

$$\theta^2_i(\phi_i, m_k, t_{i_k}) \Rightarrow write(O^2_{m_k}(t_{i_k}) \equiv (m_k, t_{i_k})) \tag{8.57}$$

Since predicate $\theta_i(x_{\theta_i}, \delta_{\theta_i}, t_{\theta_i})$ implements the equation 8.21, implications 8.56 and 8.57 are equivalent to:

$$\begin{aligned} p_1(i_k) = i \wedge t_{i_{k-1}} < t_{i_k} \wedge x_i(t_{i_{k-1}}) \neq x_i(t_{i_k}) \\ \Rightarrow \exists m_k \in \Delta^2_i, \theta^2_i(\phi_i, m_k, t_{i_k}) \\ \Rightarrow write(O^2_{m_k}(t_{i_k}) \equiv (m_k, t_{i_k})) \end{aligned} \tag{8.58}$$

Let us now impose that $m_k = \mu(x_i(t_{i_k}))$, where $\mu$ is the classification function of 8.55, equation 8.58 leads to:

$$\begin{aligned} p_1(i_k) = i \wedge t_{i_{k-1}} < t_{i_k} \wedge x_i(t_{i_{k-1}}) \neq x_i(t_{i_k}) \\ \Rightarrow \theta^2_i(\phi_i, \mu(x_i(t_{i_k})), t_{i_k}) \\ \Rightarrow write(O^2_{m_k}(t_{i_k}) \equiv (m_k, t_{i_k})) \end{aligned} \tag{8.59}$$

This means that Unary Observer $\Theta^2_i(\phi_i, \Delta^2_i)$ writes a timed observation $O^2_{m_k}(t_{i_k}) \equiv (m_k, t_{i_k})$ where $m_k = \mu(x_i(t_{i_k}))$ each time the value of the function $x_i(t)$, for the client id $i$, changes.

Let us denote $O^2_{m_k} = \{(\phi_i, m_k)\}$, the observation class linking abstract variable name $\phi_i$ to constant $m_k = \mu(x_i(t_{i_k}))$.

Let us denote $\omega^2_i(t_{n^{(2)}_i}) = \{O^2_{m_k}(t_{i_k}) \equiv (m_k, t_{i_k}), m_k \in \Delta^2_i, t_{i_k} \in \Gamma_i, i \in \Delta_{ID\_CLI}\}$, the sequences of $n^{(2)}_i \in \mathbb{N}$ timed observations $O^2_{m_k}(t_{i_k})$.

Let us denote $\Omega^2_{CLI} = \bigcup_{i \in \Delta_{ID\_CLI}} \omega^2_i$, the superposition of the four sequences of timed observations $\omega^2_i$.

Let us build the $TOT(\Delta^2_{CLI})$ Category where:

- objects are Unary Observers $\Theta^2_i(\phi_i, \Delta^2_i)$, $i \in \Delta_{ID\_CLI}$;
- morphisms are relations $f^2_{ij} : \Theta^2_i \rightarrow \Theta^2_j$, $(i, j) \in \Delta_{ID\_CLI} \times \Delta_{ID\_CLI}$.

Let us consider the level of abstraction of the $TOT(\Delta^2_{CLI})$ Category, denoted $\mathcal{L}^2_{CLI} = \langle \mathcal{S}^2_{CLI}, \mathcal{T}^2_{CLI} \rangle$, where:

- $\mathcal{S}^2_{CLI} = (\Omega^2_{CLI}, \overset{\Delta\tau_{ij}}{+})$ is the algebraic structure of the observed process $(\Phi_{CLI}(t), \Theta^2_{CLI}(\Phi_{CLI}, \Delta^2_{CLI}))$;

- $\mathcal{T}^2_{CLI} = (\Omega^2_{CLI}, \mathcal{I}^2_{CLI})$ is the observable space of the observed process $((\Phi_{CLI}(t), \Theta^2_{CLI}(\Phi_{CLI}, \Delta^2_{CLI}))$.

From observable space $\mathcal{T}^2_{CLI}$, we can build the abstract chronicle model $\mathcal{M}^2_{CLI}$ containing all temporal binary relations $r^2_{m_k m_{k+1}}(O^2_{m_k}, O^2_{m_{k+1}}, \Delta\tau_{m_k m_{k+1}})$ between observation classes $O^2_{m_k}$ and $O^2_{m_{k+1}}$:

$$\mathcal{M}^2_{CLI} = \{r^2_{m_k m_{k+1}}(O^2_{m_k}, O^2_{m_{k+1}}, \Delta\tau_{i_k j_{k+1}}), (i,j) \in \Delta_{ID\_CLI} \times \Delta_{ID\_CLI}\} \tag{8.60}$$

The abstract chronicle model $\mathcal{M}^2_{CLI}$ contains all binary sequences of successive transactions from a client $i \in \Delta_{ID\_CLI}$ to a client $j \in \Delta_{ID\_CLI}$, under the temporal constraint $\Delta\tau_{m_k m_{k+1}}$, that may be observable in the transaction database, but represented under their amounts compact form.

| **Sequence** | $N_i^{(2)}$ |
|---|---|
| $\omega^2_{1001}(t_{n^{(2)}_{1001}})$ | 51 |
| $\omega^2_{1002}(t_{n^{(2)}_{1002}})$ | 43 |
| $\omega^2_{1003}(t_{n^{(2)}_{1003}})$ | 61 |
| $\omega^2_{1004}(t_{n^{(2)}_{1004}})$ | 60 |

Table 8.4: Number $N_i^{(2)}$ of observation classes associated with sequences $\omega^2_i(t_{n^{(2)}_i})$, $i \in \Delta_{ID\_CLI}$

Table 8.4 sums up the number of observation classes associated with sequences $\omega^2_i(t_{n^{(2)}_i})$ for each client id $i \in \Delta_{ID\_CLI}$. This table allows to compute the size of the abstract chronicle model $\mathcal{M}^2_{CLI}$ according to property 20:

$$\begin{aligned}
Card(\mathcal{M}^2_{CLI}) = \; & (N^{(2)}_{1001} + N^{(2)}_{1002})^2 + (N^{(2)}_{1001} + N^{(2)}_{1003})^2 + (N^{(2)}_{1001} + N^{(2)}_{1004})^2 + \\
& (N^{(2)}_{1002} + N^{(2)}_{1003})^2 + (N^{(2)}_{1002} + N^{(2)}_{1004})^2 + \\
& (N^{(2)}_{1003} + N^{(2)}_{1004})^2 \\
Card(\mathcal{M}^2_{CLI}) = \; & (51 + 43)^2 + (51 + 61)^2 + (51 + 60)^2 + \\
& (43 + 61)^2 + (43 + 60)^2 + \\
& (61 + 60)^2 \\
Card(\mathcal{M}^2_{CLI}) = \; & 69\ 767
\end{aligned} \tag{8.61}$$

At this level of abstraction $\mathcal{L}^2_{CLI}$, the abstract chronicle model $\mathcal{M}^2_{CLI}$ contains 69 767 temporal binary relations modeling all the successive transactions between a client $i \in \Delta_{ID\_CLI}$ to a client $j \in \Delta_{ID\_CLI}$. Compared to the 3 379 879 temporal binary relations contained in the abstract chronicle model $\mathcal{M}^1_{CLI}$ at the LoA $\mathcal{L}^1_{CLI}$, the number of temporal binary relations has been reduced to $97,94\%$.

According to definition 7.4 of chapter 7, an Abstraction functor, denoted $TOT(\Delta^1_{CLI}) \rightarrow TOT(\Delta^2_{CLI})$, can be built according to the surjective classification function $\mu$, linking $TOT(\Delta^1_{CLI})$ Category to $TOT(\Delta^2_{CLI})$ Category. Such an Abstraction functor maps timed observations $O_{i_k}(t_{i_k})$ of $TOT(\Delta^1_{CLI})$ Category to its more abstract form $O^2_{m_k}(t_{i_k})$ of $TOT(\Delta^2_{CLI})$ Category:

$$\begin{array}{rcccc} TOT(\Delta^1_{CLI}) \rightarrow TOT(\Delta^2_{CLI}) & : & TOT(\Delta^1_{CLI}) & \rightarrow & TOT(\Delta^2_{CLI}) \\ & & O_{i_k}(t_{i_k}) \equiv (\delta_{i_k}, t_{i_k}) & \mapsto & O^2_{m_k}(t_{i_k}) \equiv (m_k, t_{i_k}) \end{array} \tag{8.62}$$

Let us consider the dynamic process $\Phi_{CPTE}(t) = \{\phi_j(t), j \in \Delta_{ID\_CPTE}\}$ composed of 30 undefined abstract functions $\phi_j(t), j \in \Delta_{ID\_CPTE}$. Such a dynamic process defines the set $\Phi_{CPTE} = \{\phi_j, j \in \Delta_{ID\_CPTE}\}$ of 30 variable names $\phi_j, j \in \Delta_{ID\_CPTE}$. Let us consider the independent program of observation $\Theta^2_{CPTE}(\Phi_{CPTE}, \Delta^2_{CPTE})$ observing the dynamic process $\Phi_{CPTE}(t)$. Let us consider the observed process $(\Phi_{CPTE}(t), \Theta^2_{CPTE}(\Phi_{CPTE}, \Delta^2_{CPTE}))$.

With the same reasoning, we can build the superposition $\Omega^2_{CPTE} = \bigcup_{j \in \Delta_{ID\_CPTE}} \omega^2_j$ of the 30 sequences of timed observations $\omega^2_j$. A sequence $\omega^2_j(t_{n^{(2)}_j}) = \{O^2_{m_k}(t_{j_k}) \equiv (m_k, t_{j_k}), m_k \in \Delta^2_j, t_{j_k} \in \Gamma_j, j \in \Delta_{ID\_CPTE}\}$ contains $n^{(2)}_j \in \mathbb{N}$ timed observations of the form $O^2_{m_k}(t_{j_k}) \equiv (m_k, t_{j_k})$ where $m_k = \mu(x_j(t_{j_k}))$.

Let us build the $TOT(\Delta^2_{CPTE})$ Category where:

- objects are Unary Observers $\Theta^2_j(\phi_j, \Delta^2_j)$, $j \in \Delta_{ID\_CPTE}$;
- morphisms are relations $f^2_{ij} : \Theta^2_i \rightarrow \Theta^2_j$, $(i, j) \in \Delta_{ID\_CPTE} \times \Delta_{ID\_CPTE}$.

Let us consider the level of abstraction of the $TOT(\Delta^2_{CPTE})$ Category, denoted $\mathcal{L}^2_{CPTE} = \langle \mathcal{S}^2_{CPTE}, \mathcal{T}^2_{CPTE} \rangle$, where:

- $\mathcal{S}^2_{CPTE} = (\Omega^2_{CPTE}, \overset{\Delta\tau_{ij}}{+})$ is the algebraic structure of the observed process $(\Phi_{CPTE}(t), \Theta^2_{CPTE}(\Phi_{CPTE}, \Delta^2_{CPTE}))$;
- $\mathcal{T}^2_{CPTE} = (\Omega^2_{CPTE}, \mathcal{I}^2_{CPTE})$ is the observable space of the observed process $((\Phi_{CPTE}(t), \Theta^2_{CPTE}(\Phi_{CPTE}, \Delta^2_{CPTE}))$.

From observable space $\mathcal{T}^2_{CPTE}$, we can build the abstract chronicle model $\mathcal{M}^2_{CPTE}$ containing all temporal binary relations $r^2_{m_k m_{k+1}}(O^2_{m_k}, O^2_{m_{k+1}}, \Delta\tau_{m_k m_{k+1}})$ between observation classes $O^2_{m_k}$ and $O^2_{m_{k+1}}$:

$$\mathcal{M}^2_{CPTE} = \{r^2_{m_k m_{k+1}}(O^2_{m_k}, O^2_{m_{k+1}}, \Delta\tau_{i_k j_{k+1}}), (i, j) \in \Delta_{ID\_CPTE} \times \Delta_{ID\_CPTE}\} \tag{8.63}$$

The abstract chronicle model $\mathcal{M}^2_{CPTE}$ contains all binary sequences of successive transactions from account $i \in \Delta_{ID\_CPTE}$ to account $j \in \Delta_{ID\_CPTE}$, under the temporal constraint $\Delta\tau_{m_k m_{k+1}}$, that may be observable in the transaction database, but represented under their amounts compact form.

The size of such an abstract chronicle model $\mathcal{M}^2_{CPTE}$ is then:

$$Card(\mathcal{M}^2_{CPTE}) = \sum_{\substack{(i,j) \in \Delta^2_{ID\_CPTE} \\ i \neq j}} (N_i + N_j)^2 = 404\ 797 \tag{8.64}$$

| **Sequence** | $N_j^{(2)}$ | **Sequence** | $N_j^{(2)}$ | **Sequence** | $N_j^{(2)}$ |
|---|---|---|---|---|---|
| $\omega^2_{2001}(t_{n^{(2)}_{2001}})$ | 47 | $\omega^2_{2011}(t_{n^{(2)}_{2011}})$ | 2 | $\omega^2_{2021}(t_{n^{(2)}_{2021}})$ | 1 |
| $\omega^2_{2002}(t_{n^{(2)}_{2002}})$ | 2 | $\omega^2_{2012}(t_{n^{(2)}_{2012}})$ | 2 | $\omega^2_{2022}(t_{n^{(2)}_{2022}})$ | 2 |
| $\omega^2_{2003}(t_{n^{(2)}_{2003}})$ | 2 | $\omega^2_{2013}(t_{n^{(2)}_{2013}})$ | 1 | $\omega^2_{2023}(t_{n^{(2)}_{2023}})$ | 1 |
| $\omega^2_{2004}(t_{n^{(2)}_{2004}})$ | 2 | $\omega^2_{2014}(t_{n^{(2)}_{2014}})$ | 2 | $\omega^2_{2024}(t_{n^{(2)}_{2024}})$ | 46 |
| $\omega^2_{2005}(t_{n^{(2)}_{2005}})$ | 40 | $\omega^2_{2015}(t_{n^{(2)}_{2015}})$ | 1 | $\omega^2_{2026}(t_{n^{(2)}_{2026}})$ | 2 |
| $\omega^2_{2006}(t_{n^{(2)}_{2006}})$ | 9 | $\omega^2_{2016}(t_{n^{(2)}_{2016}})$ | 2 | $\omega^2_{2027}(t_{n^{(2)}_{2027}})$ | 2 |
| $\omega^2_{2007}(t_{n^{(2)}_{2007}})$ | 57 | $\omega^2_{2017}(t_{n^{(2)}_{2017}})$ | 2 | $\omega^2_{2028}(t_{n^{(2)}_{2028}})$ | 2 |
| $\omega^2_{2008}(t_{n^{(2)}_{2008}})$ | 45 | $\omega^2_{2018}(t_{n^{(2)}_{2018}})$ | 2 | $\omega^2_{2029}(t_{n^{(2)}_{2029}})$ | 2 |
| $\omega^2_{2009}(t_{n^{(2)}_{2009}})$ | 2 | $\omega^2_{2019}(t_{n^{(2)}_{2019}})$ | 4 | $\omega^2_{2030}(t_{n^{(2)}_{2030}})$ | 2 |
| $\omega^2_{2010}(t_{n^{(2)}_{2010}})$ | 3 | $\omega^2_{2020}(t_{n^{(2)}_{2020}})$ | 3 | $\omega^2_{2031}(t_{n^{(2)}_{2031}})$ | 3 |

Table 8.5: Number $N_j^{(2)}$ of observation classes for sequences $\omega_j^2(t_{n_j^{(2)}})$, $j \in \Delta_{ID\_CPTE}$

At this level of abstraction $\mathcal{L}^2_{CPTE}$, the abstract chronicle model $\mathcal{M}^2_{CPTE}$ contains 404 797 temporal binary relations modeling all the successive transactions between an account $i \in \Delta_{ID\_CPTE}$ to an account $j \in \Delta_{ID\_CPTE}$. Compared to the 14 635 861 temporal binary relations contained in the abstract chronicle model $\mathcal{M}^1_{CPTE}$ at the LoA $\mathcal{L}^1_{CPTE}$, the number of temporal binary relations has been reduced to $97,23\%$.

Again, an Abstraction functor, denoted $TOT(\Delta^1_{CPTE}) \rightarrow TOT(\Delta^2_{CPTE})$, can be built according to the surjective classification function $\mu$, linking $TOT(\Delta^1_{CPTE})$ Category to $TOT(\Delta^2_{CPTE})$ Category. Such an Abstraction functor maps timed observations $O_{j_k}(t_{j_k})$ of $TOT(\Delta^1_{CPTE})$ Category to its more abstract form $O^2_{m_k}(t_{j_k})$ of $TOT(\Delta^2_{CPTE})$ Category:

$$\begin{array}{rccc} TOT(\Delta^1_{CPTE}) \rightarrow TOT(\Delta^2_{CPTE}) \ : & TOT(\Delta^1_{CPTE}) & \rightarrow & TOT(\Delta^2_{CPTE}) \\ & O_{j_k}(t_{j_k}) \equiv (\delta_{j_k}, t_{j_k}) & \mapsto & O^2_{m_k}(t_{j_k}) \equiv (m_k, t_{j_k}) \end{array} \tag{8.65}$$

Let us consider the dynamic process $\Phi_{TYP}(t) = \{\phi_l(t), l \in \Delta_{ID\_TYP\_EVT}\}$ composed of 40 undefined abstract functions $\phi_l(t), l \in \Delta_{ID\_TYP\_EVT}$. Such a dynamic process defines the set $\Phi_{TYP} = \{\phi_l, l \in \Delta_{ID\_TYP\_EVT}\}$ of 40 variable names $\phi_l, l \in \Delta_{ID\_TYP\_EVT}$. Let us consider the independent program of observation $\Theta^2_{TYP}(\Phi_{TYP}, \Delta^2_{TYP})$ observing the dynamic process $\Phi_{TYP}(t)$. Let us consider the observed process $(\Phi_{TYP}(t), \Theta^2_{TYP}(\Phi_{TYP}, \Delta^2_{TYP}))$.

With the same reasoning, we can build the superposition $\Omega^2_{TYP} = \bigcup_{l \in \Delta_{ID\_TYP\_EVT}} \omega^2_l$ of the 40 sequences of timed observations $\omega^2_l$. A sequence $\omega^2_l(t_{n_l^{(2)}}) = \{O^2_{m_k}(t_{l_k}) \equiv (m_k, t_{l_k}), m_k \in \Delta^2_l, t_{l_k} \in \Gamma_l, l \in \Delta_{ID\_TYP\_EVT}\}$ contains $n_l^{(2)} \in \mathbb{N}$ timed observations of the form $O^2_{m_k}(t_{l_k}) \equiv (m_k, t_{l_k})$ where $m_k = \mu(x_l(t_{l_k}))$.

Let us build the $TOT(\Delta^2_{TYP})$ Category where:

- objects are Unary Observers $\Theta^2_l(\phi_l, \Delta^2_l)$, $l \in \Delta_{ID\_TYP\_EVT}$;
- morphisms are relations $f^2_{ij} : \Theta^2_i \rightarrow \Theta^2_l$, $(i,j) \in \Delta_{ID\_TYP\_EVT} \times \Delta_{ID\_TYP\_EVT}$.

Let us consider the level of abstraction of the $TOT(\Delta^2_{TYP})$ Category, denoted $\mathcal{L}^2_{TYP} = \langle \mathcal{S}^2_{TYP}, \mathcal{T}^2_{TYP} \rangle$, where:

- $\mathcal{S}^2_{TYP} = (\Omega^2_{TYP}, \overset{\Delta\tau_{ij}}{+})$ is the algebraic structure of the observed process $(\Phi_{TYP}(t), \Theta^2_{TYP}(\Phi_{TYP}, \Delta^2_{TYP}))$;
- $\mathcal{T}^2_{TYP} = (\Omega^2_{TYP}, \mathcal{I}^2_{TYP})$ is the observable space of the observed process $((\Phi_{TYP}(t), \Theta^2_{TYP}(\Phi_{TYP}, \Delta^2_{TYP}))$.

From observable space $\mathcal{T}^2_{TYP}$, we can build the abstract chronicle model $\mathcal{M}^2_{TYP}$ containing all temporal binary relations $r^2_{m_k m_{k+1}}(O^2_{m_k}, O^2_{m_{k+1}}, \Delta\tau_{m_k m_{k+1}})$ between observation classes $O^2_{m_k}$ and $O^2_{m_{k+1}}$:

$$\mathcal{M}^2_{TYP} = \{r^2_{m_k m_{k+1}}(O^2_{m_k}, O^2_{m_{k+1}}, \Delta\tau_{i_k j_{k+1}}), (i,j) \in \Delta_{ID\_TYP\_EVT} \times \Delta_{ID\_TYP\_EVT}\} \quad (8.66)$$

The abstract chronicle model $\mathcal{M}^2_{TYP}$ contains all binary sequences of successive transactions from transaction type $i \in \Delta_{ID\_TYP\_EVT}$ to transaction type $j \in \Delta_{ID\_TYP\_EVT}$, under the temporal constraint $\Delta\tau_{m_k m_{k+1}}$, that may be observable in the transaction database, but represented under their amounts compact form.

| **Sequence** | $N_l^{(2)}$ | **Sequence** | $N_l^{(2)}$ | **Sequence** | $N_l^{(2)}$ | **Sequence** | $N_l^{(2)}$ |
|---|---|---|---|---|---|---|---|
| $\omega^2_{3001}(t_{n^{(2)}_{3001}})$ | 3 | $\omega^2_{3011}(t_{n^{(2)}_{3011}})$ | 1 | $\omega^2_{3021}(t_{n^{(2)}_{3021}})$ | 0 | $\omega^2_{3031}(t_{n^{(2)}_{3031}})$ | 8 |
| $\omega^2_{3002}(t_{n^{(2)}_{3002}})$ | 3 | $\omega^2_{3012}(t_{n^{(2)}_{3012}})$ | 0 | $\omega^2_{3022}(t_{n^{(2)}_{3022}})$ | 7 | $\omega^2_{3032}(t_{n^{(2)}_{3032}})$ | 1 |
| $\omega^2_{3003}(t_{n^{(2)}_{3003}})$ | 3 | $\omega^2_{3013}(t_{n^{(2)}_{3013}})$ | 1 | $\omega^2_{3023}(t_{n^{(2)}_{3023}})$ | 12 | $\omega^2_{3033}(t_{n^{(2)}_{3033}})$ | 1 |
| $\omega^2_{3004}(t_{n^{(2)}_{3004}})$ | 25 | $\omega^2_{3014}(t_{n^{(2)}_{3014}})$ | 0 | $\omega^2_{3024}(t_{n^{(2)}_{3024}})$ | 15 | $\omega^2_{3034}(t_{n^{(2)}_{3034}})$ | 15 |
| $\omega^2_{3005}(t_{n^{(2)}_{3005}})$ | 1 | $\omega^2_{3015}(t_{n^{(2)}_{3015}})$ | 0 | $\omega^2_{3025}(t_{n^{(2)}_{3025}})$ | 4 | $\omega^2_{3035}(t_{n^{(2)}_{3035}})$ | 14 |
| $\omega^2_{3006}(t_{n^{(2)}_{3006}})$ | 7 | $\omega^2_{3016}(t_{n^{(2)}_{3016}})$ | 0 | $\omega^2_{3026}(t_{n^{(2)}_{3026}})$ | 3 | $\omega^2_{3036}(t_{n^{(2)}_{3036}})$ | 12 |
| $\omega^2_{3007}(t_{n^{(2)}_{3007}})$ | 2 | $\omega^2_{3017}(t_{n^{(2)}_{3017}})$ | 24 | $\omega^2_{3027}(t_{n^{(2)}_{3027}})$ | 3 | $\omega^2_{3037}(t_{n^{(2)}_{3037}})$ | 7 |
| $\omega^2_{3008}(t_{n^{(2)}_{3008}})$ | 3 | $\omega^2_{3018}(t_{n^{(2)}_{3018}})$ | 6 | $\omega^2_{3028}(t_{n^{(2)}_{3028}})$ | 3 | $\omega^2_{3038}(t_{n^{(2)}_{3038}})$ | 14 |
| $\omega^2_{3009}(t_{n^{(2)}_{3009}})$ | 1 | $\omega^2_{3019}(t_{n^{(2)}_{3019}})$ | 23 | $\omega^2_{3029}(t_{n^{(2)}_{3029}})$ | 3 | $\omega^2_{3039}(t_{n^{(2)}_{3039}})$ | 24 |
| $\omega^2_{3010}(t_{n^{(2)}_{3010}})$ | 1 | $\omega^2_{3020}(t_{n^{(2)}_{3020}})$ | 6 | $\omega^2_{3030}(t_{n^{(2)}_{3030}})$ | 12 | $\omega^2_{3040}(t_{n^{(2)}_{3040}})$ | 10 |

Table 8.6: Number $N_l^{(2)}$ of observation classes for sequences $\omega^2_l(t_{n^{(2)}_l})$, $l \in \Delta_{ID\_TYP\_EVT}$

The size of the abstract chronicle model $\mathcal{M}^2_{TYP}$ is then:

$$Card(\mathcal{M}^2_{TYP}) = \sum_{\substack{(i,j) \in \Delta^2_{ID\_TYP\_EVT} \\ i \neq j}} (N_i + N_j)^2 = 231\ 640 \quad (8.67)$$

At this level of abstraction $\mathcal{L}^2_{TYP}$, the abstract chronicle model $\mathcal{M}^2_{CPTE}$ contains 231 640 temporal binary relations modeling all the successive transactions between a transaction type $i \in \Delta_{ID\_TYP\_EVT}$ to a transaction type $j \in \Delta_{ID\_TYP\_EVT}$. Compared to the 12 630 799 temporal binary relations contained in the abstract chronicle model $\mathcal{M}^1_{TYP}$ at the LoA $\mathcal{L}^1_{TYP}$, the number of temporal binary relations has been reduced to $98,16\%$.

Again, an Abstraction functor, denoted $TOT(\Delta^1_{TYP}) \rightarrow TOT(\Delta^2_{TYP})$, can be built according to the surjective classification function $\mu$, linking $TOT(\Delta^1_{TYP})$ Category to $TOT(\Delta^2_{TYP})$ Category. Such an Abstraction functor maps timed observations $O_{l_k}(t_{l_k})$ of $TOT(\Delta^1_{TYP})$ Category to its more abstract form $O^2_{m_k}(t_{l_k})$ of $TOT(\Delta^2_{TYP})$ Category:

$$\begin{array}{rccc} TOT(\Delta^1_{TYP}) \rightarrow TOT(\Delta^2_{TYP}) \quad : & TOT(\Delta^1_{TYP}) & \rightarrow & TOT(\Delta^2_{TYP}) \\ & O_{l_k}(t_{l_k}) \equiv (\delta_{l_k}, t_{l_k}) & \mapsto & O^2_{m_k}(t_{l_k}) \equiv (m_k, t_{l_k}) \end{array} \tag{8.68}$$

According to property 23, the LoA $\mathcal{L}^2$ is higher or more abstract than the LoA $\mathcal{L}^1$. Thus, each time the value of the function $x_i(t)$ changes, a timed observation $O_{i_k}(t_{i_k}) \equiv (\delta_{i_k}, t_{i_k})$ is written at the LoA 1 of abstraction, representing a transaction of the database and, at the same time, another timed observation $O^2_{m_k}(t_{i_k}) \equiv (m_k, t_{i_k})$ is written, representing the amount of this transaction into a more abstract form and residing at a more abstract level of abstraction.

Such a LoA, denoted $\mathcal{L}^2$, is composed of:

- three TOT Categories modeling the observed problem, still into three different point of view but also into a more compact form;
- three Abstraction functors linking categories belonging to the first LoA with categories belonging to the second LoA.

| **Categories** | $Card(\mathcal{M}^1)$ **at LoA** $\mathcal{L}^1$ | $Card(\mathcal{M}^2)$ **at LoA** $\mathcal{L}^2$ | **Data Reduction Rate** |
|---|---|---|---|
| Clients | 3 379 879 | 69 767 | $97,94\%$ |
| Banking Accounts | 14 635 861 | 404 797 | $97,23\%$ |
| Transaction Types | 12 630 799 | 231 640 | $98,16\%$ |

Table 8.7: Reduction rate of data to be analysed when abstracting from LoA $\mathcal{L}^1$ to LoA $\mathcal{L}^2$

It has been demonstrated in this section that the reduction rate of data to be analysed is around 98% (see table 8.7): data at the second LoA are semantically richer and syntacticly poorer than data at the first LoA. Abstraction functors allow the existence of a mapping between concrete timed observations of the first LoA and abstract timed observations of the second LoA. On one hand, a concrete timed observation of the LoA $\mathcal{L}^1$ have no existence out of its own LoA: it is definitively forgotten. On the other hand, an abstract timed observation of the LoA $\mathcal{L}^2$, thanks to the existence of an Abstraction functor, can always be linked to its more concrete form. In other word, an abstract timed observation has *always an interpretation* and can *always be reified* under its more concrete form.

Section 8.7 gives a concrete application of this concept of reification when solving the internal fraud problem.

### 8.6.4 Syntactic Model of the Observed Problem at the Third LoA

This step consists of observing binary sequences of successive timed observations of the form $\big(O(t_{i_k}) \equiv (m_k, t_{i_k}), O(t_{j_{k+1}}) \equiv (m_{k+1}, t_{j_{k+1}})\big)$ at the LoA $\mathcal{L}^2$ for categories $TOT(\Delta^2_{CLI})$, $TOT(\Delta^2_{CPTE})$ and $TOT(\Delta^2_{TYP})$. The observation of such binary sequences is based on the operation of composition of observers $\bigoplus$ (see 4.5 and 7.12) in these TOT Categories at the level of abstraction $\mathcal{L}^2$.

Let us consider the $TOT(\Delta^2_{CLI})$ Category whose objects are the four Unary Observers $\Theta^2_i(\phi_i, \Delta^2_i)$, $i \in \Delta_{ID\_CLI}$. Let us apply the operation of composition of observers on these objects:

$$\forall (i,j) \in \Delta^2_{ID\_CLI}, \Theta^3_k(\phi_k, \Delta^3) = \Theta^2_i(\phi_i, \Delta^2_i) \bigoplus \Theta^2_j(\phi_j, \Delta^2_j) \tag{8.69}$$

Since we only deal with the observation of successive timed observations, according to property 24, observer $\Theta^3_k(\phi_k, \Delta^3)$ can be decomposed into the four Abstract Binary Observers $\Theta^3_{ii}(\{\phi_{ii}\}, \{\delta^{ii}\})$, $\Theta^3_{ij}(\{\phi_{ij}\}, \{\delta^{ij}\})$, $\Theta^3_{ji}(\{\phi_{ji}\}, \{\delta^{ji}\})$ and $\Theta^3_{jj}(\{\phi_{jj}\}, \{\delta^{jj}\})$. Abstract Binary Observer $\Theta^3_{ii}(\{\phi_{ii}\}, \{\delta^{ii}\})$ observes binary sequences of successive timed observations from a client $i$ to the same client $i$: it is then useless for our concerned problem since these kind of transactions cannot be concerned by the fraud. With the same idea, Abstract Binary Observer $\Theta^3_{jj}(\{\phi_{jj}\}, \{\delta^{jj}\})$ is also useless. So, only Abstract Binary Observers $\Theta^3_{ij}(\{\phi_{ij}\}, \{\delta^{ij}\})$ and $\Theta^3_{ji}(\{\phi_{ji}\}, \{\delta^{ji}\})$ are concerned.

Considering Abstract Binary Observers $\Theta^3_{ij}(\{\phi_{ij}\}, \{\delta^{ij}\})$, equation 8.69 leads then to (the reasoning is the same with $\Theta^3_{ji}(\{\phi_{ji}\}, \delta^{ji})$ by swapping index $i$ and $j$):

$$\forall (i,j) \in \Delta^2_{ID\_CLI}, \Theta^3_{ij}(\{\phi_{ij}\}, \{\delta^{ij}\}) = \Theta^2_i(\phi_i, \Delta^2_i) \bigoplus \Theta^2_j(\phi_j, \Delta^2_j) \tag{8.70}$$

The operation 8.70 corresponds then to the operation of addition of timed observations under temporal constraints $\overset{\Delta\tau_{i_k j_{k+1}}}{+}$. At the LoA $\mathcal{L}^2$, such an operation is written:

$$O^3_{ij}(t_{j_{k+1}}) = O^2_{m_k}(t_{i_k}) \overset{\Delta\tau_{i_k j_{k+1}}}{+} O^2_{m_{k+1}}(t_{j_{k+1}}) \tag{8.71}$$

This operation corresponds to the application of the Modus Ponens with the following rule:

$$\begin{array}{c} \theta^2_i(\phi_i, m_k, t_{i_k}) \wedge \theta^2_j(\phi_j, m_{k+1}, t_{j_{k+1}}) \wedge |t_{j_{k+1}} - t_{i_k}| \in \Delta\tau_{i_k j_{k+1}} \\ \Rightarrow \theta^3_{ij}(\phi_{ij}, \delta^{ij}, t_{j_{k+1}}) \end{array} \tag{8.72}$$

where $m_k = \mu(x_i(t_{i_k}))$ and $m_{k+1} = \mu(x_j(t_{j_{k+1}}))$.

In order to respect constraints number one and two of the internal fraud constraints of definition 8.2, *(i) the debited account belongs to a bank customer* and *(ii) credited account belongs to a bank employee*, we must impose $i \neq 1003$ and $j = 1003$. In order to respect the constraint number three of the internal fraud constraints of definition 8.2, *(iii) the debited and credited amounts are the same*, we must impose abstract amounts in $\mathcal{M}$ to be of opposite signs, that is to say, $m_k + m_{k+1} = 0$ for all $k$. Constraint number four, *(iv) the debiting transaction precedes the crediting transaction*, is always respected since, by construction, $t_{i_k} < t_{j_{k+1}}$, for all $k$.

