



Université
de Toulouse

THÈSE

En vue de l'obtention du DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par :

Université Toulouse III Paul Sabatier (UT3 Paul Sabatier)

Discipline ou spécialité :

Robotique et Systèmes Embarqués

Présentée et soutenue par :

Nuno BELARD

le : lundi 21 Mai 2012

Titre :

Raisonnement sur les Modèles:
Détection et Isolation d'Anomalies dans les Systèmes de Diagnostic

Ecole doctorale :

Systèmes (EDSYS)

Unité de recherche :

Laboratoire d'Analyse et d'Architecture des Systèmes

Directeur(s) de Thèse :

M. Michel COMBACAU et M. Yannick PENCOLE

Rapporteurs :

Mme Marie-Odile CORDIER et M. Pierre MARQUIS

Membre(s) du jury :

M. Michel COMBACAU et M. Yannick PENCOLE (Co-directeurs)

Mme Marie-Odile CORDIER et M. Pierre MARQUIS (Rapporteurs)

Mme Sylvie DELPRAT (Examinatrice)

M. Didier MAQUIN (Examinateur)

M. João PAVAO MARTINS (Examinateur)

Reasoning about Models:
Detecting and Isolating Abnormalities in Diagnostic Systems

Nuno Belard

Acknowledgements

This thesis marks the end of my Ph.D. journey; a journey that was long but memorable, challenging but rewarding, difficult but exciting. It is at the end, when all is said and done, that I write these lines to those that were by my side along the richest period of my life.

My first words go to Michel Combacau and Yannick Pencolé, my academic supervisors who guided me through my research. By encouraging me and defying my crazy ideas they made me think and evolve. If, research-wise, I am more mature today, it is clearly and without any doubt due to them.

Christophe Bordry, Denys Bernard, Nathalie Bergé and Vincent Cheriére will also be staying in my mind. I will never forget that white board where Christophe, Denys and Vincent shared their ideas and let me enter their world. Along with Nathalie who kept pushing me to become stronger and stronger, they truly were my inspiration. Myriam Fedoui and Sylvie Delprat are also not to be forgotten. In an industrial context, Myriam trusted me to follow my ideas and was more honest to me than what I could have ever expected. Sylvie had the patience to go through all my work with a critical yet supportive point of view. In the hostile environment of my thoughts this is an achievement by itself.

I will also want to thank M. Didier Maquin, M. João Pavão Martins, Mme. Marie-Odile Cordier, M. Pierre Marquis and Louise Travé-Massuyès. The first four, for accepting to become a part of my jury and for sharing their insights about my work. As for Louise, by accepting me with open arms she provided me with a working environment that many would envy.

A Ph.D. is also about those whose friendship is there for you, not only in the happy and exciting moments, but also in the tough ones. While writing a full list of names is certainly an impossible task, I cannot complete the last page of my thesis without thanking Almeida, Camille, Cassiano, Cristina, Erica, Inês, Iñigo, Jose, Lucia, Mafalda, Margaux, Pauline, Ricardo, Rita, Té and Tete. Each one of them knows exactly what they mean to me. Kudos.

Finally, I will probably not have another opportunity to publicly acknowledge my family. I am honestly grateful for everything we have shared. The man I am today is a product of every instant with them. From the bottom of my heart, thank you.

Abstract

In Model-Based Diagnosis, a set of inference rules is typically used to compute diagnoses using a scientific and mathematical theory about a system under study and some observations. Contrary to the classical hypothesis, it is often the case that these Models are abnormal with respect to a series of required properties, hence affecting the quality of the computed diagnoses with possibly huge economical consequences, in particular at Airbus.

A thesis on reality and cognition is firstly used to redefine the classic framework of model-based diagnosis from a formal model-theoretic perspective. This, in turn, enables the formalisation of abnormalities and of their relation with the properties diagnoses.

With such material and the idea that an implemented diagnostic system can be seen a real-world artefact to be diagnosed, a theory of meta-diagnosis is developed, enabling the detection and isolation of abnormalities in Models of diagnostic systems and explanation in general. Such theory is then encoded in a tool, called MEDITO, and successfully tested against Airbus real-world industrial problems.

Moreover, as different heterogeneous implemented Airbus diagnostic systems, suffering from distinct abnormalities, may compute different diagnoses, methods and tools are developed for: 1) checking the consistency between subsystem-level diagnoses and 2) validating and comparing the performance of these diagnostic systems. Such work relies on an original bridge between the Airbus framework of diagnosis and its academic counterpart.

Finally, meta-diagnosis is generalised to handle meta-systems other than implemented diagnostic systems.

Résumé

Dans le cadre du diagnostic à base de Modèle, un ensemble de règles d'inférence est typiquement exploité pour calculer des diagnostics, ceci en utilisant une théorie scientifique et mathématique sur le système à diagnostiquer, ainsi qu'un ensemble d'observations. Contrairement aux hypothèses classiques, les Modèles sont souvent anormaux vis-à-vis d'un ensemble de propriétés requises. Naturellement, cela affecte la qualité des diagnostics [à Airbus].

Une théorie sur la réalité, l'information et la cognition est créée pour redéfinir, dans une perspective basée sur la théorie des modèles, le cadre classique de diagnostic à base de Modèle. Ceci rend possible la formalisation des anomalies et de leur relation avec des propriétés des diagnostics.

Avec ce travail et avec l'idée qu'un système de diagnostic implémenté peut être vu comme un objet à diagnostiquer, une théorie de méta-diagnostic est développée, permettant la détection et isolation d'anomalies dans les Modèles des systèmes de diagnostic. Cette théorie est mise en pratique à travers d'un outil, MEDITO; et est testée avec succès à travers un ensemble de problèmes industriels, à Airbus.

Comme des différents systèmes de diagnostic Airbus, souffrant d'anomalies variées, peuvent calculer des diagnostics différents, un ensemble de méthodes et outils est développé pour: 1) déterminer la cohérence entre diagnostics et 2) valider et comparer la performance de ces systèmes de diagnostic. Ce travail dépend d'un pont original entre le cadre de diagnostic Airbus et son équivalent académique.

Finalement, la théorie de méta-diagnostic est généralisée pour prendre en compte des méta-systèmes autres que des systèmes de diagnostic implémentés.

Contents

1	Introduction	1
1.1	When Models hurt explanation: academic objectives	2
1.1.1	Scientific and mathematical theories have and will always have holes . .	2
1.1.2	Observations have and will always have holes	4
1.1.3	Rules of inference may have holes	5
1.1.4	The academic objectives	5
1.2	When Models hurt Airbus diagnoses: industrial objectives	6
1.2.1	Diagnosis at Airbus	6
1.2.2	What can be improved in Airbus diagnostic scene	8
1.2.3	Industrial objectives	9
1.3	Overview	10
2	Diagnostic systems	13
2.1	A historical view of explanation and Models	14
2.1.1	Explanation in philosophy	14
2.1.2	Explanation in Artificial Intelligence	16
2.2	Building a deep foundation	17
2.2.1	On reality, information and cognition	18
2.2.2	On theories and observations	20
2.3	Towards a framework of diagnosis	22
2.3.1	On systems, components, abnormality and other concepts	22
2.3.2	Characterising diagnoses	25

2.4	Defining and relating properties in diagnostic systems and diagnoses	31
2.4.1	Defining properties	32
2.4.2	Relating properties	36
2.5	Originality and Related work	39
3	Meta-diagnosis	41
3.1	Towards a framework of meta-diagnosis	42
3.1.1	On meta-systems, meta-components and other concepts	42
3.1.2	Characterising meta-diagnoses	44
3.1.3	Some words on meta-diagnoses	46
3.2	Modelling and solving meta-diagnostic problems	49
3.2.1	The problem of ontologically false believed system description	49
3.2.2	The problem of diagnosable believed systems and sound and complete diagnostic algorithms	50
3.3	MEDITO: a logic-based meta-diagnosis tool	51
3.3.1	MEDITO's architecture	52
3.3.2	A detailed view on MEDITO's modules	53
3.3.3	Meta-diagnosing with MEDITO	57
3.4	Originality and Related work	57
4	The Airbus world	61
4.1	Towards a common framework of diagnosis at Airbus	63
4.1.1	Defining a common vocabulary of diagnosis	63
4.1.2	Defining a common subsystem-level diagnostic framework	67
4.1.3	Defining a common aircraft-level diagnostic framework	71
4.2	Checking for local consistency with CONCKER	74
4.2.1	Improving local-diagnoses' consistency at Airbus	74
4.2.2	CONCKER's architecture	75
4.2.3	A detailed view on CONCKER's modules	76
4.2.4	Checking for local consistency with CONCKER	81

4.3	Assessing performance with DPAT	82
4.3.1	Improving Airbus diagnostic performance assessment	82
4.3.2	DPAT's architecture	84
4.3.3	"Utilities" block	85
4.3.4	"Specific-knowledge Generation" block	88
4.3.5	"Test-case Generation" block	90
4.3.6	"Performance Assessment" block	91
4.3.7	Assessing performance with DPAT	97
4.4	Meta-diagnosing Airbus Models with MEDITO	98
4.4.1	An Airbus landing gear extraction and retraction system	99
4.4.2	An Airbus LGERS believed system	100
4.4.3	Meta-diagnosing an Airbus LGERS believed system with MEDITO . .	101
4.5	Originality and related work	102
5	Generalising meta-systems	105
5.1	Meta-diagnosing non-logic-based diagnostic systems	106
5.1.1	When Models hurt ARR-based MBD: a DC motor trial	106
5.1.2	From an FDI approach to a DX one: transforming the problem	110
5.1.3	Meta-diagnosis	111
5.1.4	Back to an FDI approach	112
5.2	Meta-diagnosing other meta-systems	113
5.2.1	Characterising abnormality predictions	114
5.2.2	Defining and relating properties in abnormality prediction systems and abnormality prediction results	116
5.2.3	When Models hurt prediction: a DC motor trial	118
5.2.4	Meta-diagnosis	122
5.3	Originality and Related work	124

6	A final perspective	125
6.1	Summary of the contribution	125
6.2	A glimpse into the future	126
6.2.1	The notion of knowledge	126
6.2.2	The notion and logic of explanation	127
6.2.3	The logics handled by MEDITO	127
6.2.4	The formalism of Meta-diagnosis	127
6.2.5	The selection of test-cases	128
6.2.6	The repair of Meta-systems	128
A	Model-theory	129
	Bibliography	133

Chapter 1

Introduction

Logic: The art of thinking and reasoning in strict accordance with the limitations and incapacities of the human misunderstanding.

- Ambrose Bierce

From the numerous fields of Artificial Intelligence, this work investigates one: *Models* and their usage in *scientific explanation* on the form of implemented [Airbus] *diagnostic systems*.

Ubiquitous in our everyday life, explanation has a tremendous practical value in the *real-world*. We want to understand the meaning of a word or how to drive a car. We want to know why our bedroom window is broken, why there are glass pieces all over the room and why there is a strange ball on the bed. We might even want to know why our doctor does not discover why we have headaches after every day of work. All these questions, however, correspond to two very different accounts of explanation. As Salmon puts it [92], there are “[explanations] of *what something means* or *how to do something*”, like understanding how to drive a car; and “[explanations] of *why something occurs*”, as in the strange case of the broken window. Straightforwardly, in this work I am interested in the latter account of explanation, the one that comes to our rescue with *causes* everytime there are *effects* to be explained in our lives. Hereafter, the word “explanation” will be used solely on the context of these why-questions.

As we will see further in this dissertation, several different accounts of explanation exist, both in Artificial Intelligence and in philosophy, none of which being generally accepted. Nevertheless, every single thesis on what explanation is tells us that, in order to bring up the answers we need, explanation relies on *Models*. First, we need to build a *cognitive* Model of both the question and the context: the *observations*. As so, we have to *perceive* the *explananda* - a broken window, the glass pieces, the ball - and *perceive* the context - the oxygen, the gravity and so on. Then, explanation relies on another cognitive Model, the *scientific and mathematical theories*, to reason about such observations. Finally, this reasoning process is

based on another Model of how inferences are drawn, the *rules of inference*. In a nutshell, observations, scientific and mathematical theories and rules of inference are the Models on which explanation relies; and we say that some *explanans* explain some explananda-observations if the latter can be inferred from the former and the context-observations, both at some earlier time, along with some scientific and mathematical theories.