Let us then rewrite the constant $\delta^{ij}$ as the concatenation of the abstract amounts $-m_k$ and $m_k$:

$$\delta^{ij} \equiv -m_k m_k \tag{8.73}$$

Thus, the rule 8.72 becomes:

$$\begin{array}{c} \theta^2_i(\phi_i, m_k, t_{i_k}) \wedge \theta^2_j(\phi_j, -m_k, t_{j_{k+1}}) \wedge |t_{j_{k+1}} - t_{i_k}| \in \Delta\tau_{i_k j_{k+1}} \\ \Rightarrow \theta^3_{ij}(\phi_{ij}, -m_k m_k, t_{j_{k+1}}) \end{array} \tag{8.74}$$

Thus, for each assignation $\theta^3_{ij}(\phi_{ij}, -m_k m_k, t_{j_{k+1}})$, Abstract Binary Observer $\Theta^3_{ij}(\{\phi_{ij}\}, \{\delta^{ij}\})$ writes a timed observation of the form $O^3_{ij}(t_{j_{k+1}}) \equiv (\delta^{ij} \equiv -m_k m_k, t_{j_{k+1}}) \equiv O^3_{-m_k m_k}(t_{j_{k+1}})$. Such a timed observation means that it has been observed, in the superposition $\Omega^2_{CLI} = \bigcup_{i \in \Delta_{ID\_CLI}} \omega^2_i$, at the date $t_{j_{k+1}}$, a timed observation $O^2_{m_k}(t_{i_k}) \equiv (m_k, t_{i_k})$, $i \neq 1003$, followed by a timed observation $O^2_{-m_k}(t_{j_{k+1}}) \equiv (-m_k, t_{j_{k+1}})$, $j = 1003$, in a time interval of $\Delta\tau_{i_k j_{k+1}}$. Such a timed observation corresponds then to an instance $r^2_{-m_k m_k}(O^2_{m_k}(t_{i_k}), O^2_{-m_k}(t_{j_{k+1}}))$ of the temporal binary relation $r^2_{-m_k m_k}(O^2_{m_k}, O^2_{-m_k}, \Delta\tau_{i_k j_{k+1}})$. Such an instance composes the set $\mathcal{B}^2_{CLI}$ corresponding to the behaviour of the observed process $(\Phi_{CLI}(t), \Theta^2_{CLI}(\Phi_{CLI}, \Delta^2_{CLI}))$ (see 5.6) in the $TOT(\Delta^2_{CLI})$ Category. Thus, in other words, this step consists of determining the behaviour $\mathcal{B}^2_{CLI}$ in the $TOT(\Delta^2_{CLI})$ Category. Let us recall that such a behaviour $\mathcal{B}^2_{CLI}$ is an instance of the abstract chronicle model $\mathcal{M}^2_{CLI}$ at the LoA $\mathcal{L}^2$.

Let us now give more precisely the composition of the behaviour model $\mathcal{B}^2_{CLI}$. To this aim, let us first fix the temporal constraints to:

$$\forall k \in \mathbb{N}, \Delta\tau_{i_k j_{k+1}} = [\tau^-_{i_k}; \tau^+_{j_{k+1}}] = [0; 30\ days] \tag{8.75}$$

This concretely means that only pairs of transactions whose duration is *inferior to 30 days* are concerned.

Three Abstract Binary Observers $\Theta^3_{ij}(\{\phi_{ij}\}, \{\delta^{ij}\}) \equiv \Theta^3_{ij}$ observe binary sequences of timed observations of the form
$\left(O^2_{m_k}(t_{i_k}) \equiv (m_k, t_{i_k}), O^2_{-m_k}(t_{j_{k+1}}) \equiv (-m_k, t_{j_{k+1}})\right)$ from client $i \neq 1003$ to client $j = 1003$.

Table 8.8 provides binary sequences of timed observations that has been observed by Abstract Binary Observer $\Theta^3_{1001,1003}$ from sequence $\omega^2_{1001}(t_{n^{(2)}_{1001}})$ associated to client id 1001 to sequence $\omega^2_{1003}(t_{n^{(2)}_{1003}})$ associated to the bank manager (client id 1003). This table also provides the instances of concerned temporal binary relations as well as timed observations written by this Abstract Binary Observer. The list of corresponding timestamps is given in table 8.14.

| **Observed binary sequence** | **Instance of temporal binary relation** | **Timed observation written by $\Theta^3_{1001,1003}$** |
|---|---|---|
| $((-55, t_{1001_1}), (55, t_{1003_1}))$ | $r^2_{-5555}(O_{-55}(t_{1001_1}), O_{55}(t_{1003_1}))$ | $O^3_{-5555}(t_{1003_1}) \equiv (-5555, t_{1003_1})$ |
| $((-65, t_{1001_2}), (65, t_{1003_2}))$ | $r^2_{-6565}(O_{-65}(t_{1001_2}), O_{65}(t_{1003_2}))$ | $O^3_{-6565}(t_{1003_2}) \equiv (-6565, t_{1003_2})$ |

Table 8.8: Observation of binary sequences by Abstract Binary Observer $\Theta^3_{1001,1003}$ from client id 1001 to bank manager

Second Abstract Binary Observer, $\Theta^3_{1002,1003}$, does the same job in order to find binary sequences from sequence of timed observations $\omega^2_{1002}(t_{n^{(2)}_{1002}})$ associated to client id 1002 to sequence associated to the bank manager (see table 8.9):

| **Observed binary sequence** | **Instance of temporal binary relation** | **Timed observation written by $\Theta^3_{1002,1003}$** |
|---|---|---|
| $((-61, t_{1002_1}), (61, t_{1003_3}))$ | $r^2_{-6161}(O_{-61}(t_{1002_1}), O_{61}(t_{1003_3}))$ | $O^3_{-6161}(t_{1003_3}) \equiv (-6161, t_{1003_3})$ |
| $((-62, t_{1002_2}), (62, t_{1003_4}))$ | $r^2_{-6262}(O_{-62}(t_{1002_2}), O_{62}(t_{1003_4}))$ | $O^3_{-6262}(t_{1003_4}) \equiv (-6262, t_{1003_4})$ |

Table 8.9: Observation of binary sequences by Abstract Binary Observer $\Theta^3_{1002,1003}$ from client id 1002 to bank manager

And the third, $\Theta^3_{1004,1003}$, from sequence of timed observations $\omega^2_{1004}(t_{n^{(2)}_{1004}})$ associated to client id 1004 to sequence associated to the bank manager (see table 8.10):

| **Observed binary sequence** | **Instance of temporal binary relation** | **Timed observation written by $\Theta^3_{1004,1003}$** |
|---|---|---|
| $((-51, t_{1004_1}), (51, t_{1003_5}))$ | $r^2_{-5151}(O_{-51}(t_{1004_1}), O_{51}(t_{1003_5}))$ | $O^3_{-5151}(t_{1003_5}) \equiv (-5151, t_{1003_5})$ |
| $((-51, t_{1004_2}), (51, t_{1003_6}))$ | $r^2_{-5151}(O_{-51}(t_{1004_2}), O_{51}(t_{1003_6}))$ | $O^3_{-5151}(t_{1003_6}) \equiv (-5151, t_{1003_6})$ |
| $((-52, t_{1004_3}), (52, t_{1003_7}))$ | $r^2_{-5252}(O_{-52}(t_{1004_3}), O_{52}(t_{1003_7}))$ | $O^3_{-5252}(t_{1003_7}) \equiv (-5252, t_{1003_7})$ |
| $((-52, t_{1004_4}), (52, t_{1003_8}))$ | $r^2_{-5252}(O_{-52}(t_{1004_4}), O_{52}(t_{1003_8}))$ | $O^3_{-5252}(t_{1003_8}) \equiv (-5252, t_{1003_8})$ |
| $((-52, t_{1004_5}), (52, t_{1003_9}))$ | $r^2_{-5252}(O_{-52}(t_{1004_5}), O_{52}(t_{1003_9}))$ | $O^3_{-5252}(t_{1003_9}) \equiv (-5252, t_{1003_9})$ |
| $((-54, t_{1004_6}), (54, t_{1003_{10}}))$ | $r^2_{-5454}(O_{-54}(t_{1004_6}), O_{54}(t_{1003_{10}}))$ | $O^3_{-5454}(t_{1003_{10}}) \equiv (-5454, t_{1003_{10}})$ |
| $((-55, t_{1004_7}), (55, t_{1003_{11}}))$ | $r^2_{-5555}(O_{-55}(t_{1004_7}), O_{55}(t_{1003_{11}}))$ | $O^3_{-5555}(t_{1003_{11}}) \equiv (-5555, t_{1003_{11}})$ |
| $((-55, t_{1004_8}), (55, t_{1003_{12}}))$ | $r^2_{-5555}(O_{-55}(t_{1004_8}), O_{55}(t_{1003_{12}}))$ | $O^3_{-5555}(t_{1003_{12}}) \equiv (-5555, t_{1003_{12}})$ |
| $((-55, t_{1004_9}), (55, t_{1003_{13}}))$ | $r^2_{-5555}(O_{-55}(t_{1004_9}), O_{55}(t_{1003_{13}}))$ | $O^3_{-5555}(t_{1003_{13}}) \equiv (-5555, t_{1003_{13}})$ |
| $((-55, t_{1004_{10}}), (55, t_{1003_{14}}))$ | $r^2_{-5555}(O_{-55}(t_{1004_{10}}), O_{55}(t_{1003_{14}}))$ | $O^3_{-5555}(t_{1003_{14}}) \equiv (-5555, t_{1003_{14}})$ |
| $((-61, t_{1004_{11}}), (61, t_{1003_{15}}))$ | $r^2_{-6161}(O_{-61}(t_{1004_{11}}), O_{61}(t_{1003_{15}}))$ | $O^3_{-5555}(t_{1003_{15}}) \equiv (-6161, t_{1003_{15}})$ |
| $((-61, t_{1004_{12}}), (61, t_{1003_{16}}))$ | $r^2_{-6161}(O_{-61}(t_{1004_{12}}), O_{61}(t_{1003_{16}}))$ | $O^3_{-5555}(t_{1003_{16}}) \equiv (-6161, t_{1003_{16}})$ |
| $((-61, t_{1004_{13}}), (61, t_{1003_{17}}))$ | $r^2_{-6161}(O_{-61}(t_{1004_{13}}), O_{61}(t_{1003_{17}}))$ | $O^3_{-5555}(t_{1003_{17}}) \equiv (-6161, t_{1003_{17}})$ |
| $((-61, t_{1004_{14}}), (61, t_{1003_{18}}))$ | $r^2_{-6161}(O_{-61}(t_{1004_{14}}), O_{61}(t_{1003_{18}}))$ | $O^3_{-5555}(t_{1003_{18}}) \equiv (-6161, t_{1003_{18}})$ |

Table 8.10: Observation of binary sequences by Abstract Binary Observer $\Theta^3_{1004,1003}$ from client id 1004 to bank manager

Three other Abstract Binary Observers $\Theta^3_{ji}(\{\phi_{ji}\}, \{\delta^{ji}\})$ observe binary sequences of timed observations of the form
$\left(O^2_{m_k}(t_{j_k}) \equiv (m_k, t_{j_k}), O^2_{-m_k}(t_{i_{k+1}}) \equiv (-m_k, t_{i_{k+1}})\right)$ from client $j = 1003$ to client $i \neq 1003$.

Table 8.11 shows that there is no observed binary sequence neither instance of temporal binary relations nor timed observation written by this Abstract Binary Observer $\Theta^3_{1003,1001}$ from sequence $\omega^2_{1003}(t_{n^{(2)}_{1003}})$ associated to the bank manager to sequence $\omega^2_{1001}(t_{n^{(2)}_{1001}})$ associated client id 1001.

| **Observed binary sequence** | **Instance of temporal binary relation** | **Timed observation written by $\Theta^3_{1003,1001}$** |
|---|---|---|
| None | None | None |

Table 8.11: There is no observation of binary sequence by Abstract Binary Observer $\Theta^3_{1003,1001}$ from bank manager to client id 1001

On the other side, there exists binary sequences of timed observations observed by Abstract Binary Observer $\Theta^3_{1003,1002}$ from sequence associated to the bank manager to sequence $\omega^2_{1002}(t_{n^{(2)}_{1002}})$ associated client id 1002. Table 8.12 provides these sequence as well as instances of concerned temporal binary relations and timed observations written by this Abstract Binary Observer.

And there also exists binary sequences of timed observations observed by Abstract Binary Observer $\Theta^3_{1003,1004}$ from sequence associated to the bank manager to sequence $\omega^2_{1004}(t_{n^{(2)}_{1004}})$ associated client id 1004 (see table 8.13):

Thus, the behaviour model $\mathcal{B}^2_{CLI}$ is composed of the 24 instances of temporal binary relations listed in tables 8.8 to 8.13. Table 8.14 provides the list of the database datetimes corresponding to timestamps $t_{i_k}$ present in these tables.

| Observed binary sequence | Instance of temporal binary relation | Timed observation written by $\Theta^3_{ji}(\{\phi_{ji}\},\{\delta^{ji}\})$ |
|---|---|---|
| $((-51,t_{1003_{19}}),(51,t_{1002_3}))$ | $r^2_{-5151}(O_{-51}(t_{1003_{19}}),O_{51}(t_{1002_3}))$ | $O^3_{-5151}(t_{1003_{19}}) \equiv (-5151,t_{1003_{19}})$ |
| $((-51,t_{1003_{20}}),(51,t_{1002_4}))$ | $r^2_{-5151}(O_{-51}(t_{1003_{20}}),O_{51}(t_{1002_4}))$ | $O^3_{-5151}(t_{1003_{20}}) \equiv (-5151,t_{1003_{20}})$ |

Table 8.12: There is no observation of binary sequence by Abstract Binary Observer $\Theta^3_{ji}(\{\phi_{ji}\},\{\delta^{ji}\})$ from bank manager to client id 1002

| Observed binary sequence | Instance of temporal binary relation | Timed observation written by $\Theta^3_{1003,1004}$ |
|---|---|---|
| $((-51,t_{1003_{21}}),(51,t_{1004_{15}}))$ | $r^2_{-5151}(O_{-51}(t_{1003_{21}}),O_{51}(t_{1004_{15}}))$ | $O^3_{-5151}(t_{1003_{21}}) \equiv (-5151,t_{1003_{21}})$ |
| $((-51,t_{1003_{22}}),(51,t_{1004_{16}}))$ | $r^2_{-5151}(O_{-51}(t_{1003_{22}}),O_{51}(t_{1004_{16}}))$ | $O^3_{-5151}(t_{1003_{22}}) \equiv (-5151,t_{1003_{22}})$ |
| $((-51,t_{1003_{23}}),(51,t_{1004_{17}}))$ | $r^2_{-5151}(O_{-51}(t_{1003_{23}}),O_{51}(t_{1004_{17}}))$ | $O^3_{-5151}(t_{1003_{23}}) \equiv (-5151,t_{1003_{23}})$ |
| $((-55,t_{1003_{24}}),(55,t_{1004_{18}}))$ | $r^2_{-5555}(O_{-51}(t_{1003_{24}}),O_{51}(t_{1004_{18}}))$ | $O^3_{-5555}(t_{1003_{24}}) \equiv (-5151,t_{1003_{24}})$ |

Table 8.13: There is no observation of binary sequence by Abstract Binary Observer $\Theta^3_{1003,1004}$ from bank manager to client id 1004

| Timestamp | Datetime | Timestamp | Datetime |
|---|---|---|---|
| $t_{1001_1}$ | *2009-09-28 02:03:55* | $t_{1003_1}$ | *2009-10-08 03:24:52* |
| $t_{1001_2}$ | *2009-11-17 02:21:37* | $t_{1003_2}$ | *2009-11-18 06:20:55* |
| $t_{1002_1}$ | *2009-11-19 06:26:46* | $t_{1003_3}$ | *2009-11-23 09:07:10* |
| $t_{1002_2}$ | *2009-11-19 03:16:45* | $t_{1003_4}$ | *2009-11-27 09:08:00* |
| $t_{1004_1}$ | *2009-03-09 05:07:30* | $t_{1003_5}$ | *2009-03-10 05:30:41* |
| $t_{1004_2}$ | *2009-04-22 02:43:58* | $t_{1003_6}$ | *2009-05-22 03:12:18* |
| $t_{1004_3}$ | *2009-04-08 01:25:30* | $t_{1003_7}$ | *2009-04-28 03:35:43* |
| $t_{1004_4}$ | *2009-09-10 06:58:34* | $t_{1003_8}$ | *2009-09-16 02:33:07* |
| $t_{1004_5}$ | *2009-09-16 06:48:48* | $t_{1003_9}$ | *2009-09-28 08:03:11* |
| $t_{1004_6}$ | *2009-02-13 06:05:04* | $t_{1003_{10}}$ | *2009-02-26 03:53:07* |
| $t_{1004_7}$ | *2009-02-13 04:59:56* | $t_{1003_{11}}$ | *2009-02-17 01:36:17* |
| $t_{1004_8}$ | *2009-09-16 07:46:36* | $t_{1003_{12}}$ | *2009-09-18 05:48:22* |
| $t_{1004_9}$ | *2009-09-18 12:02:32* | $t_{1003_{13}}$ | *2009-09-21 03:23:20* |
| $t_{1004_{10}}$ | *2009-09-21 04:31:59* | $t_{1003_{14}}$ | *2009-10-08 03:24:52* |
| $t_{1004_{11}}$ | *2009-03-16 07:49:24* | $t_{1003_{15}}$ | *2009-03-17 03:29:01* |
| $t_{1004_{12}}$ | *2009-10-16 04:59:22* | $t_{1003_{16}}$ | *2009-10-30 09:33:29* |
| $t_{1004_{13}}$ | *2009-11-09 09:39:00* | $t_{1003_{17}}$ | *2009-11-13 04:53:17* |
| $t_{1004_{14}}$ | *2009-11-13 08:43:05* | $t_{1003_{18}}$ | *2009-11-23 09:07:10* |
| $t_{1003_{19}}$ | *2009-11-12 02:19:11* | $t_{1002_3}$ | *2009-11-18 05:11:08* |
| $t_{1003_{20}}$ | *2009-11-24 04:31:59* | $t_{1002_4}$ | *2009-11-30 03:18:00* |
| $t_{1003_{21}}$ | *2009-01-09 02:28:34* | $t_{1004_{15}}$ | *2009-01-22 02:34:23* |
| $t_{1003_{22}}$ | *2009-06-04 03:49:44* | $t_{1004_{16}}$ | *2009-06-12 04:47:47* |
| $t_{1003_{23}}$ | *2009-12-22 04:47:11* | $t_{1004_{17}}$ | *2009-12-22 07:38:53* |
| $t_{1003_{24}}$ | *2009-03-20 03:45:22* | $t_{1004_{18}}$ | *2009-03-30 05:27:06* |

Table 8.14: List of database datetimes corresponding to timestamps present in tables 8.8 to 8.13

According to property 22, the existence of the behaviour model $\mathcal{B}^2_{CLI}$ at the LoA $\mathcal{L}^2_{CLI}$ induces an abstraction process which allows to define a higher (or more abstract) LoA denoted $\mathcal{L}^3_{CLI}$. At this third LoA, reside timed observations of the form $O^3_{ij}(t_{j_{k+1}})$ (resp. $O^3_{ji}(t_{i_{k+1}})$) produced by Abstract Binary Observer $\Theta^3_{ij}(\{\phi_{ij}\},\{\delta^{ij}\})$ (resp. $\Theta^3_{ji}(\{\phi_{ji}\},\delta^{ji})$).

Let us denote $\omega_{ij}^3(t_{n_{ij}^{(3)}}) = \{O_{ij}^3(t_{j_{k+1}}), i \neq 1003, j = 1003\} \equiv \omega_{ij}^3$, the six sequences of $n_{ij}^{(3)}$ timed observations such as $i \neq 1003$ and $j = 1003$. Let us denote $\Omega^3 = \bigcup_{i\neq 1003, j=1003} \omega_{ij}^3(t_{n_{ij}^{(3)}})$, the superposition of the six sequences $\omega_{ij}^3(t_{n_{ij}^{(3)}})$.

Let us compute the cardinal of the abstract chronicle model $\mathcal{M}_{CLI}^3$ at this level $\mathcal{L}_{CLI}^3$ of abstraction. To this aim, we must compute the number of observation classes $O_{-m_k m_k}^3 = \{(\phi_{ij}, \delta^{ij} \equiv -m_k m_k)\}$ at level $\mathcal{L}_{CLI}^3$.

| **Sequence** | $N_{ij}^{(3)}$ |
|---|---|
| $\omega_{1001,1003}^3$ | 2 |
| $\omega_{1002,1003}^3$ | 2 |
| $\omega_{1004,1003}^3$ | 5 |
| $\omega_{1003,1001}^3$ | 0 |
| $\omega_{1003,1002}^3$ | 1 |
| $\omega_{1003,1004}^3$ | 2 |

Table 8.15: Number $N_{ij}^{(3)}$ of observation classes associated with sequences $\omega_{ij}^3$, $i \neq 1003, j = 1003$

Table 8.15 sums up the number of observation classes associated with sequences $\omega_{ij}^3$, $i \neq 1003, j = 1003$. This table allows to compute the size of the abstract chronicle model $\mathcal{M}_{CLI}^3$ according to property 20:

$$\begin{aligned} Card(\mathcal{M}_{CLI}^3) = \; & (2+2)^2 + (2+5)^2 + (2+0)^2 + (2+1)^2 + (2+2)^2 + \\ & (2+5)^2 + (2+0)^2 + (2+1)^2 + (2+2)^2 + \\ & (5+0)^2 + (5+1)^2 + (5+2)^2 + \\ & (0+1)^2 + (0+2)^2 + \\ & (1+2)^2 \\ Card(\mathcal{M}_{CLI}^3) = \; & 296 \end{aligned} \tag{8.76}$$

At this level of abstraction $\mathcal{L}_{CLI}^3$, the abstract chronicle model $\mathcal{M}_{CLI}^3$ contains 296 temporal binary relations modeling all the successive timed observations of the form $\left(O_{-m_k m_k}^3(t_k), O_{-m_{k+1} m_{k+1}}^3(t_{k+1})\right)$ and respecting the internal fraud constraints of definition 8.2. Compared to the 69 767 temporal binary relations contained in the abstract chronicle model $\mathcal{M}_{CLI}^2$ at the LoA $\mathcal{L}_{CLI}^2$, the number of temporal binary relations has been reduced to $99,58\%$. The operation of abstraction from LoA $\mathcal{L}_{CLI}^2$ to LoA $\mathcal{L}_{CLI}^3$ demonstrates again that data at the third LoA are semantically richer and syntacticly poorer than data at the second LoA.

Let us then build the $TOT(\Delta_{CLI}^3)$ Category whose :

- objects are Abstract Binary Observers $\Theta_{ij}^3(\{\phi_{ij}\}, \{\delta^{ij}\})$ and $\Theta_{mn}^3(\{\phi_{mn}\}, \{\delta^{mn}\})$, $(i,j,m,n) \in \Delta_{ID\_CLI}^4$.
- morphisms are relations $f_{ijmn}^3 : \Theta_{ij}^3 \to \Theta_{mn}^3$, $(i,j,m,n) \in \Delta_{ID\_CLI}^4$.

According to the definition 7.4, an Abstraction functor denoted $TOT(\Delta_{CLI}^2) \to TOT(\Delta_{CLI}^3)$ can be built. This Abstraction functor links categories $TOT(\Delta_{CLI}^2)$ to $TOT(\Delta_{CLI}^3)$ and maps binary sequences of timed observations $\left(O_{m_k}^2(t_{i_k}) \equiv (m_k, t_{i_k}), O_{m_{k+1}}^2(t_{j_{k+1}}) \equiv (-m_k, t_{j_{k+1}})\right)$ with a timed observation of the form $O_{-m_k m_k}^3(t_{j_{k+1}}) \equiv (-m_k m_k, t_{j_{k+1}})$:

$$\begin{array}{rcccc} TOT(\Delta^2_{CLI}) \rightarrow TOT(\Delta^3_{CLI}) & : & TOT(\Delta^2_{CLI}) & \rightarrow & TOT(\Delta^3_{CLI}) \\ & & ((m_k, t_{i_k}), (-m_k, t_{j_{k+1}})) & \mapsto & (-m_k m_k, t_{j_{k+1}}) \end{array} \quad (8.77)$$

Timed observations at the level $\mathcal{L}^3_{CLI}$ of abstraction are of the form $O^3_{-m_k m_k}(t_{j_{k+1}}) \equiv (-m_k m_k, t_{j_{k+1}})$. By construction, constants $-m_k m_k$ does not depend on the category we consider: they are the same whatever the observed process is. As a consequence, the superposition $\Omega^3$ of timed observation $O^3_{-m_k m_k}(t_{j_{k+1}})$, at this third LoA, is the same whatever we choose to observe the process associated to clients, accounts or transaction types. The difference comes from the way this superposition is produced, that is to say, from the number of Abstract Binary Observers involved to produce such a superposition.

For example, when considering the category of the clients, Abstract Binary Observer $\Theta^3_{1001,1003}$ observes binary sequences $((-55, t_{1001_1}), (55, t_{1003_1}))$ and $((-65, t_{1001_2}), (65, t_{1003_2}))$ in the superposition $\Omega^3$ (see table 8.8), writing respectively timed observations $O^3_{-5555}(t_{1003_1})$ and $O^3_{-6565}(t_{1003_2})$. When considering the category of accounts, binary sequence $((-55, t_{1001_1}), (55, t_{1003_1}))$ is observed by Abstract Binary Observer $\Theta^3_{2008,2001}$ while binary sequence $((-65, t_{1001_2}), (65, t_{1003_2}))$ is observed by Abstract Binary Observer $\Theta^3_{2009,2024}$. Nevertheless, both Abstract Binary Observers write the same timed observations i.e. $O^3_{-5555}(t_{1003_1})$ and $O^3_{-6565}(t_{1003_2})$. As a consequence, instances of temporal binary relations $r^2_{-5555}(O_{-55}(t_{1001_1}), O_{55}(t_{1003_1}))$ and $r^2_{-6565}(O_{-65}(t_{1001_2}), O_{65}(t_{1003_2}))$ are also the same.

Let us then consider the $TOT(\Delta^3_{CPTE})$ Category whose objects are Abstract Binary Observers $\Theta^3_{ij}(\{\phi_{ij}\}, \{\delta^{ij}\})$ where $(i, j) \in \Delta_{ID\_CPTE} \times \Delta_{ID\_CPTE}$. Let us then consider also the $TOT(\Delta^3_{TYP})$ Category whose objects are Abstract Binary Observers $\Theta^3_{ij}(\{\phi_{ij}\}, \{\delta^{ij}\})$ where $(i, j) \in \Delta_{ID\_TYP\_EVT} \times \Delta_{ID\_TYP\_EVT}$.

Table 8.16 lists the concerned Abstract Binary Observers (ABS) in the category of the account ids and in the category of the event type ids for observed binary sequences and instances of temporal binary relations of tables 8.8 to 8.13.

Thus, we can affirm that behaviour models associated to clients, accounts and transaction types, at the LoA $\mathcal{L}^2$, are the same:

$$\mathcal{B}^2_{CLI} = \mathcal{B}^2_{CPTE} = \mathcal{B}^2_{TYP} = \mathcal{B}^2 \quad (8.78)$$

Furthermore, there exists, in the the $TOT(\Delta^3_{CPTE})$ Category, 13 different Abstract Binary Observers corresponding to 13 sequences $\omega^3_{ij}(t_{n^{(3)}_{ij}})$, $(i, j) \in \Delta_{ID\_CPTE} \times \Delta_{ID\_CPTE}$ of timed observations $O^3_{-m_k m_k}(t_{j_{k+1}})$. The superposition $\Omega^3$, in this category, is then a superposition of 13 sequences.

And there exists, in the the $TOT(\Delta^3_{TYP})$ Category, 17 different Abstract Binary Observers corresponding to 17 sequences $\omega^3_{ij}(t_{n^{(3)}_{ij}})$, $(i, j) \in \Delta_{ID\_TYP\_EVT} \times \Delta_{ID\_TYP\_EVT}$ of timed observations $O^3_{-m_k m_k}(t_{j_{k+1}})$. The superposition $\Omega^3$, in this category, is then a superposition of 17 sequences.

Let us compute the cardinal of the abstract chronicle model $\mathcal{M}^3_{CPTE}$ at the level $\mathcal{L}^3_{CPTE}$ of abstraction. To this aim, les us compute the number of observation classes $O^3_{-m_k m_k} = \{(\phi_{ij}, \delta^{ij} \equiv$

| Observed binary sequence | Instance of temporal binary relation | Concerned ABS in account id category | Concerned ABS in transaction type id category |
|---|---|---|---|
| $((-55, t_{1001_1}), (55, t_{1003_1}))$ | $r^2_{-5555}(O_{-55}(t_{1001_1}), O_{55}(t_{1003_1}))$ | $\Theta^3_{2008,2001}$ | $\Theta^3_{3034,3023}$ |
| $((-65, t_{1001_2}), (65, t_{1003_2}))$ | $r^2_{-6565}(O_{-65}(t_{1001_2}), O_{65}(t_{1003_2}))$ | $\Theta^3_{2009,2024}$ | $\Theta^3_{3038,3023}$ |
| $((-61, t_{1002_1}), (61, t_{1003_3}))$ | $r^2_{-6161}(O_{-61}(t_{1002_1}), O_{61}(t_{1003_3}))$ | $\Theta^3_{2005,2028}$ | $\Theta^3_{3034,3017}$ |
| $((-62, t_{1002_2}), (62, t_{1003_4}))$ | $r^2_{-6262}(O_{-62}(t_{1002_2}), O_{62}(t_{1003_4}))$ | $\Theta^3_{2011,2027}$ | $\Theta^3_{3038,3010}$ |
| $((-51, t_{1004_1}), (51, t_{1003_5}))$ | $r^2_{-5151}(O_{-51}(t_{1004_1}), O_{51}(t_{1003_5}))$ | $\Theta^3_{2007,2001}$ | $\Theta^3_{3004,3023}$ |
| $((-51, t_{1004_2}), (51, t_{1003_6}))$ | $r^2_{-5151}(O_{-51}(t_{1004_2}), O_{51}(t_{1003_6}))$ | $\Theta^3_{2007,2001}$ | $\Theta^3_{3004,3005}$ |
| $((-52, t_{1004_3}), (52, t_{1003_7}))$ | $r^2_{-5252}(O_{-52}(t_{1004_3}), O_{52}(t_{1003_7}))$ | $\Theta^3_{2007,2001}$ | $\Theta^3_{3004,3040}$ |
| $((-52, t_{1004_4}), (52, t_{1003_8}))$ | $r^2_{-5252}(O_{-52}(t_{1004_4}), O_{52}(t_{1003_8}))$ | $\Theta^3_{2020,2001}$ | $\Theta^3_{3038,3039}$ |
| $((-52, t_{1004_5}), (52, t_{1003_9}))$ | $r^2_{-5252}(O_{-52}(t_{1004_5}), O_{52}(t_{1003_9}))$ | $\Theta^3_{2020,2001}$ | $\Theta^3_{3038,3039}$ |
| $((-54, t_{1004_6}), (54, t_{1003_{10}}))$ | $r^2_{-5454}(O_{-54}(t_{1004_6}), O_{54}(t_{1003_{10}}))$ | $\Theta^3_{2007,2001}$ | $\Theta^3_{3034,3039}$ |
| $((-55, t_{1004_7}), (55, t_{1003_{11}}))$ | $r^2_{-5555}(O_{-55}(t_{1004_7}), O_{55}(t_{1003_{11}}))$ | $\Theta^3_{2007,2001}$ | $\Theta^3_{3034,3023}$ |
| $((-55, t_{1004_8}), (55, t_{1003_{12}}))$ | $r^2_{-5555}(O_{-55}(t_{1004_8}), O_{55}(t_{1003_{12}}))$ | $\Theta^3_{2007,2001}$ | $\Theta^3_{3034,3039}$ |
| $((-55, t_{1004_9}), (55, t_{1003_{13}}))$ | $r^2_{-5555}(O_{-55}(t_{1004_9}), O_{55}(t_{1003_{13}}))$ | $\Theta^3_{2010,2001}$ | $\Theta^3_{3038,3039}$ |
| $((-55, t_{1004_{10}}), (55, t_{1003_{14}}))$ | $r^2_{-5555}(O_{-55}(t_{1004_{10}}), O_{55}(t_{1003_{14}}))$ | $\Theta^3_{2010,2001}$ | $\Theta^3_{3038,3023}$ |
| $((-61, t_{1004_{11}}), (61, t_{1003_{15}}))$ | $r^2_{-6161}(O_{-61}(t_{1004_{11}}), O_{61}(t_{1003_{15}}))$ | $\Theta^3_{2007,2001}$ | $\Theta^3_{3004,3039}$ |
| $((-61, t_{1004_{12}}), (61, t_{1003_{16}}))$ | $r^2_{-6161}(O_{-61}(t_{1004_{12}}), O_{61}(t_{1003_{16}}))$ | $\Theta^3_{2010,2003}$ | $\Theta^3_{3038,3017}$ |
| $((-61, t_{1004_{13}}), (61, t_{1003_{17}}))$ | $r^2_{-6161}(O_{-61}(t_{1004_{13}}), O_{61}(t_{1003_{17}}))$ | $\Theta^3_{2010,2024}$ | $\Theta^3_{3038,3023}$ |
| $((-61, t_{1004_{14}}), (61, t_{1003_{18}}))$ | $r^2_{-6161}(O_{-61}(t_{1004_{14}}), O_{61}(t_{1003_{18}}))$ | $\Theta^3_{2010,2028}$ | $\Theta^3_{3038,3017}$ |
| $((-51, t_{1003_{19}}), (51, t_{1002_3}))$ | $r^2_{-5151}(O_{-51}(t_{1003_{19}}), O_{51}(t_{1002_3}))$ | $\Theta^3_{2024,2019}$ | $\Theta^3_{3004,3017}$ |
| $((-51, t_{1003_{20}}), (51, t_{1002_4}))$ | $r^2_{-5151}(O_{-51}(t_{1003_{20}}), O_{51}(t_{1002_4}))$ | $\Theta^3_{2024,2005}$ | $\Theta^3_{3036,3035}$ |
| $((-51, t_{1003_{21}}), (51, t_{1004_{15}}))$ | $r^2_{-5151}(O_{-51}(t_{1003_{21}}), O_{51}(t_{1004_{15}}))$ | $\Theta^3_{2001,2007}$ | $\Theta^3_{3031,3023}$ |
| $((-51, t_{1003_{22}}), (51, t_{1004_{16}}))$ | $r^2_{-5151}(O_{-51}(t_{1003_{22}}), O_{51}(t_{1004_{16}}))$ | $\Theta^3_{2001,2007}$ | $\Theta^3_{3004,3023}$ |
| $((-51, t_{1003_{23}}), (51, t_{1004_{17}}))$ | $r^2_{-5151}(O_{-51}(t_{1003_{23}}), O_{51}(t_{1004_{17}}))$ | $\Theta^3_{2024,2007}$ | $\Theta^3_{3024,3023}$ |
| $((-55, t_{1003_{24}}), (55, t_{1004_{18}}))$ | $r^2_{-5555}(O_{-51}(t_{1003_{24}}), O_{51}(t_{1004_{18}}))$ | $\Theta^3_{2001,2007}$ | $\Theta^3_{3036,3035}$ |

Table 8.16: Observed binary sequences, instances of temporal binary relations and their concerned Abstract Binary Observer in the $TOT(\Delta^3_{CPTE})$ Category and in the $TOT(\Delta^3_{TYP})$ Category

$-m_k m_k)\}$ at level $\mathcal{L}^3_{CPTE}$.

| Sequence | $N^{(3)}_{ij}$ | Sequence | $N^{(3)}_{ij}$ |
|---|---|---|---|
| $\omega^3_{2008,2001}$ | 1 | $\omega^3_{2010,2003}$ | 1 |
| $\omega^3_{2009,2024}$ | 1 | $\omega^3_{2010,2024}$ | 1 |
| $\omega^3_{2005,2028}$ | 1 | $\omega^3_{2010,2028}$ | 1 |
| $\omega^3_{2011,2027}$ | 1 | $\omega^3_{2024,2019}$ | 1 |
| $\omega^3_{2007,2001}$ | 4 | $\omega^3_{2024,2005}$ | 1 |
| $\omega^3_{2020,2001}$ | 1 | $\omega^3_{2024,2007}$ | 1 |
| $\omega^3_{2010,2001}$ | 1 | | |

Table 8.17: Number $N^{(3)}_{ij}$ of observation classes associated with sequences $\omega^3_{ij}(t_{n^{(3)}_{ij}})$, $(i,j) \in \Delta_{ID\_CPTE} \times \Delta_{ID\_CPTE}$

Table 8.17 sums up the number of observation classes associated with sequences $\omega^3_{ij}(t_{n^{(3)}_{ij}})$, $(i,j) \in \Delta_{ID\_CPTE} \times \Delta_{ID\_CPTE}$. This table allows to compute the size of the abstract chronicle model $\mathcal{M}^3_{CPTE}$ according to property 20:

$$\begin{aligned} Card(\mathcal{M}^3_{CPTE}) = & \sum_{\substack{(i,j)\in\Delta^2_{ID\_CPTE} \\ i\neq j}} (N_i + N_j)^2 \\ Card(\mathcal{M}^3_{CPTE}) = & \ 564 \end{aligned} \tag{8.79}$$

At this level of abstraction $\mathcal{L}^3_{CPTE}$, the abstract chronicle model $\mathcal{M}^3_{CPTE}$ contains 564 temporal binary relations modeling all the successive timed observations of the form $\left(O^3_{-m_k m_k}(t_k), O^3_{-m_{k+1} m_{k+1}}(t_{k+1})\right)$ and respecting the internal fraud constraints of definition 8.2. Compared to the 404 797 temporal binary relations contained in the abstract chronicle model $\mathcal{M}^2_{CPTE}$ at the LoA $\mathcal{L}^2_{CPTE}$, the number of temporal binary relations has been reduced to $99,86\%$. The operation of abstraction from LoA $\mathcal{L}^2_{CPTE}$ to LoA $\mathcal{L}^3_{CPTE}$ demonstrates again that data at the third LoA are semantically richer and syntacticly poorer than data at the second LoA.

Now, let us compute the cardinal of the abstract chronicle model $\mathcal{M}^3_{TYP}$ at the level $\mathcal{L}^3_{TYP}$ of abstraction.

| **Sequence** | $N^{(3)}_{ij}$ | **Sequence** | $N^{(3)}_{ij}$ |
|---|---|---|---|
| $\omega^3_{3034,3023}$ | 1 | $\omega^3_{3034,3023}$ | 1 |
| $\omega^3_{3038,3023}$ | 1 | $\omega^3_{3004,3039}$ | 1 |
| $\omega^3_{3034,3017}$ | 1 | $\omega^3_{3038,3017}$ | 1 |
| $\omega^3_{3038,3010}$ | 1 | $\omega^3_{3004,3017}$ | 1 |
| $\omega^3_{3004,3023}$ | 4 | $\omega^3_{3036,3035}$ | 1 |
| $\omega^3_{3004,3035}$ | 1 | $\omega^3_{3031,3023}$ | 1 |
| $\omega^3_{3004,3040}$ | 1 | $\omega^3_{3024,3023}$ | 1 |
| $\omega^3_{3038,3039}$ | 1 | $\omega^3_{3036,3035}$ | 1 |
| $\omega^3_{3034,3039}$ | 1 | | |

Table 8.18: Number $N^{(3)}_{ij}$ of observation classes associated with sequences $\omega^3_{ij}(t_{n^{(3)}_{ij}})$, $(i,j) \in \Delta_{ID\_TYP\_EVT} \times \Delta_{ID\_TYP\_EVT}$

Table 8.18 sums up the number of observation classes associated with sequences $\omega^3_{ij}(t_{n^{(3)}_{ij}})$, $(i,j) \in \Delta_{ID\_TYP\_EVT} \times \Delta_{ID\_TYP\_EVT}$. This table allows to compute the size of the abstract chronicle model $\mathcal{M}^3_{TYP}$ according to property 20:

$$\begin{aligned} Card(\mathcal{M}^3_{TYP}) = & \sum_{\substack{(i,j)\in\Delta^2_{ID\_TYP\_EVT} \\ i\neq j}} (N_i + N_j)^2 \\ Card(\mathcal{M}^3_{TYP}) = & \ 906 \end{aligned} \tag{8.80}$$

At this level of abstraction $\mathcal{L}^3_{TYP}$, the abstract chronicle model $\mathcal{M}^3_{TYP}$ contains 906 temporal binary relations modeling all the successive timed observations of the form $\left(O^3_{-m_k m_k}(t_k), O^3_{-m_{k+1} m_{k+1}}(t_{k+1})\right)$ and respecting the internal fraud constraints of definition 8.2. Compared to the 231 640 temporal binary relations contained in the abstract chronicle model $\mathcal{M}^2_{TYP}$ at the LoA $\mathcal{L}^2_{TYP}$, the number of temporal binary relations has been reduced to $99,61\%$. The operation of abstraction from LoA $\mathcal{L}^2_{TYP}$ to LoA $\mathcal{L}^3_{TYP}$ demonstrates again that data at the third LoA are semantically richer and syntacticly poorer than data at the second LoA.

According to the definition 7.4, two Abstraction functors denoted $TOT(\Delta^2_{CPTE}) \rightarrow TOT(\Delta^3_{CPTE})$ and $TOT(\Delta^2_{TYP}) \rightarrow TOT(\Delta^3_{TYP})$ can be built.

Abstraction functor $TOT(\Delta^2_{CPTE}) \rightarrow TOT(\Delta^3_{CPTE})$ links categories $TOT(\Delta^2_{CPTE})$ to $TOT(\Delta^3_{CPTE})$ and maps binary sequences of timed observations $\left(O^2_{m_k}(t_{i_k}) \equiv (m_k, t_{i_k}), O^2_{m_{k+1}}(t_{j_{k+1}}) \equiv (-m_k, t_{j_{k+1}})\right)$

with a timed observation of the form $O^3_{-m_k m_k}(t_{j_{k+1}}) \equiv (-m_k m_k, t_{j_{k+1}})$:

$$\begin{array}{rccc} TOT(\Delta^2_{CPTE}) \rightarrow TOT(\Delta^3_{CPTE}) & : & TOT(\Delta^2_{CPTE}) & \rightarrow & TOT(\Delta^3_{CPTE}) \\ & & ((m_k, t_{i_k}), (-m_k, t_{j_{k+1}})) & \mapsto & (-m_k m_k, t_{j_{k+1}}) \end{array} \quad (8.81)$$

Abstraction functor $TOT(\Delta^2_{TYP}) \rightarrow TOT(\Delta^3_{TYP})$ links categories $TOT(\Delta^2_{TYP})$ to $TOT(\Delta^3_{TYP})$ and maps binary sequences of timed observations $\left(O^2_{m_k}(t_{i_k}) \equiv (m_k, t_{i_k}), O^2_{m_{k+1}}(t_{j_{k+1}}) \equiv (-m_k, t_{j_{k+1}})\right)$ with a timed observation of the form $O^3_{-m_k m_k}(t_{j_{k+1}}) \equiv (-m_k m_k, t_{j_{k+1}})$:

$$\begin{array}{rccc} TOT(\Delta^2_{TYP}) \rightarrow TOT(\Delta^3_{TYP}) & : & TOT(\Delta^2_{TYP}) & \rightarrow & TOT(\Delta^3_{TYP}) \\ & & ((m_k, t_{i_k}), (-m_k, t_{j_{k+1}})) & \mapsto & (-m_k m_k, t_{j_{k+1}}) \end{array} \quad (8.82)$$

Thus, the LoA $\mathcal{L}^3$ is composed of:

- three TOT Categories containing the same sequence $\Omega^3$ of timed observations;
- three Abstraction functors linking categories belonging to the second LoA with categories belonging to the third LoA.

| **Categories** | $Card(\mathcal{M}^2)$ **at LoA** $\mathcal{L}^2$ | $Card(\mathcal{M}^3)$ **at LoA** $\mathcal{L}^3$ | **Data Reduction Rate** |
|---|---|---|---|
| Clients | 69 767 | 296 | $99,58\%$ |
| Banking Accounts | 404 797 | 564 | $99,86\%$ |
| Transaction Types | 231 640 | 906 | $99,61\%$ |

Table 8.19: Reduction rate of data to be analysed when abstracting from LoA $\mathcal{L}^2$ to LoA $\mathcal{L}^3$

It has been demonstrated in this section that the reduction rate of data to be analysed is more than 99% (see table 8.19): data at the third LoA are semantically richer and syntacticly poorer than data at the second LoA. Abstraction functors allow the existence of a mapping between abstract timed observations of the second LoA and abstract timed observations of the third LoA. These Abstraction functors allow to keep the link between timed observations of the third LoA and timed observations of the second LoA. Keeping such a link is the key in order to be able to reify the data from a higher LoA to a lower LoA. Again, section 8.7 gives a concrete application of this concept of reification when solving the internal fraud problem.

Now all elements are in place to build the gradient of abstraction (GoA) of the observed problem.

### 8.6.5 Gradient of Abstraction of the Observed Problem

Let us then sum up elements we have:

- at the first LoA $\mathcal{L}^1$:
  - the LoA $\mathcal{L}^1_{CLI} = \langle \mathcal{S}^1_{CLI}, \mathcal{T}^1_{CLI} \rangle$ of the $TOT(\Delta^1_{CLI})$ Category and its behaviour model $\mathcal{B}^1_{CLI}$ as an instance of the abstract chronicle model $\mathcal{M}^1_{CLI}$;
  - the LoA $\mathcal{L}^1_{CPTE} = \langle \mathcal{S}^1_{CPTE}, \mathcal{T}^1_{CPTE} \rangle$ of the $TOT(\Delta^1_{CPTE})$ Category and its behaviour model $\mathcal{B}^1_{CPTE}$ as an instance of the abstract chronicle model $\mathcal{M}^1_{CPTE}$;

  - the LoA $\mathcal{L}^1_{TYP} = \langle \mathcal{S}^1_{TYP}, \mathcal{T}^1_{TYP} \rangle$ of the $TOT(\Delta^1_{TYP})$ Category and its behaviour model $\mathcal{B}^1_{TYP}$ as an instance of the abstract chronicle model $\mathcal{M}^1_{TYP}$;
  - Modeling functors $TOT(\Delta^1_{CLI}) \rightarrow TOT(\Delta^1_{CPTE})$, $TOT(\Delta^1_{CPTE}) \rightarrow TOT(\Delta^1_{TYP})$ and $TOT(\Delta^1_{TYP}) \rightarrow TOT(\Delta^1_{CLI})$;
  - Modeling functors $TOT(\Delta^1_{CPTE}) \rightarrow TOT(\Delta^1_{CLI})$, $TOT(\Delta^1_{TYP}) \rightarrow TOT(\Delta^1_{CPTE})$ and $TOT(\Delta^1_{CLI}) \rightarrow TOT(\Delta^1_{TYP})$.

  Behaviour models $\mathcal{B}^1_{CLI}$, $\mathcal{B}^1_{CPTE}$ and $\mathcal{B}^1_{TYP}$ are obviously not given since the aim of this chapter is to compute these behaviour models at a higher LoA. Nevertheless, they do exist.

- at the second LoA $\mathcal{L}^2$:

  - the LoA $\mathcal{L}^2_{CLI} = \langle \mathcal{S}^2_{CLI}, \mathcal{T}^2_{CLI} \rangle$ of the $TOT(\Delta^2_{CLI})$ Category and its behaviour model $\mathcal{B}^2_{CLI}$ as an instance of the abstract chronicle model $\mathcal{M}^2_{CLI}$;
  - the LoA $\mathcal{L}^2_{CPTE} = \langle \mathcal{S}^2_{CPTE}, \mathcal{T}^2_{CPTE} \rangle$ of the $TOT(\Delta^2_{CPTE})$ Category and its behaviour model $\mathcal{B}^2_{CPTE}$ as an instance of the abstract chronicle model $\mathcal{M}^2_{CPTE}$;
  - the LoA $\mathcal{L}^2_{TYP} = \langle \mathcal{S}^2_{TYP}, \mathcal{T}^2_{TYP} \rangle$ of the $TOT(\Delta^2_{TYP})$ Category and its behaviour model $\mathcal{B}^2_{TYP}$ as an instance of the abstract chronicle model $\mathcal{M}^2_{TYP}$;
  - Abstract functors $TOT(\Delta^1_{CLI}) \rightarrow TOT(\Delta^2_{CLI})$, $TOT(\Delta^1_{CPTE}) \rightarrow TOT(\Delta^2_{CPTE})$ and $TOT(\Delta^1_{TYP}) \rightarrow TOT(\Delta^2_{TYP})$.

  And it has been seen that $\mathcal{B}^2_{CLI} = \mathcal{B}^2_{CPTE} = \mathcal{B}^2_{TYP} = \mathcal{B}^2$.

- at the third LoA $\mathcal{L}^3$:

  - the LoA $\mathcal{L}^3_{CLI} = \langle \mathcal{S}^3_{CLI}, \mathcal{T}^3_{CLI} \rangle$ of the $TOT(\Delta^3_{CLI})$ Category and its the abstract chronicle model $\mathcal{M}^3_{CLI}$;
  - the LoA $\mathcal{L}^3_{CPTE} = \langle \mathcal{S}^3_{CPTE}, \mathcal{T}^3_{CPTE} \rangle$ of the $TOT(\Delta^3_{CPTE})$ Category and its abstract chronicle model $\mathcal{M}^3_{CPTE}$;
  - the LoA $\mathcal{L}^3_{TYP} = \langle \mathcal{S}^3_{TYP}, \mathcal{T}^3_{TYP} \rangle$ of the $TOT(\Delta^3_{TYP})$ Category and its abstract chronicle model $\mathcal{M}^3_{TYP}$;
  - Abstract functors $TOT(\Delta^2_{CLI}) \rightarrow TOT(\Delta^3_{CLI})$, $TOT(\Delta^2_{CPTE}) \rightarrow TOT(\Delta^3_{CPTE})$ and $TOT(\Delta^2_{TYP}) \rightarrow TOT(\Delta^3_{TYP})$.

Thus, the gradient of abstraction of the observed problem is, according to definition 7.5, the collection $\mathcal{G}$ of LoAs:

$$\mathcal{G} = \{\mathcal{L}^1_{CLI}, \mathcal{L}^1_{CPTE}, \mathcal{L}^1_{TYP}, \mathcal{L}^2_{CLI}, \mathcal{L}^2_{CPTE}, \mathcal{L}^2_{TYP}, \mathcal{L}^3_{CLI}, \mathcal{L}^3_{CPTE}, \mathcal{L}^3_{TYP}\} \tag{8.83}$$

Such a GoA does present Abstractions functors linking two consecutive categories as well as behaviour models.

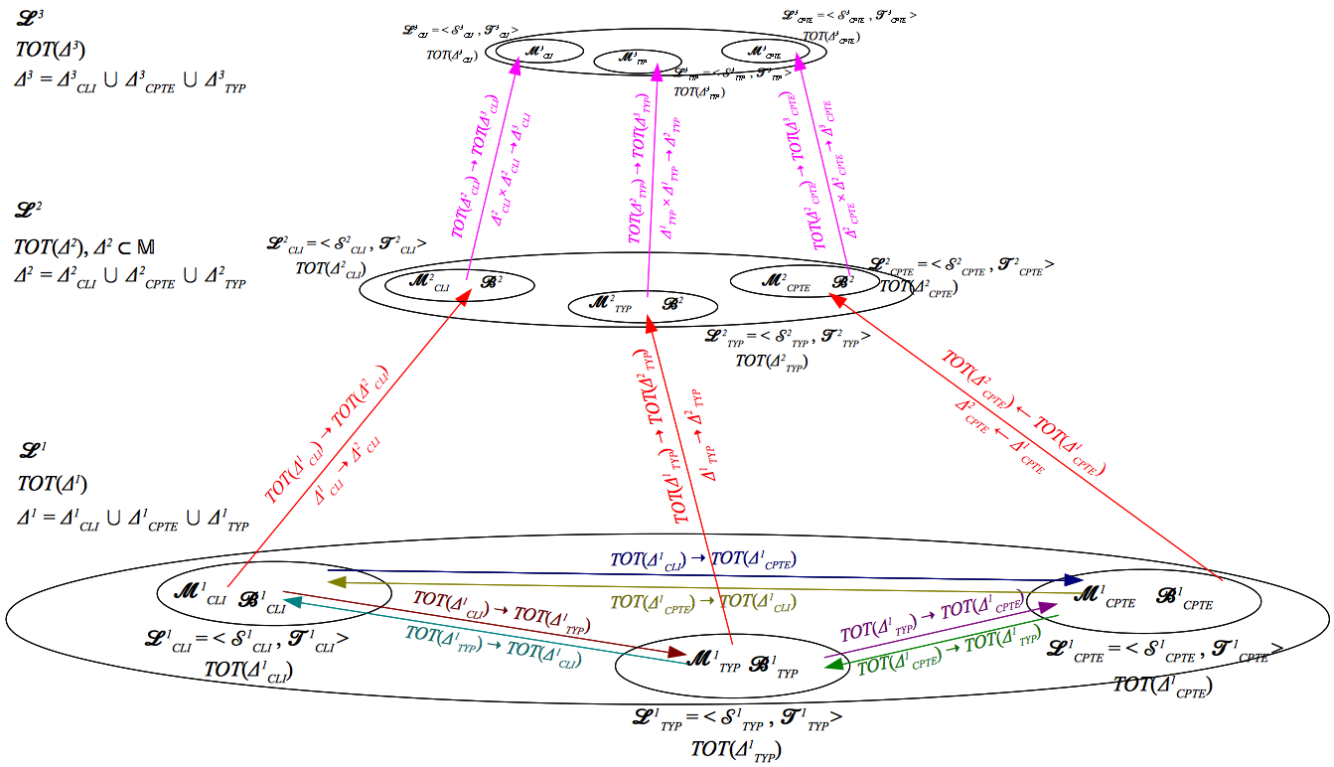

Figure 8.10: Illustration of the Gradient of Abstraction $\mathcal{G}$ of the Observed Problem

## 8.7 Interpretation Step

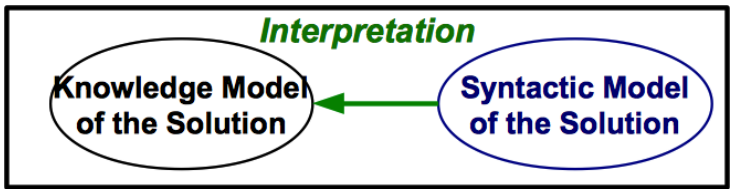


Figure 8.11: Interpretation step in the TOM4 Methodology

### 8.7.1 Reification Process

Let us recall that (see section 8.6.4), at the level $\mathcal{L}^3$ of abstraction, the superposition $\Omega^3$ contains the same timed observations of the form $O^3_{-m_k m_k}(t_k)$ whatever the category of clients, accounts or transaction types is considered. This superposition is composed of the 24 abstract timed observations listed in tables 8.8 to 8.13. Each of these 24 abstract timed observations $O^3_{-m_k m_k}(t_k)$ respects the internal fraud constraints of definition 8.2 at this level $\mathcal{L}^3$ of abstraction. Now, we need to know if the corresponding transactions at the concrete LoA $\mathcal{L}^1$ does respect those constraints or not. To this aim, we use the mapping made by Abstraction functors between categories of different level of abstractions.

Let us consider the $TOT(\Delta^3_{CLI})$ Category at the level $\mathcal{L}^3_{CLI}$ of abstraction.

Let us consider the abstract timed observation $O^3_{-5252}(t_{10003_7}) \equiv (-5252, t_{10003_7})$ of table 8.10. Thanks to Abstraction functor $TOT(\Delta^2_{CLI}) \rightarrow TOT(\Delta^3_{CLI})$, we have access to the following mapping (see 8.77):

$$\begin{array}{rcccc} TOT(\Delta^2_{CLI}) \rightarrow TOT(\Delta^3_{CLI}) & : & TOT(\Delta^2_{CLI}) & \rightarrow & TOT(\Delta^3_{CLI}) \\ & & ((-52, t_{1004_3}), (52, t_{1003_7})) & \mapsto & (-5252, t_{1003_7}) \end{array} \tag{8.84}$$

Thanks to Abstraction functor $TOT(\Delta^1_{CLI}) \rightarrow TOT(\Delta^2_{CLI})$, we have access to the following mappings (see 8.62):

$$\begin{array}{rcccc} TOT(\Delta^1_{CLI}) \rightarrow TOT(\Delta^2_{CLI}) & : & TOT(\Delta^1_{CLI}) & \rightarrow & TOT(\Delta^2_{CLI}) \\ & & O_{1004_3}(t_{1004_3}) \equiv (\delta_{1004_3}, t_{1004_3}) & \mapsto & O^2_{-52}(t_{1004_3}) \equiv (-52, t_{1004_3}) \\ & & O_{1003_7}(t_{1003_7}) \equiv (\delta_{1003_7}, t_{1003_7}) & \mapsto & O^2_{52}(t_{1003_7}) \equiv (52, t_{1003_7}) \end{array} \tag{8.85}$$

Let un now focus on constants $\delta_{1004_3}$ and $\delta_{1003_7}$:

$$\begin{array}{c} \delta_{1004_3} = \text{"}1004|2007|3004|258| - 2092,00\text{"} \\ \delta_{1003_7} = \text{"}1003|2001|3040|1826|2500,00\text{"} \end{array} \tag{8.86}$$

The transaction amount associated to constant $\delta_{1004_3}$, *-2 092,00*, is not the opposite of the transaction amount associated to constant $\delta_{1003_7}$, *2 500,00*. In this case, this transaction does not respect the internal fraud constraints and thus can not be selected to be a potential fraudulent transaction.

Now, let us consider the abstract timed observation $O^3_{-5252}(t_{1003_8}) \equiv (-5252, t_{1003_8})$ of table 8.10. Thanks to Abstraction functor $TOT(\Delta^2_{CLI}) \rightarrow TOT(\Delta^3_{CLI})$, we have access to the following mapping:

$$\begin{array}{rcccc} TOT(\Delta^2_{CLI}) \rightarrow TOT(\Delta^3_{CLI}) & : & TOT(\Delta^2_{CLI}) & \rightarrow & TOT(\Delta^3_{CLI}) \\ & & ((-52, t_{1004_4}), (52, t_{1003_8})) & \mapsto & (-5252, t_{1003_8}) \end{array} \tag{8.87}$$

Thanks to Abstraction functor $TOT(\Delta^1_{CLI}) \rightarrow TOT(\Delta^2_{CLI})$, we have access to the following mappings:

$$\begin{array}{rcccc} TOT(\Delta^1_{CLI}) \rightarrow TOT(\Delta^2_{CLI}) & : & TOT(\Delta^1_{CLI}) & \rightarrow & TOT(\Delta^2_{CLI}) \\ & & O_{1004_4}(t_{1004_4}) \equiv (\delta_{1004_4}, t_{1004_4}) & \mapsto & O^2_{-52}(t_{1004_4}) \equiv (-52, t_{1004_4}) \\ & & O_{1003_8}(t_{1003_8}) \equiv (\delta_{1003_8}, t_{1003_8}) & \mapsto & O^2_{52}(t_{1003_8}) \equiv (52, t_{1003_8}) \end{array} \tag{8.88}$$

Let un now focus on constants $\delta_{1004_4}$ and $\delta_{1003_8}$:

$$\begin{array}{c} \delta_{1004_4} = \text{"}1004|2020|3038|644| - 2000,00\text{"} \\ \delta_{1003_8} = \text{"}1003|2001|3039|2345|2000,00\text{"} \end{array} \tag{8.89}$$

The transaction amount associated to constant $\delta_{1004_4}$, *-2 000,00*, is the opposite of the transaction amount associated to constant $\delta_{1003_8}$, *2 000,00*. In this case, this transaction does respect the internal fraud constraints and thus can be selected to be a potential fraudulent transaction. Thus, binary sequence $(O_{1004_4}(t_{1004_4}) \equiv (\delta_{1004_4}, t_{1004_4}), O_{1003_8}(t_{1003_8}) \equiv (\delta_{1003_8}, t_{1003_8}))$ represents a potential fraudulent transaction of 2000,00€from client 1004 to bank manager.

These both examples show how works the reification process: doing so with the 24 abstract timed observations of the superposition $\Omega^3$, we can build, in fine, the set $\mathcal{S}^1_{CLI}$ containing all binary sequences of timed observations of the form $\big(O_{i_k}(t_{i_k}) \equiv (\delta_{i_k}, t_{i_k}), O_{j_{k+1}}(t_{j_{k+1}}) \equiv (\delta_{j_{k+1}}, t_{j_{k+1}})\big)$ where $i \neq 1003$ and $j = 1003$ respecting the internal fraud constraints. The binary sequences represent all potential fraudulent transactions from a client $i \neq 1003$ to the bank manager.

Thus, in fine, this set $\mathcal{S}^1_{CLI}$ is composed of the *eight* following binary sequences of timed observations:

$$\begin{array}{c}
\mathcal{S}^1_{CLI} = \\
\{(O_{1001_1}(t_{1001_1}) \equiv (\delta_{1001_1}, t_{1001_1}), O_{1003_1}(t_{1003_1}) \equiv (\delta_{1003_1}, t_{1003_1}))\,, \\
(O_{1001_2}(t_{1001_2}) \equiv (\delta_{1001_2}, t_{1001_2}), O_{1003_2}(t_{1003_2}) \equiv (\delta_{1003_2}, t_{1003_2}))\,, \\
(O_{1004_4}(t_{1004_4}) \equiv (\delta_{1004_4}, t_{1004_4}), O_{1003_8}(t_{1003_8}) \equiv (\delta_{1003_8}, t_{1003_8}))\,, \\
(O_{1004_9}(t_{1004_9}) \equiv (\delta_{1004_9}, t_{1004_9}), O_{1003_{13}}(t_{1003_{13}}) \equiv (\delta_{1003_{13}}, t_{1003_{13}}))\,, \\
(O_{1004_{10}}(t_{1004_{10}}) \equiv (\delta_{1004_{10}}, t_{1004_{10}}), O_{1003_{14}}(t_{1003_{14}}) \equiv (\delta_{1003_{14}}, t_{1003_{14}}))\,, \\
(O_{1004_{11}}(t_{1004_{11}}) \equiv (\delta_{1004_{11}}, t_{1004_{11}}), O_{1003_{15}}(t_{1003_{15}}) \equiv (\delta_{1003_{15}}, t_{1003_{15}}))\,, \\
(O_{1004_{13}}(t_{1004_{13}}) \equiv (\delta_{1004_{13}}, t_{1004_{13}}), O_{1003_{17}}(t_{1003_{17}}) \equiv (\delta_{1003_{17}}, t_{1003_{17}}))\,, \\
(O_{1003_{23}}(t_{1003_{23}}) \equiv (\delta_{1003_{23}}, t_{1003_{23}}), O_{1004_{17}}(t_{1004_{17}}) \equiv (\delta_{1004_{17}}, t_{1004_{17}}))\}
\end{array} \tag{8.90}$$

Where:

| **Constant** | **Timestamp** |
|---|---|
| $\delta_{1001_1} \equiv$ "1001\|2008\|3034\|1168\| − 5000, 00" | $t_{1001_1} =$ *2009-09-28 02:03:55* |
| $\delta_{1003_1} \equiv$ "1003\|2001\|3023\|2441\|5000, 00" | $t_{1003_1} =$ *2009-10-08 03:24:52* |
| $\delta_{1001_2} \equiv$ "1001\|2009\|3038\|1206\| − 50000, 00" | $t_{1001_2} =$ *2009-11-17 02:21:37* |
| $\delta_{1003_2} \equiv$ "1003\|2024\|3023\|2600\|50000, 00" | $t_{1003_2} =$ *2009-11-18 06:20:55* |
| $\delta_{1004_4} \equiv$ "1004\|2020\|3038\|644\| − 2000, 00" | $t_{1004_4} =$ *2009-09-10 06:58:34* |
| $\delta_{1003_8} \equiv$ "1003\|2001\|3039\|2345\|2000, 00" | $t_{1003_8} =$ *2009-09-16 02:33:07* |
| $\delta_{1004_9} \equiv$ "1004\|2010\|3038\|658\| − 5000, 00" | $t_{1004_9} =$ *2009-09-18 12:02:32* |
| $\delta_{1003_{13}} \equiv$ "1003\|2001\|3039\|2379\|5000, 00" | $t_{1003_{13}} =$ *2009-09-16 02:33:07* |
| $\delta_{1004_{10}} \equiv$ "1004\|2010\|3038\|662\| − 5000, 00" | $t_{1004_{10}} =$ *2009-09-21 04:31:59* |
| $\delta_{1003_{14}} \equiv$ "1003\|2001\|3023\|2441\|5000, 00" | $t_{1003_{14}} =$ *2009-10-08 03:24:52* |
| $\delta_{1004_{11}} \equiv$ "1004\|2007\|3004\|192\| − 17000, 00" | $t_{1004_{11}} =$ *2009-03-16 07:49:24* |
| $\delta_{1003_{15}} \equiv$ "1003\|2001\|3039\|1675\|17000, 00" | $t_{1003_{15}} =$ *2009-03-17 03:29:01* |
| $\delta_{1004_{13}} \equiv$ "1004\|2010\|3038\|756\| − 10000, 00" | $t_{1004_{13}} =$ *2009-11-09 09:39:00* |
| $\delta_{1003_{17}} \equiv$ "1003\|2024\|3023\|2591\|10000, 00" | $t_{1003_{17}} =$ *2009-11-13 04:53:17* |
| $\delta_{1003_{23}} \equiv$ "1003\|2024\|3024\|2726\| − 1000, 00" | $t_{1003_{23}} =$ *2009-12-22 04:47:11* |
| $\delta_{1004_{17}} \equiv$ "1004\|2007\|3023\|815\|1000, 00" | $t_{1004_{17}} =$ *2009-12-22 07:38:53* |

Table 8.20: Table of constants and timestamps corresponding to potentially fraudulent transactions in the client category

Any binary sequence of timed observations $\big(O_{i_k}(t_{i_k}), O_{j_{k+1}}(t_{j_{k+1}})\big)$ is an instance $r_{i_k j_{k+1}}(O_{i_k}(t_{i_k}), O_{j_{k+1}}(t_{j_{k+1}}))$ of a temporal binary relation $r_{i_k j_{k+1}}(O_{i_k}, O_{j_{k+1}}, \Delta\tau_{i_k j_{k+1}})$ contained in the abstract chronicle model $\mathcal{M}^1_{CLI}$. Thus, the set $\mathcal{S}^1_{CLI}$ is an instance of this chronicle model $\mathcal{M}^1_{CLI}$ and is then, by definition, the behaviour model of the observed process associated to the category of clients and respecting the internal fraud constraints of definition 8.2. We have seen (see table 8.7) that the abstract chronicle model $\mathcal{M}^1_{CLI}$ contains 3 379 879 temporal binary relations. Finding the 8 instances respecting the internal fraud constraints among these 3 379 879

temporal binary relations would have been a very hard work without operating a process of abstraction and reification of data. This demonstrates how such this abstraction and reification method is a powerful tool.

Let us consider the $TOT(\Delta^3_{CPTE})$ Category at the level $\mathcal{L}^3_{CLI}$ of abstraction. Thanks to Abstraction functors $TOT(\Delta^2_{CPTE}) \rightarrow TOT(\Delta^3_{CPTE})$ and $TOT(\Delta^1_{CPTE}) \rightarrow TOT(\Delta^2_{CPTE})$, another reification process can be operated leading to the following set $\mathcal{S}^1_{CPTE}$ of binary sequences of timed observations:

$$\begin{array}{c} \mathcal{S}^1_{CPTE} = \\ \{(O_{2008_1}(t_{2008_1}) \equiv (\delta_{2008_1}, t_{2008_1}), O_{2001_1}(t_{2001_1}) \equiv (\delta_{2001_1}, t_{2001_1}))\,, \\ (O_{2009_2}(t_{2009_2}) \equiv (\delta_{2009_2}, t_{2009_2}), O_{2024_2}(t_{2024_2}) \equiv (\delta_{2024_2}, t_{2024_2}))\,, \\ (O_{2020_4}(t_{2020_4}) \equiv (\delta_{2020_4}, t_{2020_4}), O_{2001_8}(t_{2001_8}) \equiv (\delta_{2001_8}, t_{2001_8}))\,, \\ (O_{2010_9}(t_{2010_9}) \equiv (\delta_{2010_9}, t_{2010_9}), O_{2001_{13}}(t_{2001_{13}}) \equiv (\delta_{2001_{13}}, t_{2001_{13}}))\,, \\ (O_{2010_{10}}(t_{2010_{10}}) \equiv (\delta_{2010_{10}}, t_{2010_{10}}), O_{2001_{14}}(t_{2001_{14}}) \equiv (\delta_{2001_{14}}, t_{2001_{14}}))\,, \\ (O_{2007_{11}}(t_{2007_{11}}) \equiv (\delta_{2007_{11}}, t_{2007_{11}}), O_{2001_{15}}(t_{2001_{15}}) \equiv (\delta_{2001_{15}}, t_{2001_{15}}))\,, \\ (O_{2010_{13}}(t_{2010_{13}}) \equiv (\delta_{2010_{13}}, t_{2010_{13}}), O_{2024_{17}}(t_{2024_{17}}) \equiv (\delta_{2024_{17}}, t_{2024_{17}}))\,, \\ (O_{2024_{23}}(t_{2024_{23}}) \equiv (\delta_{2024_{23}}, t_{2024_{23}}), O_{2007_{17}}(t_{2007_{17}}) \equiv (\delta_{2007_{17}}, t_{2007_{17}}))\} \end{array} \quad (8.91)$$

Where:

| **Constant** | **Timestamp** |
|---|---|
| $\delta_{2008_1} \equiv$ "1001\|2008\|3034\|1168\| − 5000, 00" | $t_{2008_1} =$ *2009-09-28 02:03:55* |
| $\delta_{2001_1} \equiv$ "1003\|2001\|3023\|2441\|5000, 00" | $t_{2001_1} =$ *2009-10-08 03:24:52* |
| $\delta_{2009_2} \equiv$ "1001\|2009\|3038\|1206\| − 50000, 00" | $t_{2009_2} =$ *2009-11-17 02:21:37* |
| $\delta_{2024_2} \equiv$ "1003\|2024\|3023\|2600\|50000, 00" | $t_{2024_2} =$ *2009-11-18 06:20:55* |
| $\delta_{2020_4} \equiv$ "1004\|2020\|3038\|644\| − 2000, 00" | $t_{2020_4} =$ *2009-09-10 06:58:34* |
| $\delta_{2001_8} \equiv$ "1003\|2001\|3039\|2345\|2000, 00" | $t_{2001_8} =$ *2009-09-16 02:33:07* |
| $\delta_{2010_9} \equiv$ "1004\|2010\|3038\|658\| − 5000, 00" | $t_{2010_9} =$ *2009-09-18 12:02:32* |
| $\delta_{2001_{13}} \equiv$ "1003\|2001\|3039\|2379\|5000, 00" | $t_{2001_{13}} =$ *2009-09-16 02:33:07* |
| $\delta_{2010_{10}} \equiv$ "1004\|2010\|3038\|662\| − 5000, 00" | $t_{2010_{10}} =$ *2009-09-21 04:31:59* |
| $\delta_{2001_{14}} \equiv$ "1003\|2001\|3023\|2441\|5000, 00" | $t_{2001_{14}} =$ *2009-10-08 03:24:52* |
| $\delta_{2007_{11}} \equiv$ "1004\|2007\|3004\|192\| − 17000, 00" | $t_{2007_{11}} =$ *2009-03-16 07:49:24* |
| $\delta_{2001_{15}} \equiv$ "1003\|2001\|3039\|1675\|17000, 00" | $t_{2001_{15}} =$ *2009-03-17 03:29:01* |
| $\delta_{2010_{13}} \equiv$ "1004\|2010\|3038\|756\| − 10000, 00" | $t_{2010_{13}} =$ *2009-11-09 09:39:00* |
| $\delta_{2001_{17}} \equiv$ "1003\|2024\|3023\|2591\|10000, 00" | $t_{2024_{17}} =$ *2009-11-13 04:53:17* |
| $\delta_{2024_{23}} \equiv$ "1003\|2024\|3024\|2726\| − 1000, 00" | $t_{2024_{23}} =$ *2009-12-22 04:47:11* |
| $\delta_{2007_{17}} \equiv$ "1004\|2007\|3023\|815\|1000, 00" | $t_{2007_{17}} =$ *2009-12-22 07:38:53* |

Table 8.21: Table of constants and timestamps corresponding to potentially fraudulent transactions in the account category

Again the abstraction and reification process associated to the category of accounts allows to easily find the 8 instances of temporal binary relations respecting the fraud constraints among the 14 635 861 temporal binary relations composing the abstract chronicle model $mathcalM^1_{CPTE}$.