Presenting a more than sufficient reason for one to embrace the study of Models, explanation is certainly far from being the only motivation. In parallel with it, Models play a central role both in *prediction*, *planning* and *learning*; the main Artificial Intelligence fields that require reasoning about time [97]. If explanation is seen as a reasoning process where, by and large, some observations over a period of time and some scientific and mathematical theories are used to determine a description of the real-world at some earlier time, then we can see: prediction as a task where some observations over a period of time and some scientific and mathematical theories are used to determine a future description of the real-world; planning as a task where some observations over a period of time and some scientific and mathematical theories are used to determine a sequence of *actions* that enable one to get to a goal real-world description; and learning as a task where some observations over a period of time are used to determine new scientific and mathematical theories.

In a straightforward manner, although in this book explanation will be used as a framework for reasoning about Models, I think that several of the obtained results can be applied to other Artificial Intelligence fields. Concerning the remainder of this chapter, I will start by diving much deeper into the explanation world, explore its Model-related problems in connection with implemented diagnostic systems and map those problems into Airbus industrial context. This work's objectives will follow directly from the problems stated. Finally, an overview of this book and the dependency between chapters will be presented.

1.1 When Models hurt explanation: academic objectives

As affirmed above (and justified in Section 2.1), although there are many different notions of explanation, they all rely on Models, i.e. scientific and mathematical theories, observations and rules of inference. What happens then when Models have “problems”?

In this section, I intuitively expose some of the many “problems” Models may have; and argue that, no matter what notion of explanation, Model-based diagnosis and Artificial Intelligence in general need tools both to detect and to isolate these and many other “problems”.

1.1.1 Scientific and mathematical theories have and will always have holes

Imagine a logician, Mr. Tortuga, inhabiting an island where little to no technology exists. One fine day, Tortuga finds a very strange object known to us as a car. After some days studying the artefact, he turns the car key and the car starts. Eventually, by repeatedly turning the car key and observing the consequent effects, he builds up a theory on how the car works: “turning a normal key implies the car start”; or $\neg \text{Ab}(\text{Key}) \wedge \text{Turn}(\text{Key}) \Rightarrow \text{Start}(\text{Car})$.

Happy to have found a working relic, Tortuga uses the car for some time. One cursed morning, however, he turns the key and the car does not start, i.e. $\text{Turn}(\text{Key}) \wedge \neg \text{Start}(\text{Car})$. Our main character is now facing a why-question (or explanation problem, or diagnostic problem): why does the car not start? From what he knows, he naturally explains the observed effect with an abnormality in the keys. Indeed, $\{\neg \text{Ab}(\text{Key}) \wedge \text{Turn}(\text{Key}) \Rightarrow \text{Start}(\text{Car}), \text{Turn}(\text{Key}) \wedge \neg \text{Start}(\text{Car})\} \models \text{Ab}(\text{Key})$. This, however, proves to be wrong. Tortuga then tries to take the doors off, swap the tires, and so on - after all he knows nothing about how cars work. Without any chance, our hero eventually gives up.



Suppose that, for the sake of the argument, Tortuga takes his car to Mr. Zapato, a very well known illegal technology dealer. In a minute, Tortuga is told that the car is missing a substance called gasoline. The theory on how the car works quickly evolves in Tortuga's mind: "turning a normal key and having gasoline implies the car start"; or $\neg \text{Ab}(\text{Gas}) \wedge \neg \text{Ab}(\text{Key}) \wedge \text{Turn}(\text{Key}) \Rightarrow \text{Start}(\text{Car})$. And from that day on, when wanting to take a ride, Tortuga always checked for the presence of gasoline before turning the key.

The situation with the gasoline keeps happening with the battery, the spark plugs and many other car parts. Eventually, Tortuga finds out that:

- There are and there may always be unknown conditions among the premisses of an effect in a scientific and mathematical theory. For example, Tortuga once turned the key and the car did not start. When turning it again, surprisingly, he observed the opposite effect. Until this day he could not understand the condition that was missing in the first try: an automatic reset of the engine control unit temporarily disabled the spark plugs' electricity and, consequently, the ability of the car to start.
- It is too expensive to indiscriminately check for every condition whenever an observation contradicts an expected effect. For instance, Tortuga once realized that it was almost always an empty tank that caused the car not to start. From that day on, he always checked for the presence of gasoline whenever the car failed to start.

In Artificial Intelligence terms, what our hero is facing is a well known problem: the *qualification problem*, i.e. how to express and reason with all the known and unknown preconditions required for an action to produce an effect, introduced by McCarthy and Hayes in [76].

Just as the qualification problem, many other problems affect scientific and mathematical theories. The *frame problem* [77, 76, 95] concerns producing a scientific and mathematical theory expressing information about what is changed and, especially, what remains unchanged by an action or event. It can be illustrated, for instance, by Tortuga's need to know that pushing the accelerator pedal does not stop the car. The *ramification problem* [95] is characterised by our intrinsic inability to produce a scientific and mathematical theory containing all the indirect effects of an action or event. For example, our main character knows that having the headlights on for a long period of time will have the desired effect of illuminating his house in the dark. Conversely, he does not know that it will also have the undesired consequence of inducing a discharge in the battery and a consequent impossibility of starting the car. The *incompleteness problem* - which can be viewed as a hard qualification problem - concerns generating a scientific and mathematical theory that contains all of the relevant effects and their preconditions. It is illustrated by Tortuga's lack of knowledge on how to stop his car the first time he used it. The *ontologically false theories' problem* (cf. Chapter 2) is certainly the most annoying one. It corresponds to having scientific and mathematical theories representing things that are not as so in reality [106]. Tricked by Mr. Zapato, our hero once thought that clapping his hands twice when inside the car caused the car to start.

The qualification, frame, ramification, incompleteness and ontologically false theories' problems are just some of the many known holes in scientific and mathematical theories affecting explanation; but there can exist, in theory, many more unknown ones. Moreover, all these problems are closely related, and some even imply others¹. Finally, since the objective of Artificial Intelligence is to encode these theories in a robot's "mind", there can also be holes in this encoding. All in all, scientific and mathematical theories have and will always have holes.

1.1.2 Observations have and will always have holes

Let us go back to Mr. Tortuga and his car. One fine day, Tortuga starts the car and observes that, contrary to what was usual, the "change oil" light was bright red, i.e. $\text{Red}(\text{OilLight})$. Knowing that a low level of oil produces that effect, he immediately checks the oil level. After many hours inspecting the car and a lot of money spent with Mr. Zapato's services, Tortuga finds out that everything is fine with his relic: the red colour was an optical effect produced by the red windows in his garage, i.e. $\neg \text{Red}(\text{OilLight})$.

This time, Tortuga is facing the *problem of ontologically false observations* (cf. Chapter 2), intuitively corresponding to gathering observations that are not as so in reality [106].

The problem of ontologically false observations is not the only problem affecting one's observations. The *problem of uncertain observations* [23, 84] concerns collecting precise measures of the observables (what can be observed) in a scientific and mathematical theory.

¹I will expose this in Chapter 2 and use such fact to justify treating only some of these holes.

For example, Tortuga needs to determine if the “change oil” light is red or, as usual, white; and he may doubt between such values. The *problem of incomplete observations* [23, 84] concerns capturing all the observations needed to draw precise conclusions using scientific and mathematical theories. Once again, Tortuga needs to observe a car that did not start, a turned key, a normal battery and so on, just to conclude that the spark plugs were abnormal.

As with scientific and mathematical theories, observations have many known holes - shown, for instance, in the problems of ontologically false, uncertain and incomplete observations - and possibly many more unknown ones. If we add to these, once again, the possible errors in the encoding of observations in an Artificial Intelligence agent, there should remain no doubt that observations have and will always have holes.

1.1.3 Rules of inference may have holes

In the same way as observations and scientific and mathematical theories, rules of inference may also be exposed to some holes.

The *soundness problem* [74, 56] is the problem of inferring truths that are not logically entailed by observations together with scientific and mathematical theories. As for its dual, the *completeness problem* [74, 56] is concerned with inferring all the truths entailed by scientific and mathematical theories and observations. For example, if $\{\neg\text{Ab}(\text{Key}) \wedge \text{Turn}(\text{Key}) \Rightarrow \text{Start}(\text{Car}), \text{Turn}(\text{Key}) \wedge \neg\text{Start}(\text{Car})\} \models \text{Ab}(\text{Key})$; then the inference of $\neg\text{Ab}(\text{Key})$ from $\neg\text{Ab}(\text{Key}) \wedge \text{Turn}(\text{Key}) \Rightarrow \text{Start}(\text{Car})$ and $\text{Turn}(\text{Key}) \wedge \neg\text{Start}(\text{Car})$ would be unsound, and the inference of \emptyset from these same sentences would be incomplete.

Even if these rules-of-inference-related issues are not as problematic as the ones mentioned in the previous subsections, for the soundness and completeness of a set of inference rules in a given logic can be mathematically proved, they need to be considered for two reasons. First, as with observations and scientific and mathematical theories, the encoding of rules of inference in a robot’s “mind” is error-prone. Second, there are real-life situations where it is impossible to mathematically prove the soundness and completeness of such rules (eg. when they constitute a black-box for the user).

1.1.4 The academic objectives

In the previous subsections I argued that Models, i.e. scientific and mathematical theories, observations and rules of inference, are condemned to always suffer from several abnormalities (referred to as “holes”) clearly affecting explanation. Model-based diagnosis is a particular form of explanation, where one is interested in nothing but those causes of some effects that correspond to abnormalities in certain system’s components (detailed in Section 2.3). This being so, the following is the main academic objective of this dissertation:

Academic objectives:

- To reason about Models for detecting and isolating abnormalities in implemented diagnostic systems (and explanation in general).

1.2 When Models hurt Airbus diagnoses: industrial objectives

This chapter begun with a general overview of explanation, followed by my argument that scientific and mathematical theories, observations and rules of inference, ubiquitous in every account of explanation, will always suffer from abnormalities. This motivated the main academic objective of this book. It is now time to move from the academic field to the industrial one, understand how Model-based diagnosis at Airbus suffers from Model abnormalities, and be confronted to the industrial objectives behind this dissertation.

1.2.1 Diagnosis at Airbus

Just like nuclear power plants or satellites, aircraft are complex systems with tens of interacting subsystems and thousands of underlying parts; some being complex systems on their own.

Naturally, as the aircraft life progresses, these parts are subjected to different types of stress; stress which, eventually, originate *faults*, i.e. abnormal conditions, or non-permitted deviations regarding some fundamental properties of the components. Faults leading to *failures*, i.e. the permanent interruption of the aircraft's (or one of its subsystems) ability to perform some required functions under specified operating conditions, maintenance appears as a fundamental activity to preserve aircraft airworthiness [65] (These concepts will be dissected in Chapter 4).

From a maintenance standpoint, the first approach to preserve this aircraft airworthiness is to perform scheduled activities, such as structural inspections or electronic tests; aiming at confirming and restoring the safety and reliability levels of the aircraft at minimum cost. Nevertheless, in between scheduled maintenance activities, operational failures often appear and put the aircraft dangerously close to unsafe situations. This is when unscheduled maintenance activities take place, aiming at restoring, as quickly as possible, some aircraft functions.

To decrease the time spent by mechanics during unscheduled activities, several constituents of aircraft subsystems are designed to be quickly replaced. These *Line Replaceable Units* are, thus, the atomic components in the unscheduled maintenance process.