Let us consider the $TOT(\Delta^3_{TYP})$ Category at the level $\mathcal{L}^3_{CLI}$ of abstraction. Thanks to Abstraction functors $TOT(\Delta^2_{TYP}) \rightarrow TOT(\Delta^3_{TYP})$ and $TOT(\Delta^1_{TYP}) \rightarrow TOT(\Delta^2_{TYP})$, another

reification process can be operated leading to the following set $\mathcal{S}^1_{TYP}$ of binary sequences of timed observations:

$$\begin{gathered}\mathcal{S}^1_{TYP} = \\ \{(O_{3034_1}(t_{3034_1}) \equiv (\delta_{3034_1}, t_{3034_1}), O_{3023_1}(t_{3023_1}) \equiv (\delta_{3023_1}, t_{3023_1}))\,, \\ (O_{3038_2}(t_{3038_2}) \equiv (\delta_{3038_2}, t_{3038_2}), O_{3023_2}(t_{3023_2}) \equiv (\delta_{3023_2}, t_{3023_2}))\,, \\ (O_{3038_4}(t_{3038_4}) \equiv (\delta_{3038_4}, t_{3038_4}), O_{3039_8}(t_{3039_8}) \equiv (\delta_{3039_8}, t_{3039_8}))\,, \\ (O_{3038_9}(t_{3038_9}) \equiv (\delta_{3038_9}, t_{3038_9}), O_{3039_{13}}(t_{3039_{13}}) \equiv (\delta_{3039_{13}}, t_{3039_{13}}))\,, \\ (O_{3038_{10}}(t_{3038_{10}}) \equiv (\delta_{3038_{10}}, t_{3038_{10}}), O_{3023_{14}}(t_{3023_{14}}) \equiv (\delta_{3023_{14}}, t_{3023_{14}}))\,, \\ (O_{3004_{11}}(t_{3004_{11}}) \equiv (\delta_{3004_{11}}, t_{3004_{11}}), O_{3039_{15}}(t_{3039_{15}}) \equiv (\delta_{3039_{15}}, t_{3039_{15}}))\,, \\ (O_{3038_{13}}(t_{3038_{13}}) \equiv (\delta_{3038_{13}}, t_{3038_{13}}), O_{3023_{17}}(t_{3023_{17}}) \equiv (\delta_{3023_{17}}, t_{3023_{17}}))\,, \\ (O_{3024_{23}}(t_{3024_{23}}) \equiv (\delta_{3024_{23}}, t_{3024_{23}}), O_{3023_{17}}(t_{3023_{17}}) \equiv (\delta_{3023_{17}}, t_{3023_{17}}))\}\end{gathered} \tag{8.92}$$

Where:

| **Constant** | **Timestamp** |
|---|---|
| $\delta_{3034_1} \equiv$ "1001\|2008\|3034\|1168\| − 5000, 00" | $t_{3034_1} =$ *2009-09-28 02:03:55* |
| $\delta_{3023_1} \equiv$ "1003\|2001\|3023\|2441\|5000, 00" | $t_{3023_1} =$ *2009-10-08 03:24:52* |
| $\delta_{3038_2} \equiv$ "1001\|2009\|3038\|1206\| − 50000, 00" | $t_{3038_2} =$ *2009-11-17 02:21:37* |
| $\delta_{3023_2} \equiv$ "1003\|2024\|3023\|2600\|50000, 00" | $t_{3023_2} =$ *2009-11-18 06:20:55* |
| $\delta_{3038_4} \equiv$ "1004\|2020\|3038\|644\| − 2000, 00" | $t_{3038_4} =$ *2009-09-10 06:58:34* |
| $\delta_{3039_8} \equiv$ "1003\|2001\|3039\|2345\|2000, 00" | $t_{3039_8} =$ *2009-09-16 02:33:07* |
| $\delta_{3038_9} \equiv$ "1004\|2010\|3038\|658\| − 5000, 00" | $t_{3038_9} =$ *2009-09-18 12:02:32* |
| $\delta_{3039_{13}} \equiv$ "1003\|2001\|3039\|2379\|5000, 00" | $t_{3039_{13}} =$ *2009-09-16 02:33:07* |
| $\delta_{3038_{10}} \equiv$ "1004\|2010\|3038\|662\| − 5000, 00" | $t_{3038_{10}} =$ *2009-09-21 04:31:59* |
| $\delta_{3023_{14}} \equiv$ "1003\|2001\|3023\|2441\|5000, 00" | $t_{3023_{14}} =$ *2009-10-08 03:24:52* |
| $\delta_{3004_{11}} \equiv$ "1004\|2007\|3004\|192\| − 17000, 00" | $t_{3004_{11}} =$ *2009-03-16 07:49:24* |
| $\delta_{3039_{15}} \equiv$ "1003\|2001\|3039\|1675\|17000, 00" | $t_{3039_{15}} =$ *2009-03-17 03:29:01* |
| $\delta_{3038_{13}} \equiv$ "1004\|2010\|3038\|756\| − 10000, 00" | $t_{3038_{13}} =$ *2009-11-09 09:39:00* |
| $\delta_{3039_{15}} \equiv$ "1003\|2024\|3023\|2591\|10000, 00" | $t_{3023_{17}} =$ *2009-11-13 04:53:17* |
| $\delta_{3024_{23}} \equiv$ "1003\|2024\|3024\|2726\| − 1000, 00" | $t_{3024_{23}} =$ *2009-12-22 04:47:11* |
| $\delta_{3023_{17}} \equiv$ "1004\|2007\|3023\|815\|1000, 00" | $t_{3023_{17}} =$ *2009-12-22 07:38:53* |

Table 8.22: Table of constants and timestamps corresponding to potentially fraudulent transactions in the transaction type category

Again the abstraction and reification process associated to the category of transaction types allows to easily find the 8 instances of temporal binary relations respecting the fraud constraints among the 12 630 799 temporal binary relations composing the abstract chronicle model $\mathcal{M}^1_{TYP}$.

There exists another way to get the sets $\mathcal{S}^1_{CPTE}$ and $\mathcal{S}^1_{TYP}$. Given the Modeling functor $TOT(\Delta^1_{CLI}) \rightarrow TOT(\Delta^1_{CPTE})$ defined at the level $\mathcal{L}^1$ of abstraction, we can build the set from the set $\mathcal{S}^1_{CPTE}$ from the set $\mathcal{S}^1_{CLI}$. For example, thanks to this Modeling functor, we have access to the following mappings:

$$\begin{array}{lcccc} TOT(\Delta^1_{CLI}) \rightarrow TOT(\Delta^1_{CPTE}) & : & TOT(\Delta^1_{CLI}) & \rightarrow & TOT(\Delta^1_{CPTE}) \\ & & O_{1001_1}(t_{1001_1}) \equiv (\delta_{1001_1}, t_{1001_1}) & \mapsto & O_{2008_1}(t_{2008_1}) \equiv (\delta_{2008_1}, t_{2008_1}) \\ & & O_{1003_1}(t_{1003_1}) \equiv (\delta_{1003_1}, t_{1003_1}) & \mapsto & O_{2001_1}(t_{2001_1}) \equiv (\delta_{2001_1}, t_{2001_1}) \end{array} \tag{8.93}$$

This mappings allow to build the binary sequence $(O_{2008_1}(t_{2008_1}) \equiv (\delta_{2008_1}, t_{2008_1}), O_{2001_1}(t_{2001_1}) \equiv (\delta_{2001_1}, t_{2001_1}))$ which is an element of $\mathcal{S}^1_{CPTE}$. Doing so with all the timed observations of $\mathcal{S}^1_{CPTE}$, we can build the whole set $\mathcal{S}^1_{CPTE}$. And, with the same reasoning, from Modeling functor $TOT(\Delta^1_{CLI}) \rightarrow TOT(\Delta^1_{TYP})$, we can build the set $\mathcal{S}^1_{TYP}$.

Thus, were determined sets $\mathcal{S}^1_{CLI}$, $\mathcal{S}^1_{CPTE}$ and $\mathcal{S}^1_{TYP}$ containing the eight binary sequences of timed observations at the level $\mathcal{L}^1$ of abstraction representing the potential fraudulent transactions in respectively the categories of clients, accounts and transaction types. Now, we need to provide a representation of this potential fraudulent transactions that can be *interpretable* by humans. This this the aim of the next section.

### 8.7.2 Knowledge Model of the Solution

To represent the behaviour model $\mathcal{B} = \{r_{i_k j_{k+1}}(O_{i_k}(t_{i_k}), O_{j_{k+1}}(t_{j_{k+1}}))\}$, being an instance of an abstract chronicle model $\mathcal{M} = \{r_{i_k j_{k+1}}(O_{i_k}, O_{j_{k+1}}, \Delta\tau_{i_k j_{k+1}})\}$ of a given observed process, we use an adaptation of a *Kripke structure* to the TOT framework.

A Kripke structure is a usual mathematical tool used to provide an interpretation of Temporal Logic formulas [CGP99]. To this aim, a Kripke structure represents the behavior of a dynamic process with a special kind of finite state machine where the nodes represent the *reachable states* $s_i$ of the machine and the edges represent state transitions:

**Definition 8.3** *Kripke Structure [CGP99]*
*Let AP be a set of atomic propositions, constants and predicate symbols of the first order predicate calculus.*
*A Kripke structure defined over AP is a 4-tuple K = (S, I, R, L) consisting of:*

- *a finite set* $S = \{s_i, i \in [1, n_S], n_S \in \mathbb{N}^*\}$ *of states* $s_i$*;*
- *a set* $I \subseteq S$ *of initial states;*
- *a transition relation R defined over* $S \times S$*:*

$$\begin{array}{rcccc} R & : & S & \rightarrow & S \\ & & s_i & \mapsto & s_j \end{array} \tag{8.94}$$

*such that R is left-total:*

$$\forall s_i \in S,\ \exists s_j \in S \text{ such that } (s_i, s_j) \in R \tag{8.95}$$

- *a labeling function L defined over* $S \times 2^{AP}$*, mapping each state* $s_i$ *to a set* $L(s_i)$ *of logical properties that holds in this state:*

$$\begin{array}{rcccc} L & : & S & \rightarrow & 2^{AP} \\ & & s_i & \mapsto & L(s_i) \end{array} \tag{8.96}$$

The *interpretation* of a given state $s_i$ is then provided by $L$ under the form of the set $L(s_i)$.

This definition leads to the notion of *path* in a Kripke structure $K$:

**Definition 8.4** *Path in a Kripke Structure*
*A path in a Kripke structure $K = (S, I, R, L)$ is a sequence of states $P(K) = (s_i)_{s_i \in S}$ such that for each $i > 0$, $R(s_i, s_{i+1})$ holds.*

This definition leads to the notion of *word* of a path $W(K)$:

**Definition 8.5** *Word of a Path*
*A word $W(K)$ of a path $P(K)$ in a Kripke structure $K = (S, I, R, L)$ is a sequence of sets of atomic propositions $W(K) = (L(s_i))_{s_i \in S}$.*

In a Kripke structure a path is a sequence of *states* while in the TOT framework a path is a sequence of *temporal binary relations*. This implies that the notion of *states* $s_i$ in a Kripke structure maps the notion of *temporal binary relations* $r_{i_k j_{k+1}}(O_{i_k}, O_{j_{k+1}}, \Delta\tau_{i_k j_{k+1}})$ between two observation classes $O_{i_k}$ and $O_{j_{k+1}}$ in the TOT framework. The *temporal and logical constraints* associated with the instance $r_{i_k j_{k+1}}(O_{i_k}(t_{i_k}), O_{j_{k+1}}(t_{j_{k+1}}))$ of a temporal binary relation $r_{i_k j_{k+1}}(O_{i_k}, O_{j_{k+1}}, \Delta\tau_{i_k j_{k+1}})$ play the same role than the logical properties $L(s_i)$ that hold in the state $s_i$. In a Kripke structe a word being a sequence of labels, this implies that the notion of word in the TOT framework is then a sequence of instances of temporal binary relations, that is to say, maps the notion of behaviour model.

Table 8.23 sums up the correspondances between a Kripke structure elements and TOT elements.

| **Kripke Structure** | **TOT framework** |
|---|---|
| State $s_i$ | Temporal binary relation $r_{i_k j_{k+1}}(O_{i_k}, O_{j_{k+1}}, \Delta\tau_{i_k j_{k+1}})$ |
| Label $L(s_i)$ | Instance of a temporal binary relation $r_{i_k j_{k+1}}(O_{i_k}(t_{i_k}), O_{j_{k+1}}(t_{j_{k+1}}))$ |
| Path $P(K) = (s_i)_{s_i \in S}$ | Abstract chronicle model $\mathcal{M} = \{r_{i_k j_{k+1}}(O_{i_k}, O_{j_{k+1}}, \Delta\tau_{i_k j_{k+1}})\}$ |
| Word $W(K) = (L(s_i))_{s_i \in S}$ | Behaviour model $\mathcal{B} = \{r_{i_k j_{k+1}}(O_{i_k}(t_{i_k}), O_{j_{k+1}}(t_{j_{k+1}}))\}$ |

Table 8.23: Adaptation of a Kripke structure in the TOT Framework

Now it is possible to represent the potential fraudulent transactions in the category of clients.

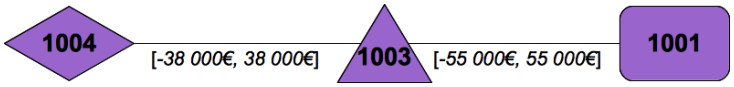


Figure 8.12: Kripke structure representing the fraud scheme under the point of view of clients

This figure represents movements of money between client ids 1001 and 1002 with the manager of the bank (client id 1003). Each shape represents a client id and is associated with an observation class. The diamond labeled 1004 represents the client id 1004 and is associated with the observation class denoted $O_{1004} = \{x_{1004}, \Delta^1_{1004}\}$. The triangle labeled 1003 represents the bank manager and is associated with the observation class denoted $O_{1003} = \{x_{1003}, \Delta^1_{1003}\}$. The rectangle labeled 1001 represents client id 1001 and is associated with the observation class denoted $O_{1001} = \{x_{1001}, \Delta^1_{1001}\}$.

The line linking client id 1004 with the bank manager represents all instances of the form $r_{1004_k 1003_{k+1}}(O_{1004_k}(t_{1004_k}), O_{1003_{k+1}}(t_{1003_{k+1}}))$ and $r_{1003_k 1004_{k+1}}(O_{1003_k}(t_{1003_k}), O_{1004_{k+1}}(t_{1004_{k+1}}))$ contained in the behaviour model $\mathcal{B}^1_{CLI}$. The temporal and logical constraints are represented

with a label of the form of an interval [-38 000€,38 000€] meaning that 38 000€ have been transfered from client 1004 to the bank manager in a sliding period of 30 days.

The line linking client id 1001 with the bank manager represents all instances of the form $r_{1001_k 1003_{k+1}}(O_{1001_k}(t_{1001_k}), O_{1003_{k+1}}(t_{1003_{k+1}}))$ and $r_{1003_k 1001_{k+1}}(O_{1003_k}(t_{1003_k}), O_{1001_{k+1}}(t_{1001_{k+1}}))$ contained in the behaviour model $\mathcal{B}^1_{CLI}$. Here, temporal and logical constraints mean that 55 000€ have been transfered from client 1001 to the bank manager in a sliding period of 30 days.

Thus, figure 8.12 allows, in an eye blink, to understand that the bank manager have potentially stolen 93 000€ to his clients.

The point of view of accounts is also interesting because it allows to understand which accounts have been used by the manager to bring these 93 000€ into his own accounts. Figure 8.13 provides the Kripke fraud scheme under the point of view to account ids.

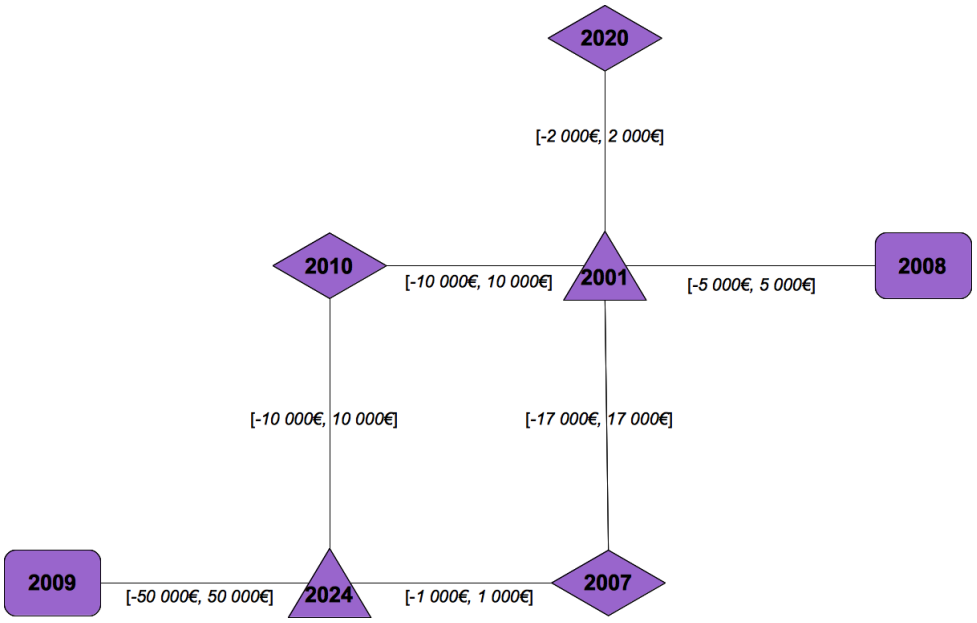


Figure 8.13: Kripke structure representing the fraud scheme under the point of view of accounts

For example, we know from this scheme that the 55 000€ transfered from client 1001 correspond to a transfert of 5 000€ from account 2008 (which belongs to client 1001) to account 2001 (which belongs to bank manager) and to another transfert of 50 000€ from account 2009 (which belongs to client 1001) to account 2024 (which belongs to bank manager).

Finally, the fraud scheme according to the point of view of transaction types can also be provided (see figure 8.14):

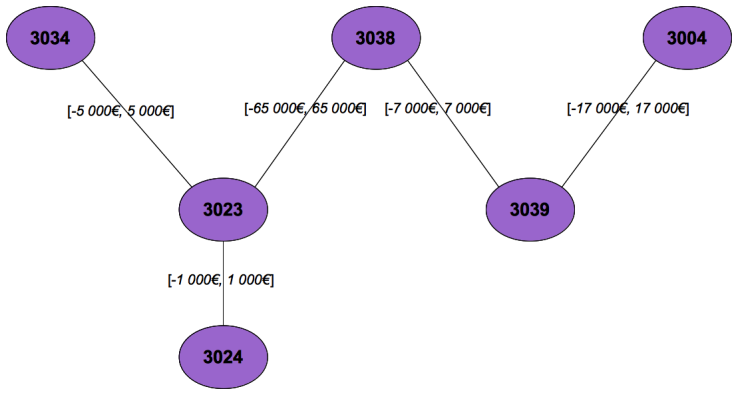


Figure 8.14: Kripke structure representing the fraud scheme under the point of view of transaction types

Again fraud scheme 8.13 under the point of view of accounts and fraud scheme 8.14 under the point of view of transaction types can be built thanks to Modeling functors $TOT(\Delta^1_{CLI}) \rightarrow TOT(\Delta^1_{CPTE})$ and $TOT(\Delta^1_{CLI}) \rightarrow TOT(\Delta^1_{TYP})$. As said in chapter 7, section 7.3.1, a Modeling functor is a tool allowing to change the point of view under which the system is observed.

## 8.8 TOM4FFS Algorithm

This section provides a brief description about how program TOM4FFS works.

Algorithm Timed Observations Mining for Fraud Fighting System (TOM4FFS) has been implemented in Java langage.

Inputs of the TOM4FFS program are the transaction databases and the relations between clients ids, account ids and transaction type ids.

TOM4FFS builds then:

- elements characterizing the first LoA:
  - observation classes;
  - unary observers for the categories of clients, accounts and transaction types: these unary observers all work in parallel, observing their own piecewise function and writing timed observations of the form $(\delta_k, t_k)$;
  - Modeling functors between these categories: there are built thanks to relations given as inputs.
- elements characterizing the second LoA:
  - observation classes;
  - unary observers for the categories of clients, accounts and transaction types: these unary observers all work in parallel, observing their own piecewise function and writing abstract timed observations of the form $(m_k, t_k)$;
  - Abstraction functors between categories of first LoA to second LoA.
- elements characterizing the third LoA:

- observation classes;
- abstract binary observers for the categories of clients, accounts and transaction types: these abstract unary observers all work in parallel, observing binary sequences of timed observations $((m_k, t_k), (-m_k, t_{k+1}))$ and writing abstract timed observations of the form $(-m_k m_k, t_{k+1})$;
- Abstraction functors between categories of second LoA to third LoA.

Each element built by TOM4FFS is given a unique name thanks to the syntactic arithmetization introduced in 7.4 allowing TOM4FFS to deal only with $TOT(\mathbb{Z})$ categories. TOM4FFS implements the reification process each time the internal fraud constraints are respected at the third LoA: TOMFFS builds then the sets of potential fraudulent transactions in real time. Once all sets of potential fraudulent transactions are built, TOM4FFS displays the fraud schemes introduced in the preceding section.

TOM4FFS is then an on-line and real time program, being able to handle more than 4 billions transactions a day, allowing to reduce the complexity of the problem from $O(n^2)$ to $O(n)$, and having the ability of representing the fraud technique at simultaneously three levels of abstraction: client, account and transaction type levels.

## 8.9 Conclusion

Considering properties 31 and 35 of chapter 7, considering a $TOT^i(\mathbb{Z})$ Category at the level $\mathcal{L}^i$ of abstraction, the TOM4A Methodology is based on the following equivalences:

- Representation $\leftrightarrow$ Abstraction process $\leftrightarrow$ Sum of objects in the $TOT^i(\mathbb{Z})$ Category;
- Interpretation $\leftrightarrow$ Reification process $\leftrightarrow$ Product of objects in the $TOT^i(\mathbb{Z})$ Category.

This chapter shows that the TOM4A Methodology, defined as a $TOT(\mathbb{Z})$ based recursive Abstraction-Reification reasoning, can be implemented in a basic computer. In practice, Modeling functors and Abstraction functors have been implemented in the Tom4K Java platform as the TOM4FFS Algorithm.

If not general, such an approach is nevertheless sufficient for many various concrete problems as it has been done for the diagnosis system Sachem of the Arcelor-Mittal group [LeG04], [LeG06], the modeling of the StMicroelectronics manufacturing road [BLGB06], [PVP10] or the modeling of human activities [PGAP12], [PLPA11], [LBP15] for examples. These applications show that the main difficulty is to define the first LoA, the LoA and to provide a human interpretation of the abstract constants. Up to our knowledge, it is the first time that such a successive abstraction and reification process is concretely implemented in a computer.

As a final note and in order to kill all suspense, let us say that the fraud schemes presented in this chapter have all been validated by the concerned French bank: all fraud transactions that were only considered as potential are actually real!

CHAPTER 9

# Conclusion

## 9.1 Synthesis

This document concerns the development of a theoretical mathematical framework to provide a technology able to manage some of the problematics of the *Big Data Flows* domain, which is characterized by (i) *temporal properties* and (ii) *high number of dimensions* of the informational space where it is defined.

More precisely, we propose to combine the Newell ontological point of view [New81] and the Floridi epistemological point of view [Flo08] about *abstraction* to build tools that *transform models by abstraction* by the mean of an adequate set of *functors* according to the Category theory of Samuel Eilenberg and Saunders Mac Lane [ML71].

This kind of tools is called *Problem Solving Methods* (PSM) in the field of Knowledge Engineering. Such a PSM relies on a *real time abstraction reasoning* process to produce, on line, two dual effects:

1. *Decreasing* the flow data amount, that is to say, coming back to a data flow *normal level* that a common computer can manage.

2. *Increasing* the semantic richness carried by the information flow, that is to say, resuming a lot of *semantically poor* data into an equivalent but *richer* one.

The price to pay for such a *semantic enrichment* of information is the loss of *syntactic* data, that is to say, accepting a *forgetting* process. This justifies the use of the Category Theory to control the *oversight phenomenon*.

## 9.2 Contributions

Chapter 2 of this document introduces these theories to conclude on the feasibility of such a PSM when two conditions are satisfied:

1. Only *models* are concerned by the PSM and

2. The *oversight phenomenon* is controlled rationally with the notion of functor of the Category theory.

Therefore, to this aim, it is necessary to provide a mathematical core to the theories of Newel and Floridi: this is the role of the TOT, the Timed Observations Theories of Le Goc [LeG06], introduced in the chapter 3.

Chapter 4 formally introduces theoretical concepts used in the TOT framework:

- neutral observation, observation of a timestamp, observation of a constant;
- deduction of a predicate assignation from two predicate assignations: the application of the Modus Ponens to a given rule is the basis of such an assignation and is the origin of the abstraction process in the TOT framework;
- addition operation of two timed observations under temporal constraints: some algebraic properties of such an operation are demonstrated (existence of a neutral element, commutativity and associativity). This operation allows the construction of any algebraic structure.
- composition of observers: extends the notion of timed observations addition to two sequences of timed observations. This composition is essential when considering TOT Categories building since it plays the role of composition of objects in a category as well as there exists a composition of morphisms in a such a category.

These concepts thus defined allows the building of Abstract Unary Observers and Abstract Binary Observers. Abstract Binary Observers play a vital role in the process of abstraction and are key elements in the construction of categories since they are category objects.

Chapter 5 formally presents the abstraction process in the TOT framework. We demonstrate here that a superposition of $m$ sequences of timed observations can be modeled thanks to a collection of matrix of Abstract Binary Observers. This allows to build the algebraic structure and observable space associated to any observed process. From these algebraic and observable elements, is built the abstract chronicle model of this observed process whose behaviour model is its instance. Definitions of level of abstraction (LoA) and gradient of abstraction (GoA) are also given demonstrating their coherences with Floridi's notions introduced in chapter 2. The existence of a behaviour model generates an abstraction process allowing to affirm that timed observation plays the role of *paradigm* in the TOT framework: this particular property allows to repeat the abstraction process in a recursive way.

Chapter 6 highlights the existence of a homomorphism linking algebraic structure associated to Dirac's comb mathematical framework to algebraic structure associated to TOT mathematical framework demonstrating that TOT concept of *Unary Observer* plays the role of sampler in TOT framework. TOT sampler being based on the TOT *spatial discretization principle*, a *Unary Observer* is the *first* brick of the theoretical construction of an abstraction based PSM. This first brick is our first contribution. It is to note that appendix A proposes a concrete example of TOT *Unary Observer* that has been used in different applications today. This appendix has been joined to this document aiming at showing how to build a concrete TOT *Unary Observer*.

Chapter 7 is then dedicated to build an adequate category, the $TOT(\mathbb{Z})$ Category, that is used to formulate the proposed abstraction based PSM. This construction is our second contribution. The interest of the $TOT(\mathbb{Z})$ Category is to allow the building of adequate *Modeling* and *Abstraction* functors, which are the basis of the proposed abstraction reasoning. This chapter shows that the $TOT(\mathbb{Z})$ Category allows the definition of an *operational* notion of level of abstraction that is compatible with the theories of Newell and Floridi.

Finally, the required mathematical tools having been defined, it is possible to define a *specific* abstraction based PSM, which constitutes our third and last contribution: the AR-PSM (Recursive Abstraction-Reification Based Problem Solving Method) called TOM4A (Timed Observations Methodology for Abstraction, cf. chapter 8).

The TOM4A Methodology is based on the following equivalences at a Level of Abstraction $\mathcal{L}^i$:

- Representation $\leftrightarrow$ Abstraction process $\leftrightarrow$ Sum of objects in the $TOT^i(\mathbb{Z})$ Category;
- Interpretation $\leftrightarrow$ Reification process $\leftrightarrow$ Product of objects in the $TOT^i(\mathbb{Z})$ Category.

It has been shown that the Abstraction process reduces to around 98% the size of data to be analysed.

A complete application of such a TOM4A Methodology has been provided for detecting and modeling the complex problem of internal frauds in the banking industry. Another application of the AR-PSM principles can be found in the appendix B.

Another good example is the Sachem knowledge based system of the Arcelor Mittal Group, validated in October 1996 by the Chairman and Chief Executive Officer of Arcelor Group, M. F. Mer. The publication [LeG04] and the Sachem's US patten [LBD+03] or Sachem's European patent [LLB+08] show that the design of the Sachem system is based on a PSM such as the TOM4A AR-PSM: Sachem's *real time abstraction reasoning* process reduces to 50.000 the size of the input data flow, a vector of above 1.450 real numbers each minute. And the economical success of this system is sufficient to attest the concrete efficiency of such an approach.

## 9.3 Perspectives

Our contribution is the first mathematical formalization of the Newell and Floridi theories.

The TOT concerned dynamic processes and its applications to various domains generally provide interesting results. This explains our insistence about the fact that *all* the propositions formulated in this document *have been implemented* in the Tom4K Java platform, which has been, notably, used to solve the fraud detection in the banking transaction domain reported in [VLGBR16] and in chapter 8.

As a consequence, one of the most important prolongation of our work is to apply TOM4A to much more numerous and various kind of problems than those cited here. Indeed, this is the only way to assess the concrete value of such an approach. And obviously, finding new problematics to evaluate TOM4A is a very difficult task because, there is no comparable works in the scientific world in general, and Artificial Intelligence community in particular.

Currently, Researchers in Artificial Intelligence (AI) can fall into two broad categories: the Artificial Neural Network and Bayesian group, representing the numerical approach of AI, and the formal logic group, representing the symbolic approach of AI. However, the TOT notion of *timed observation* has been designed to combine both numerical calculations and symbolic computation in a unique and coherent approach of automatic learning and modeling of dynamic processes. With such a perspective in mind, the closest work comes Qualitative Physics concept or Continuous Dynamic Processes theory. And the fact is that currently, there are not many AI researchers working in these areas right now.

On the other hand, the Discrete Event System (DES) community proposes similar concepts than those of *Timed Observation* and *Discernible State* of the TOT. But, up to our opinion, the notions of *state* and *discrete event* in a *Finite State Machine* are not adequate to build levels of abstraction according to Newell's or Floridi's point of views because it seems impossible to merge together (or to add) states and discrete events together.

Of course, the aim of this document is not to demonstrate this point. We believe then that the second important prolongation of this work is to clearly position TOM4A in the Artificial Intelligence, the DES and the Continuous Dynamic Systems scientific domains.

An important consequence of our work concerns the Timed Observations Theory itself. The notion of $TOT(\mathbb{Z})$ Category and Level of Abstraction $\mathcal{L}^i$ entail the need for a new formalization of the whole theory in order to provide a clear and unified description of it. This constitutes an important work that needs to be done.

To finish, let us cite *Françoise DOUAY-SOUBLIN* [DS87] when exposing the conception of analogies according to Aristote:

- *Aristote, while insisting on the correlations between series that entails an identical structure (part-to-all relationship, from symbol to symbolized, of temporal order), actually makes the analogical structure based on the prior existence of series already constituted: body parts, animal species, the list of emblems, the pantheon of the gods, the hours of the day, the ages of life.*

- *Conventional or "natural", these series have put or still put in order the symbolic universe, and the history of symbolic forms, especially pre-scientific (Curtius, De Bruynhe, Yates) has well highlighted the extraordinary analog potential the quadri-, tri-, and bi-categorizations: the square, the triangle, and the couple, of which one of the most remarkable effects is to organize in quadrennial, tri-, or bi-hierarchical quadri-, tri-, or bi-polar figures the setting in comparison with the long series: climates, stars, metals, diseases...*

- *In the form and efficiency of a CWFA (cognitively well-formed analogy), the importance of categorization and that of the predicative relation are somehow inversely proportional: if the series and their hierarchies are in place, the predicate can be elided; this is what happens in all the examples of analogy in the strict sense chosen by Aristotle, which can be reduced to the formula "A is for B that C it is for D" because they operate in strongly connected series.*

- *This is very clear in that of the French authors who has the most and best cultivated these kinds of proportional analogies, the "La Rochefoucauld of the Maxims": only those that simultaneously implement strictly binary relations of inclusion or order (quality, high degree) and bi-categorizations well established in the universe of reference, in this case morality (the body and the spirit, the merit and the beauty, the vice and the virtue, ...) assume the so-called canonical form, with elided prediction:*

  - *Good grace is for the body what common sense is for mind.*
  - *The elevation is for the merit what the adornment is for the beautiful people.*
  - *Wisdom is for the soul what health is for the body.*

Reasoning by analogy is the basis of human understanding and this reasoning is based on the capacity for abstraction. Indeed, the formula *"A is for B what C it is for D"* can only work under two necessary conditions:

1. Concepts A and C, as B and D, must belong to the same abstraction $\mathcal{A}(X)$ so that the relations $\mathcal{A}(A) \leftrightarrow \mathcal{A}(C)$ and $\mathcal{A}(B) \leftrightarrow \mathcal{A}(D)$ can be possible;
2. Clocks related to the instances $\mathcal{A}(A)$ and $\mathcal{A}(C)$, as well as those of $\mathcal{A}(B)$ and $\mathcal{A}(D)$, must be sufficiently correlated.

Therefore, a promising work perspective would be to confirm that any analogy reasoning is underlied by an abstraction reasoning.

# Bibliography


[Ahb10a] A. Ahbad. *Contribution to Bayesian Networks Learning from Timed Data to Diagnose Continuous Dynamic Process.* PhD thesis, Aix-Marseille University, Marseille (France), June 2010.

[Ahb10b] A. Ahbad. *Contribution to Bayesian Networks Learning from Timed Data to Diagnose Continuous Dynamic Process.* PhD thesis, Aix-Marseille University, Marseille (France), June 2010.