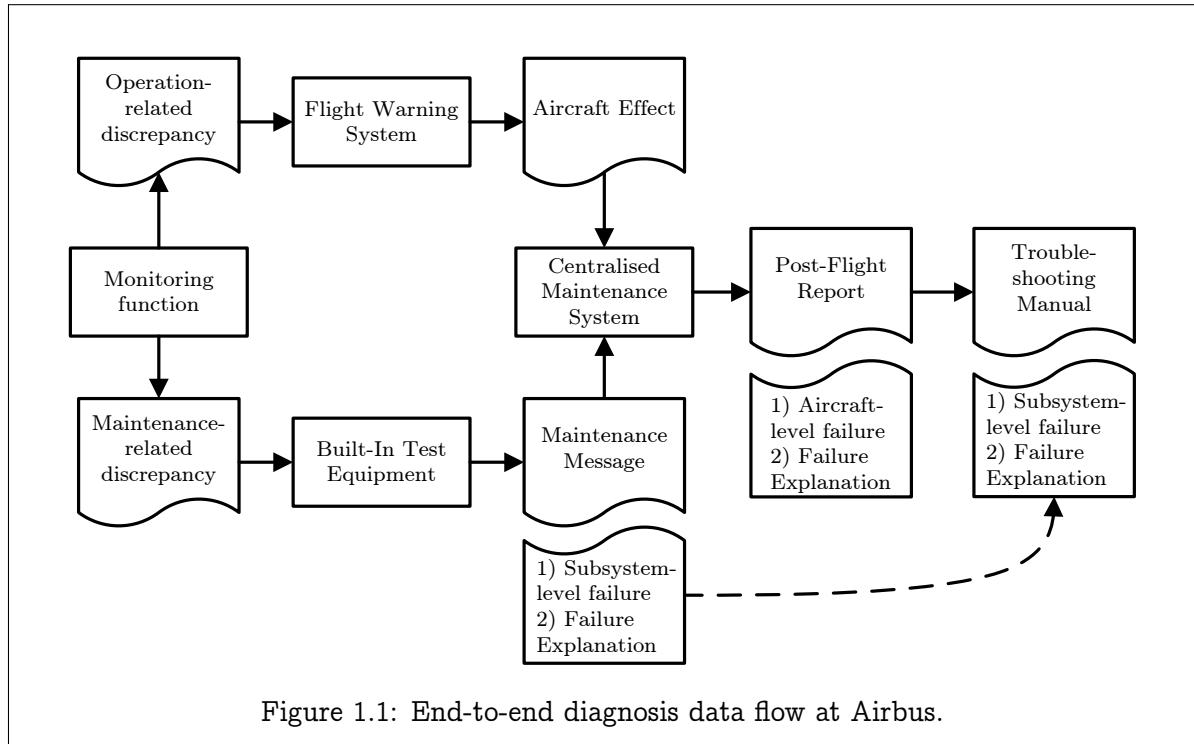
As stated before, having modular and quickly replaceable components is a key-factor for reducing unscheduled maintenance time and costs. Nevertheless, since there are thousands of Line Replaceable Units in an aircraft like the Airbus A380 and there can be tens or hundreds contributing to a single aircraft function, it is virtually impossible for a line mechanic to always infer what units to change and how to change them, when facing a given failure. This is why Airbus relies on automatic diagnostic means and manual standardized troubleshooting steps to help mechanics accomplish their goals of getting the aircraft back to nominal conditions; by providing them with “fast, accurate, and unambiguous identifications of faulty Line Replaceable Units and recommendations for corrective action” [64].

Concerning automatic diagnostic means, Airbus uses a semi-centralised diagnostic architecture. Generally speaking, sets of Line Replaceable Units are overseen by *monitoring functions* in each aircraft subsystem. Whenever a *discrepancy* between a subsystem's normal and current behaviour is detected by the monitoring function, a signal is sent either/both to: 1) the

Flight Warning System, if the discrepancy can be symptomatic of a failure with an operational impact to the flight crew; or/and 2) the *Built-In Test Equipment*, a subsystem-level diagnostic agent, if the discrepancy can be symptomatic of a failure with a line maintenance impact. Note that such discrepancies are not necessarily symptomatic of a failure (such decision is a responsibility of the Built-In Test Equipments and of the Flight Warning System).

As for the Flight Warning System, it uses the reported discrepancies along with some aircraft operational conditions to decide whether a safety-related failure is present in the aircraft; and whether the flight crew should be informed, in which case they are sent an *Aircraft Effect*.

The Built-In Test Equipments, in an unrelated but similar manner, use the reported discrepancies along with the aircraft operational conditions to decide whether some maintenance-related subsystem-level failures are present in its subsystem. If this is the case, a report - called *Maintenance Message* - is produced to indicate the presence of those failures; and, depending on the diagnostic philosophy specific to each aircraft, to indicate what Line Replaceable Units' faults, subsystem-external failures or operational conditions caused such failures.



Both Aircraft Effects and Maintenance Messages are then sent to the Centralised Maintenance System whose algorithms' main objectives are: 1) to isolate the Maintenance Messages containing the root failures that gave rise to every other failure in the aircraft, 2) associate such root failures, according to some logic, with the Aircraft Effects or with some potential *Maintenance Effects* (effects noticed by the maintenance crew) that might have been caused by them, and 3) explain those root failures in a *Post-Flight Report* as a logical combination of Line Replaceable Units' faults and operational conditions ². To do so, the Centralised Maintenance System relies on a series of databases that constitute its knowledge.

²Note that although the objective of the Centralised Maintenance System's algorithms is to explain each

Following the computation of an automatic diagnostic result, the manual troubleshooting takes place as a final isolation step towards an end-to-end diagnostic result (post-troubleshooting). In fact, associated to every reportable failure in the aircraft (and, hence, every Maintenance Message possibly emitted by a Built-In Test Equipment) there is a Troubleshooting Manual procedure that enables the mechanic to: 1) confirm the presence of the failure, 2) isolate the operational condition or the Line Replaceable Unit's fault causing the failure, and 3) perform corrective actions. As so, if mechanics want, for example, to eliminate the presence of a given Aircraft Effect, all they have to do is, in theory, follow the Troubleshooting Manual procedures associated by transitivity to the Aircraft Effect through some root failures.

Before ending this subsection, let me transpose the introduced Airbus-world concepts, depicted in Figure 1.1, to the vocabulary of the previous section:

- The monitoring functions act as observers providing discrepancies (observations) to the Built-In Test Equipments and the Flight Warning System.
- The Built-In Test Equipments act as local agents performing Model-based diagnosis on some subsystems. They use their own algorithms (rules of inference), knowledge bases (scientific and mathematical theories) and the discrepancies sent by the monitoring functions (explananda-observations) ³ to produce Maintenance Messages (local diagnoses).
- The Centralised Maintenance System acts as a global agent performing Model-based diagnosis on the whole aircraft. It uses its own algorithms (rules of inference), knowledge bases (scientific and mathematical theories), Maintenance Messages (context-observations) and Global Effects ⁴ (explananda-observations) ⁵ to produce a Post-Flight Report (aircraft-level automatic diagnosis).
- The Troubleshooting Manual procedures enrich the contents of Maintenance Messages and, by extension, of Post-Flight reports.

1.2.2 What can be improved in Airbus diagnostic scene

Although, as stated, theory presents a perfect end-to-end unscheduled maintenance process, things are somehow different in practice and:

- The number of ontologically true discrepancies computed by monitoring functions can be improved (holes in observations).
- The validity and precision of Maintenance Messages computed by Built-In Test Equipments algorithms can be improved (holes in scientific and mathematical theories, observations and rules of inference).
- The validity and precision of Post-Flight Reports computed by the Centralised Maintenance System algorithms can be improved (holes in scientific and mathematical theories, observations and rules of inference).

Aircraft Effect or potential Maintenance Effect in terms of a logical combination of root Maintenance Messages each of which explained by a logical combination of Line Replaceable Units' faults and operational conditions; this does not mean that the syntax of the results produced by the algorithms is exactly as stated.

³There are more BITE-level observations other than monitoring functions' discrepancies.

⁴Hereafter we will use the notion of Global Effect to refer to Aircraft Effects and Maintenance Effects.

⁵As in the BITE case, there are more observations other than Maintenance Messages and Global Effects.

- The Troubleshooting Manual procedures can be improved to ensure they are always associated to a Maintenance Message and both always have consistent contents (holes in scientific and mathematical theories).

Accordingly, airlines would benefit from these improvements to ensure that no situations exist where there are either/both too many explananda of a given *Global Effect* or the real causes behind the Global Effect are not among the explananda. Since these situations may induce aircraft delays with important economic consequences, Airbus is interested in avoiding them by increasing the performance of its implemented diagnostic systems.

It is at this point that Airbus diagnosis scene could benefit from automatic means to:

- detect and isolate abnormalities in Models of implemented Airbus diagnostic systems (naturally inheriting from its academic counterpart).
- ensure Troubleshooting Manual procedures are consistent with Maintenance Messages.
- validate and compare different configurable Built-In Test Equipments/Centralised Maintenance System algorithms and knowledge bases and Troubleshooting Manual procedures performances using heterogeneous real-world multi-program ⁶ data; to ensure the produced automatic/end-to-end diagnoses enable mechanics to repair nothing but the correct components causing the Effects as quickly as possible.
- obtain a return of experience on the performance of the computed diagnoses.

Moreover, having a common framework of diagnosis would enable the standard formalisation of different aircraft program concepts.

1.2.3 Industrial objectives

The Airbus end-to-end maintenance process and implemented diagnostic systems were presented; and it was argued that Airbus could benefit from automatic means to detect and to isolate abnormalities in such systems' Models, to ensure the consistency of Troubleshooting Manual procedures and Maintenance Messages, and to validate and compare the performance of its implemented diagnostic systems. The industrial objectives follow these observations.

Industrial objectives:

- To define a common framework of diagnosis for every studied Airbus aircraft program - A380, A400M, A350 and research.
- To define a method, and specify and develop a tool, to detect and to isolate Model abnormalities in the Centralised Maintenance System.
- To define a method, and specify and develop a tool, to check for the consistency between Troubleshooting Manual procedures and Maintenance Messages.
- To define a method, and specify and develop a tool, to validate and compare different configurable Built-In Test Equipments/Centralised Maintenance System algorithms and knowledge bases and Troubleshooting Manual procedures performances using heterogeneous real-world multi-program data.

⁶At Airbus, a program is a class of aircraft with similar characteristics, as the A380 or the A350 program.

These industrial objectives are in straight connection with the academic one and both nourish each other. Finally, due to its complexity, I suspect systems such as nuclear power plants, satellites or cars may suffer from these same problems. Hence, the results obtained may be of use in such areas.

1.3 Overview

It is now time to provide an overview of the remainder chapters of this book; a book which is both vertical and wide in scope, ranging from fundamental problems in philosophical and Artificial Intelligence accounts of explanation and Model-based diagnosis to the industrial V&V process of implemented Airbus diagnostic systems.

Chapter 2. This chapter focuses on the Artificial Intelligence classical account of Model-based diagnosis and the exposure of its intrinsic issues. I initiate the chapter by presenting the reader with a historical view of explanation, both in Artificial Intelligence and philosophy. Then, motivated by the lack of a fundamental framework for reasoning about the Models used by explanation, i.e. scientific and mathematical theories, observations and rules of inference; I establish a thesis on reality, cognition, information and their connections with these Models. This is, in my opinion, a necessary starting line to the deep investigation of this field. From its foundation, Reiter's classical diagnostic framework [90] is then introduced from a different perspective. Furthermore, some problems of Model-based diagnosis are revealed and a series of diagnostic system's Models and diagnostic results' properties, as well as the relations between them are uncovered. Finally, I justify my contribution by comparing it to others in the literature.

Chapter 3. From an extensive account of Model-based diagnosis, I move to reasoning about abnormalities in Models of implemented diagnostic systems and explanation. This chapter starts with an original theory of meta-diagnosis, providing the theoretical tools to tackle the problem; as well as with a process for modelling meta-diagnostic problems. I then argue that such theory shares the syntax of Reiter's classical diagnostic framework, although with different semantics. Exploiting this observation, I specify and develop a meta-diagnosis tool, MEDITO, and put it to use against some toy problems. Finally, the novelty of this contribution is justified with an overview of some related works in the literature.

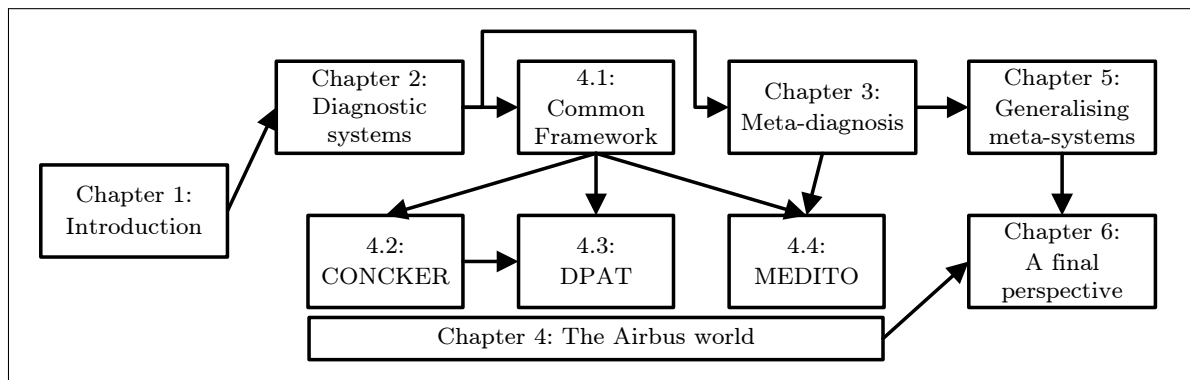
Chapter 4. Moving to the industrial world, the fourth chapter of this dissertation tackles the core of the industrial problems. First, Airbus is provided with a clear definition of its diagnoses and implemented diagnostic systems in the framework of Chapter 2. This constitutes the common framework of diagnosis for every studied Airbus aircraft program - A380, A400M, A350 and research - enabling the comparison of diagnostic methods and results. Then, using this framework, I bring down the problem of defining a method and specifying and developing a tool to check for the consistency between Troubleshooting Manual procedures and Maintenance Messages. This is how CONCKER, a CONSistency ChecKER, appears to the Airbus world. With CONCKER introduced, I then focus on problem of automatically validating and comparing different configurable Built-In

Test Equipments/Centralised Maintenance System algorithms and knowledge bases and Troubleshooting Manual procedures performances using heterogeneous real-world multi-program data. The problem is separated in pieces and all of the previous work is used to tackle it by building a Diagnosis Performance Assessment Tool, DPAT, and defining its underlying method. Finally, MEDITO is used, in an Airbus context, to show how it can detect and isolate Model abnormalities in the Centralised Maintenance System. As in the previous chapters, I justify the originality of our contributions by providing, as far as known, same-scope works in the related literature.