[AL01] M Alavi and DE Leidner. Knowledge management and knowledge management systems. *Conceptual Foundations and Research Issues.*, 2001.

[Ben38] F. Benford. The law of anomalous numbers. *Proceedings of the American Philosophical Society*, 1938.

[Ben10] N. Benayadi. *Contribution à la découverte de connaissances à partir de données datées.* PhD thesis, Univertité Paul Cezanne Aix-Marseille III, 2010.

[BLGB06] N. Benayadi, M. Le Goc, and P. Bouché. Discovering manufacturing process from timed data: the bjt4r algorithm. In *Second International IEEE Workshop on Mining Complex Data (MCD'06)*, IEEE International Conference on Data Mining (ICDM'06). IEEE, December 2006.

[BLGC08] P. Bouché, M. Le Goc, and J. Coinu. A Global Model of Sequences of Discrete Event Class Occurrences. In *Proceedings of the Tenth International Conference on Enterprise Information Systems (ICEIS 2008)*, volume AIDSS, pages 173–180, 2008.

[Bou05] P. Bouché. *Une approche stochastique de modélisation de séquences d'événements discrets pour le diagnostic des systèmes dynamiques.* PhD thesis, Aix-Marseille University, 2005.

[Bre94] B. Bredeweg. *The CommonKads Library for Expertise Modelling, chapter Model-based diagnosis and prediction, p.121-153.* IOPress, 1994.

[CGP99] E. Clarke, O. Grumberg, and D. Peled. Model checking. *MIT Press*, page 14, 1999.

[CGTT93] L. Chittaro, L. Guida, L. Tasso, and E. Toppano. Functional and teological knowledge in the multi-modeling approach for reasoning about physical systems: A case study in diagnosis. In *IEEE Transactions on Systems, Man, and Cybernetics*, volume 23, pages 1718–1751, 1993.

[CR99] L. Chittaro and R. Ranon. Diagnosis of multiple faults with flow-based functional models: the functional diagnosis with efforts and flows approach. *Reliability Engineering and System Safety, 64(2):137-150, 1999.*, 1999.

[Dag01] P. Dagues. *Diagnostic, Intelligence Artificielle et Reconnaissance des Formes*, chapter Théorie logique du diagnostic à base de modèles, pages 11–104. 2001.

[Dam99] A. Damasio. The feeling of what happens: Body and emotion in the making of consciousness, harcourt. 1999.

[Dam05] A. Damasio. Descartes' error: Emotion, reason, and the human brains. 2005.

[DS87] F. Douay-Soublin. *La contre-analogie, réflexion sur la récusation de certaines analogies pourtant bien formées cognitivement.* G.T.A. Recueil de textes, 1987.

[Flo08] L. Floridi. The method of levels of abstraction. *Minds and Machines*, 18:303–329, September 2008.

[Flo10] L. Floridi. Levels of abstraction and the turing test. *Keybenetes*, 39(3):423–440, 2010.

[Flo17] Luciano Floridi. The logic of design as a conceptual logic of information. *Minds and Machines*, June 2017.

[FT98] G. Fauconnier and M. Turner. Conceptual integration networks. *Cognitive Science*, (22):133–187, 1998.

[FT03] G. Fauconnier and M. Turner. Conceptual blending, form and meaning. *Recherches en communication, n° 19 (2003).*, (19):57–86, 2003.

[Göd31] K. Gödel. *On Formally Undecidable Propositions of Principia Mathematica and Related Systems I.* PhD thesis, Vienna University, 1931.

[Hal74] F. Halbwachs. *La pensée physique.* Zeithos, Delachaux, Nestlé, 1974.

[Hue85] G. Huet. *Initiation à la Théorie des Catégories*, volume Fonctionnalité, Structures de Calcul et Programmation. Université Paris VII, 1985.

[LA12] M. Le Goc and A. Ahdab. *Learning Bayesian Networks From Timed Observations.* LAP LAMBERT Academic Publishing GmbH & Co. KG, 2012.

[LBD+03] Marc Le Goc, Michel Barles, Norbert Dolenc, François-Marie Lessaffre, and Claude Thirion. Procedure for controlling a complex dynamic process. US Patent 6 560 585 B1 (Filing Date: 07/30/2000), May 2003.

[LBP15] Marc Le Goc, Fabien Barthelot, and Eric Pascual. Emergence of regularities in the stochastic behavior of human. In *IEEE International Conference on Data Mining Workshop, ICDMW 2015, Atlantic City, NJ, USA, November 14-17, 2015*, pages 381–388. IEEE Computer Society, July 2015.

[LeG04] Marc LeGoc. Sachem. a real time intelligent diagnosis system based on the discrete event paradigm. *Simulation, The Society for Modeling and Simulation International Ed.*, 80(11):591–617, Novembre 2004.

[LeG06] Marc LeGoc. *Notion d'observation pour le diagnostic des processus dynamiques: Application à Sachem et à la découverte de connaissances temporelles.* Hdr, Aix-Marseille University, Faculté des Sciences et Techniques de Saint Jérôme, novembre 2006.

[LGFCT13] M. Le Goc, I. Fakhfakh, C. Curt, and L. Torres. Hydraulic dam safety assessment with the timed observations theory. *R and C*, 2013.

[LGG04] Marc Le Goc and Michel Gaeta. Modeling Strutures in Generic Space, a Condition for Adaptiveness of Monitoring Cognitive Agent. *Journal of Intelligent and Robotics Systems*, 41(2-3):113–140, January 2004.

[LGV17] M. Le Goc and F. Vilar. Operationalization of the blending and the levels of abstraction theories with the timed observations theory. In *Proceedings of the 9th International Conference on Agents and Artificial Intelligence (ICAART 2017)*, February 2017.

[LLB+08] Marc Le Goc, François-Marie Lessaffre, Michel Barles, Claude Thirion, and Norbert Dolenc. Method for controlling a complex dynamic process. European Patent EP 1 069 486 B2 (Filing Date: 06/30/2000), October 2008.

[LT98] Marc Le Goc and Claude Thirion. The sachem experience on artificial neural networks application. In *ECSC Workshop on the Applications Artificial Neural Network Systems in the Steel Industry*, January 1998.

[LTT98] Marc Le Goc, Claude Thouzet, and Claude Thirion. The sachem experience on artificial neural networks application. In *Fourth International Conference on Neural Networks and their Applications (Neurap'98)*, pages 315–321, May 1998.

[Mél12] B. Mélès. Pratique mathématique et lectures de hegel, de jean cavaillès à william lawvere. *Philosophia Scientiae*, pages 153–182, 2012.

[Mer83] J. Merker. De la théorie des catégories à l'usage des modèles en science. Merker's Lecturer Notes, 1983.

[ML71] S. Mac Lane. *Categories for the Working Mathematician.* Springer Science, 1971.

[New81] Alan Newell. The knowledge level. *AI Magazine*, 2(2):1–20, 1981.

[NK98] I. Nonaka and N. Konno. The concept of "ba": Building a fundation for knowledge creation. *California Management Review*, 40(3):40–54, 1998.

[Non91] I. Nonaka. The knowledge-creating company. 1991.

[Non94] I. Nonaka. Combining knowledge-based method and possibility theory for assessing dam performance. 1994.

[PGAP12] L. Pomponio, M. Le Goc, A. Anfosso, and E. Pascual. Levels of Abstraction for Behavior Modeling in the GerHome Project. *International Journal of E-Health and Medical Communications*, 3(3):12–28, 2012.

[PLG14] L. Pomponio and M. Le Goc. Reducing the gap between experts' knowledge and data: The tom4d methodology. *Data & Knowledge Engineering*, DOI 10.1016/j.datak.2014.07.006, July 2014.

[PLPA11] L. Pomponio, M. Le Goc, E. Pascual, and Alain Anfosso. Discovering Models of Human's Behavior from Sensor's Data. In *Workshop Proceedings of the 7th International Conference on Intelligent Environments, Nottingham, UK. 25-26th of July 2011*, volume 10 of *Ambient Intelligence and Smart Environments*, pages 17–28. IOS Press, 2011.

[Pol66] M. Polanyi. The tacit dimensione. 1966.

[Pom12] L. Pomponio. *Definition of a Human-Machine Learning Process from Timed Observations: Application to the Modelling of Human Behaviour for the Detection of Abnormal Behaviour of Old People at Home.* PhD thesis, Aix-Marseille University, Marseille (France), June 2012.

[PVP10] Marc Le Goc Pamela Viale, Nabil Benayadi and Jacques Pinaton. Discovery of large scale manufacturing process models from timed data. In *Proceeding of the 5th International Conference on Software and Data Technologies (ICSoft 2010)*, 2010.

[SAA+00] G. Schreiber, H. Akkermans, A. Anjewierden, R. de Hoog, N. Shadbolt, W. Van de Velde, and B. Wielinga. *Knowledge Engineering and Management: The CommonKADS Methodology.* MIT Press, 2000.

[SBF98] R Studer, VR Benjamins, and D Fensel. Knowledge engineering: Principles and methods. 1998.

[Sha84] C. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27:379–423, 623–656, July, October 1984.

[Vau08] A. Vautier. *Fouille de Données Sans Information a Priori sur la Structure de la Connaissance.* PhD thesis, Université de Rennes, 2008.

[VLGBR16] F. Vilar, M. Le Goc, P. Bouche, and P. Rolland. Discovering internal fraud models in a stream of banking transactions. *ICAART*, 2016.

[ZGF06a] C. Zanni, M. Le Goc, and C. Frydman. A conceptual framework for the analysis, classification and choice of knowledge-based diagnosis systems. *KES - International Journal of Knowlegde-Based and Intelligent Engineering Systems, 10(2):113-138*, 2006.

[ZGF06b] C. Zanni, M. Le Goc, and C. Frydman. A conceptual framework for the analysis, classification and choice of knowledge-based diagnosis systems. *KES - International Journal of Knowlegde-Based and Intelligent Engineering Systems, Kluwer Academic Publishers*, 10(2):113–138, 2006.

# Appendices

CHAPTER A

# Unary Observers

## A.1 Introduction

A *unary observer*, denoted $\Theta(\Delta, \Psi)$ where $\Delta = \{\delta_i\} \subseteq \mathbb{N}$ is a set of constants $\delta_i$ and $\Psi = \{\psi_i\} \subseteq \mathfrak{R}$ is a set of thresholds $\psi_i$, is a program made to write a timed observation $O(t_k)$ each time the value of a particular function $x(t)$ enters an interval $[\psi_i, +\infty[$ at the time $t_k$.

When recorded in a database, a timed observation $O(t_k)$ is represented by a couple $(\delta_i, t_k)$ mapping a constant $\delta_i$ with a timestamp $t_k$:

$$O(t_k) \equiv (\delta_i, t_k) \tag{A.1}$$

In a general way, the constant $\delta_i$ is a string but, in this framework, it is set to an integer, $\delta_i \in \mathbb{N}$.

## A.2 Theoretical Unary Observer

The *theoretical unary observer* $\Theta(\{\delta\}, \{\psi\})$ implements the *Spatial Discretization Principle* of a time function $x(t)$ (see 3.5):

$$\forall t_{k-1} \in \mathfrak{R}, t_k \in \mathfrak{R}, x(t_{k-1}) < \psi \wedge x(t_k) \geq \psi \Rightarrow write((\delta, t_k)) \tag{A.2}$$

The *theoretical unary observer* $\Theta(\{\delta\}, \{\psi\})$ aims at writing a time observation $O(t_k) \equiv (\delta, t_k)$ each time the value $x(t_k)$ of the function $x(t)$ enters the interval $[\psi, +\infty[$.

The interval $[\psi, +\infty[$ is named the *value range* and constant $\delta$ is considered as the name of the interval $[\psi, +\infty[$.

According to the TOT, any unary observer has to verify the following properties:

- absence of memory: decision to write a given occurrence of a time observation $O(t_k) \equiv (\delta_i, t_k)$ at $t_k$ only depends on the entrance of the value $x(t_k)$ in the range $[\psi, +\infty[$. As a consequence, this decision doesn't depend on any occurrence of $O(t_j), t_j < t_k$ written in the past;

- constants' independence: the choice of a constant $\delta_i \in \Delta$ must only depend on the range $[\psi_i, +\infty[$ mapping the constant $\delta_i$. This choice doesn't depend on the other constants $\delta_j \in \Delta, j \neq i$;

- no simultaneous observations: $\forall t_i, t_j \in \Gamma, t_i \neq t_j$.

The *theoretical unary observer* $\Theta(\{\delta\},\{\psi\})$ is the Time Observation Theory's sampling device and extends Dirac's periodical sampling device (see chapter **??**). It produces a sequence of timed observations $\omega(t_k) = \{O(t_1), O(t_2), ..., O(t_k)\}$ and generates a set of timestaps $\Gamma = \{t_1, t_2..., t_k\}$ called a *stochastic clock*.

Let us consider the example of the function $x(t)$ seen on figure A.1 representing the evolution over time of a horse's odd during a race.


Figure A.1: Horse odd evolution over time

Data are recorded from time 20 : 35 : 25 until 21 : 07 : 25 every 10 seconds. Figure A.1 represents then a set of 193 couples (time, odd).

Let us consider the unary observer $\Theta^+(\{\delta^+\},\{\psi\})$ implementing the following equation:

$$\forall t_{k-1} \in \Re, t_k \in \Re, x(t_{k-1}) < \psi \wedge x(t_k) \geq \psi \Rightarrow write((\delta^+, t_k)) \tag{A.3}$$

Writing the constant $\delta^+$ at time $t_k$ indicates an upward crossing of the threshold $\psi$ by the function $x(t)$. This means that the function $x(t)$ enters the interval $I^+ \equiv [\psi, +\infty[$. The writing of the constant $\delta^+$ is then related to the interval $I^+ \equiv [\psi, +\infty[$:

$$\delta^+ \leftrightarrow I^+ \equiv [\psi, +\infty[ \tag{A.4}$$

Let us now set the threshold $\psi$ to the real value 1.90, $\psi = 1.90$. The unary observer $\Theta^+(\{\delta^+\},\{\psi\})$, applied on the function $x(t)$ represented on figure A.1, produces then a sequence of timed observations $\omega^+(t_5^+) = \{O(t_1^+), O(t_2^+), O(t_3^+), O(t_4^+), O(t_5^+)\}$ and a stochastic clock $\Gamma^+ = \{t_1^+, t_2^+, t_3^+, t_4^+, t_5^+\}$, as seen on figure A.2, where:

- $O(t_1^+) \equiv (\delta^+, t_1^+)$ and $t_1^+ = 20:44:35$;
- $O(t_2^+) \equiv (\delta^+, t_2^+)$ and $t_2^+ = 20:44:55$;
- $O(t_3^+) \equiv (\delta^+, t_3^+)$ and $t_3^+ = 20:52:25$;

- $O(t_4^+) \equiv (\delta^+, t_4^+)$ and $t_4^+ = 20:55:15$;
- $O(t_5^+) \equiv (\delta^+, t_5^+)$ and $t_5^+ = 21:07:15$.

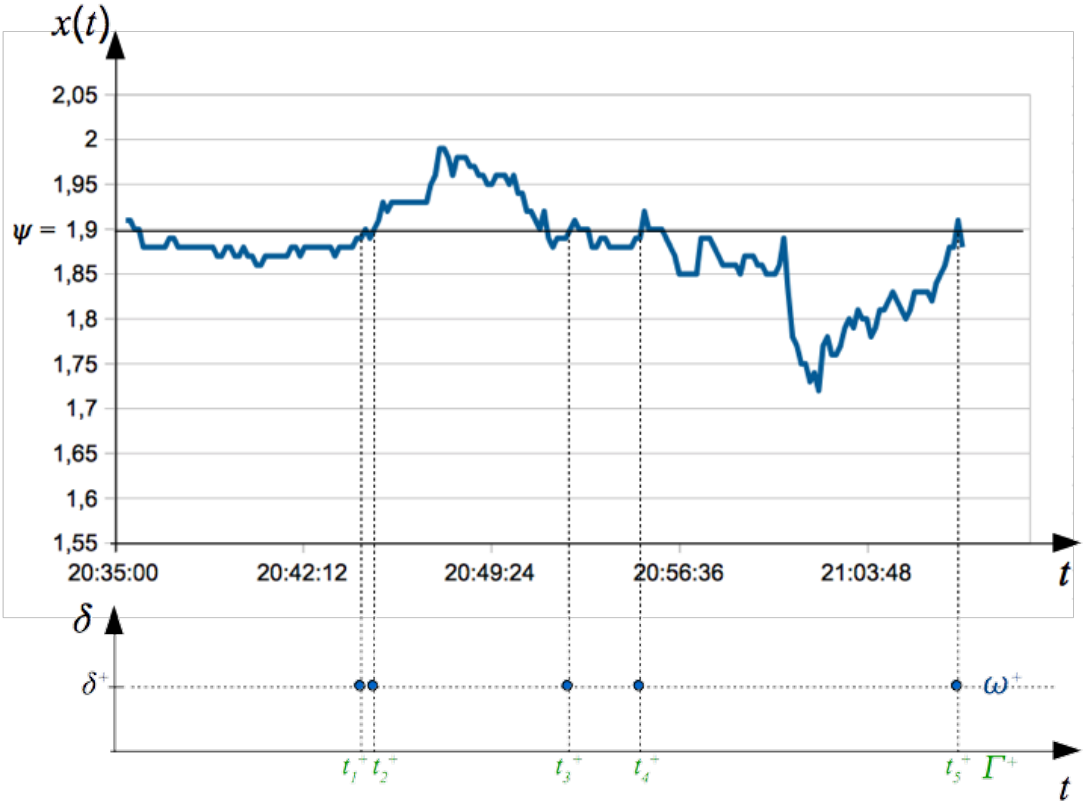


Figure A.2: Sequence of timed observations and stochastic clock produced by $\Theta^+(\{\delta^+\}, \{\psi\})$ where $\psi = 1.90$

The *symmetrical unary observer* of $\Theta^+(\{\delta^+\}, \{\psi\})$ is a program denoted $\Theta^-(\{\delta^-\}, \{\psi\})$ implementing the following equation:

$$\forall t_{k-1} \in \Re, t_k \in \Re, x(t_{k-1}) \geq \psi \wedge x(t_k) < \psi \Rightarrow write((\delta^-, t_k)) \tag{A.5}$$

Writing the constant $\delta^-$ at time $t_k$ indicates a downward crossing of the threshold $\psi$ by the function $x(t)$. This means that the function $x(t)$ enters the interval $I^- \equiv ]-\infty, \psi[$. The writing of the constant $\delta^-$ is then related to the interval $I^- \equiv ]-\infty, \psi[$:

$$\delta^- \leftrightarrow I^- \equiv ]-\infty, \psi[ \tag{A.6}$$

The unary observer $\Theta^-(\{\delta^-\}, \{\psi\})$, applied on the function $x(t)$ represented on figure A.1, produces then a sequence of timed observations $\omega^-(t_6^-) = \{O(t_1^-), O(t_2^-), O(t_3^-), O(t_4^-), O(t_5^-), O(t_6^-)\}$ and a stochastic clock $\Gamma^- = \{t_1^-, t_2^-, t_3^-, t_4^-, t_5^-\}$, as seen on figure A.3, where:

- $O(t_1^-) \equiv (\delta^-, t_1^-)$ and $t_1^- = 20:36:05$;
- $O(t_2^-) \equiv (\delta^-, t_2^-)$ and $t_2^- = 20:44:45$;
- $O(t_3^-) \equiv (\delta^-, t_3^-)$ and $t_3^- = 20:51:35$;

- $O(t_4^-) \equiv (\delta^-, t_4^-)$ and $t_4^- = 20:53:15$;
- $O(t_5^-) \equiv (\delta^-, t_5^-)$ and $t_5^- = 20:56:05$;
- $O(t_6^-) \equiv (\delta^-, t_6^-)$ and $t_6^- = 21:07:25$.

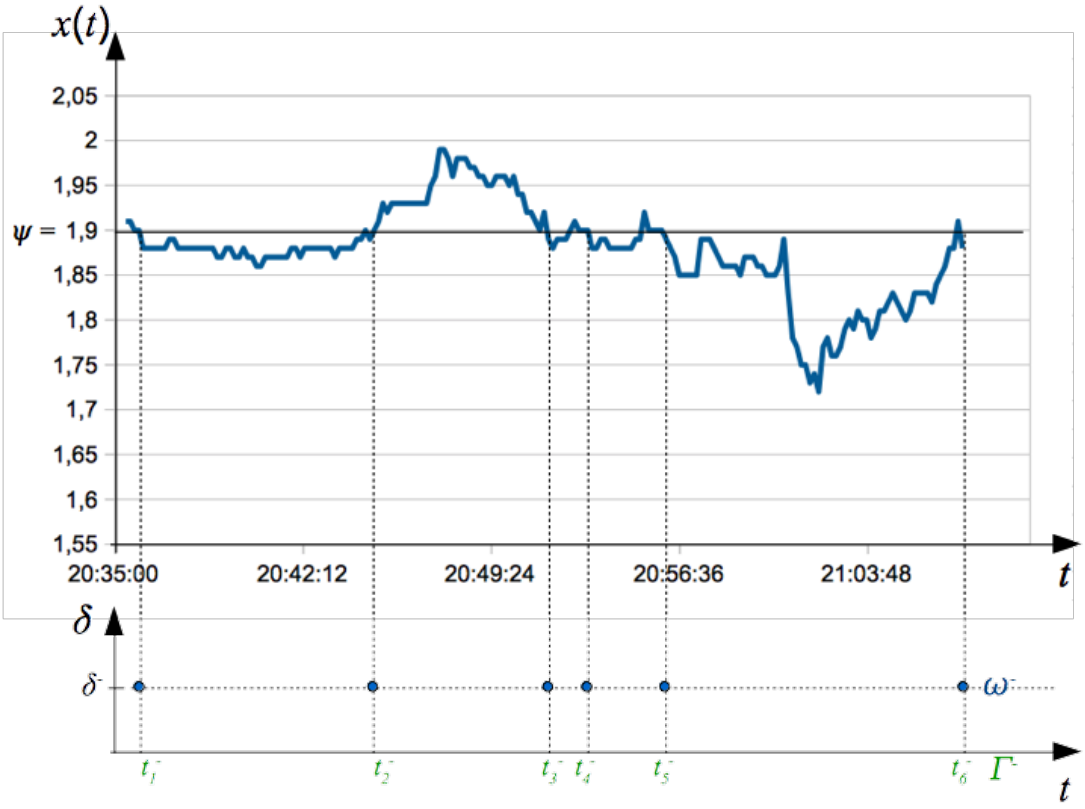


Figure A.3: Sequence of timed observations and stochastic clock produced by $\Theta^-(\{\delta^-\}, \{\psi\})$ where $\psi = 1.90$

The unary observer $\Theta^-(\{\delta^-\}, \{\psi\})$ is the *symmetrical unary observer* of $\Theta^+(\{\delta^+\}, \{\psi\})$.

Table A.1 sums up the interpretation made from timed observations generated by both theoretical unary observers:

| Timed Observation | Interpretation |
|---|---|
| $(\delta^-, t_1^-)$ | At $t_1^- = 20:36:05$, $x(t)$ enters $I^- \equiv ]-\infty, \psi[$ |
| $(\delta^+, t_1^+)$ | At $t_1^+ = 20:44:35$, $x(t)$ enters $I^+ \equiv [\psi, +\infty[$ |
| $(\delta^-, t_2^-)$ | At $t_2^- = 20:44:45$, $x(t)$ enters $I^- \equiv ]-\infty, \psi[$ |
| $(\delta^+, t_2^+)$ | At $t_2^+ = 20:44:55$, $x(t)$ enters $I^+ \equiv [\psi, +\infty[$ |
| $(\delta^-, t_3^-)$ | At $t_3^- = 20:51:35$, $x(t)$ enters $I^- \equiv ]-\infty, \psi[$ |
| $(\delta^+, t_3^+)$ | At $t_3^+ = 20:52:25$, $x(t)$ enters $I^+ \equiv [\psi, +\infty[$ |
| $(\delta^-, t_4^-)$ | At $t_4^- = 20:53:15$, $x(t)$ enters $I^- \equiv ]-\infty, \psi[$ |
| $(\delta^+, t_4^+)$ | At $t_4^+ = 20:55:15$, $x(t)$ enters $I^+ \equiv [\psi, +\infty[$ |
| $(\delta^-, t_5^-)$ | At $t_5^- = 20:56:05$, $x(t)$ enters $I^- \equiv ]-\infty, \psi[$ |
| $(\delta^+, t_5^+)$ | At $t_5^+ = 21:07:15$, $x(t)$ enters $I^+ \equiv [\psi, +\infty[$ |
| $(\delta^-, t_6^-)$ | At $t_6^- = 21:07:25$, $x(t)$ enters $I^- \equiv ]-\infty, \psi[$ |

Table A.1: Interpretation made from timed observations generated by both theoretical unary observers

The *theoretical unary observer* $\Theta(\{\delta^-, \delta^+\}, \{\psi\})$ is then the *superposition* of two *symmetrical unary observers*, $\Theta^-(\{\delta^-\}, \{\psi\})$ and $\Theta^+(\{\delta^+\}, \{\psi\})$, respectively implementing the following equations:

$$\begin{cases} \forall t_{k-1} \in \Re, t_k \in \Re, x(t_{k-1}) \geq \psi \wedge x(t_k) < \psi \Rightarrow write((\delta^-, t_k)) \\ \forall t_{k-1} \in \Re, t_k \in \Re, x(t_{k-1}) < \psi \wedge x(t_k) \geq \psi \Rightarrow write((\delta^+, t_k)) \end{cases} \tag{A.7}$$

The *theoretical unary observer* $\Theta(\{\delta^-, \delta^+\}, \{\psi\})$ builds two disjoint sets of $\Re$, $I^- \equiv ]-\infty, \psi[$ and $I^+ \equiv [\psi, +\infty[$:

$$\begin{cases} I^- \bigcup I^+ = \Re \\ I^- \bigcap I^+ = \emptyset \end{cases} \tag{A.8}$$

linking each constant $\delta^-$ and $\delta^+$ to respectively $I^-$ and $I^+$:

$$\begin{cases} \delta^- \leftrightarrow I^- \equiv ]-\infty, \psi[ \\ \delta^+ \leftrightarrow I^+ \equiv [\psi, +\infty[ \end{cases} \tag{A.9}$$

The *theoretical unary observer* $\Theta(\{\delta^-, \delta^+\}, \{\psi\})$ generates a sequence of timed observations $\omega(t_k)$ and a stochastic clock $\Gamma$ such as:

$$\begin{cases} \omega(t_k) = \{\omega^-(t_k^-), \omega^+(t_k^+)\}, t_k = max(t_k^-, t_k^+) \\ \Gamma = \{\Gamma^-, \Gamma^+\} \end{cases} \tag{A.10}$$

## A.3 Concrete Unary Observer

The decision to cross a threshold is only taken from a unique value $x(t_k)$. The *theoretical unary observer* $\Theta(\{\delta\}, \{\psi\})$ is not suitable for noisy signals. In a pratical way, this decision must be taken from a set of values $w(t_k) = \{x(t_{k-(n-1)}), x(t_{k-(n-2)}), ..., x(t_k)\}$, called *observation window*, containing the last $n$ known values of the function $x(t)$.

A realistic criteria of decision is built on the following general principal:

- a threshold $\psi$ is crossed by a function $x(t)$ when the majority of the values $x(t_i)$ contained in the *observation window* $w(t_k)$ are superior to the threshold $\psi$.

Let us introduce a *criteria of decision* $c(w(t_k))$ based on a measure $\mu(w(t_k))$ made on the *observation window* $w(t_k)$ and a *decision threshold* denoted $\psi_d$ such as:

$$\mu(w(t_k)) \geq \psi_d \Rightarrow c(w(t_k)) = true \tag{A.11}$$

The measure $\mu(w(t_k))$ can be as complex as needed but must be computed for each new value $x(t_k)$ so the following property has to be respected:

- $x(t_k)$ enters the range of values $[\psi, +\infty[$ if and only if $x(t_{k-1})$ was outside that range at the previous timestamp $t_{k-1}$.

For instance, a measure $\mu(w(t_k))$ can compute the percentage of values of $x(t_k)$ which are superior to the threshold $\psi$:

$$\mu(w(t_k)) = \frac{Card(w_d(t_k))}{Card(w(t_k))} \tag{A.12}$$

where $w_d(t_k) = \{x(t_i), t_{k-(n-1)} \leq t_i \leq t_k)\}$, called the *decision window*, is the set of values $x(t_i)$ superior to the threshold $\psi$:

$$\forall x(t_i) \in w(t_k), x(t_i) \geq \psi \Rightarrow x(t_i) \in w_d(t_k) \tag{A.13}$$

A *concrete unary observer* is a program $\Theta^+(\{\delta^+\}, \{\psi\}, \psi_d, n)$ implementing the following equations:

$$\begin{cases} \forall x(t_i) \in w(t_k), x(t_i) \geq \psi \Rightarrow x(t_i) \in w_d^+(t_k) \\ \forall t_{k-1}, t_k \in \Re, \frac{Card(w_d^+(t_{k-1}))}{Card(w^+(t_{k-1}))} < \psi_d \wedge \frac{Card(w_d^+(t_k))}{Card(w^+(t_k))} \geq \psi_d \Rightarrow write((\delta^+, t_k)) \end{cases} \tag{A.14}$$

Let us consider the function $x(t)$ introduced on figure A.1. Let us set the *decision threshold* to 80%, $\psi_d = 0.8$, and the size $n$ of the *observation window* to 30, $n = 30$.

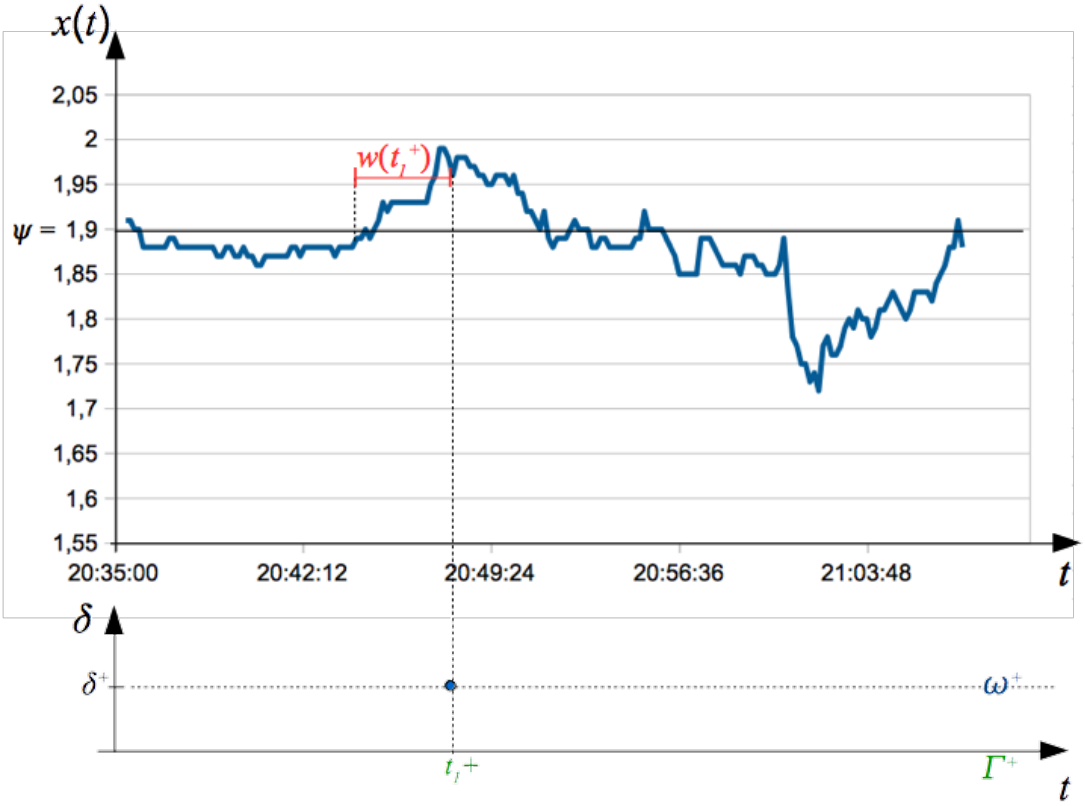


Figure A.4: Sequence of timed observations and stochastic clock produced by a concrete unary observer $\Theta^+(\{\delta^+\}, \{\psi\}, \psi_d, n)$ where $\psi = 1.90$, $\psi_d = 0.8$ and $n = 30$

The *concrete unary observer* $\Theta^+(\{\delta^+\}, \{\psi\}, \psi_d, n)$, applied on the function $x(t)$, produces then a sequence of only one timed observation $\omega(t_1^+) = \{O(t_1^+)\}$ and a stochastic clock $\Gamma^+ = \{t_1^+\}$, as seen on figure A.4, where:

- $O(t_1^+) \equiv (\delta^+, t_1^+)$ and $t_1^+ = 20:48:35$.

According to such a *concrete unary observer*, the function $x(t)$ enters the interval $I^+ \equiv [\psi, +\infty[$

at time $t_1^+ = 20:48:35$.

The *symmetrical unary observer* of $\Theta^+(\{\delta^+\},\{\psi\},\psi_d,n)$ is a program denoted $\Theta^-(\{\delta^-\},\{\psi\},\psi_d,n)$ implementing the following equations:

$$\begin{cases} \forall x(t_i) \in w(t_k), x(t_i) < \psi \Rightarrow x(t_i) \in w_d^-(t_k) \\ \forall t_{k-1}, t_k \in \Re, \frac{Card(w_d^-(t_{k-1}))}{Card(w(t_{k-1}))} < \psi_d \wedge \frac{Card(w_d^-(t_k))}{Card(w(t_k))} \geq \psi_d \Rightarrow write((\delta^-, t_k)) \end{cases} \quad \text{(A.15)}$$

The *symmetrical concrete unary observer* $\Theta^-(\{\delta^-\},\{\psi\},\psi_d,n)$, applied on the function $x(t)$, produces then a sequence of only one timed observation $\omega(t_1^-) = \{O(t_1^-)\}$ and a stochastic clock $\Gamma^- = \{t_1^-\}$, as seen on figure A.5, where:

- $O(t_1^-) \equiv (\delta^-, t_1^-)$ and $t_1^- = 20:57:55$.

According this *symmetrical concrete unary observer*, the function $x(t)$ enters the interval $I^- \equiv [-\infty, \psi[$ at time $t_1^- = 20:57:55$.

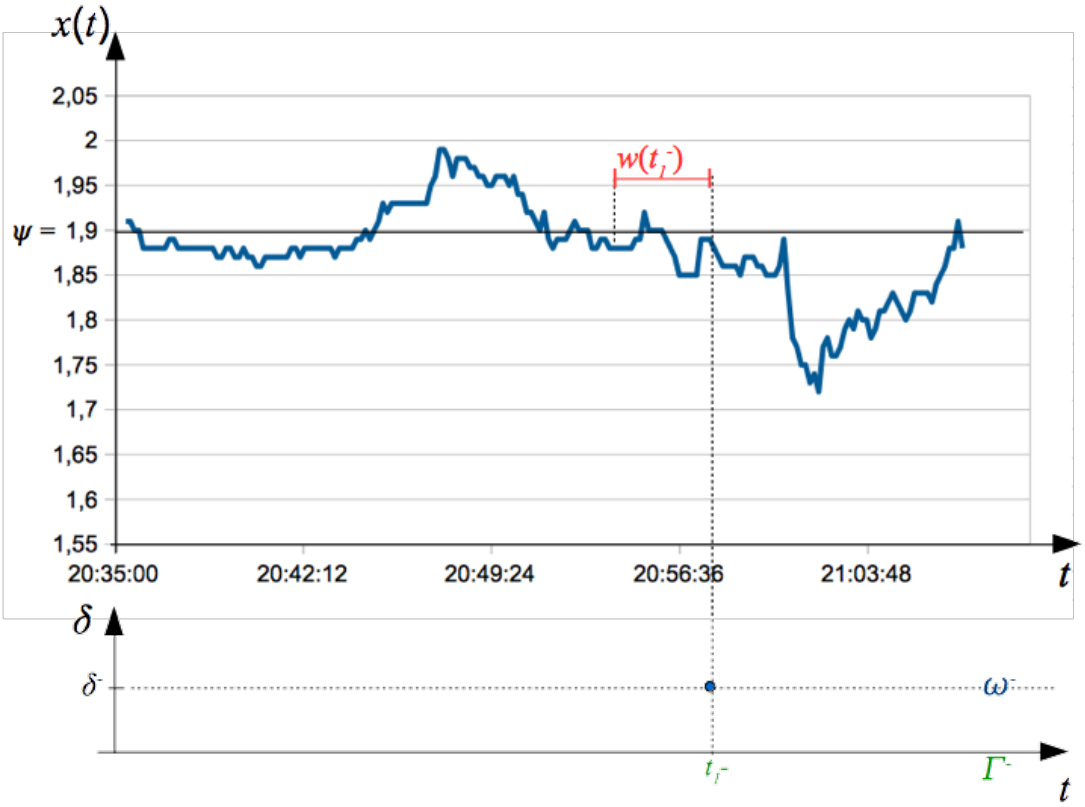


Figure A.5: Sequence of timed observations and stochastic clock produced by the symmetrical concrete unary observer $\Theta^-(\{\delta^-\},\{\psi\},\psi_d,n)$ where $\psi = 1.90$, $\psi_d = 0.8$ and $n = 30$

Table A.2 sums up the interpretation made from timed observations generated by both concrete unary observers:

The *concrete unary observer* $\Theta(\{\delta^-,\delta^+\},\{\psi\},\psi_d,n)$ is then the *superposition* of two *symmetrical concrete unary observers*, $\Theta^-(\{\delta^-\},\{\psi\},\psi_d,n)$ and $\Theta^+(\{\delta^+\},\{\psi\},\psi_d,n)$, respectively implementing the following equations:

| Timed Observation | Interpretation |
|---|---|
| $(\delta^+, t_1^+)$ | At $t_1^+ = 20:48:35$, $x(t)$ enters $I^+ \equiv [\psi, +\infty[$ |
| $(\delta^-, t_2^-)$ | At $t_2^- = 20:57:55$, $x(t)$ enters $I^- \equiv ]-\infty, \psi[$ |

Table A.2: Interpretation made from timed observations generated by both concrete unary observers

$$\begin{cases} \forall x(t_i) \in w(t_k), x(t_i) < \psi \Rightarrow x(t_i) \in w_d^-(t_k) \\ \forall t_{k-1}, t_k \in \mathfrak{R}, \frac{Card(w_d^-(t_{k-1}))}{Card(w(t_{k-1}))} < \psi_d \wedge \frac{Card(w_d^-(t_k))}{Card(w(t_k))} \geq \psi_d \Rightarrow write((\delta^-, t_k)) \\ \forall x(t_i) \in w(t_k), x(t_i) \geq \psi \Rightarrow x(t_i) \in w_d^+(t_k) \\ \forall t_{k-1}, t_k \in \mathfrak{R}, \frac{Card(w_d^+(t_{k-1}))}{Card(w^+(t_{k-1}))} < \psi_d \wedge \frac{Card(w_d^+(t_k))}{Card(w^+(t_k))} \geq \psi_d \Rightarrow write((\delta^+, t_k)) \end{cases} \tag{A.16}$$

The *concrete unary observer* $\Theta(\{\delta^-, \delta^+\}, \{\psi\}, \psi_d, n)$ builds two disjoint sets of $\mathfrak{R}$, $I^- \equiv ]-\infty, \psi[$ and $I^+ \equiv [\psi, +\infty[$:

$$\begin{cases} I^- \bigcup I^+ = \mathfrak{R} \\ I^- \bigcap I^+ = \emptyset \end{cases} \tag{A.17}$$

linking each constant $\delta^-$ and $\delta^+$ to respectively $I^-$ and $I^+$:

$$\begin{cases} \delta^- \leftrightarrow I^- \equiv ]-\infty, \psi[ \\ \delta^+ \leftrightarrow I^+ \equiv [\psi, +\infty[ \end{cases} \tag{A.18}$$

The *concrete unary observer* $\Theta(\{\delta^-, \delta^+\}, \{\psi\}, \psi_d, n)$ generates a sequence of timed observations $\omega(t_k)$ and a stochastic clock $\Gamma$ such as:

$$\begin{cases} \omega(t_k) = \{\omega^-(t_k^-), \omega^+(t_k^+)\}, t_k = max(t_k^-, t_k^+) \\ \Gamma = \{\Gamma^-, \Gamma^+\} \end{cases} \tag{A.19}$$

A more general definition of a *concrete unary observer* is given here, splitting the set $\mathfrak{R}$ into $m+1$ disjoint sets.

A *concrete unary observer* is a program $\Theta(\Delta, \Psi, \psi_d, n)$ where:

- $\Psi = \{\psi_1, ..., \psi_m\} \subseteq \mathfrak{R}$ is a set of $m$ thresholds;
- $\Delta = \{\delta_0, ..., \delta_m\} \subseteq \mathbb{N}$ is a set of $m+1$ constants;
- $\psi_d \in \mathfrak{R}$ is a *decision threshold*;
- $n$ is the size of the *observation window* $w(t_k)$.

Each constant $\delta_i$ links a range of values defined by an interval $I_i$ such as:

- $\delta_0 \leftrightarrow I_1 \equiv ]-\infty, \psi_1[$;
- $\delta_1 \leftrightarrow I_2 \equiv [\psi_1, \psi_2[$;
- ...;
- $\delta_i \leftrightarrow I_i \equiv [\psi_i, \psi_{i+1}[$;

- ...;
- $\delta_m \leftrightarrow I_{m+1} \equiv [\psi_m, +\infty[$.

Such a *concrete unary observer* applied on function $x(t)$ produces:

- a superposition $\Omega(t_k) = \{\omega^i(t_k)\}$ of $m$ sequences of timed observations $\omega^i(t_k) = \{O^i(t_1), O^i(t_2), ..., O^i(t_k)\}$;
- a set $\Gamma = \{\Gamma^i\}$ of $m$ stochastic clocks $\Gamma^i = \{t_1^i, t_2^i, ..., t_k^i\}$.

Each timestamp $t_j^i$ means that the function $x(t)$ enters the interval $I_i = [\psi_i, \psi_{i+1}[$ at time $t = t_j^i$. This sampling device can be used on any dynamic process $X(t) = \{x^i(t)\}$.

## A.4 Piecewise Functions

This section deals with the spatial segmentation of piecewise functions.

$$\forall k \in \mathbb{Z}, \forall a_k \in \mathbb{N}^*, \forall t_k \in \mathfrak{R}, \quad \begin{array}{ccccc} x & : & \mathfrak{R} & \rightarrow & \mathbb{N}^* \\ & & t & \mapsto & \sum_{k=-\infty}^{+\infty} a_k.H(t-t_k) \end{array} \tag{A.20}$$

The function $H(t)$ is the Heaviside step function:

$$H(t-t_k) = \begin{cases} 0 & \text{if } t < t_k \\ 1 & \text{if } t \geq t_k \end{cases} \tag{A.21}$$

The derivative $H^{'}(t)$ of the Heaviside function is the Dirac distribution $\delta(\{0\})$:

$$H^{'}(t) = \delta(\{0\}) \tag{A.22}$$

### A.4.1 Observing a Piecewise Function

Let us consider the spatial discretization equation A.2 designed for stepwise functions:

$$\forall t_{k-1}, t_k \in \mathfrak{R}, x(t_{k-1}) \neq x(t_k) \Rightarrow write(O(t_k)) \text{ where } O(t_k) \equiv (x(t_k), t_k) \tag{A.23}$$

As $x(t) = \sum_{k=-\infty}^{+\infty} a_k.H(t-t_k)$, a timed observation is only written when $x(t_{k-1}) \neq x(t_k)$ i.e. when the value of $x(t)$ changes:

$$x(t_k) = a_k.H(t_k - t_k) = a_k \tag{A.24}$$

And:

$$O(t_k) \equiv (x(t_k), t_k) \equiv (a_k, t_k) \tag{A.25}$$

A theoretical unary observer observing a piecewise function $x(t) = \sum_{k=-\infty}^{+\infty} a_k.H(t-t_k)$ is a program denoted $\Theta(\mathbb{N}, \{0\})$, set with the set of constants contained in $\mathbb{N}$ and a 0 threshold, implementing the following equation:

$$\forall t_{k-1}, t_k \in \mathfrak{R}, x(t_{k-1}) \neq x(t_k) \Rightarrow write(O(t_k)) \text{ where } O(t_k) \equiv (a_k, t_k) \tag{A.26}$$

Such an observer generates a sequence of timed observations:

$$\omega(t_k) = \{(a_1, t_1), ..., (a_i, t_i), ..., (a_k, t_k)\} \tag{A.27}$$

and a stochastic clock:

$$\Gamma = \{t_1, ..., t_i, ..., t_k\} \tag{A.28}$$

The Dirac distribution defined by its action on any test function $\varphi$ as:

$$\langle \delta, \varphi \rangle = \int_{-\infty}^{+\infty} \delta(t)\varphi(t)\mathrm{d}t = \varphi(0) \tag{A.29}$$

The Dirac delta function is the neutral element of the convolution product applied on any function $x(t)$:

$$\forall t_k \in \mathfrak{R}, (x * \delta)(t_k) = \int_{-\infty}^{+\infty} x(t).\delta(t_k - t)\,\mathrm{d}t = x(t_k) \tag{A.30}$$

Let us apply equation A.30 to the Heaviside function $x(t) = a_k.H(t - t_k)$:

$$\forall t_k \in \mathfrak{R}, (x * \delta)(t_k) = \int_{-\infty}^{+\infty} x(t).\delta(t_k - t)\,\mathrm{d}t = x(t_k) = a_k.H(0) = a_k \tag{A.31}$$

Let us consider the stochastic clock $\Gamma$ (see equation A.28) generated by the unary observer $\Theta(\mathbb{N}, \{0\})$. Let us now consider the Dirac distribution $\delta(\Gamma)$ whose support is the stochastic clock $\Gamma$. Applying the convolution product between the piecewise function $x(t) = \sum_{k=-\infty}^{+\infty} a_k.H(t-t_k)$ and $\delta(\Gamma)$ generates then the sequence of values $\{a_0, a_1, ..., a_k\}$. The sequence of timed observations $\omega(t_k) = \{(a_1, t_1), ..., (a_i, t_i), ..., (a_k, t_k)\}$ generated by the unary observer $\Theta(\mathbb{N}, \{0\})$ on a piecewise function $x(t) = \sum_{k=-\infty}^{+\infty} a_k.H(t - t_k)$ is *isomorphic* to the sequence of values $\{a_0, a_1, ..., a_k\}$ generated by the convolution product $x(t_k) = x(t) * \delta(\Gamma)$ where:

$$x(t) = \sum_{k=-\infty}^{+\infty} a_k.H(t - t_k) \tag{A.32}$$

Any sequence $\omega(t_k) = \{(a_1, t_1), ..., (a_i, t_i), ..., (a_k, t_k)\}$ can be then interpreted as the result of the convolution of a piecewise function $x(t) = \sum_{k=-\infty}^{+\infty} a_k.H(t - t_k)$ with a Dirac comb $\delta(\Gamma)$ whose support is the stochastic clock $\Gamma$.

### A.4.2 Observing the Derivative of a Piecewise Function

Let us now consider the following equation:

$$\forall t_{k-1}, t_k \in \mathfrak{R}, x(t_{k-1}) \neq x(t_k) \Rightarrow write(O(t_k)) \text{ where } O(t_k) \equiv (x(t_k) - x(t_{k-1}), t_k) \tag{A.33}$$

As $x(t) = \sum_{k=-\infty}^{+\infty} a_k.H(t - t_k)$, a timed observation is only written when $x(t_{k-1}) \neq x(t_k)$ i.e. when the value of $x(t)$ changes:

$$x(t_k) - x(t_{k-1}) = a_k.H(t_k - t_k) - a_{k-1}.H(t_{k-1} - t_{k-1}) = a_k - a_{k-1} \tag{A.34}$$

And:

$$O(t_k) \equiv (x(t_k), t_k) \equiv (a_k - a_{k-1}, t_k) \tag{A.35}$$

A theoretical unary observer observing the derivative of a piecewise function $x(t) = \sum_{k=-\infty}^{+\infty} a_k.H(t-t_k)$ is a program denoted $\Theta^{'}(\mathbb{N}, \{0\})$, set with the set of constants contained in $\mathbb{N}$ and a 0 threshold, implementing the following equation:

$$\forall t_{k-1}, t_k \in \Re, x(t_{k-1}) \neq x(t_k) \Rightarrow write(O(t_k)) \text{ where } O(t_k) \equiv (a_k - a_{k-1}, t_k) \tag{A.36}$$

Such an observer generates a sequence of timed observations:

$$\omega^{'}(t_k) = \{(a_1 - a_0, t_1), ..., (a_i - a_{i-1}, t_i), ..., (a_k - a_{k-1}, t_k)\} \tag{A.37}$$

and a stochastic clock:

$$\Gamma^{'} = \{t_1, ..., t_i, ..., t_k\} \tag{A.38}$$

Let us consider the stochastic clock $\Gamma^{'}$ (see equation A.38) generated by the unary observer $\Theta^{'}(\mathbb{N}, \{0\})$. Let us now consider the Dirac distribution $\delta(\Gamma^{'})$ whose support is the stochastic clock $\Gamma^{'}$. Let us consider the piecewise function $d(t) = \sum_{k=-\infty}^{+\infty}(a_k - a_{k-1}).H(t - t_k)$. Applying the convolution product between the piecewise function $d(t) = \sum_{k=-\infty}^{+\infty}(a_k - a_{k-1}).H(t - t_k)$ and $\delta(\Gamma^{'})$ generates then the sequence of values $\{a_1 - a_0, ..., a_i - a_{i-1}, ..., a_k - a_{k-1}\}$. The sequence of timed observations $\omega(t_k) = \{(a_1 - a_0, t_1), ..., (a_i - a_{i-1}, t_i), ..., (a_k - a_{k-1}, t_k)\}$ generated by the unary observer $\Theta^{'}(\mathbb{N}, \{0\})$ on a piecewise function $d(t) = \sum_{k=-\infty}^{+\infty}(a_k - a_{k-1}).H(t - t_k)$ is *isomorphic* to the sequence of values $\{a_1 - a_0, ..., a_i - a_{i-1}, ..., a_k - a_{k-1}\}$ generated by the convolution product $d(t_k) = d(t) * \delta(\Gamma^{'})$ where:

$$d(t) = \sum_{k=-\infty}^{+\infty} (a_k - a_{k-1}).H(t - t_k) \tag{A.39}$$

The value $d_k = \frac{a_k - a_{k-1}}{t_k - t_{k-1}}$ is the linear coefficient of the line whose equation is:

$$\forall t \in [t_{k-1}; t_k[, x_k(t) = d_k.t + \frac{a_{k-1}.t_k - a_k.t_{k-1}}{t_k - t_{k-1}} \tag{A.40}$$

This equation can be rewritten:

$$\forall t \in [t_{k-1}; t_k[, x_k(t) = d_k.t + \alpha_k \text{ where } \alpha_k \equiv \frac{a_{k-1}.t_k - a_k.t_{k-1}}{t_k - t_{k-1}} \tag{A.41}$$

Such a line passes from the point $x(t_{k-1})$ to $x(t_k)$ belonging to the function $x(t) = \sum_{k=-\infty}^{+\infty} a_k.H(t-t_k)$ (see figure A.6). In other words, the unary observer $\Theta^{'}(\mathbb{N}, \{0\})$, applied on the piecewise function $x(t) = \sum_{k=-\infty}^{+\infty} a_k.H(t - t_k)$, generates a sequence of timed observations $\omega^{'}(t_k) = \{(a_1 - a_0, t_1), ..., (a_i - a_{i-1}, t_i), ..., (a_k - a_{k-1}, t_k)\}$ which, given the value $a_0$, is necessary and sufficient to describe the function $x(t) = \sum_{k=-\infty}^{+\infty} a_k.H(t - t_k)$ under the form of a sequence of lines $x_k(t) = d_k.t + \alpha_k, t \in [t_{k-1}; t_k[$ linking the points $x(t_k)$.

Since $a_k = x(t_k)$ and $a_{k-1} = x(t_{k-1})$, the linear coefficient $d_k$ of the line $x_k(t)$ is:

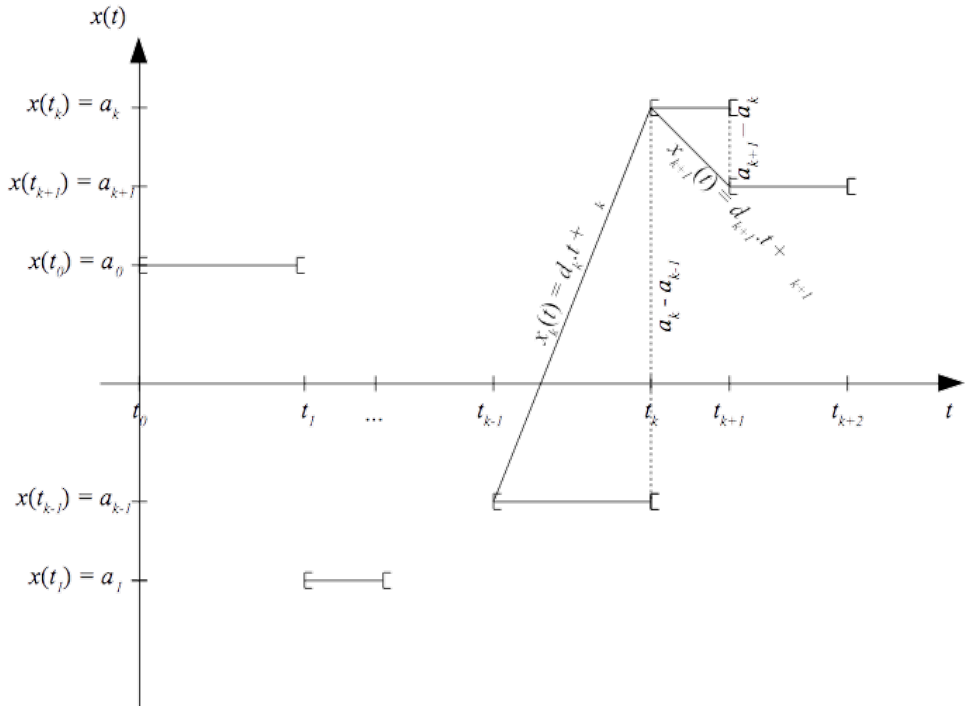


Figure A.6: Derivative of a piece wise function

$$d_k = \frac{x(t_k) - x(t_{k-1})}{t_k - t_{k-1}} \tag{A.42}$$

If the duration $t_k - t_{k-1}$ is small enough, the linear coefficient $d_k$ tends to the derivative $x^{'}(t_k)$ of the function $x(t)$ at the time $t_k$:

$$\lim_{t_k - t_{k-1} \to 0} d_k = \lim_{t_k - t_{k-1} \to 0} \frac{x(t_k) - x(t_{k-1})}{t_k - t_{k-1}} = x^{'}(t_k) \tag{A.43}$$

This property allows then to interprete timed observations $O(t_k) \equiv (a_k - a_{k-1}, t_k)$ as a measure of the derivative of $x^{'}(t_k)$ of a piecewise function $x(t) = \sum_{k=-\infty}^{+\infty} a_k.H(t - t_k)$.

Any sequence $\omega^{'}(t_k) = \{(a_1 - a_0, t_1), ..., (a_i - a_{i-1}, t_i), ..., (a_k - a_{k-1}, t_k)\}$ can be then interpreted as the result of the convolution product $x^{'}(t_k) = x^{'}(t) * \delta(\Gamma^{'})$ where $x^{'}(t) = \sum_{k=-\infty}^{+\infty}(a_k - a_{k-1}).H(t - t_k)$ is the derivative of the piecewise function $x(t) = \sum_{k=-\infty}^{+\infty} a_k.H(t - t_k)$.

### A.4.3 Piecewise Function Evolution

The *Tetrahedron of States* formalized by Rosenberg and Karnopp [REFFF] describes the relations between four generalized continuous variables from Newtonian Physics (see figure A.7):

- the effort $e$;
- the flow $f$;
- the impulse $p$;
- the displacement $q$.

Let us consider the unary observer $\Theta(\mathbb{N}, \{0\})$, implementing equation A.26, applied on a piecewise function $x(t) = \sum_{k=-\infty}^{+\infty} a_k.H(t - t_k)$. Such a unary observer generates a stochastic

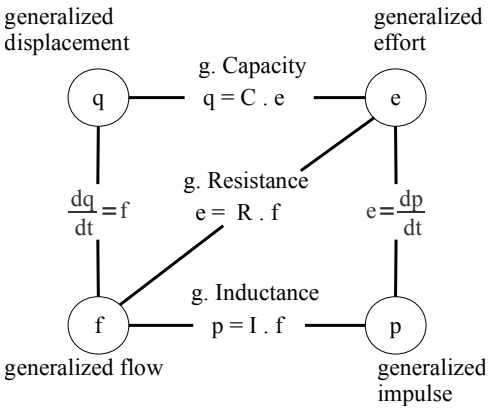


Figure A.7: Tetrahedron of states

clock $\Gamma = \{t_k, t_k \in \Re\}$ and a sequence of timed observations $\omega(t_k) = \{O(t_k), t_k \in \Gamma\}$ where $O(t_k) \equiv (a_k, t_k)$.

Let us consider the *observation window* $w(t_i) = \{\{x(t_{i-(n-1)}), x(t_{i-(n-2)}), ..., x(t_i)\}\}$, $t_i \geq t_k$, containing the $n$ last known values of $x(t)$. On this *observation window*, the observer $\Theta(\mathbb{N}, \{0\})$ generates, at the time $t_i$, a sequence of timed observations $\omega(i) = \{O(t_{k-(m-1)}), ..., O(t_k)\} \subseteq \omega(t_k)$ containing the $m$ most recent observations of $\omega(t_k)$.

In the *Tetrahedron of States* framework, the *flow* $\lambda(i)$ of a sequence of timed observations $\omega(t_k)$ at time $t_i, t_i \geq t_k$ is the ratio between the number of timed observations in $\omega(i)$ and the number of values contained in the *observation window* $w(t_i)$:

$$\lambda(i) = \begin{cases} \frac{Card(\omega(i))}{Card(w(t_i))} & \text{if } Card(w(t_i)) \neq 0 \\ 0 & \text{if } Card(w(t_i)) = 0 \end{cases} \tag{A.44}$$

The flow $\lambda(i)$ is then a $[0; 1]$-valued function that can be interpreted as a percentage. Let us consider the concrete unary observer $\Theta_\lambda(\Delta_\lambda, \Psi_\lambda, n_\lambda)$, where $\Delta_\lambda = \{0, 1, 2\}$ and $\Psi_\lambda = \{\psi_1, \psi_2\}$ implementing equations A.14 and A.15, applied on $\lambda(i)$. Each constant of $\Delta_\lambda$ links a range of values defined by an interval $I_i$ such as:

- $0 \leftrightarrow I_1 \equiv ]-\infty, \psi_1[$;
- $1 \leftrightarrow I_2 \equiv [\psi_1, \psi_2[$;
- $2 \leftrightarrow I_3 \equiv [\psi_2, +\infty[$.

Such a unary observer generates a stochastic clock $\Gamma_\lambda = \{t_i, t_i \in \Re\}$ and a sequence of timed observations of the form:

$$\omega_\lambda(t_i) = \{O_\lambda(t_i) \equiv (\delta_\lambda, t_i), \delta_\lambda \in \Delta_\lambda, t_i \in \Gamma_\lambda\} \tag{A.45}$$

The thresholds $\psi_1$ and $\psi_2$ must correctly be built so the following interpretations can be made:

- a timed observation of the form $O_\lambda(t_i) \equiv (0, t_k)$ means that the number of timed observations in $\omega(t_k)$ is *weak*;
- a timed observation of the form $O_\lambda(t_i) \equiv (1, t_k)$ means that the number of timed observations in $\omega(t_k)$ is *normal*;

- a timed observation of the form $O_\lambda(t_i) \equiv (2, t_k)$ means that the number of timed observations in $\omega(t_k)$ is *high*.

On figure A.8 are represented the *flow* $\lambda(i)$ computed from the function $x(t)$ seen on figure A.1 and the sequence of timed observations generated by the concrete unary observer $\Theta_\lambda(\Delta_\lambda, \Psi_\lambda, n_\lambda)$ applied on that flow $\lambda(i)$ whose parameters are set to $\psi_1 = 0.20$, $\psi_2 = 0.50$ and $n_\lambda = 30$.

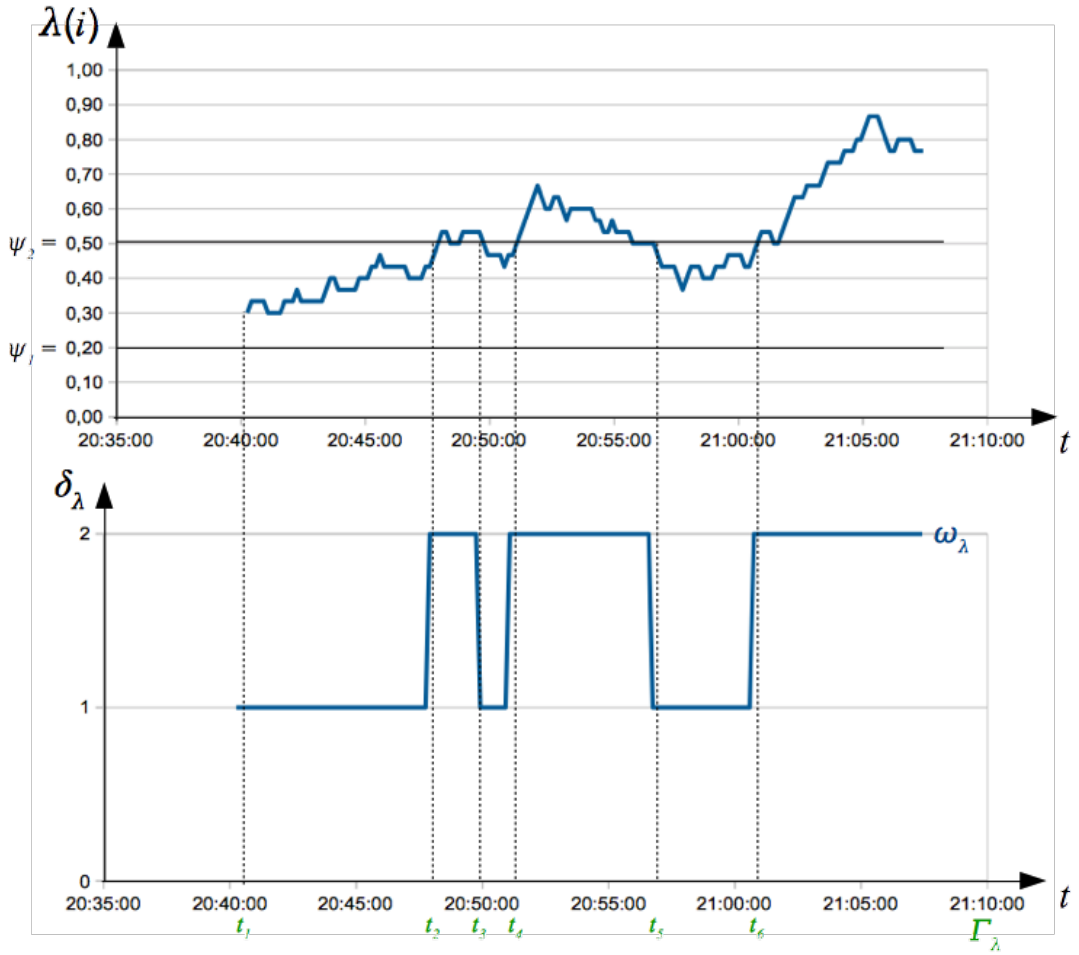


Figure A.8: Sequence of timed observations $\omega_\lambda(t_i)$ generated by a concrete unary observer applied of the *flow* $\lambda(i)$ computed from the function $x(t)$

Table A.5 sums up the interpretation made from timed observations generated by such a unary observer $\Theta_\lambda(\Delta_\lambda, \Psi_\lambda, n_\lambda)$:

| Timed Observation | Interpretation |
|---|---|
| $(1, t_1)$ | At $t_1 = 20:45:15$, the number of timed observations in $\omega(t_k)$ is *normal* |
| $(2, t_2)$ | At $t_2 = 20:47:55$, the number of timed observations in $\omega(t_k)$ is *high* |
| $(1, t_3)$ | At $t_3 = 20:49:55$, the number of timed observations in $\omega(t_k)$ is *normal* |
| $(2, t_4)$ | At $t_4 = 20:51:05$, the number of timed observations in $\omega(t_k)$ is *high* |
| $(1, t_5)$ | At $t_5 = 20:56:45$, the number of timed observations in $\omega(t_k)$ is *normal* |
| $(2, t_6)$ | At $t_6 = 21:00:45$, the number of timed observations in $\omega(t_k)$ is *high* |

Table A.3: Interpretation made from timed observations generated by concrete unary observers $\Theta_\lambda(\Delta_\lambda, \Psi_\lambda, n_\lambda)$

### A.4.3.1 Effort of a Sequence of Timed Observations

Let us consider the unary observer $\Theta^{'}(\mathbb{N}, \{0\})$, implementing equation A.36, applied on a piecewise function $x(t) = \sum_{k=-\infty}^{+\infty} a_k.H(t - t_k)$. Such a unary observer generates a stochastic clock $\Gamma = \{t_k, t_k \in \mathfrak{R}\}$ and a sequence of timed observations $\omega(t_k) = \{O(t_k), t_k \in \Gamma\}$ where $O(t_k) \equiv (a_k - a_{k-1}, t_k)$.

Let us consider the *observation window* $w(t_i) = \{\{x(t_{i-(n-1)}), x(t_{i-(n-2)}), ..., x(t_i)\}\}$, $t_i \geq t_k$, containing the $n$ last known values of $x(t)$. On this *observation window*, the observer $\Theta^{'}(\mathbb{N}, \{0\})$ generates, at the time $t_i$, a sequence of timed observations $\omega(i) = \{O(t_{k-(m-1)}), ..., O(t_k)\} \subseteq \omega(t_k)$ containing the $m$ most recent observations of $\omega(t_k)$. By construction, $\omega(i)$ defines a sequence of values $\{a_{k-(m-1)} - a_{k-m}, ..., a_k - a_{k-1}\}$ such as:

$$\sum_{i=0}^{m-1} (a_k - a_{k-i}) = a_k - a_{k-m} \tag{A.46}$$

In the *Tetrahedron of States* framework, the *effort* $\mu(i)$ of a sequence of timed observations $\omega(t_k)$ generated by a unary observer observing the derivative of a piecewise function during the *observation window* $w(t_i), t_i \geq t_k$ is:

$$\mu(i) = \begin{cases} Card(\omega(i)) \cdot \frac{|a_{k-m}|}{a_k - a_{k-m}} & \text{if } a_k - a_{k-m} \neq 0 \text{ and } a_{k-m} \neq 0 \\ Card(\omega(i)) \cdot \frac{1}{a_k - a_{k-m}} & \text{if } a_k - a_{k-m} \neq 0 \end{cases} \tag{A.47}$$

The *effort* $\mu(i)$ is then a $\mathfrak{R}$-valued function. If $\mu(i) > 0$, this can be interpreted as a percentage of an *upward* evolution of the function $x(t)$ relatively to the reference value $x(t_{k-m}) = a_{k-m}$. If $\mu(i) < 0$, this can be interpreted as a percentage of a *downward* evolution of the function $x(t)$ relatively to the reference value $x(t_{k-m}) = a_{k-m}$.

Let us consider the concrete unary observer $\Theta_\mu(\Delta_\mu, \Psi_\mu, n_\mu)$, where $\Delta_\mu = \{-2, -1, 0, 1, 2\}$ and $\Psi_\mu) = \{\psi_{-2}, \psi_{-1}, \psi_1, \psi_2\}$ implementing equations A.14 and A.15, applied on $\mu(i)$. Each constant of $\Delta_\mu$ links a range of values defined by an interval $I_i$ such as:

- $-2 \leftrightarrow I_{-2} \equiv ]-\infty, \psi_{-2}[$;
- $-1 \leftrightarrow I_{-1} \equiv [\psi_{-2}, \psi_{-1}[$;
- $0 \leftrightarrow I_0 \equiv [\psi_{-1}, \psi_1[$;
- $1 \leftrightarrow I_2 \equiv [\psi_1, \psi_2[$;
- $2 \leftrightarrow I_3 \equiv [\psi_2, +\infty[$.

Such a unary observer generates a stochastic clock $\Gamma_\mu = \{t_i, t_i \in \mathfrak{R}\}$ and a sequence of timed observations of the form:

$$\omega_\mu(t_i) = \{O_\mu(t_i) \equiv (\delta_\mu, t_i), \delta_\mu \in \Delta_\mu, t_i \in \Gamma_\mu\} \tag{A.48}$$

The thresholds $\psi_i \in \Psi_\mu$ must correctly be built so the following interpretations can be made:

- a timed observation of the form $O_\mu(t_k) \equiv (-2, t_k)$ means a *strong downward effort*;
- a timed observation of the form $O_\mu(t_k) \equiv (-1, t_k)$ means a *downward effort*;

- a timed observation of the form $O_\mu(t_k) \equiv (0, t_k)$ means a *stable effort*;
- a timed observation of the form $O_\mu(t_k) \equiv (1, t_k)$ means an *upward effort*;
- a timed observation of the form $O_\mu(t_k) \equiv (2, t_k)$ means a *strong upward effort*.

On figure A.9 are represented the *effort* $\mu(i)$ computed from the function $x(t)$ seen on figure A.1 and the sequence of timed observations generated by the concrete unary observer $\Theta_\mu(\Delta_\mu, \Psi_\mu, n_\mu)$ applied on that *effort* $\mu(i)$ whose parameters are set to $\psi_{-2} = -0.60$, $\psi_{-1} = -0.20$, $\psi_1 = 0.20$, $\psi_2 = 0.60$ and $n_\mu = 30$.