Chapter 5. The industrial objectives of this dissertation achieved, I focus one again on meta-systems and on the unproven claim regarding the meta-diagnosis thesis' generality. In a nutshell, this chapter is concerned with providing evidence for the usage of meta-diagnosis to detect and to isolate abnormalities in Models of 1) non-logical-based diagnostic systems, and 2) meta-systems other than implemented diagnostic systems.

Chapter 6. This last chapter is focused on the contributions made in this book, discusses their Achilles' heel and finished with a series perspectives opened by this dissertation.

The dependency between chapters and sections is depicted hereafter.



Finally, this dissertation is deeply related to model theory. Hence, the following words, whose definitions are provided in Appendix A, will be reserved to such context: logic, tuple, signature, structure, signature interpretation, correct structure interpretation, substructure, extension, model ⁷, isomorphism, theory, satisfiability, entailment, syntax, semantics, inference, logical implication, consequence, theorem and complete theory.

⁷Note that the capitalised word “Model” will be distinguished from the uncapitalised one “model”. The former will always refer to one of the three Models, i.e. scientific and mathematical theories, observations and rules of inference; while the latter will define a structure in which a given sentence or theory is true.

Chapter 2

Diagnostic systems

All our knowledge has its origin in our perceptions.

- Leonardo da Vinci

As presented in the introductory chapter, the main objective of this dissertation is to provide Artificial Intelligence and Airbus with means for reasoning about abnormalities in Models (scientific and mathematical theories, observations and rules of inference) of implemented diagnostic systems and explanation in general. The prerequisites for achieving such goal are both a formal and clear definition of a well-founded framework of Model-based diagnosis, and a formal account of Model abnormalities and of their relations with diagnoses and explanans in general. The present chapter addressed these subjects.

In the first section, the reader is presented with a historical view of explanation, both in Artificial Intelligence and philosophy. Two reasons form the rational behind such exposé. First, it aims at making it clear that there are many different notions of explanation, both in Artificial Intelligence and in philosophy, without a generally accepted conceptual and technical account. Second, it highlights the fact that every single account of explanation relies on Models, i.e. scientific and mathematical theories, observations and rules of inference, as a means to bring up the answers we need.

Being Models essential for Model-based diagnosis (and explanation in general), the second section reveals my thesis on reality, cognition, information and their connections with them. In my opinion, establishing a solid foundation of diagnostic systems is a fundamental step into achieving our objectives; a step which, to my best knowledge, is missing in almost every account of Model-based diagnosis.

In the third section, this thesis is used to expose the classical logical framework of Model-based diagnosis in Artificial Intelligence from a different perspective; a perspective which is based on the difference between the real-world and the cognitive concepts of, for instance, system, component, abnormality, function, input, output or replaceability.

Having a well-founded framework of Model-based diagnosis, the fourth section is devoted both to the formal study of Models' properties in the case of diagnostic systems, transforming the informally exposed Model "holes" into a set of well-defined mathematical definitions; and to the establishment of a series of relations between such properties and those of the computed diagnoses (or explanans in general).

Finally, in the last section I relate this chapter to many works in philosophy, science and mathematics; and use such connections as a means to express the originality of my contribution.

2.1 A historical view of explanation and Models

Explanation is far from being a simple modern subject. Accordingly, and in an attempt to provide the reader with a solid background, this section offers a brief overview on the major historical accounts of Model-based diagnosis and explanation in general, both in philosophy and Artificial Intelligence.

2.1.1 Explanation in philosophy

Aristotle exposed what can be considered nowadays as one of the first general accounts of explanation in his discussion of the four causes [6] [7]. With the idea that explanation relies on causation, Aristotle introduced the notion of *aition* (or explanatory factor), i.e. whatever appropriately occurs in response to a [rephrased] why-question. With many possible senses, *aitia* might be: 1) material, i.e. on the form "x is what y is made out of", 2) formal, i.e. "x is what it is to be y", 3) efficient, i.e. on the form "x is what produces y", and 4) final, i.e. on the form "x is what y is for". In the 4th century B.C., causation and explanation were already in a deep connection.

Between Aristotle and the present days, many philosophers continued to address the problem of scientific explanation. In the 1st century B.C., Lucretius presented an explanation of the origin of the universe. What is important for us is that, in his path, he came across two fundamental philosophical problems: determinism and free will; in straight connection with causation and scientific explanation. According to Lucretius, determinism is false for it implies not having free will, which, in his opinion, humans do have.

Fast-forwarding, in the 17th and 18th centuries other philosophers came to change our vision of causation and scientific explanation. Laplace, with his "demon", told us that "Given (...) an intelligence which could comprehend all the forces of which nature is animated and the respective situation of the beings who compose it - an intelligence sufficiently vast to submit these data to analysis - (...) nothing would be uncertain and the future, the past, would be present to its eyes" [72]. Hence, Laplace defended determinism over free will and stated that if an intelligence could access every observation about the world, then it would be able to predict all the future and explain all the past. By comparing Laplace's account with the introductory definition of both prediction and explanation tasks, we can see that Laplace's account has a deep implication of the nature of observations and scientific and mathematical

theories. In his opinion, if an intelligence existed capable of assessing all observations and scientific and mathematical theories of the universe then it would be able to fully infer the past and the future, for these Models completely determine the universe. I encourage the reader to retain this thought; a thought which was shared by both Leibnitz [110] and Kant [66] in the form of the principle of sufficient reason, according to which “nothing happens without a reason”. Finally, and also in these centuries, Hume brought up another difficulty to the quest of finding a general accepted account of scientific explanation. According to him, causal connections are purely cognitive and do not exist in the real-world. As so, we believe that there is a causal relation when we perceive 1) a temporal priority between causes and effects, 2) a spatio-temporal contiguity of causes to effects, and 3) the fact that the occurrence of the causes always produces the occurrence of the effects [92]. This is how he accepted both determinism and free will (compatibilism).

As we moved away from Laplace, Leibniz, Kant, Hume and many others, and entered the 20th century, new data came into the scientific explanation equation: the Heisenberg uncertainty principle and quantum mechanics, relativity and the Einstein-Podolsky-Rosen paradox [44]. First of all, Heisenberg uncertainty principle guarantees that no Laplace demon could have access to every observations in the universe. After all, according to such principle, no observation can be unambiguously gathered. Moreover, since quantum mechanics, if accepted, describes an indeterministic world, then even Laplace’s demon’s scientific and mathematical theories could be at risk ¹. On the other hand, the Einstein-Podolsky-Rosen paradox states that quantum mechanics provides an incomplete description of reality. Therefore, the question of determinism and indeterminism remained opened. If we add to this the Duhem-Quine thesis [51], stating that scientific theories are underdetermined by physical evidence, we get a rough environment for explanation.

After the EPR paradox, many philosophers worked on new accounts of scientific explanation and its underlying Models. From the existing explanation-related works between the 1930’s and nowadays, let us focus on those of Popper, Hempel, Oppenheim and Kuhn.

Introduced by Popper in [89] and formalized by Hempel and Oppenheim in [59], the Deductive-Nomological model (D-N model) of scientific explanation marks the transition between the attitude towards explanation in the beginning of the century and the modern one. According to this model, “a D-N explanation of a particular event is a valid [classical] deductive argument whose conclusion states that the event-to-be-explained did occur. Its premises must include essentially at least one general law” [92]. Thus, if for instance someone asked why a the length of a flagpole shadow is x , then a D-N explanation could be that the height of the flagpole is y and the angle between the sun’s rays and the flagpole is α ; since from these facts and the “laws of physics” one can certainly deduce the length of the shadow ². Firstly, let me point the readers’ attention to the similarity between the D-N model and the rough definition of explanation provided in the beginning of the chapter. Secondly, notice that the D-N model: 1) does not impose a temporal asymmetry (or at least simultaneity) between causes and effects, and 2) imposes the use of classical deduction instead of an inference in an arbitrary

¹In [42] Earman proved that Newton physics are not deterministic, a required hypothesis for Laplace, but that was not the paradigm in Laplace, Leibniz, Kant and Hume’s era.

²The flagpole example was introduced by Bromberg in [21].

logic. The rational behind these differences is two-fold. Concerning the temporal symmetry of the D-N model, it has been criticized by numerous philosophers [92]. For example, the D-N model tells us that we can explain the height of the pole from the length of the shadow, which is far from being accepted. As for the deduction argument, it is far more trickier. In fact, classical deduction presents many “problems” for scientific explanation, such as, for instance, monotony. I will come back to these problems later in this dissertation.

The Hempel-Oppenheim D-N model provided a “consensus”³ in the scientific explanation community. Nevertheless, as recognized by Hempel, it was not enough; maybe because it clearly fails to include quantum mechanics due to the classical-deduction problem. This is why in *Deductive-Nomological vs. Statistical Explanation* [57] and *Aspects of Scientific Explanation* [58] Hempel introduced a new type of model in parallel with the D-N one: the Inductive-Statistical (I-S) model; according to which induction substitutes deduction in D-N models with a certain degree of probability. But even the sum of these two models had some issues, as discussed in [92], among which the same time symmetry problems.

All in all, explanation has been studied for centuries in philosophy without a single accepted account for it. Coming across several issues with causation, determinism, free will, time symmetry, unprovability of models, underdetermination and much more; explanation is far from being a trivial subject. And yet, explanation is ubiquitous in our daily lives.

2.1.2 Explanation in Artificial Intelligence

If we leave the realm of philosophy and enter the much more recent Artificial Intelligence one, explanation and its underlying Models gain another interesting contour: not only we want to understand how to provide scientific explanations, but we also want robots to explain.

Accordingly, let us focus, Model-based-diagnosis-wise, on the explanation world in AI. In [37], de Kleer provided us with what can be seen as one of the roots of Model-based diagnosis: a “simple theory of troubleshooting” and its “unexpected complexities”. As so, in 1975, the problems of incompleteness of expert knowledge - which we can see as the incompleteness of scientific and mathematical theories in each expert’s mind - and the uncertainty of measures - corresponding to the accuracy of observations in this dissertation’s framework - appeared for the first time.

Following de Kleer, authors such as Davis, Genesereth and Shirley [35, 50, 96] shed some light on the community. However, it was not until Reiter [90] and de Kleer and Williams [40] seminal works that a clear and accepted formalism for Model-based diagnosis appeared. In 1987 we had, for the first time, a theory of diagnosis from the first principles distinguishing:

- the artefact to be diagnosed in a real-world context,
- a logical theory for representing the artefact’s structure and behaviour,
- some observations obtained from the real-world and mapped into a logical theory, and
- an algorithm for computing diagnoses.

³The quotation marks account for the fact that in philosophy and science there is rarely a total consensus.

As for the notion of explanation itself, Reiter, de Kleer and Williams encoded the idea that an explanan is any state of the world satisfiable with some scientific and mathematical theories (the artefact representation) and with the observations gathered ⁴. Although not in phase with the philosophical accounts, for instance in the temporal aspects, these works provided a first step towards the formal treatment of explanation (as diagnosis) in Artificial Intelligence ⁵.

Naturally, many works followed these ones. On the one hand, Aleliunas, Console, Goebel, Poole, Theseider Dupré and Torasso [87, 27] coined the notion of abductive diagnosis; where explanans are seen as any states of the world that, along with some scientific and mathematical theories representing the artefact to be diagnosed, entail all the observations gathered. On the other hand, Console, de Kleer, Dressler, Hamscher, Poole, Struss, Torasso and Williams [29, 53, 41, 85, 101] worked on the scientific and mathematical theories side, and on how to integrate models of good/faulty behaviour into consistency-based and abduction-based diagnoses.