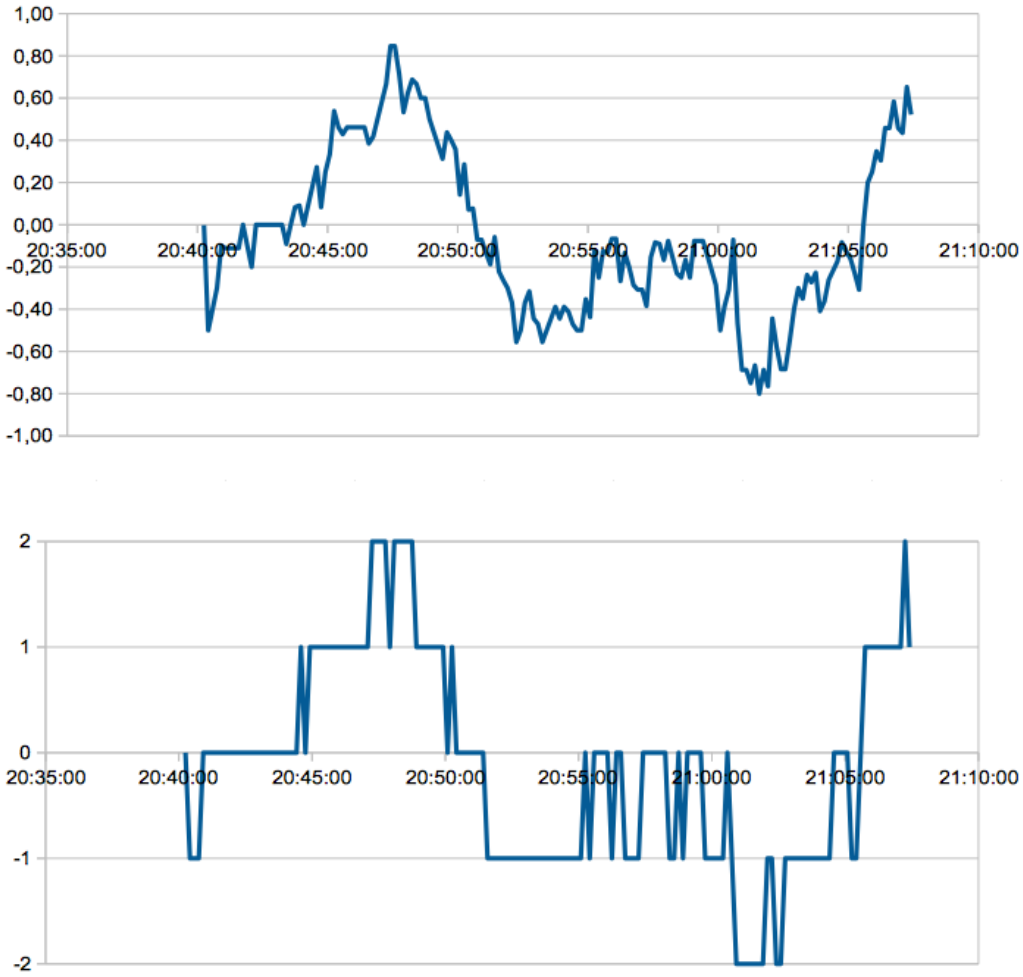


Figure A.9: Sequence of timed observations $\omega_\mu(t_i)$ generated by a concrete unary observer applied of the *effort* $\mu(i)$ computed from the function $x(t)$

### A.4.3.2 Power of a Sequence of Timed Observations

Let us consider the unary observer $\Theta'(\mathbb{N}, \{0\})$, implementing equation A.36, applied on a piecewise function $x(t) = \sum_{k=-\infty}^{+\infty} a_k.H(t - t_k)$. Such a unary observer generates a stochastic clock $\Gamma = \{t_k, t_k \in \Re\}$ and a sequence of timed observations $\omega(t_k) = \{O(t_k), t_k \in \Gamma\}$ where $O(t_k) \equiv (a_k - a_{k-1}, t_k)$.

Let us consider the *observation window* $w(t_i) = \{\{x(t_{i-(n-1)}), x(t_{i-(n-2)}), ..., x(t_i)\}\}$, $t_i \geq t_k$, containing the $n$ last known values of $x(t)$. On this *observation window*, the observer $\Theta'(\mathbb{N}, \{0\})$

generates, at the time $t_i$, a sequence of timed observations $\omega(i) = \{O(t_{k-(m-1)}), ..., O(t_k)\} \subseteq \omega(t_k)$ containing the $m$ most recent observations of $\omega(t_k)$. By construction, $\omega(i)$ defines a sequence of values $\{a_{k-(m-1)} - a_{k-m}, ..., a_k - a_{k-1}\}$ such as:

$$\sum_{i=0}^{m-1}(a_k - a_{k-i}) = a_k - a_{k-m} \tag{A.49}$$

The *power* $\nu(i)$ of a sequence of timed observations $\omega(t_k)$ is the product of the *flow* $\lambda(i)$ and the *effort* $\mu(i)$:

$$\nu(i) = \lambda(i) \times \mu(i) \tag{A.50}$$

| | $\lambda$ | | |
|---|---|---|---|
| $\mu$ | **0** | **1** | **2** |
| **-2** | 0 | -3 | -4 |
| **-1** | 0 | -1 | -2 |
| **0** | 0 | 0 | 0 |
| **1** | 0 | 1 | 2 |
| **2** | 0 | 3 | 4 |

Table A.4: Building of constants

| Timed Observation | Interpretation |
|---|---|
| $(1, t_1)$ | At $t_1 = 20:45:15$, the number of timed observations in $\omega(t_k)$ is *normal* |
| $(2, t_2)$ | At $t_2 = 20:47:55$, the number of timed observations in $\omega(t_k)$ is *high* |
| $(1, t_3)$ | At $t_3 = 20:49:55$, the number of timed observations in $\omega(t_k)$ is *normal* |
| $(2, t_4)$ | At $t_4 = 20:51:05$, the number of timed observations in $\omega(t_k)$ is *high* |
| $(1, t_5)$ | At $t_5 = 20:56:45$, the number of timed observations in $\omega(t_k)$ is *normal* |
| $(2, t_6)$ | At $t_6 = 21:00:45$, the number of timed observations in $\omega(t_k)$ is *high* |

Table A.5: Timed Observations From Flow Interpretation

The *effort* $\nu(i)$ is then a $\Re$-valued function. Let us consider the concrete unary observer $\Theta_\nu(\Delta_\nu, \Psi_\nu, \psi_d, n)$, where $\Delta_\nu = \{-4, -3, -2, -1, 0, 1, 2, 3, 4\}$ and $\Psi_\nu) = \{\psi_{-4}, \psi_{-3}, \psi_{-2}, \psi_{-1}, \psi_1, \psi_2, \psi_3, \psi_4\}$ implementing equations A.14 and A.15, applied on $\nu(i)$. Each constant of $\Delta_\nu$ links a range of values defined by an interval $I_i$ such as:

- $-4 \leftrightarrow I_{-4} \equiv ]-\infty, \psi_{-4}[$;
- $-3 \leftrightarrow I_{-3} \equiv [\psi_{-4}, \psi_{-3}[$;
- $-2 \leftrightarrow I_{-2} \equiv [\psi_{-3}, \psi_{-2}[$;
- $-1 \leftrightarrow I_{-1} \equiv [\psi_{-2}, \psi_{-1}[$;
- $0 \leftrightarrow I_0 \equiv [\psi_{-1}, \psi_1[$;
- $1 \leftrightarrow I_2 \equiv [\psi_1, \psi_2[$;
- $2 \leftrightarrow I_2 \equiv [\psi_2, \psi_3[$;
- $3 \leftrightarrow I_2 \equiv [\psi_3, \psi_4[$;

- $4 \leftrightarrow I_3 \equiv [\psi_4, +\infty[$.

Such a unary observer generates a stochastic clock $\Gamma_\nu = \{t_i, t_i \in \Re\}$ and a sequence of timed observations of the form:

$$\omega_\nu(t_i) = \{O_\nu(t_i) \equiv (\delta_\nu, t_i), \delta_\nu \in \Delta_\nu, t_i \in \Gamma_\nu\} \tag{A.51}$$

The thresholds $\psi_i \in \Psi_\nu$ must correctly be built so the following interpretations can be made:

- a timed observation of the form $O_\nu(t_k) \equiv (-4, t_k)$ means a *very strongly negative power*;
- a timed observation of the form $O_\nu(t_k) \equiv (-3, t_k)$ means a *strongly negative power*;
- a timed observation of the form $O_\nu(t_k) \equiv (-2, t_k)$ means a *negative power*;
- a timed observation of the form $O_\nu(t_k) \equiv (-1, t_k)$ means a *weakly negative power*;
- a timed observation of the form $O_\nu(t_k) \equiv (0, t_k)$ means a *weak power*;
- a timed observation of the form $O_\nu(t_k) \equiv (1, t_k)$ means a *weakly positive power*;
- a timed observation of the form $O_\nu(t_k) \equiv (2, t_k)$ means a *positive power*;
- a timed observation of the form $O_\nu(t_k) \equiv (3, t_k)$ means a *strongly positive power*;
- a timed observation of the form $O_\nu(t_k) \equiv (4, t_k)$ means a *very strongly positive power*.

This leads to get informations about the evolution of the function $x(t)$. The *power* $\nu(i) = \lambda(i) \times \mu(i)$ and $\lambda(i) \geq 0$, so:

$$\begin{cases} \nu(i) > 0 \Leftrightarrow \mu(i) > 0 \\ \nu(i) < 0 \Leftrightarrow \mu(i) < 0 \end{cases} \tag{A.52}$$

If $\mu(i) > 0$ that can be interpreted as a percentage of an *upward* evolution of the function $x(t)$, during the *observation window* $w(t_k)$, relatively to the reference value $x(t_{k-m}) = a_{k-m}$. If $\mu(i) < 0$ that can be interpreted as a percentage of a *downward* evolution of the function $x(t)$, during the *observation window* $w(t_k)$, relatively to the reference value $x(t_{k-m}) = a_{k-m}$.

So, during the *observation window* $w(t_k)$:

- a timed observation of the form $O_\nu(t_k) \equiv (-4, t_k)$ means the function $x(t)$ is *very strongly decreasing*;
- a timed observation of the form $O_\nu(t_k) \equiv (-3, t_k)$ means the function $x(t)$ is *strongly decreasing*;
- a timed observation of the form $O_\nu(t_k) \equiv (-2, t_k)$ means the function $x(t)$ is *decreasing*;
- a timed observation of the form $O_\nu(t_k) \equiv (-1, t_k)$ means the function $x(t)$ is *weakly decreasing*;
- a timed observation of the form $O_\nu(t_k) \equiv (0, t_k)$ means the function $x(t)$ is *flat*;

- a timed observation of the form $O_\nu(t_k) \equiv (1, t_k)$ means the function $x(t)$ is *weakly increasing*;
- a timed observation of the form $O_\nu(t_k) \equiv (2, t_k)$ means the function $x(t)$ is *increasing*;
- a timed observation of the form $O_\nu(t_k) \equiv (3, t_k)$ means the function $x(t)$ is *strongly increasing*;
- a timed observation of the form $O_\nu(t_k) \equiv (4, t_k)$ means the function $x(t)$ is *very strongly increasing.*

On figure A.10 are represented the *power* $\nu(i)$ and the sequence of timed observations generated by the concrete unary observer $\Theta_\nu(\Delta_\nu, \Psi_\nu, n_\nu)$ applied on that *power* $\nu(i)$ whose parameters are set to $\psi_{-4} = -0.30$, $\psi_{-3} = -0.12$, $\psi_{-2} = -0.10$, $\psi_{-1} = -0.04$, $\psi_1 = 0.04$, $\psi_2 = 0.10$, $\psi_3 = 0.12$, $\psi_4 = 0.30$ and $n_\nu = 30$.

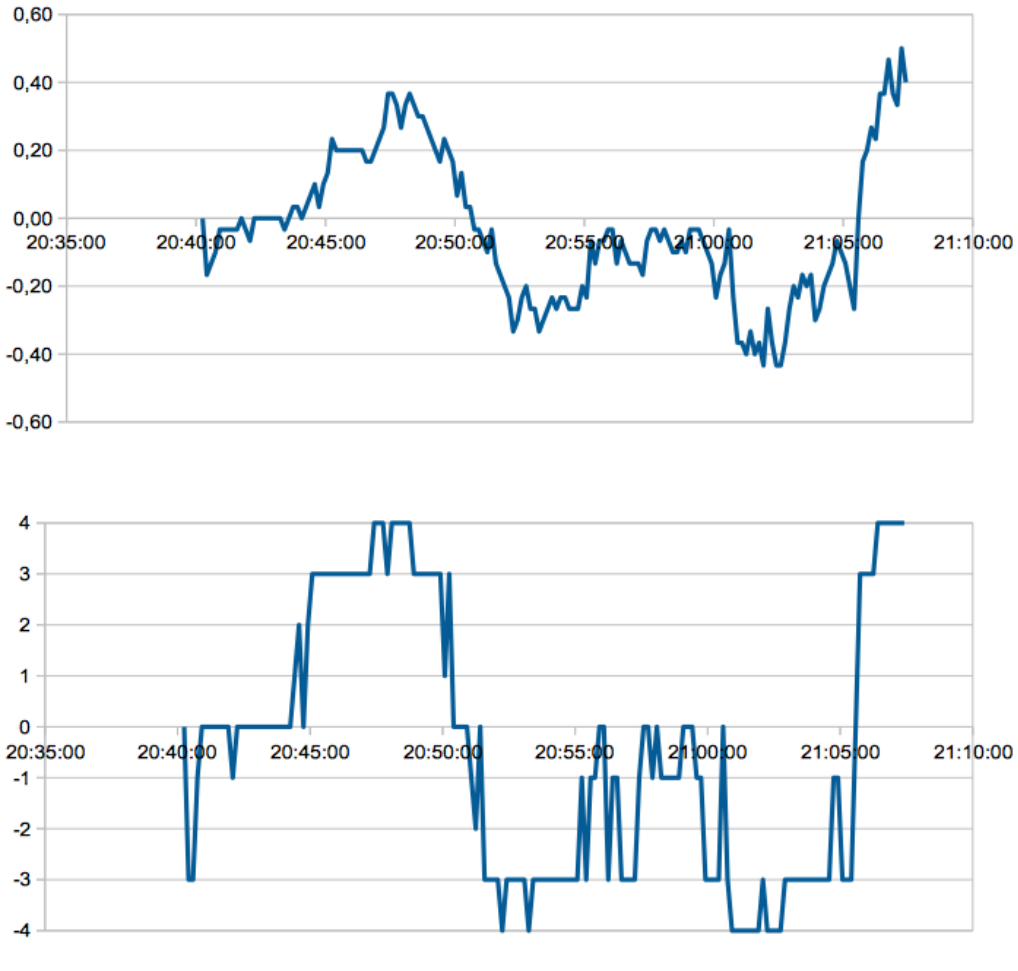


Figure A.10: Sequence of timed observations $\omega_\nu(t_i)$ generated by a concrete unary observer applied of the *power* $\nu(i)$ computed from the function $x(t)$

CHAPTER B

# Operationalization of the Blending and the Levels of Abstraction Theories with the Timed Observations Theory

## B.1 INTRODUCTION

With the always increasing amount of data collected over things connected on information networks, the need for data and knowledge analysis became a crucial stake for most of the industrial and service activities, including the research activity itself. The main difficulty with data and knowledge analysis resides in the introduction, in a controlled way, of semantics in the syntactic patterns provided by human analysts with the eventual help of Statistic Learning or Data Mining algorithms. There is then a crucial need for models able to guide a rationale interpretation of data providing from humans or machines.

To this aim, [FT98] proposed a theory of *Conceptual Integration Networks*, also called the *Blending Theory*. This theory defines a common conceptual operation, the *blending of conceptual spaces*, to provide a meaning and a way to compress the representations that are useful for knowledge memorization and manipulation. Blending of different conceptual spaces *plays a fundamental role in the construction of meaning in everyday life, in the arts and sciences, and especially in the social and behavioral sciences* [FT03]. The essence of the conceptual blending operation is to establish a new conceptual space through the matching between the contain of different conceptual spaces. Fauconnier and Turner suggest that *the capacity for complex conceptual blending is the crucial capacity needed for thought and language* [FT03].

Another but complementary point of view is proposed in [Flo08, Flo10] to address the problem of defining the nature of natural, human or artificial agents with the notion of *Level of Abstraction*. The *Levels of Abstraction Theory* aims to clarify implicit assumptions and to allow the resolution of possible conceptual confusions with the comparisons between different point of view about the same phenomenon (concrete or abstract). Similarly to the *Blending Theory*, it *provides a detailed and controlled way of comparing analyses and models* [Flo08] with the introduction of multiple levels of abstraction in conceptual analysis. It constitutes then a crucial and powerful tool to address the analysis and the modeling of the phenomenon under consideration. Floridi argued that *for discrete systems, whose observables take on only finitely-many values, the method is indispensable* [Flo08].

These two theories share common goals but develop different ways to achieve them, and both lack of mathematical foundations. The aim of this paper is to propose an adequate mathematical

framework that provides for the first time, up to our knowledge, a strong formal foundation to these theories. This mathematical framework is build on the *Timed Observations Theory* (TOT, [LeG06]), which constitutes the basis of a new abstraction approach called the *Timed Observation Method for Abstraction* (Tom4A). Clearly, our long term goal is to develop software tools able to discover and to model knowledge representations from sets of timed data so that the human interpretation is intuitive, immediate and independent of the learning and the modeling tools.

To make the mathematical framework as simple and intuitive as possible, the main concepts are illustrated with a running example of three speakers discussing about a vehicle (cf. Section B.2), the original text coming from the web site of the *Society for the Philosophy of Information* (http://www.socphilinfo.org/node/150). Section B.3 provides the principles and the formal modeling tools of the TOT that will be used all along this paper. Section B.4 describes the building of the conceptual spaces of the three speakers. Section B.5 introduces the blending process to build the common model used by the speakers to understand together. Section B.6 define the notion of generic conceptual space. This section ends the introduction of the basic modeling elements of the *Blenbing Theory*. Section B.7 introduces the basis of the *Levels of Abstraction Theory* that will be used to add an inference structure to the blended and the generic conceptual spaces and to organize them with both a *disjoint* and a *nested* gradients of abstraction. This section shows also that the formalized notion of *gradients of abstraction* constitute a powerful tool to capture and to represent the meaning in a coherent and formal way. Finally, section B.8 proposes a short synthesis of Tom4A approach, and provide some insights about our future works.

## B.2 RUNNING EXAMPLE

In this section, only the factual elements of the running example are given in *verbatim*, its analysis according to the *Levels of Abstraction Theory* being available in http://www.socphilinfo.org/node/150:

*Suppose we join Alice, Bob, and Carol earlier on at the party. They are in the middle of a conversation. We do not know the subject of their conversation, but we are able to hear this much:*

- *Alice observes that its (whatever "it" is) old engine consumed too much, that it has a stable market value but that its spare parts are expensive.*
- *Bob observes that its engine is not the original one, that its body has been recently re-painted but that all leather parts are very worn;*
- *Carol observes that it has an anti-theft device installed, is kept garaged when not in use, and has had only a single owner;*

The point to notice in this conversation is the fact the three speakers *observe* properties about an unknown, for us, *system*. The three speakers exchanges then *observations* about the system. Since Aristotle, we know that this type of discourse can be resumed with a set of apophantic formulas of the form *Subject1-Copula-Subject2*. Each observation is then a *proposition*, that is to says a *relation* between two subjects, the nature of the relation being defined with a verb (the copula) linking the two subjects. For example, Alice's observation *it has a stable market*

*value* is a proposition that can be represented with the binary predicate *is* linking the subject $MarketValue$ and the other subject *stable*: $is(MarketValue, stable)$. So, the conversation can be re-written to make clear the different observations of each speakers:

1. Alice: its engine is old.
2. Alice: its engine consumed too much.
3. Alice: it has a stable market value.
4. Alice: its spare parts are expensive.
5. Bob: its engine is not the original one.
6. Bob: its body has been recently re-painted.
7. Bob: all leather parts are very worn.
8. Carol: its anti-theft device is installed.
9. Carol: it is kept garaged when not in use.
10. Carol: it has had only a single owner.

The conversation is then an exchange of 10 propositions where each but the $10^{th}$ can be easily formalized with a binary predicate. Because of the use of the *when* connector, the observation 9 links two propositions: $is(it, not\_in\_use)$ and $is\_kept(it, garaged)$. Yet, numerous of these observations use verbs conjugated to the past, meaning that the observations have a time reference. Temporal versions of the first order Predicate Logic would then be used to formalize such observations but the interpretation of the resulting formulas is accessible for only specialists of these logics.

To keep an intuitive interpretation of the formulas, Tom4A uses the modeling principles of the Timed Observations Theory that are introduced in the next section. These principles will be applied (i) to build the conceptual spaces of the three speakers, (ii) to define of the corresponding blended and generic spaces and (iii) to model the blended space with a Gradient of Abstraction (GoA) according to [Flo08]).

## B.3 MODELING WITH THE TOT

The Timed Observations Theory (TOT) provides a mathematical framework to model dynamic processes from timed data. The TOT is currently the mathematical basis of the TOM4L (Timed Observation Mining For Learning) Knowledge Discovering from Databases process [LBP15, LA12], and the TOM4D (Timed Observation Modeling for Diagnosis) Knowledge Engineering methodology [PLG14]. The Tom4A method aims at introducing levels of abstraction and generic spaces [LGG04] in TOM4L and Tom4D according to the notion of *conceptual equivalence* [ZGF06b].

The aim of the TOT is to model an *observed process* defined as a couple $(X(t), \Theta(X, \Delta))$ where $X(t)$ is an arbitrarily constituted set $X(t) = \{x_1(t), ..., x_{n_X}(t)\}$ of $n_X$ timed functions

$x_i(t)$ of continuous time $t$ (the dynamic process), $X = \{x_1, x_2, ..., x_{n_X}\}$ is the set of the $n_X$ variable names $x_i$ corresponding to each time functions $x_i(t)$ and $\Theta(X, \Delta)$ is an *observation program* implemented in a human or a computer, the set $\Delta = \{\delta_j\}$ being a set of *constant* values. A dynamic process $X(t)$ is said to be *observed* by a program $\Theta(X, \Delta)$ when this latter aims at writing *timed observations* describing the *modifications* over time of the functions $x_i(t)$ of $X(t)$:

**Definition B.1** *Timed Observation*
*Let $\Gamma = \{t_k\}_{t_k \in \Re}$ be a set of arbitrary timestamps $t_k$ at which $\Theta(X, \Delta)$ observes a time function $x_i(t) \in X(t)$ and $\theta(x_\theta, \delta_\theta, t_\theta)$ be a predicate implicitly determined by $\Theta(X, \Delta)$;*

*A timed observation $(\delta_j, t_k) \in \Delta \times \Gamma$ made on $x_i(t)$ is the assignation of values $x_i$, $\delta_j$ and $t_k$ to the predicate $\theta(x_\theta, \delta_\theta, t_\theta)$ such that $\theta(x_i, \delta_j, t_k)$.*

For example, Alice's observation *it has a stable market value* is represented with the timed observation $(stable, t_k)$, $t_k$ being the (unknown) timestamps of the instant where Alice pronounces this sentence during the conversation. So, Alice play the role of the observation program $\Theta(X, \Delta)$ and the assigned ternary predicate $is(MarketValue, Stable, t_k)$, corresponding to $\theta(x_i, \delta_j, t_k)$, provides a *meaning* to the timed observation $(stable, t_k)$. The explicit link between a variable $x_i$ $(MarketValue)$ and a constant $\delta_j$ $(stable)$ is made with the notion of *observation class*:

**Definition B.2** *Observation Class*
*Let $X = \{x_i\}_{i=1...n_{n_X}}$ be the set of variable names corresponding to $X(t)$ and $\Delta = \{\delta_j\}$ a set of constant values an observation program $\Theta(X, \Delta)$ can use.*

*An observation class $O_k = \{..., (x_i, \delta_j), ...\}$ for $\Theta(X, \Delta)$ is a subset of $X \times \Delta$.*

Any association establishing a mapping $\Delta \mapsto X$ for each $\delta_j$ of $\Delta$ can be made. The simplest way, and the most used, to define observation classes is the use of singletons $O_j = \{(x_i, \delta_j)\}$ where the pair $(x_i, \delta_j)$ is the unique element the set $O_j$. For example, the observation class $O_s^A = \{(MarketValue, stable)\}$ has been implicitly used by Alice to reason about the system (i.e. *it*). It is then obvious that doing so, all but the observation 9 (*it is kept garaged when not in use*) are occurrences of a particular observation class, the observations 9 linking together two occurrences of two different observation classes:

1. Alice, *its engine is old*:
   $O_4^A(t_1) \equiv (old, t_1)$, $O_4^A = \{x_4^A, old\}$.

2. Alice, *its engine consumed too much*:
   $O_6^A(t_2) \equiv (too\_much, t_2)$, $O_6^A = \{x_6^A, too\_much\}$.

3. Alice, *it has a stable market value*:
   $O_2^A(t_3) \equiv (stable, t_3)$, $O_2^A = \{x_2^A, stable\}$.

4. Alice, *its spare parts are expensive*:
   $O_8^A(t_4) \equiv (expensive, t_4)$, $O_8^A = \{x_8^A, expensive\}$.

5. Bob, *its engine is not the original one*:
   $O_1^B(t_5) \equiv (original, t_5)$, $O_1^B = \{x_1^B, original\}$.

6. Bob, *its body has been recently re-painted*:
   $O_3^B(t_6) \equiv (recently, t_6)$, $O_3^B = \{x_3^B, recently\}$.

7. Bob, *all leather parts are very worn*:
   $O_5^B(t_7) \equiv (very\_worn, t_7)$, $O_5^B = \{x_5^B, very\_worn\}$.

8. Carol, *its anti-theft device is installed*:
   $O_1^C(t_8) \equiv (installed, t_8)$, $O_1^C = \{x_1^C, installed\}$.

9. Carol, *it is kept garaged when not in use*:
   $O_3^C(t_{10}) \equiv (garaged, t_{10})$, $O_3^C = \{x_3^C, garaged\}$,
   $O_2^C(t_{11}) \equiv (not\_in\_use, t_{11})$, $O_2^C = \{x_2^C, not\_in\_use\}$.

10. Carol, *it has had only a single owner*:
    $O_5^C(t_{12}) \equiv (single, t_{12})$, $O_5^C = \{x_5^C, single\}$.

Carol's observation number 9 defines two timed observations, $O_2^C(t_{11}) \equiv (not\_in\_use, t_{11})$ and $O_3^C(t_{10}) \equiv (garaged, t_{12})$, corresponding to two observation classes $O_2^C = \{x_2^C, not\_in\_use\}$ and $O_3^C = \{x_3^C, garaged\}$. One of the interests of the timestamps $t_k$ of a timed observation is to provide a temporal reference in a flow of observation. Carol's meaning of the term *when* being unclear, the timestamps allows to provide a meaning to it: the *when* is can be interpreted as a reference to the past. In other words, the observation 9 is interpreted as: *when it is not in use, it is kept garaged*): the status of the usage of *it* must be defined *before* the assertion of the location. The TOT defines the notion of *timed binary relation* to represent a *sequential* relation between two observations classes $O_i$ and $O_j$:

**Definition B.3** *Temporal Binary Relation*
*A temporal binary relation* $r_{ij}(O_i, O_j, [\tau_{ij}^-, \tau_{ij}^+])$, $\tau_{ij}^- \in \Re$, $\tau_{ij}^+ \in \Re$, *is an oriented relation between two observation classes* $O_i$ *and* $O_j$ *that is timed constrained with the* $[\tau_{ij}^-, \tau_{ij}^+]$ *interval.*

This definition leads to define Carol's observation number 9 with the timed binary relation $r_{23}^C(O_2^C, O_3^C, [0, \tau_{23}^+])$, $\tau_{23}^- = 0$ meaning that the end of the usage of *it* can coincide with the beginning of the put in the garage (i.e. may be $t_{10} = t_{11}$). This example suffers to provide an intuitive comprehension of Tom4D's operational definition of *knowledge* (cf. [PLG14] for a justification of this definition):

**Definition B.4** *Any relation logically consistent with a binary temporal relation of the form* $r_{ij}(C_i, C_j, [\tau_{ij}^-, \tau_{ij}^+])$ *is a piece of knowledge.*

A model being an organized set of knowledge representations, the knowledge under consideration is a set of binary relations between time functions $x_i(t)$, constants $\delta_i$ and stochastic clocks $\Gamma_i$ (cf. Figure B.1).

The notion of *dynamic function* play a pivot role in the Tom4D modeling methodology. Figure B.2 shows a graphical representation of the dynamic function $x_{i2}(t) = f(x_{i1}(t))$. The timed functions $x_{i1}(t)$ and $x_{i2}(t)$ are linked to a particular *component* $c_i$, itself being a part of the *container* of all the components, i.e. the *system* $S$. The dynamic function $x_{i2}(t) = f(x_{i1}(t))$

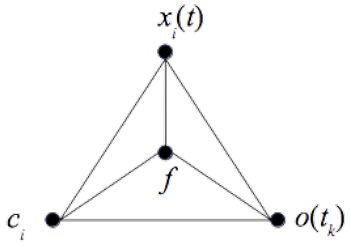


Figure B.1: Basic Concepts of TOM4D models

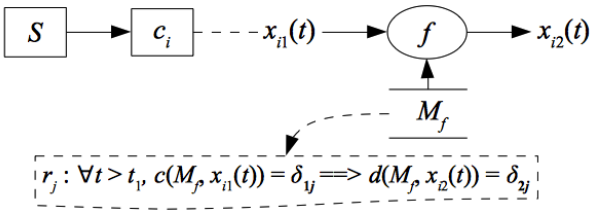


Figure B.2: Tom4D's Representation of a Dynamic Function

is defined over the Cartesian product $\Delta_{x_{i1}} \times \Delta_{x_{i2}}$ of the definition domain of the timed functions $x_{i1}(t)$ and $x_{i2}(t)$ respectively and implements a set of *decision rule* of the form:

$$\begin{aligned} &\forall \delta_{1j} \in \Delta_{x_{i1}}, \exists \delta_{2j} \in \Delta_{x_{i2}}, \\ &x_{i1}(t_k) = \delta_{1j} \overset{f}{\Longrightarrow} x_{i2}(t_k) = \delta_{2j}. \end{aligned} \tag{B.1}$$

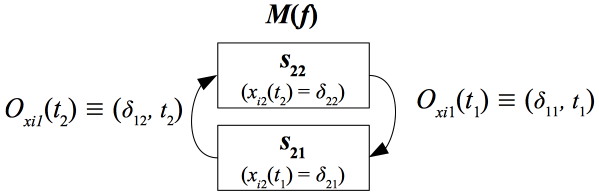


Figure B.3: Finite State Machine Model of a Tom4D Function

Such a set of decision rule specifies the *Finite State Machine* (FSM) of figure B.3 constituting the *behavioral model* of the $f$ dynamic function with $\Delta_{x_{i1}} = \{\delta_{11}, \delta_{11}\}$ and $\Delta_{x_{i2}} = \{\delta_{21}, \delta_{21}\}$. According to Tom4D, a rectangle represents a *discernible state* $s_{ij}$ labeled with a proposition about the value of one or more functions at a particular timestamps, $x_{i2}(t_1) = \delta_{21}$ for $s_{21}$ for example. An arrow represents a *transition* between two discernible states. Such a transition is conditioned with an occurrence of a particular observation class, $o_{x_{i1}} \equiv (\delta_{11}, t_1)$ for example. This means that the dynamic function $f$ implements the ternary predicate *equals* $= (x_i, \delta_j, t_k)$ of definition B.1. In other words, the semantics of the ternary predicate $\theta(x_\theta, \delta_\theta, t_\theta)$ of definition B.1 is given by the following two simple decision rules:

$$\begin{aligned} r_1 &: \forall t \geq t_1, x_{i1}(t_1) = \delta_{11} \implies x_{i2}(t) = \delta_{21} \\ r_2 &: \forall t \geq t_2, x_{i1}(t_2) = \delta_{12} \implies x_{i2}(t) = \delta_{22} \end{aligned} \tag{B.2}$$

Clearly, with boolean sets, such a FSM is not necessary, these two basic decision rules are sufficient. But generally speaking, as figure B.2 shows, a Tom4D dynamic function $x_2(t_k) = f(x_1(t_k))$ implements a *decision model* $M_f$ implementing a set of decision rules of the following general form: $r_j : \forall t \geq t_1,\ c(M_f, x_{i1}(t_1)) = \delta_{1j} \implies d(M_f, x_{i2}(t)) = \delta_{2j}$.

This formalism is necessary and sufficient for providing a formal meaning to the 11 observations of the speakers, and for building a semantic model of their conversation. The next section shows that the three speakers make a very frequent usage of simple functions that implement a two-states' FSM of the form of figure B.3.

## B.4 SPEAKERS' CONCEPTUAL SPACE

According to [FT98], *mental spaces are small conceptual packets constructed as we think and talk, for purposes of local understanding and action.* They are *very partial assemblies containing elements, and structured by frames and cognitive models. They are interconnected, and can be modified as thought and discourse unfold.*

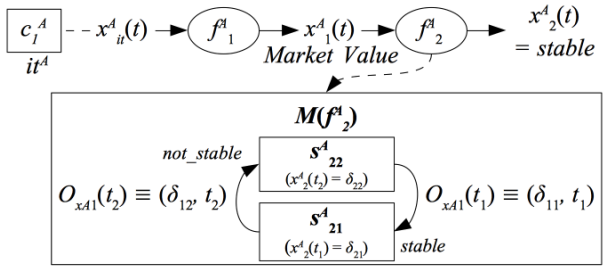


Figure B.4: Observation Number 3 (Alice)

To apply this notion, let us consider again Alice's observation *it has a stable market value.* According to the Tom4D methodology, this observation can be formalized with a *dynamic function* $x_1^A(t) = f_2^A(x_2^A(t))$ (cf. figure B.4) where $x_1^A(t)$ is the *time function* representing the *Market Value* evolution over time, and $x_2^A(t)$ is the *time function* representing Alice's assessments about the *Market Value*. The *variable* $x_1^A$ denotes then Alice's *Market Value* concept. At the particular instant where she is speaking, Alice's evaluation of the *value* of $x_2^A(t)$ equals *stable*. By construction, the definition domain of the variable $x_2^A$ is then *at least* a boolean set $\Delta_{x_2^A} = \{$ *stable*, *not_stable* $\}$, the constant *not_stable* meaning anything but *stable*. This justifies the two states FSM implemented in the $f_2^A$ assessment function.

The definition domain of the variable $x_1^A$ is unknown. Nevertheless, if we interpret the concept of the *Market Value* with a usual dictionary, we can deduce that the dimension of $x_1^A$ is an amount of money in a particular currency. This means that the definition domain of $x_1^A$ is the set $N$ of the natural numbers representing a number of cents in the implicit currency: $x_1^A \in N$. In other words, Alice's assessment function $f_2^A$ is defined over the Cartesian product $N \times \Delta_{x_2^A}$. Now, clearly, the values of the variable $x_1^A$ *must be* provided by a dynamic *measurement* function $x_1^A(t) = f_1^A(x_{it}^A(t))$. Such a function is either implemented in Alice's mind or, more surely, Alice make an implicit reference to an external function aiming at providing the *Market Value* of Alice's system $it^A$. This explains the relation, denoted with a dotted line, between the *component* labeled $c_1^A$ and the variable $x_{it}^A$ of figure B.4. This component representing the term *its* in Alice's

observation *it has a stable market value*, it formalizes Alice's notion of the system about which she talks. The component $c_1^A$ is then the container of all the components of the system. The relations between the system and its components will then be represented with a dotted line.

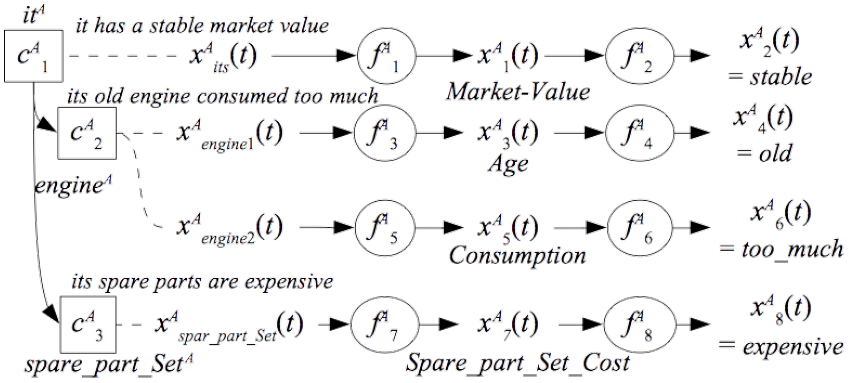


Figure B.5: Alice's Conceptual Space

Doing so for its four observations, it is simple to build a formal model of Alice's conceptual space as given in figure B.5. This figure shows that Alice's observations concerned a system $c_1^A$ is made of two components $c_2^A$, its *engine* and $c_3^A$, its *spare parts*. Two time functions are linked with the component $c_2^A$ representing Alice's notion of *engine*: $x^A_{engine1}(t)$ which is the input of the dynamic function $f_3^A$ that *counts* the *age* of $c_2^A$, and $x^A_{engine1}(t)$, the input of $f_5^A$ that *measure* the *consumption* of $c_2^A$. The component $c_3^A$ represents Alice's notion of the set of *spare part* of the system $c_1^A$. The time function $x^A_{spare\_parts\_Set\_Cost}(t) = f_7^A(x^A_{spare\_parts\_Set}(t))$ is a measurement function similar to $f_1^A$, the dynamic functions $f_4^A$, $f_6^A$ and $f_6^A$ being assessment functions similar to $f_2^A$. Obviously, these assessment functions use different decision models (the decision rules haven't been represented to simplify the figure B.5).

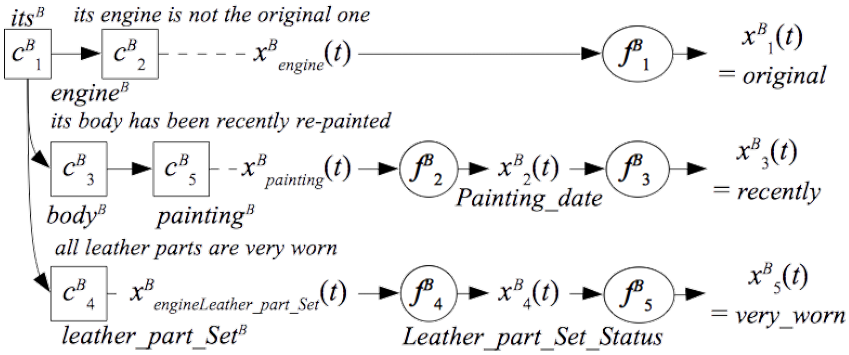


Figure B.6: Bob's Conceptual Space

Figure B.6 shows Bob's conceptual space that has been made with the same method. To understand Bob's observations, a conceptual space made with two assessment functions, $x_3^B(t) = f_3^B(x_2^B(t))$ and $x_5^B(t) = f_5^B(x_4^B(t))$, must be build where $x_2^B(t)$ represents the current painting timestamps of the system's *body* and $x_4^B(t)$ the status of the *leather parts*. The function $x_1^B(t) = f_1^B(x^B_{engine}(t))$ is a an *assertion* function allowing Bob to assert the *original* status of what Bob names the *engine* $c_2^B$. An assertion function *can* directly provide a fact (or a property) from a time function, at the opposite of an assessment function which *must* operate on the values computed with a measurement function as $f_4^B$ for $f_5^B$. The dynamic function $x_2^B(t) = f_2^B(x^B_{painting}(t))$ is

a *dating* function that provides the timestamps of the most recent painting of the system body. Finally, Bob's notion of the system is the following set of components: $C^B = \{c_1^B, c_2^B, c_3^B, c_4^B, c_5^B\}$, $c_5^B$ being linked with $c_3^B$.

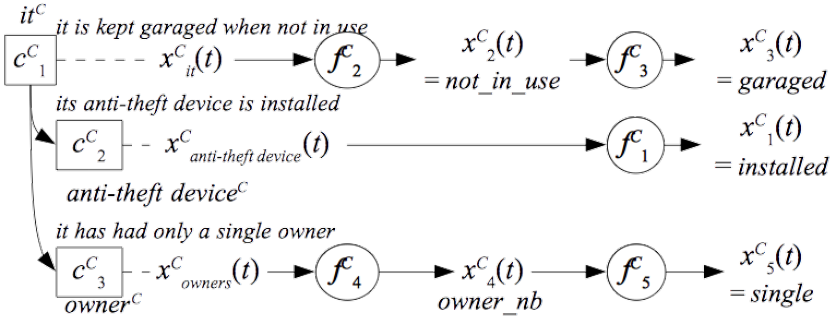


Figure B.7: Carol's Conceptual Space

Similarly, the same method leads to Carol's conceptual space of figure B.7. The interpretation of Carol's observation number 9 (i.e. *when it is not in use, it is kept garaged*) leading to the timed binary relation $r_{23}^C(O_2^C, O_3^C, [0, \tau_{23}^+])$, it is represented with two successive assertion functions: the first, $x_2^C(t) = f_2^C(x_{it}^C(t))$ asserts the status of the *usage* of the system $c_1^C$, the second, $x_3^C(t) = f_3^C(x_2^C(t))$, asserts the *location* of $C_1^C$ according to the values of $x_2^C(t)$. The two others observations 8 and 10 of Carol (i.e. *its anti-theft device is installed* and i.e. *it has had only a single owner* respectively) are modeled with the assertion functions $x_1^C(t) = f_1^C(x^C_{\text{anti-theft devise}}(t))$ and $x_5^C(t) = f_5^C(x_4^C(t))$. The particularity of the observation 10 is that to assert that $c_1^C$ has had only one owner, the function $x_5^C(t) = f_5^C(x_4^C(t))$ needs the computing of the owner number. This is then the role of the *counting* function $x_4^C(t) = f_4^C(x^C_{\text{owners}}(t))$.

The conceptual space of Alice, Bob and Carol are those built by each of these speakers to produce their observations. The building of a Blended space is now required to to understands together the 10 observations.

## B.5 BLENDED CONCEPTUAL SPACE

*Blending* is the usual name of the *conceptual integration* operation aiming to *project* at least two different conceptual spaces into a third one, the blended conceptual space: *conceptual integration-like framing or categorization-is a basic cognitive operation that operates uniformly at different levels of abstraction and under superficially divergent contextual circumstances* [FT98]. To cite again Fauconnier and Turner, *Projection is the backbone of analogy, categorization, and grammar* and they consider that it is *an established and fundamental finding of cognitive science that structure mapping and metaphorical projection play a central role in the construction of reasoning and meaning.* To build a blended conceptual space from the conceptual spaces of Alice, Bob and Carol, we must make the following hypothesis: Alice, Bob and Carol understand together. In other words, they speaks about the same system. It is to be noticed that nothing in the conversation confirms this hypothesis. But we need it to build a blended and a generic conceptual space.

With this hypothesis, Tom4A's formalization make very simple the conceptual integration operation because the 10 are independents. Figure B.8 shows the structure of the blended

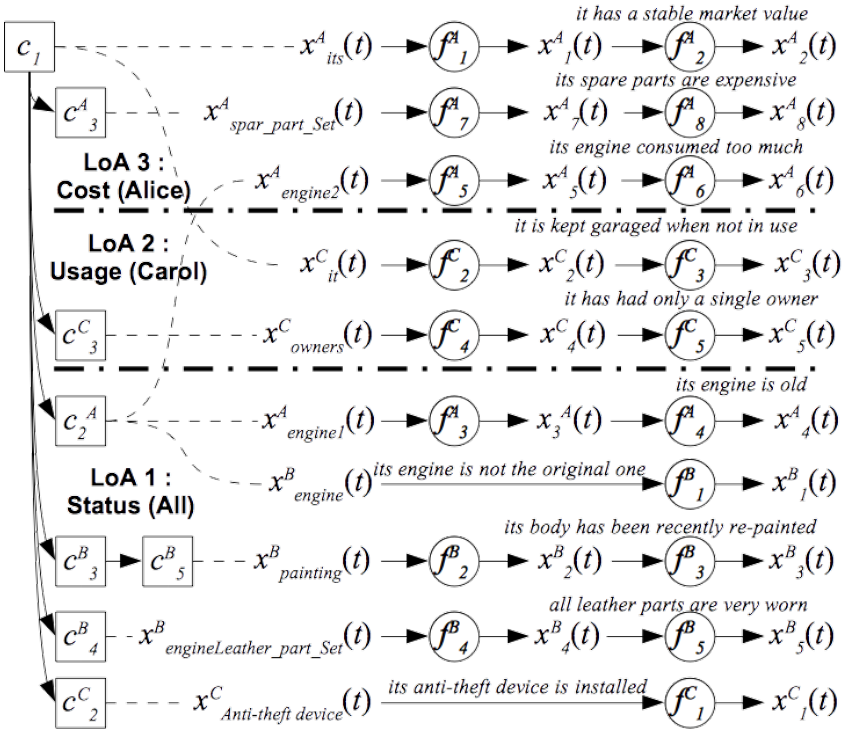


Figure B.8: Blended Conceptual Space

conceptual space. The exponents have been kept to clarify the links between the individual conceptual space and the resulting blended space after the projections of Alice's space firstly, next Bob's one and Carol's space lastly:

- The system is now represented with a unique component : $C_1 \equiv C_1^A \equiv C_1^B \equiv C_1^C$.
- The components of $C_1$ is the fusion of the Alice, Bob and Carol component sets : $C = \{c_1, c_2^A, c_2^C, c_3^A, c_3^B, c_3^C, c_4^B, c_5^B\}$.
- The time function's set is the fusion of the Alice, Bob and Carol time function's sets : $X(t) = \{x_{its}^A(t), x_1^A(t), ..., x^C_{\text{anti-theft devise}}(t), x_1^C\}$.
- The dynamic function's set is also the fusion of the individual dynamic function's sets : $F = \{f_1^A, f_2^A, ..., f_4^B, f_5^B, f_1^C\}$.

In figure B.8, the 10 observations of the conversation have been organized in three abstraction levels. The lowest contains the more concrete observations of Alice, Bob and Carol: those concerning the *status* of the system $c_1$. The abstraction's level of the middle concerns its *usage* and contains Carol's observations number 9 and 10 only. The highest abstraction level concerns the *cost usage* of the system $c_1$. Only Alice made observations at this level of abstraction.

## B.6 GENERIC CONCEPTUAL SPACE

One of the interesting features of Fauconnier and Turner's theory is the notion of *generic conceptual space*. This usual notion in the domain of Knowledge Engineering is of the main importance to model a knowledge corpus (cf. the CommonKads methodology [SAA+00] or [PLG14] for

a detailed illustration). Tom4D's Knowledge Engineering methodology allows to build generic conceptual spaces according to a notion of *conceptual equivalence* [ZGF06b] between different *knowledge roles*. *A knowledge role is an abstract label that indicates the role that the domain knowledge to which the label is attached plays in an inference process* [Bre94]. So, the basic idea of the *conceptual equivalence* is that when two different concepts play the same role in a reasoning process, they can be considered as *conceptually equivalent*.

Let us consider together Alice's observation number 3 (*it has a stable market value*) and Bob's observation number 7 (*all leather parts are very worn*). Figures B.5 and B.6 shows that these two observations uses two *assessment* functions, $f_2^A$ and $f_5^B$, and two *measurement* functions, $f_1^A$ and $f_4^B$. It is obvious that, in Alice's and Bob's reasoning, the functions $f_2^A$ and $f_5^B$ plays the same *role*: to assess something about the system. Similarly, the role of $f_1^A$ and $f_4^B$ is to measure the level of some time function. It is then clear that the time functions $x_2^A(t)$ and $x_3^B(t)$, although basically different, play the same role in Alice's and Bob's reasoning. The same analysis holds for the others time functions. As a consequence, these two observations can be represented with the same pattern made of *type* of function linking type of *variable* (cf. figure B.9). A set of such patterns is called the *functional network*. It is build from the projection from a *concrete conceptual space*, typically a blended space, to the space of the function's types. To build the figure B.9, let us define the type of functions used by our three speakers.

The type of an *assessment* function is called *Discretization*: this is the function's type of the dynamic functions $f_2^A$, $f_4^A$, $f_6^A$, $f_8^A$, $f_3^B$, $f_5^B$ and $f_5^C$. The *Discretization* function's type corresponds to the *Quantization* operation in the Discrete Event Systems community. It is represented with a function of the form $x_2^D = f_D(\Psi, x_1^D)$ where $\Psi$ is a set $\Psi = \{\psi_i\}$ *thresholds* values $\psi_i$. The definition domain of $f_D$ is $\Delta_{x_1^D} \times \Delta_{x_2^D}$ where $\Delta_{x_1^D}$ is a *cardinal* set and $\Delta_{x_2^D}$ is an *ordinal* set or a set without any topology. The constants $\delta_i^{x_2^D}$ of $\Delta_{x_2^D}$ denotes *ranges* of values (i.e. intervals) in $\Delta_{x_1^D}$ so that the number of elements in $\Delta_{x_2^D}$ is equals to the numbers of thresholds values $\psi_i$ in $\Psi$ plus one. As a consequence, any function mapping a cardinal set to an ordinal set or an a-topology set can be represented with a $f_D$ function type. In the running example, the time functions $x_2^A(t)$, $x_8^A(t)$, $x_6^A(t)$, $x_5^C(t)$, $x_4^A(t)$, $x_3^B(t)$ and $x_5^B(t)$ are linked with the variable's type $x_2^D$.

The type of a *assertion* function is a *Classification* function: this concerns the dynamic functions $f_1^B$, $f_1^C$, $f_2^C$ and $f_3^C$. A classification function implements a reasoning that uses a set $R_f$ of classification rules of the form ($N$ denotes the set of natural numbers):

$$\forall x_1^C \in \Delta_{x_1^C}, \exists n \in N, x_1^C = \delta_i^{x_1^C} \implies x_2^C = n \tag{B.3}$$

In this equation, $n$ denotes a particular class so that $x_2^C = n$ means that the class corresponding to the value of $x_1^C$ is the $n^{th}$ class. A classification function is then represented with a function of the form $x_2^C = f_C(R_f, x_1^C)$, its definition domain being $\Delta_{x_1^C} \times N$ where $\Delta_{x_1^C}$ is any type of set. Any function mapping a set to $N$ can be represented with a $f_C$ function type. The time functions $x_2^C(t)$, $x_3^C(t)$, $x_1^B(t)$ and $x_1^C(t)$ are then linked with the variable's type $x_2^C$.

The type of a *measurement* function is a *Modeling* function. It concerns the dynamic functions $f_1^A$, $f_5^A$, $f_7^A$ and $f_4^B$. It is represented with a function of the form $x_2^M = f_M(M_f, x_1^M)$ where $M_f$ is a model providing the value of $x_2^M$ given those of $x_1^M$. The definition domain of $f_M$ is $\Delta_{x_1^M} \times \Delta_{x_2^M}$ where $\Delta_{x_1^M}$ and $\Delta_{x_2^M}$ are *cardinal* sets. Any function mapping two ordinal sets can be represented

with a $f_M$ function type. The time functions $x_1^A(t)$, $x_7^A(t)$, $x_5^A(t)$ and $x_4^B(t)$ are then linked with the variable's type $x_2^M$.

The type of a *counting* function is a *Numbering* function ($f_3^A$ and $f_4^C$). A numbering function is a function of the form $x_2^N = f_N(P_f, x_1^N)$ where $P_f$ is a counting process (i.e. a Poisson process or a discrete Markov counting model for examples). The definition domain of $f_N$ is $\Delta_{x_1^N} \times \Delta_{x_2^N}$ where $\Delta_{x_1^N}$ and $\Delta_{x_2^N}$ are two *ordinal* sets: any function mapping two ordinal sets can be represented with a $f_N$ function type. The time functions $x_4^C(t)$ and $x_3^A(t)$ are then linked with the variable's type $x_2^N$.

The last type of function is the *dating* functions and concerns only the $f_2^B$ dynamic function. This type is called *Time-Stamping* function. It is represented with a function of the form $x_2^T = f_T(T_f, x_1^T)$ where $T_f$ is a time stamping process providing the timestamps $t_k$ of the current value of $x_1^T$). The definition domain of $f_T$ is $\Delta_{x_1^T} \times \Delta_{x_2^T}$ where $\Delta_{x_1^T}$ can be any kind of set, $\Delta_{x_2^T}$ being an *ordinal* set. So, any function mapping a set to an ordinal set can be represented with a $f_T$ function type. Only the time function $x_2^B(t)$ is linked with the variable's type $x_2^N$.

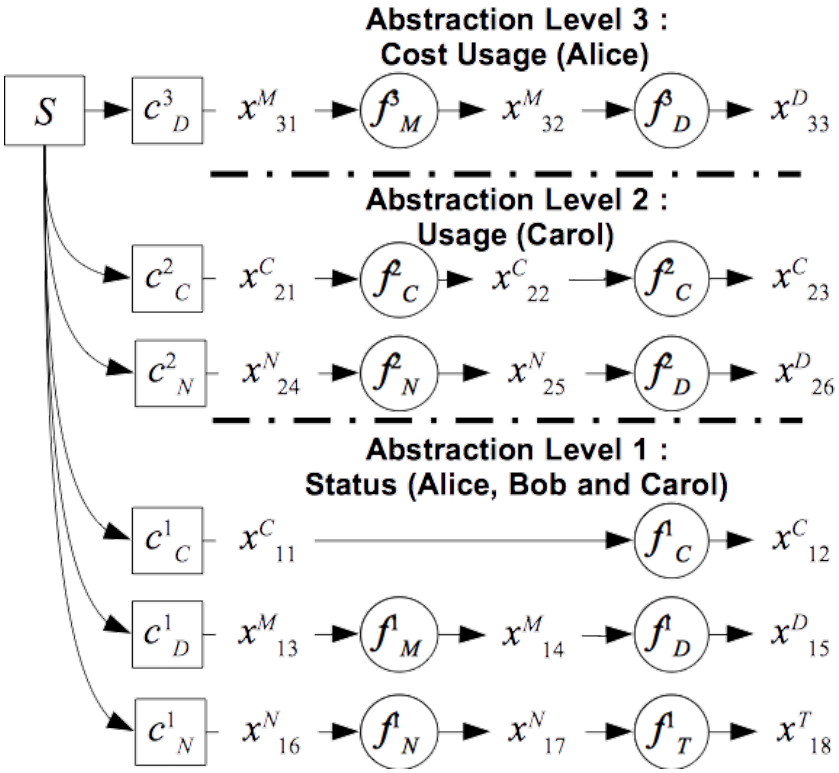


Figure B.9: Generic Conceptual Space

Mapping the dynamic and the time functions of the blended space of figure B.8 with the corresponding types leads to the functional network of figure B.9. The components have also been associated with *abstract components* so that $C_1$ is linked with the $S$ component, $C_3^A$ is linked with $C_D^3$, $C_3^C$ is linked with $C_N^2$, $C_2^A$ is linked with $C_C^3$, $C_C^1$ and $C_D^1$, the pair $(C_3^B, C_5^B)$ is linked with $C_1^N$, $C_4^B$ with $C_D^1$ and finally, $C_2^C$ with $C_C^1$. An abstract component specifies the properties or the constraints that a concrete component must satisfy to be considered as an instance of this abstract component. The reference to any concrete element in the blend being contained in the projection from the blended to the generic conceptual space, the functional network is more compact than the blend.

An important remark is that, when forgetting the projection, there is no way to come back from the generic to the blended space. In other words, it is impossible to build Alice, Bob and Carol observations with the only functional network of figure B.9: the blended space of figure B.8 is a specific instantiation of the functional network of figure B.9. This remark is the basis of the formalization of Floridi's theory of gradients of levels of abstraction.

## B.7 LEVELS OF ABSTRACTION

Fauconnier and Turner's theory aims at studying the *creation* of specific *structures* that *emerge* out of the blending operation. For this theory, the notion of abstraction is then an inherent property of the objects them self (cf. the three levels of abstraction in figures B.8 and B.9).

Nevertheless, in [New81], Newell builds an *ontological* notion of abstraction levels, which is characterized by the following properties: each abstraction level describes a *system* that transforms a *medium* through its *components* which provide primitive treatments, and *laws of compositions* of these components which provide a *structure* to the system. As a consequence, the working of the system is described with *laws of behavior* which establish how the system behavior depends on the behavior of the components and their composition. Newell apply this notion to describe an information system with four levels of abstraction: the *physic* level (electromagnetic waves), the *circuits* level (transistors), the *logic* level (boolean algebra) and the *symbol* level (program), from the most concrete (continue space) to the most abstract (purely discrete space). Of this report, Newell proposes the existence of the *Knowledge Level* that it places above these ones. The Knowledge Level is characterized by the following fact: there is no *laws of composition* because behavior is governed by a *Principle of Rationality*. At the knowledge level, a system is an *agent* whose components are *goals*, *actions* and a *body* (i.e. a knowledge corpus); and which processes its input informations to determine the (output) actions to take in order to reach its goals.

On an another hand, Floridi's uses an *epistemological* point of view to develop its *Method of Levels of Abstraction* in [Flo08, Flo10]. A *level of abstraction (LoA) is a finite but non-empty set of observables* [Flo08, p. 10], and the word *system* refers to the object of study, a process in science or engineering or a domain of discourse. The *behaviour of a system, at a given LoA, is defined to consist of a predicate whose free variables are observables at that LoA. The substitutions of values for observables that make the predicate true are called the system behaviours.* A *Level of Abstraction* is then a particular organization of *variable*, *observable*, *behavior* and *transition rules* between *values. A moderated LoA is defined to consist of a LoA together with a behaviour at that LoA*, [Flo08, p. 11].

The *Method of Levels of Abstraction* organizes LoA in *Gradient of Abstraction* (GoA). A GoA allows to vary the LoA to make observations at different granularity levels: the higher the level of abstraction, the fewer but richer the information. *The quantity of information in a model varies with the LoA: a lower LoA, of greater resolution of finer granularity, produces a model that contains more information than a model produced at a higher, or more abstract, LoA*, [Flo08, p. 18]. Foridi's theory distinguishes two kinds of GoA: *disjoint GoAs*, where the LoA are independent together, and *nested GoAs* where each LoA incrementally describes the same phenomena.

Tom4A defines three moderated LoA's. The most concrete is called the *Observation Level*: the blended space of figure B.8 constitutes the moderated LoA for the conversation of the running example of section B.2 at the observation level of abstraction. It formally describes the *observations* of the speakers according to the TOT mathematical framework. It is made with a set of *binary relations* linking concrete components, time functions and dynamic functions constituting respectively the *Structural Model*, the *Behavioral Model* and the *Functional Model* of the *observed process* $(X(t), \Theta(X, \Delta))$ (cf. section B.3 and [PLG14] for a detailed example).

The intermediate level of abstraction is called the *Computing Level*: the functional network of figure B.9 is the moderated LoA for the conversation at the computing level of abstraction. It formally describes the types of computing that are required to create *timed observations* from an observed process. In other words, the Computing Level contains the necessary and sufficient corpus of knowledge to *specifies* the programs that could generate the timed observations of the observation level. Again, it is made of *binary relations* linking *types* of components, variables and functions, describing the *logical* approach to build timed observations at the Observation Level.

The last level of abstraction, the highest according to Tom4A, is the *Reasoning Level*: it is made with an inference structure describing the reasoning of an observer. It formally describes the reasoning allowing the use of the type of functions of the computing level to achieve a particular goal. It is made of *binary relations* linking *knowledge roles* and type of *inferences*, describing the reasoning *steps* that are required to achieve a goal defined with a particular problem to solve.

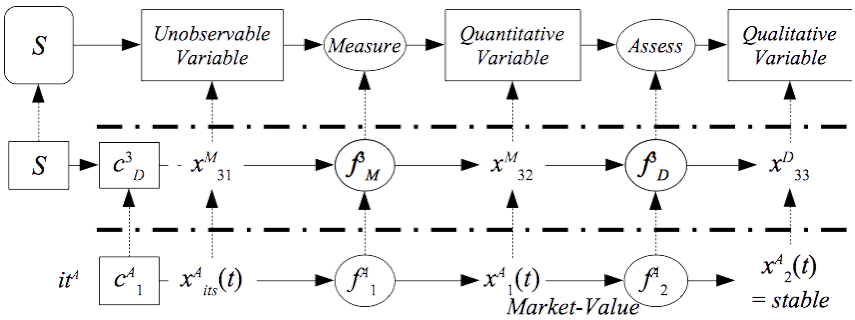


Figure B.10: Alice's Observation 3 at three LoA

To build a model at this level of abstraction, let us consider again Alice's observation *it has a stable market value*. The first point to notice is that Alice uses the term *market value* to build its observation. This term has been represented with the time function $x^A_2(t)$ (cf. figure B.10), which is a *Discretization function* represented with the variable's type $x^D_{33}$. At the Knowledge Level, the role of a discretization function is to transform a *Quantitative Variable* in a *Qualitative Variable*. Such a transformation aims at defining the level of a quantitative variable regard to thresholds (cf. section B.6). In the same spirit, the role of a modeling function $f^3_M$ is to provide a quantitative evaluation of the *Unobservable Variable* $x^M_{31}$ which characterizes, at the Knowledge Level, the *phenomena* of the evolution of the time function $x^A_{its}(t)$. The role of *System* at the Knowledge Level is then those of a *transfer function*, graphically represented a rectangle with round corners, that provides values for each of its variables. Finally, according to Tom4A, the complete meaning of Alice's observation number 3 is given in figure B.10.

Doing the same for all the observations of the conversation leads to the figure B.11. In this figure, the blended, the generic and the inference conceptual spaces have been organized

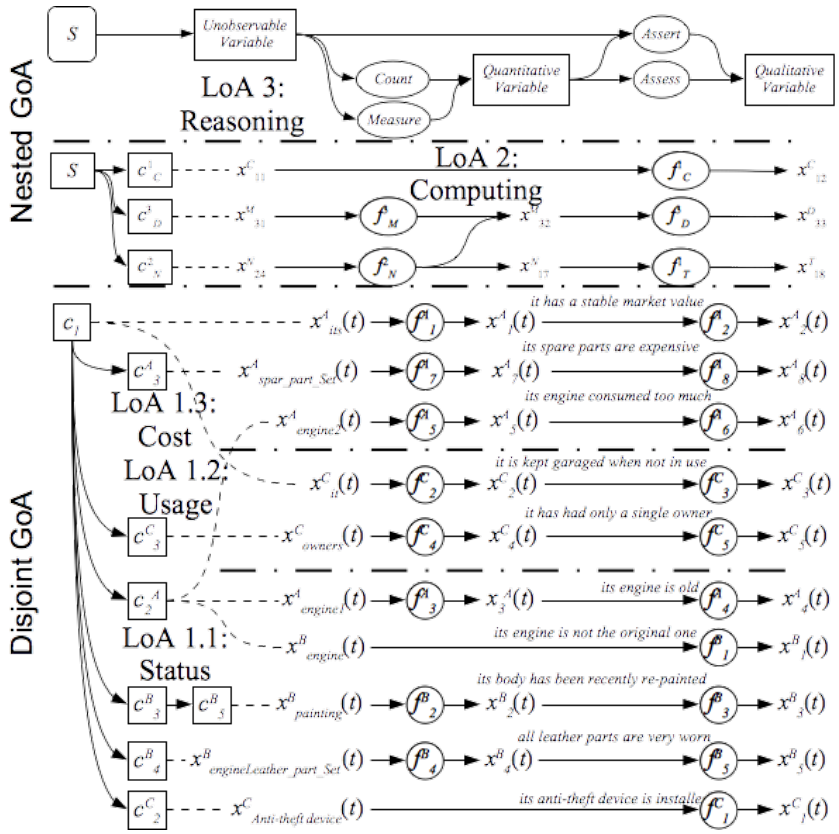


Figure B.11: Gradients of Abstraction of the Conversation

in two *gradient of abstraction* (GoA): a *disjoint GoA* which constitutes the lowest abstraction level, the *Observation Level*, and contains the Blended Conceptual Space of figure B.8, and a *nested GoA* made of the *Observation Level*, the *Computing Level* and the *Reasoning Level*. The Generic Conceptual Level of figure B.9 is represented with the intermediate abstraction level, the *Computing Level*. The effect of the conceptual equivalence appears clearly: even if they aim at representing the same thing, the functional network of figure B.11 is much more compact than the Generic Conceptual Level of figure B.9.

Up to our knowledge, the *Reasoning Level* has no counter part in the Blending Theory. The fundamental interest of this LoA appears in figure B.11: it allows to identify the common aim of the speakers to explicit some properties of a (still unknown) system in order to state its qualities and defects.

## B.8 CONCLUSION

This paper proposes a formal framework, the Tom4A method (Timed Observations Method for Abstraction), that provides for the first time, up to our knowledge, a strong mathematical foundation to both the Blending theory [FT98] and the Method of Abstraction theory of [Flo08]. This mathematical framework is build on the *Timed Observations Theory* (TOT). Tom4A completes the Tom4D Knowledge Engineering methodology (Timed Observations Methodology for Diagnosis, [PLG14]) and the Tom4L Knowledge Discovery in Databases process (Timed Observations Mining for Learning, [LBP15, LA12]), also based on the TOT. The basic concepts of Tom4A are progressively introduced with a running example, a conversation between three speakers,

whose original text comes from the web site of the *Society for the Philosophy of Information* (http://www.socphilinfo.org/node/150). This example provides for the first time, still up to our knowledge, the first conceptual models of a conversation under the formal form of two gradients of abstraction, defining the meaning of this conversation.

Our long term goal is to develop software tools able to discover and to model knowledge representations from sets of timed data so that the human interpretation is intuitive, immediate and independent of the learning and the modeling tools. The next step of this work is then to propose a new formalization of the analogical reasoning based on the combination of the TOT and the Category Theory [ML71].

# Résumé

Ce document concerne le développement d'un cadre mathématique spécifiant une technologie capable de prendre en charge quelques unes des problématiques relevant du domaine des *Grands Flux de Données* (*Big Data Flows*). Nous proposons de combiner le point de vue ontologique de Newell et celui épistémologique de Floridi d'*abstraction* pour construire des outils de *transformation de modèles* au moyen d'un ensemble adéquats de *foncteurs* au sens de la théorie des Catégories de Samuel Eilenberg et Saunders Mac Lane. La méthode de résolution de problème proposée est basée sur un *raisonnement d'abstraction temps réel* qui produit, en ligne, une réduction d'un grand nombre de données sémantiquement pauvres en une donnée unique équivalente mais sémantiquement plus riche. Le prix à payer pour un tel *enrichissement sémantique* de l'information est la perte d'information *syntaxique* (i.e. le phénoméne d'oubli).

Nos contributions sont les suivantes: (i) la démonstration que le concept d'*Observateur Unaire* de la Théorie des Observations Datées (TOT) de Le Goc joue le même rôle qu'un échantillonneur de Dirac, (ii) la construction de la catégorie $TOT(\mathbb{Z})$, adéquate à la formulation du processus d'abstraction proposé et (iii) la conception de la méthode de résolution de probléme TOM4A (Timed Observations Methodology for Abstraction) dont une application concrète est présentée visant à découvrir et modéliser le problème complexe de la fraude interne dans le domaine bancaire.

En synthèse, la catégorie $TOT^i(\mathbb{Z})$ à un niveau d'abstraction arbitraire $\mathcal{L}^i$, $i \neq 0$, engendre une structure algébrique particulière ainsi qu'un espace observable particulier aux données observées et permet la modélisation de ces données via un modèle de chronique abstrait $\mathcal{M}^i$ spécifique à ce niveau d'abstraction. Ce document démontre les équivalences suivantes: Représentation $\leftrightarrow$ Abstraction $\leftrightarrow$ Somme dans la catégorie $TOT^i(\mathbb{Z})$ et Interprétation $\leftrightarrow$ Réification $\leftrightarrow$ Produit dans la catégorie $TOT^i(\mathbb{Z})$.



Mots clés : Abstraction, Réification, Morphisme, Théorie des Catégories, Processus Dynamiques, Ingénierie des Connaissances.



# Abstract

This document concerns the development of a theoretical mathematical framework to provide a technology able to manage some of the problematics of the *Big Data Flows* domain. We propose to combine Newell's ontological and Floridi's epistemological point of views of *abstraction* to build tools that *transform models* by the mean of an adequate set of *functors* according to Samuel Eilenberg and Saunders Mac Lane's Category Theory. The proposed Problem Solving Method relies on a *real time abstraction reasoning* process to resume, on line, a lot of *semantically poor* data into an equivalent but *richer* one. The price to pay for such an information *semantic enrichment* is the loss of *syntactic* data (i.e. the *oversight phenomenon*).

Our contributions are (i) to prove that Le Goc's Timed Observations Theory (TOT) concept of *Unary Observer* plays the same role as Dirac's sampler, (ii) the construction of the $TOT(\mathbb{Z})$ Category that is adequate to formulate the proposed abstraction based PSM and (iii) the design of TOM4A (Timed Observations Methodology for Abstraction), a *specific* recursive abstraction-reification based PSM whose a concrete application has been provided for detecting and modeling the complex problem of internal frauds in the banking industry.

Finally, a $TOT^i(\mathbb{Z})$ Category at an arbitrary Level of Abstraction (LoA) $\mathcal{L}^i$, $i \neq 0$, induces particular algebraic structure $\mathcal{S}^i$ and observable space $\mathcal{T}^i$ to the observed data and model them with a specific abstract chronicle model $\mathcal{M}^i$ at this LoA. This document demonstrates then the following equivalences: Representation $\leftrightarrow$ Abstraction process $\leftrightarrow$ Sum of objects in the $TOT^i(\mathbb{Z})$ Category and Interpretation $\leftrightarrow$ Reification process $\leftrightarrow$ Product of objects in the $TOT^i(\mathbb{Z})$ Category.



Keywords: Abstraction, Reification, Morphism, Category Theory, Dynamic Process, Knowledge Engineering.



LIS, Laboratoire d'Informatique et Systèmes, UMR CNRS 7020
Falculté de Saint Jérôme, Avenue Escadrille Normandie Niemen
13197 Marseille Cedex 20