In 1991, the notion of explanation behind Model-based diagnosis evolved once more with the work of Console and Torasso [30]. By taking all the previous contributions into account, these authors proposed a view of explanation as a reasoning process where *explanans* explain some explananda-observations if the latter can be logically implied from the former and some context-observations, along with some scientific and mathematical theories representing the artefact to be diagnosed. Although without explicit temporal constraints or a strategy to separate context from explananda observations, this notion of explanation certainly approaches the one introduced in the beginning of the present chapter.

From Console and Torasso's work until nowadays, several authors explored and enriched the Model-based diagnosis world in many different axes. There have been many studies in the development of diagnostic algorithms [122, 121]; in the exploration of techniques for managing complexity [78, 26]; in the direct or indirect inclusion of time [22]; among many other fields. Conversely, and despite the ubiquity of, for instance, invalid and imprecise diagnoses, few works focused on Models and their "problems", a central part of Model-based diagnosis.

2.2 Building a deep foundation

In the previous section, a historical view of explanation and Model-based diagnosis has been provided. With such background, one is now able to understand that there are many different notions of explanation without a generally accepted conceptual and technical account. Moreover, it should also be clear that every one of them relies on Models. Hence, I devote this section to providing a deep foundation of Models and, by transitivity, implemented diagnostic systems (and explanation in general). Such thesis relies on the concepts of reality, information and cognition. In order to highlight the model-theoretic roots of such thesis, every word used in this section on a model-theoretic context will be written in **this font** (eg. **structure**). I encourage the reader to refer to Appendix A for more details on model theory.

⁴The technical view over Reiter, de Kleer and Williams will be a part Chapter 2.

⁵In [90], Reiter tells us that his framework can accommodate a variety of logics from temporal monotonic to non-monotonic ones. However, besides some brief notes on the intrinsic nonmonotony of diagnosis, he gives no clues on the characteristics these logics should have to integrate a correct notion of explanation.

2.2.1 On reality, information and cognition

Human beings, in the same manner as robots and its encoded diagnostic systems, are a part of what I will hereafter refer to as *reality* or *real-world*. This seemingly obvious statement has more to it than what meets the eye. In fact, let me introduce it as the first pile of our foundation and justify its importance along the present chapter.

Philosophers, scientists and mathematicians have struggled for years to define this “reality” without a well accepted result. Hence, I will just assume that reality exists and that we are a part of it; without ever trying to define it. Moreover, let me borrow a number of Wheeler’s ideas [117] and introduce some properties one generally *thinks* reality must have.

To start with, Wheeler’s “No super-turtles! No tower of turtles!” principle tells us that reality must exist by itself and one should stop looking for ever-prior laws (super-turtles). This being so, as Wheeler puts it, “to endlessness no alternative is evident but a loop, such a loop as this: Physics gives rise to observer-participancy; observer-participancy gives rise to information; information gives rise to physics.”

Wheeler’s principle brings with it a series of useful data. First, we notice a special kind of “objects” ⁶, the observer-participants, and a straightforwardly related distinction between reality and *mind* or the real-world and the *cognitive-world*. After all, Wheeler’s “observer-participants” differ from “physics” due to their consciousness and cognitive capabilities. This is the second pile of our foundation.

The second interesting aspect behind the “No super-turtles! No tower of turtles!” principle is the role of *information* as a narrow bridge between reality and cognition. Information is a mathematical **structure** that can be seen as reality’s quasi-dual in the cognitive-world. This is our third pile and I will return to it in a few paragraphs to detail what information is.

With the triple reality, information and cognition introduced, we can move a step further into establishing deep foundation for reasoning (mind) with and about real-world (reality) applications and, in particular, implemented diagnostic systems. Let me do so by introducing the concept of perception.

Perception is the process through which observer-participants **interpret** a **substructure** of information corresponding to a sensed part of reality. Being based on each observer-participant **interpretation**, *percepts*, the cognitive-world resultant **structure** of perception, are shaped by hers **theories** and goals. As for *observations*, these are, simply put, **theories** that aim at **axiomatising** (or at least at being **satisfied** in) percepts. A classical example of how percepts based on the same **substructure** of information may be different is conveyed by optical illusions such as the “Knight” (cf. Figure 2.1); perceived by some as a series of pixels forming a pattern and by others as a knight riding his horse ⁷.

With it, the concept of perception brings another Wheeler-related topic: the “no question, no answer” principle through which the existence of information (answer) is predicated to

⁶The “^o” account for my intrinsic obligation to use a cognitive term when referring to the real-world.

⁷As Wheeler puts it: “what we call reality consists ... of a few iron posts of observation between which we fill a *papier-maché* of imagination and theory” [115].

each observer-participant perception (question) [117]. As so, and in model-theoretic terms, information is the **structure** underlying “every” observer-participant percept.



Figure 2.1: An optical illusion presenting a classical example of how the same perceived sensed part of reality may originate more than one percept.

The reader may feel this definition to be somewhat ambiguous. After all, *who* are “every” observer participants?

I will start to answer this question by borrowing some famous words from Einstein: “time and space are modes by which we think and not conditions in which we live”. This, supported by Wheeler’s “super-Copernican” principle which states that no particular point in time is special, tells us that “every” observer-participant are those that exist (or are a part of the real-world) in the limit of time, space or any other cognitive-axis known or unknown to us.

To sum up, the reader has been presented with the three piles of our foundation: reality, information and cognition; related through observer-participants perceptions and actions. In this context, the following main principles have been exposed:

Principle 1: Observer-participants and robots are a part of physics: they exist in reality.

Principle 2: There is a distinction between the real-world and the cognitive-world. Observer-participants differ from other parts of physics; for they exist in the former, but reason about and own their consciousness and cognition to latter.

Principle 3: Information is a mathematical **structure** (of the cognitive-world), hereon noted Ψ , underlying every observer-participant's - those that exist in the real-world in the limits of time, space and possibly other unknown axes - percepts and representing their sensed part of reality.

Principle 4: Perception is the process through which observer-participants **interpret** a **substructure** of information corresponding to their sensed part of reality. Percepts are the result of perception. As for observations, they are **theories** that aim at **axiomatising** (or at least at being **satisfied** in) percepts.

2.2.2 On theories and observations

In the previous section I introduced the concept of perception and how a **substructure** of information corresponding to some sensed part of reality is **interpreted** by observer-participants using their own cognitive-world-based **theories** and goals. Let me now focus on these **theories**.

A **theory** is a set of sentences in a **language**, usually put together to describe a fact or phenomenon. Being cognitive in essence, depending on the object of its description **theories** can be classified as being either *scientific* or *mathematical*; with the former being concerned with the real-world, e.g. quantum mechanics, and the latter with the cognitive-world, e.g. set theory. In a nutshell, mathematical **theories** lack of a connection with reality, a connection which is in the nature of scientific **theories**.

Concerning its building process, mathematical **theories**, evolve through a method in which:

1. a set of **axioms** is invented;
2. some **theorems** are *conjectured*; and
3. if these **theorems** can be [logically] **proved** by a sound and complete proof system, then they are *logically true* and are added to the **theory**.

Scientific **theories**, on the other hand, are formed using the *scientific method*, where:

1. observations are gathered and analysed through the existing scientific **theories**;
2. if no existing **theory** is **consistent** with the observations, then a new **theory** is *conjectured* to fit these empirical data; and
3. the new **theory** is tested against new observations. If no empirical tests contradict the new **theory** then it is *assumed* to be *ontologically true*.

These methods to build up mathematical and scientific **theories** are responsible for their main characteristics. Firstly, the reader may note the discriminate usage of the concepts *logical truth* and *ontological truth*. Let me borrow some words from Tarski to provide a proper informal introduction to the subject: "[an ontological] true sentence is one which says that the state of affairs is so and so, and the state of affairs is indeed so and so" [106]. Hence, the famous sentence "All men are mortal", for instance, is assumed as being an ontological truth, since as far as we know, all men are indeed mortal. However, the also famous sentence "All birds fly" is not an ontological truth since according to our perception, there are some birds that do not fly. As for logical **truths**, they represent all the **axioms** and **theorems** (obtained

using a sound and complete proof system) in a **theory**. The notion of ontological truth of a **theory** is formally defined below; where, as defined in Appendix A, $\text{Mod}(\cdot)$ denotes the set of **models** of a given **theory**, $\mathcal{M} \subseteq \mathcal{P}$ represents the fact that \mathcal{M} is a **substructure** of \mathcal{P} , $\mathcal{M} \rightleftharpoons \mathcal{P}$ the fact that \mathcal{M} is **isomorphic** to \mathcal{P} and $\Gamma(\mathcal{M}, \mathcal{P})$ a correct interpretation of \mathcal{M} in \mathcal{P} .

Definition 1 (Ontological truth of a theory). *Let Ω be the set of all structures, $\Psi \in \Omega$ the structure of information and Γ a correct interpretation. A theory T is an ontological truth iff $\exists \mathcal{M} \in \text{Mod}(T) \exists \mathcal{P}, \mathcal{R} \in \Omega \exists \Gamma(\mathcal{M}, \mathcal{P}) (\mathcal{P} \subseteq \mathcal{R}) \wedge (\mathcal{R} \rightleftharpoons \Psi)$. If T is an ontologically true theory, then so are every sentences in T .*

The relation between logical **truths** and mathematical **theories**, and ontological truths and scientific **theories** should now be clear. As so, let me now introduce a fundamental theorem:

Theorem 1. *Let T be an ontologically true theory in a classical logic \mathcal{L} , and φ a sentence in that same logic. If $T \models_{\mathcal{L}} \varphi$, then φ is ontologically true.*

Proof.

1. In a classical logic, $T \models \varphi$ if every model of T is a model of φ .
2. T being ontologically true, it has a model \mathcal{M} such that $\exists \mathcal{P}, \mathcal{R} \in \Omega \exists \Gamma(\mathcal{M}, \mathcal{P}) (\mathcal{P} \subseteq \mathcal{R}) \wedge (\mathcal{R} \rightleftharpoons \Psi)$.
3. 1) and 2) $\Rightarrow \varphi$ has a model \mathcal{M} such that $\exists \mathcal{P}, \mathcal{R} \in \Omega \exists \Gamma(\mathcal{M}, \mathcal{P}) (\mathcal{P} \subseteq \mathcal{R}) \wedge (\mathcal{R} \rightleftharpoons \Psi)$, i.e. φ is ontologically true.

■

Theorem 1 is the formal rational behind our intuition that the consequences of ontological truths are ontological truths. To my knowledge an original theorem, it has more to it than meets the eye. First, using the direct interpretation, fields such as prediction, explanation or planning are offered a formal condition guaranteeing the ontological truth of **entailed statements** in classical **logics**. Secondly, the contraposition of Theorem 1 provides a much more important result: while both mathematical and scientific **theories** can be logically **proved**, insofar as logical **truths** can be entailed from other logical truths in a given logic; scientific theories can also be *ontologically disproved* by new observations. This single characteristic in the essence of scientific theories is what lies behind Popper’s “Falsifiability, or refutability, or testability” [88], behind Hume’s intuition in his problem of induction [63] and behind Einstein’s insight: “No amount of experimentation can ever prove me right; a single experiment can prove me wrong” [43]; courtesy of the straight connection between these **theories**, information and observer-participants. As it was stated before, information is the **structure** underlying every observer-participants percepts. Since, for example, “future” observer-participants can, in theory, perceive their sensed part of reality through a **substructure** of information different from “present” and “past” observer-participants, then a “present” **theory** can always, in principle, be ontologically disproved by “future” observer-participants’ observations. Being observations, through percepts, shaped by our **theories** and goals, and being **theories** impossible to be proved ontologically true, then observations themselves can never be proved ontologically true. This proof-lacking-loop is the actual rational nourishing most of our work in this dissertation; and is the reason why the problems of ontologically false theories and

observations mentioned in Chapter 1 are the most annoying ones: ontological truth of theories and observations is and will always be an assumption. I will come back to this in Section 2.4.

To sum up, the reader have been presented with three main principles that, when added to the ones exposed in the previous subsection, form the seven principles of our foundations:

Principle 5: Mathematical **theories**, having no connection to the real-world, can be **proved** [logically] **true**.

Principle 6: Scientific **theories**, connected to the real-world, can be **proved** [logically] **true**, but can never be proved ontologically true.

Principle 7: Observations, shaped by scientific **theories** through perception, inherit from these **theories**' incapability of being proved ontologically true.

2.3 Towards a framework of diagnosis

In the last section I introduced the concepts of reality, information and cognition, and argued that these notions are related through observer-participants' perception and action. I then presented scientific and mathematical theories; for observer-participants' observations are forged in these theories' moulds. Finally, I argued that both observations and scientific theories can never be proved ontologically true. This is how the seven principles of our foundations were built; and how two of the three Models were discussed, being rules of inference discarded of reference for its already precise definition in proof theory and logics. This being said, it is now time to jump to the Model-based Diagnosis (MBD) world to introduce a framework of diagnosis relying on these three Models and distinguishing reality and cognition.

2.3.1 On systems, components, abnormality and other concepts

Diagnosis is usually defined as a reasoning process which aims at determining the normal and abnormal components of a system, typically for replacing the abnormal ones [90, 40, 39]. Some fundamental questions arise straight away when one is aware of the seven principles:

Question 1: Since the notions of component and system belong to the cognitive-world, what do they represent in reality? In other words, what is the real-world equivalent of a component and a system?

Question 2: What is it for a component to be abnormal? Is the notion of abnormality absolute or is it relative to an observer-participant's perception?

In my opinion, it is impossible to start a proper discussion on Model-based diagnosis without a clear answer to these questions. Before doing so, let me assert that the following notions of *system*, *component*, *abnormality*, *function*, *input*, *output* and so on are all from the cognitive-world: they designate what we see as constants in physics necessary for us to reason in the real-world.

Addressing the first question, the notion of *system* can be defined by relying on *parameters*, *goals* and on the concept of perception (cf. Principle 4):

Definition 2 (Parameter). *A parameter represents a sensed part of reality perceived as a flow of physics⁸. Its valuation is the perceived flow rate.*

Definition 3 (Goal). *A goal is a [possibly empty] set of parameters each of which with an admissible value range, and a [possibly empty] set of mathematical relations - also called functions - that must hold between them [25].*

Definition 4 (System). *A system represents a sensed part of reality perceived as a part of physics with: 1) an identity [118, 119, 49, 73, 54, 98] and 2) a [possibly empty] set of assigned goals.*

This notion of system, based on the framework of the previous section, is connected to the equivalent ones from physics and systems engineering. Concerning the former, we can read in [94] that a system is “a small portion of the universe” and that a “system boundary [is] an imaginary surface (not necessarily coinciding with a physical surface) that divides the universe into the system and the environment surrounding the system”. Clearly, this is our perception of a part of reality with an identity. As for the systems engineering perspective, in [3] a system is defined as “a construct or collection of different elements that together produce results not obtainable by the elements alone. The elements, or parts, can include people, hardware, software, facilities, policies, and documents; that is, all things required to produce system-level results. The results include system-level qualities, properties, characteristics, functions, behaviour, and performance”; and in [82] a system is seen as “a combination of interacting elements organized to achieve one more stated purposes” with an element being “a member of a set of elements that constitutes a system”. In systems engineering, a system is thus defined, by and large, as a set of interacting elements with a goal. This is clearly linked to Definition 4, especially if one takes into account the notion of “component” defined in a few paragraphs.

Moving a step further, from all the parameters perceived by an observer-participant, those that correspond to flows of physics inside the system are called *internal* system-parameters; while those that represent flows of physics into and from a given system are named *boundary* system-parameters. Boundary system-parameters corresponding to flows of physics into a system are also called *input* system-parameters; while those associated to flows of physics from a system are referred to as *output* system-parameters. It is at this point that the notion of *context*, depending on the concept of perception (cf. Principle 4), is useful; for the input and output character of system-parameters is not static in its nature:

Definition 5 (Context). *A context of a system represents a perceived partition of boundary system parameters into input and output system parameters.*

From the concept of context, let us move to the one of component; a concept that relies on the notions of replaceability and perception (cf. Principle 4):

Definition 6 (Replaceability). *A sensed part of reality is perceived by an observer-participant as being replaceable iff, when interchanged through her actions by another*

⁸In today’s paradigm, this “flow of physics” is usually seen as an exchange of mass, momentum or energy [19], related in Einstein’s special theory of relativity.

*sensed part of reality perceived as being identical, the substructure of information interpreted by her remains the same*⁹.

Definition 7 (Component). *A component represents a sensed part of reality perceived as a part of physics: 1) with an identity, 2) that is a part of a system and 3) that is replaceable.*

From the definitions of system and component it follows, naturally, that:

- a component becomes a system if it is assigned with a goal, and
- a system becomes a component (usually of another system) if it is replaceable.

With these notions in mind, one is now finally able to understand how the provided definition of system perfectly fits the systems engineering one. As for the second question, note that since components can be seen as systems if one assigns them goals, the notion of abnormality can be defined as follows:

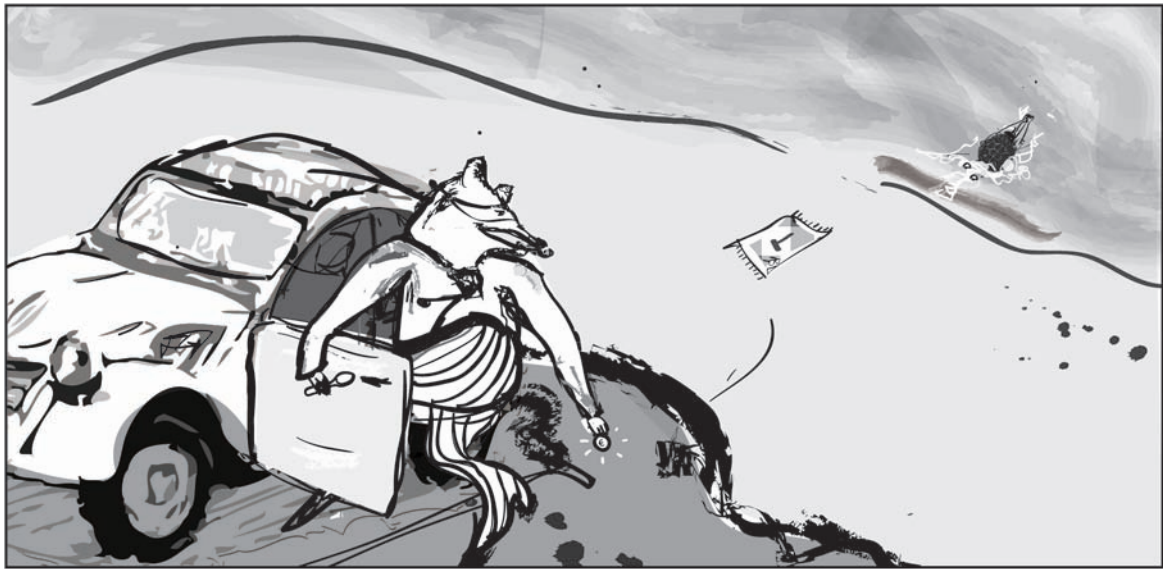
Definition 8 (Abnormality). *When assigned with a goal, a component is: normal, if the goal is met; and abnormal otherwise. A goal is met whenever all its underlying parameter values are within range and all its underlying relations hold [91, 25].*

Let us go back to Mr. Tortuga to illustrate most of these concepts. Suppose that our hero's car contains a very old jukebox that works as follows: 1) the user inserts a coin in the jukebox, 2) the jukebox gives the user an iron token, 3) the user chooses a song and inserts the token back in the jukebox, and 4) the song is played. Moreover, suppose also that, when understanding the way his jukebox worked, Tortuga identified each iron token as a component of the jukebox, each one being also a sub-system whose goal is to have fixed weight, size and marks enabling the jukebox to work. One fine day, Tortuga parks his car near the beach and goes out for a long swim in the ocean. Meanwhile, suppose that a token left outside of the jukebox gets severely oxidized. When coming back to his car, Tortuga still perceives the token as being the same token as before. After all, he is still able to perceive that sensed part of reality as having the same or an indiscernible identity as the token he left unoxidised before leaving the car. However, when putting the token in the jukebox, Tortuga realizes that the jukebox is unable to play a song when that particular token is inserted: the token must be abnormal. For the sake of the argument, suppose the same situation but imagine that, while Tortuga swims, Mr. Zapato goes to his car and forges the token into a spoon. This time, when Tortuga enters his vehicle he finds no token, immediately concluding that it was lost (although he finds a very strange spoon in the passenger's seat). Mr. Tortuga example illustrates the notions of system and component. Moreover, it shows us that identity is a very important concept and illustrates the difference between this notion and that of abnormality.

Finally, imagine that the jukebox in Tortuga's car has a keyboard enabling the song selection and a screen showing a series of songs and their associated lyrics. In this case, Tortuga perceives the data flowing from the keyboard to the jukebox as an input system-parameter

⁹The *identity of indiscernibles* and the *indiscernibility of identicals* are two principles naturally related to this discussion. The reader is encouraged to refer to [110, 17, 1, 33, 48] or [55] for more information.

and the data flowing from the screen as an output system-parameter. No other combination of input and output system-parameters is perceived by Tortuga since he simply cannot “write letters on the screen so that they get pressed on the keyboard”. Consider, for example, the car battery. If, on the one hand, it is true that Tortuga uses the current flowing from the battery to charge his electronic compass, in which case he may perceive the electrons flowing out of the negative pole to be an output system-parameter; on the other hand it is also true that Tortuga charges the car battery from time to time, in which case he may perceive the electrons flowing into the negative pole to be an input system-parameter. What actually differentiates both situations is the context of utilisation. This last example illustrates the notions of context and input and output system-parameters.



2.3.2 Characterising diagnoses

With the concepts of system, component and abnormality introduced, it is now viable to expose the classical logical framework of Model-based diagnosis while assuring a clear distinction between the real and the cognitive worlds ¹⁰. Such framework arises from the works of Console, de Kleer, Mackworth, Reiter, Struss, Torasso and Williams [90, 40, 30, 39, 100] and is enriched through some definitions; definitions whose justification will arise with the developments in the remainder of the present chapter.

The classical logical framework of Model-based diagnosis relies, unsurprisingly, on the three Models, i.e. scientific and mathematical theories, observations and rules of inference, and on the concept of explanation. Hence, it is natural for a description of such framework to focus on each these Models separately and converge towards what a diagnosis is. Before starting this description, note that I will hereafter use the language and inference rules of first-order logic with equality; for it is already defined in literature and has an expressive power necessary to tackle the examples treated in this dissertation. Nevertheless, let me also strongly assert, as

¹⁰Some of the work presented in this section was published in [13].

Reiter did in [90], that the results below apply to a series of other logics. This is extremely important since, as we will see further in this dissertation, the whole notion of explanation depends on this fact. I encourage the reader to retain this statement.

The first Model: believed systems

As for the first Model, it appears in the classical logical framework of Model-based diagnosis in the form of believed systems, relying on the notions of system, component and abnormality (cf. Definitions 4, 7 and 8):

Definition 9 (Believed system). *A believed system is a pair $(SD, COMPS)$ where:*

- *SD , the believed system description, is a set of first-order sentences. Semantically, it is a scientific and mathematical theory describing one or more systems.*
- *$COMPS$, the believed system components, is a finite set of constants. Semantically, each element c of $COMPS$ represents a component whose abnormality one wants to determine.*

Hence, believed systems are, intuitively, a scientific and mathematical theory describing a system diagnosis-wise; with a set of distinguished elements determining the components of such system. Being a scientific and mathematical theory, believed system descriptions typically characterise what is usually referred to (in Model-based diagnosis) as one's structural, behavioural, functional and teleological knowledge about the system and its underlying components [25, 91]. In a nutshell, believed system descriptions represent, by and large, some system goals, how the system components interact and how the system behaves when certain components are normal and abnormal, all this depending on the system context. This is where certain distinguished predicates come to our help. Start with the abnormality predicate relying on the notions of component and abnormality (cf. Definitions 7 and 8):

Definition 10 (Abnormality predicate). *$Ab(.)$, the abnormality predicate, is a unary predicate. If $c \in COMPS$ is a believed system component, then $Ab(c)$ and $\neg Ab(c)$ are well-formed-formulas. Semantically, $Ab(.)$ represents the abnormality of a component; while $\neg Ab(.)$ represents its normality.*

A tool is now available to assert whether a certain believed system component is normal or abnormal. However, it is still impossible to define how a component's normal and abnormal behaviour relates to values of the system parameters. Believed system parameters and their values come to our rescue at this point, relying on the concepts of system and parameter (cf. Definitions 4 and 2). They correspond to the Struss' concepts of local variables and its values in a certain situation [100].

Definition 11 (Believed system parameters). *P , the believed system parameters, is a finite set of constants semantically representing the system parameters. Moreover:*

- *$BP \subseteq P$, represents the boundary system-parameters, and*
- *$IntP \subseteq P$, represents the internal system-parameters.*

Definition 12 (Valuation function). $v(.)$, the valuation function, is a unary function. If $p \in P$ is a believed system parameter, then $v(p)$ is a term. Semantically, the valuation of a believed system parameter represents the value of a parameter.

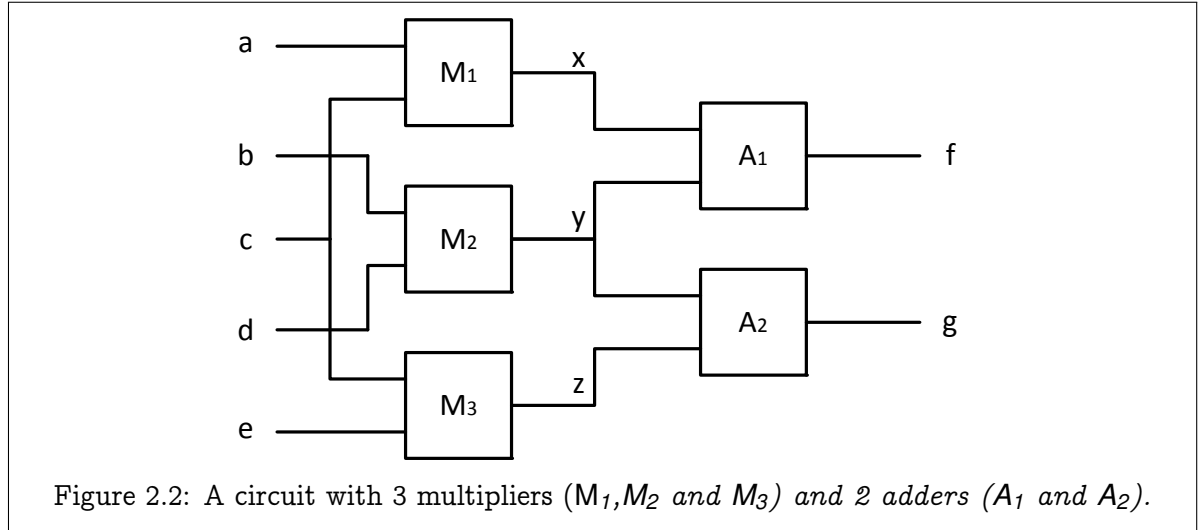
With the help of believed system parameters and their respective values, it is now almost possible to correctly guarantee the description of a component's behaviour in a system. Nevertheless, boundary system-parameters are dynamic entities that may change with the system context. This being so, the [to my best knowledge] original notion of believed system context needs to be introduced in our framework, naturally relying on Definition 5.

Definition 13 (Believed system contexts). CXT , the believed system contexts, is a set of constants, where each element $m \in CXT$ maps a partition $\{IP, OP\}$ of $BP \in P$. Semantically, believed system contexts represent system contexts and IP and OP represent, respectively, the input and output system-parameters for some context.

Definition 14 (Context presence predicate). $CP(.)$, the context presence predicate, is a unary predicate. If $m \in CXT$ is a believed system context, then $CP(m)$ and $\neg CP(m)$ are well-formed-formulas. Semantically, $CP(.)$ represents the presence of a context.

Believed system descriptions, using all the other notions introduced above, provide us with the first of the three Models necessary to reason in Model-based diagnosis and explanation in general: the scientific and mathematical theories. Such notions can be illustrated through a well known example: Davis Polybox [35]; an example which will serve as a basis for some other discussions throughout the present dissertation.

Example 1. Consider the classic circuit of Figure 2.2 introduced by Davis in [35].



If one supposes that the observer-participant building a believed system description can perceive the existence of all the data flows, then the set of parameters chosen by her could be: $P = \{a, b, c, d, e, f, g, x, y, z\}$. Suppose also that the observer participant perceives only

one context to be possible, the one in which a, b, c, d and e are input parameters and f and g are output ones. If that context is represented by the constant m_1 , then $CXT = \{m_1\}$.

With P and CXT defined, one may imagine that the observer-participant represents the circuit by a believed system where $COMPS = \{M_1, M_2, M_3, A_1, A_2\}$; and SD is a theory formed by the following first-order sentences extended with the appropriate axioms for arithmetic and so on.¹¹:

$$\begin{aligned} M_1desc: & \quad CP(m_1) \wedge \neg Ab(M_1) \Rightarrow (v(x) = (v(a) + 1) * v(c)) \\ M_2desc: & \quad CP(m_1) \wedge \neg Ab(M_2) \Rightarrow (v(y) = v(b) * v(d)) \\ M_3desc: & \quad CP(m_1) \wedge \neg Ab(M_3) \Rightarrow (v(z) = v(c) * v(e)) \\ A_1desc: & \quad CP(m_1) \wedge \neg Ab(A_1) \Rightarrow (v(f) = v(x) + v(y)) \\ A_2desc: & \quad CP(m_1) \wedge \neg Ab(A_2) \Rightarrow (v(g) = v(y) + v(z)) \end{aligned}$$



The second Model: believed system observations

The second Model, observations, is viewed in the classical logical framework of Model-based diagnosis in the form of believed system observations, relying on the concepts on system, parameter, context and observation (cf. Definitions 4, 2 and 5 and Principle 4):

Definition 15 (Believed system observations). *The believed system observations, is a set $OBS = OBS_P \cup OBS_{CXT}$ where:*

- OBS_P , the believed system parameter observations, is a finite set of first-order sentences. Sentences are usually on the form $\forall [\neg](v(p)=b_i)$ with each b_i being an observed possible value of $p \in BP$. Semantically, believed system parameter observations represent observations of boundary system-parameter values¹².
- OBS_{CXT} , the believed system context observations, is a finite set of first-order sentences. Semantically, believed system context observations represent observations of some system contexts.

Believed system observations are, by and large, observations about the system context and parameter values. A continuation of Davis Polybox's example illustrated this concept.

Example 1 (continued). *With a believed system defined, one may imagine a situation where an observer-participant observes the present context to be m_1 and the values of parameters a, b, c, d, e, f and g to be, respectively, 1, 2, 3, 4, 5, 11 and 22. In this case, $OBS_{CXT} = \{CP(m_1)\}$ and OBS_P is the theory formed by the first-order sentences below:*

$$\begin{aligned} Obs1: & \quad v(a) = 1 & Obs2: & \quad v(b) = 2 \\ Obs3: & \quad v(c) = 3 & Obs4: & \quad v(d) = 4 \\ Obs5: & \quad v(e) = 5 & Obs6: & \quad v(f) = 11 \\ Obs7: & \quad v(g) = 22 \end{aligned}$$

¹¹Note that the first sentence in SD is ontologically false since it does not truly represent the multiplier in the circuit. This will be explored later on this dissertation.

¹²Note that for a system parameter to be observed it needs to be accessible from outside the system. Thus, internal system-parameters cannot be observed *per se*.

The notion of explanation

In the introductory chapter, I stated that “*explanans* explain some explananda-observations if the latter can be inferred from the former and the context-observations, both at some earlier time, along with some scientific and mathematical theories.”. This notion of explanation is quite attractive for the following reasons:

1. it does not commit on what explanans are among the many known and unknown conditions necessary for an explananda-observation to be registered. By postponing such choice through the non-imposition of a fixed logic through which inferences are drawn, it accommodates, for instance, Hempel’s Deductive-Nomological and Inductive-Statistical models of explanation (cf. Section 2.1).
2. it complements both Hempel’s and Console and Torraso’s accounts of explanation with the imposition of a time asymmetry between explanans and context-observations, and explananda-observations; a condition that enables solving, for instance, the flagpole problem (cf. Section 2.1).

For these arguments, together with the lack of a consensus on what explanation is both in philosophy and Artificial Intelligence, such notion of explanation will be retained hereafter. I encourage the reader, nevertheless, to keep in mind that although it has been postponed, one still has to commit to a logic behind explanation.

With an account of explanation in our hands, it is now possible to map it into the classical logical framework of Model-based diagnosis. First, note that every *explanation problem* is defined through a series of explananda-observations one wants to explain based on some context-observations and scientific and mathematical theories, that is, two of the three Models. Formally speaking:



Definition 16 (Diagnostic problem). *DP*, diagnostic problem, is a tuple $(SD, COMPS, OBS)$.

Note that this concept of diagnostic problem is analogous to the notions of “abduction problem” in [30] and of “system” in [39]. It is at this point that another interesting aspect of explanation comes to play. In fact, among the believed system observations, some must correspond to explananda-observations and some other to context-observations. Moreover not every believed system observation can be an explananda-observation in the context of Model-based diagnosis since, for instance, input system parameters values can never be explained through the description of the system - after all, they are inputs. Despite such fact, both [39] and [30] do not provide a “guideline” for restricting the choice of explananda-observations. Hence, I contribute with an answer to such question.

Definition 17 (Correctly chosen diagnostic problem). Let $DP = (SD, COMPS, OBS)$ be a diagnostic problem, let CXT be the set of believed system contexts and let $OBS_{exp} \subseteq OBS_P \subseteq OBS$ be the subset of the believed system parameter observations corresponding to explananda-observations. *DP* is said to be correctly chosen iff for every parameter p evaluated in a sentence of OBS_{exp} , $p \in OP$ for every context $m \in CXT$.



Intuitively and as envisaged, if a diagnostic problem is correctly chosen then believed system input parameter observations will not be a part of explananda-observations. With a diagnostic problem and a strategy for correctly restraining the choice of explananda-observations in Model-based diagnosis, it is now time to move towards a solution to such problem; a solution relying on the notion of health state:

Definition 18 (Health state). *Let $\Delta \subseteq COMPS$. $\sigma(\Delta, COMPS \setminus \Delta)$, a health state, is the conjunction $[\bigwedge_{c \in \Delta} Ab(c)] \wedge [\bigwedge_{c \in (COMPS \setminus \Delta)} \neg Ab(c)]$.*

Health states are, by and large, statements asserting the normality or abnormality of every believed system component. It is now possible to define what a diagnosis is as an answer to a diagnostic problem. In order to do so, one has to finally commit to a logic behind explanation. Hereafter, and as stated in the beginning of the present chapter, I will be using first-order logic with equality; for it is already defined in literature and has an expressive power necessary to tackle the examples treated in this dissertation. However, my choice shall not be seen as a hard directive toward the choice of a logic. In fact I even encourage the reader to refer to the works of Shoham [97], Shanahan [95] or Thielscher [108] that show us, for instance, that the qualification problem [76] cannot be solved with a classical logic. Since the objective of this dissertation is not that of providing the reader with the best strategy towards reasoning *with* abnormalities in Models of explanation but, instead, it concerns reasoning *about* such abnormalities; I ask the reader to retain that first-order logic with equality was chosen for the sake of simplicity without, however, impacting the results obtained.

With this being said, a diagnosis is defined as follows [30, 39]:


Definition 19 (Diagnosis). *Let $DP = (SD, COMPS, OBS)$ be a diagnostic problem to be solved, and let $\{OBS_{exp}, OBS_{cons}\}$ be a partition of OBS_P in OBS . A diagnosis for DP is a set of health states (called, in this case, diagnosis hypotheses) σ such that:*


 $SD \cup OBS_{CXT} \cup OBS_{cons} \cup \{\sigma\}$
is satisfiable, and

$$SD \cup OBS_{CXT} \cup OBS_{cons} \cup \{\sigma\} \models OBS_{exp}.$$

Note that, when $OBS_{exp} = \emptyset$, $OBS_{cons} = OBS$ and we obtain “consistency-based” diagnoses. Conversely, when $OBS_{exp} \neq \emptyset$, we obtain “abduction-based” diagnoses. Recall such concepts from Section 2.1.

Since representing all the diagnosis hypotheses for a given diagnostic problem can be a daunting task, the notions of partial and kernel diagnosis were introduced [39]:

Definition 20 (Partial diagnosis hypothesis). *A partial diagnosis hypothesis for a diagnostic problem DP is a satisfiable conjunction Q of Ab -literals ($Ab(.)$ or $\neg Ab(.)$) such that for every satisfiable conjunction R of Ab -literals covered by Q , $SD \cup OBS_{CXT} \cup OBS_{cons} \cup \{R\}$ is satisfiable and $SD \cup OBS_{CXT} \cup OBS_{cons} \cup \{R\} \models OBS_{exp}$.* 

Definition 21 (Kernel diagnosis hypothesis). *A kernel diagnosis hypothesis is a partial diagnosis hypothesis with the property that the only partial diagnosis which covers it is itself.* 

Example 1 (continued). *Eight consistency-based kernel diagnosis hypotheses (using definitions 19, 20 and 21) exist in this case:*

$$\begin{array}{ll}
 \{Ab(A_1) \wedge Ab(A_2)\} & \{Ab(A_2) \wedge Ab(M_2)\} \\
 \{Ab(A_2) \wedge Ab(M_1)\} & \{Ab(A_1) \wedge Ab(M_3)\} \\
 \{Ab(A_1) \wedge Ab(M_2)\} & \{Ab(M_2) \wedge Ab(M_3)\} \\
 \{Ab(M_1) \wedge Ab(M_3)\} & \{Ab(M_1) \wedge Ab(M_2)\}
 \end{array}$$

The third Model: diagnostic algorithms

While it is true that Definition 19 provides clear semantics to the notion of diagnosis; since it is given in terms of all models, it has little computational interest. Proof-theory comes to our rescue with a syntactic approach to logic, and hence an attractive computational environment. This is where diagnostic algorithms come to play, relying on Definition 19.

Definition 22 (Diagnostic algorithm). *A diagnostic algorithm, noted A , is an algorithm with an underlying set of inference rules aiming at computing diagnoses.*

Reiter in [90] provides us an example of a diagnostic algorithm explicitly relying on a theorem-prover. Model-theoretic and proof-theoretic diagnoses, are thus, distinguished as follows:

Definition 23 (model-theoretic and proof-theoretic diagnoses).

- *A model-theoretic diagnosis, D_{M-T} , is the set of diagnostic hypotheses respecting Definition 19.*
- *A proof-theoretic diagnosis, D_{P-T} , is the set of diagnostic hypotheses computed by a diagnostic algorithm A (\vdash_A).*

With diagnostic algorithms, the third of the three Models has been introduced in the scope of Model-based diagnosis and it is finally possible to formally define diagnostic systems:

Definition 24 (Diagnostic system). *A diagnostic system is a tuple $(SD, COMPS, OBS, A)$.*

2.4 Defining and relating properties in diagnostic systems and diagnoses

In the introductory chapter, the main question to be answered by this dissertation was raised: how to reason about abnormalities in Models of implemented diagnostic systems. As we moved to the present chapter, it was stated that tackling such problem required a deep understanding of diagnostic systems, which was exactly what has been aspired in the last two sections. Moreover, I promised that some eternal weaknesses of Model-based diagnosis would be revealed through a series of diagnostic systems' and diagnoses' properties whose absence is usually considered abnormal; as well as the relations between them. It is now time to address this subject ¹³.

¹³Most of the work presented in this section was published in [13] and [14].

2.4.1 Defining properties

As stated before, believed system descriptions (scientific and mathematical theories), believed system observations (observations), diagnostic algorithms (rules of inference) and diagnoses (explanans) usually require some properties whose absence is considered abnormal. Hence, some of these properties will now be exposed, being the reader advised for the fact that I do not aim at building an exhaustive list; for this is far from the main objectives of this book.

Diagnoses properties

Two important properties of model-theoretic and proof-theoretic diagnoses (cf. Definition 23) are now presented: *validity* and *certainty*; the notion of true health state:

Definition 25 (True health state). σ_T , the true health state, is a health state with the property that $\{\sigma_T\}$ is an ontologically true theory (cf. Definition 1).

Definition 26 (Validity of a diagnosis). Let σ_T the true health state, let $\#(.)$ be the cardinality function and let $Hamm(a,b)$ be the Hamming distance between a and b [52]. A diagnosis, D , is said to be valid iff $\sigma_T \in D$; and invalid otherwise. Moreover, the degree of validity is:

$$1 - \frac{\min(Hamm(\sigma_T, d_1), \dots, Hamm(\sigma_T, d_n))}{\#(COMPS)}$$

where each $d_i \in D$ is a diagnosis hypothesis.

Definition 27 (Certainty of a diagnosis). Let $\#(.)$ be the cardinality function. A diagnosis, D , is said to be certain iff $\#(D) = 1$; and uncertain otherwise. Furthermore, the degree of uncertainty is:

$$\frac{2^{\#(COMPS)} - \#(D)}{2^{\#(COMPS)} - 1}$$

As indicated by Principle 6 (cf. Subsection 2.2.2) it is impossible to prove that a health state is a true health state and one can only assume it. I will come back to this major point in Chapter 3, but I encourage the reader to retain this fact. As for the validity and certainty of diagnoses, in order to illustrate both concepts suppose a framework with $COMPS = \{c_1, c_2\}$ and a true health state $Ab(c_1) \wedge \neg Ab(c_2)$. Suppose also two diagnoses: $D_1 = \{Ab(c_1) \wedge Ab(c_2)\}$ and $D_2 = \{Ab(c_1) \wedge \neg Ab(c_2), \neg Ab(c_1) \wedge \neg Ab(c_2)\}$. In this case D_1 is a certain but invalid diagnosis; while D_2 is an uncertain but valid one. Validity can be seen as the property guaranteed by “valid models” in [100] over the past observations; or the always implicitly assumed property behind monotonic approaches in Model-based diagnosis. As for certainty, it is the property guaranteed by “diagnosable systems” in [28]. Having valid and certain diagnoses is extremely interesting for most real life diagnostic applications. In the aeronautic industry, for example, invalid diagnoses may lead the aircraft maintenance team to replace the wrong components; and uncertain diagnoses naturally increase the time of repair.

Believed system observations' properties

Let us focus on a single property of believed system observations: *ontological truth*. Informally, ontologically true believed system observations assure a correct perception of reality. On a formal manner, and using the material from Definition 1:

Definition 28 (Ontological truth of believed system observations). *The believed system observations are an ontological truth iff OBS is an ontologically true theory.*

First of all, note that this property formalises a “hole” in observations, as described in Section 1.1: the problem of ontologically false observations; a problem illustrated by Mr. Tortuga’s incorrect perception of a red “change oil” light. This property, together with the ontological truth of believed system descriptions which we will discuss in a few paragraphs, is the most important property of implemented diagnostic systems; for having ontologically true believed system observations is a necessary although insufficient condition for determining valid diagnoses. This will be proved in Subsection 2.4.2. Finally, the reader may notice that Principle 7 (cf. Subsection 2.2.2) determines the impossibility to prove the ontological truth of observations.

Believed system descriptions' properties

From believed system observations' properties, let us move to believed system descriptions' ones: *satisfiability*, *ontological truth*, *completeness*, *coverage* and *diagnosability*.

Definition 29 (Satisfiability of a believed system description). *A believed system description is satisfiable if the theory SD has a model.*

This apparently unflavoured property is extremely interesting. In fact, if a believed system description is not satisfiable it cannot represent an artefact in the real-world, for, by Definition 1, it must have a model. Moreover, courtesy of Gödel’s completeness theorem (see [62] for details), a first-order theory has a model iff it is consistent. Thus, a syntactic check on a first-order theory enables the assessment of this property.

From satisfiability let us move to the ontological truth of believed system description; a property assuring the correct representation of reality.

Definition 30 (Ontological truth of a believed system description). *A believed system description is an ontological truth iff for all ontological true OBS, $SD \cup OBS$ is an ontologically true theory (cf. Definition 1).*

The ontological truth of believed system descriptions formalises the problem of ontologically false scientific and mathematical theories exposed in Section 1.1; a problem illustrated by Mr. Zapato’s prank leading Tortuga to think that clapping his hands twice when inside the car causes the car to start. This property is also in strict connection with the qualification problem as we will see further in this section.

As stated before, together with the ontological truth of believed system observations, such property is one of the most important ones in implemented diagnostic systems. This is as so for the same reasons as before: it is a necessary although insufficient condition for determining valid diagnoses. Before ending this discussion, note that Principle 6 (cf. Subsection 2.2.2) imposes the impossibility to prove the ontological truth of scientific and mathematical theories.

Let me now introduce a third property of interest in believed system descriptions: completeness.

Definition 31 (Completeness of a believed system description). *Let $m \in CXT$ be a believed system context partitioning P into $\{IP, OP\}$. Moreover, let Φ be a set containing first-order sentences of the form $v(ip) = b$ assigning a value for every $ip \in IP$. Finally, let σ be a health state. In this case, a believed system description is said to be complete iff for every Φ in every believed system context m and for every different health state σ :*

*If $SD \cup CP(m) \cup \Phi \cup \{\sigma\}$ is satisfiable,
then $SD \cup CP(m) \cup \Phi \cup \{\sigma\}$ is a complete theory.*

Intuitively, the completeness property is a measure of how many “loose strings” there are in the believed system description. When this property is verified, one is guaranteed to have a representation of how all the believed system output parameters are related to the input ones, for every believed system context, health state and input parameter values. Nevertheless, one is not guaranteed to have an ontologically true representation. Together, the completeness and ontological truth properties formalise another “hole” in scientific and mathematical theories described in Section 1.1: the qualification problem, illustrated by Tortuga’s inability to know and to check all the preconditions necessary for his car to start. Finally, one interesting aspect of this property is that it can, in theory, be measured; for it relies on two mathematical concepts: satisfiability and completeness of mathematical theories; concepts which can be [logically] proved (for instance using Vaught’s test [74]).

Diagnosability is the last property of believed system descriptions to be exposed.

Definition 32 (Diagnosability of a believed system description). *Let $m \in CXT$ be a believed system context and P the set of believed system parameters. Moreover, let Θ be a theory containing first-order sentences of the form $v(p) = b$ assigning a value for every $p \in BP$. A believed system description is said to be diagnosable iff there is a $\Phi \subseteq \Theta$ in every possible believed system context m such that:*

If $SD \cup CP(m) \cup \Phi$ is satisfiable, then D_{M-T} for $(SD, COMPS, \Phi)$ is certain.

Diagnosability is a very important property of believed system descriptions for it guarantees the certainty of the computed diagnoses. Moreover, diagnosability and completeness are two faces of the same coin. While completeness guarantees that, given a believed system context, a health state and every believed system input parameter value, every believed system output parameter value can be determined; diagnosability guarantees that given a believed system context and every believed system input and output parameter values, a certain diagnosis can be determined.

Diagnostic problems' properties

It is now possible to expose two diagnostic problems' properties: *satisfiability* and *observability*.

Definition 33 (Satisfiability of a diagnostic problem). *A diagnostic problem is said to be satisfiable if the theory $SD \cup OBS_{CXT} \cup OBS_P$ has a model.*

Being stronger than the satisfiability of the believed system description, the satisfiability of the diagnostic problem (corresponding to the “remark 1” of [39]) has an added interest attached since it helps measuring the adequacy of the believed system description to the believed system observations. More precisely, if a diagnostic problem is not satisfiable then either the believed system description or the believed system observations (or both) must be ontologically false.

As for the observability of a diagnostic problem, it is a good measure of how much information the diagnostic system has to perform a diagnosis.

Definition 34 (Observability of a diagnostic problem). *A diagnostic problem $DP = (SD, COMPS, OBS)$ is said to be:*

- *weakly observed iff OBS_{CXT} contains one and only one observed believed system context and every believed system input parameter in IP is given one and only one value in OBS_P .*
- *strongly observed iff OBS_{CXT} contains one and only one observed believed system context and every believed system boundary parameter in BP is given one and only one value in OBS_P .*

This concept can easily be illustrated through Davis Polybox and the development done in Example 1. In such example, the diagnostic problem is strongly observed, since every input and output parameter is observed for that single believed system context.

Diagnostic algorithms' properties

Following the introduction of some properties of diagnostic problems, let us focus on those of diagnostic algorithms: *soundness* and *completeness*; formalising two “holes” in rules of inference previously described in Section 1.1: the soundness and completeness problems.

Definition 35 (Soundness and completeness of a diagnostic algorithm). *Let T and φ be, respectively, a theory and a sentence in a logic \mathcal{L} . A diagnostic algorithm A is sound iff:*

$$\text{If } (T \vdash_A \varphi), \text{ then } (T \models_{\mathcal{L}} \varphi)$$

and complete iff:

$$\text{If } (T \models_{\mathcal{L}} \varphi), \text{ then } (T \vdash_A \varphi)$$

Intuitively, a diagnostic algorithm is sound if every truth it determines (or infers) is entailed by the starting premises. Conversely, a complete diagnostic algorithm is one that determines (or infers) all the truths entailed by some starting premises. These properties are crucial for implemented diagnostic systems, for they are the last necessary condition - together with the ontological truth of believed system descriptions and observations - guaranteeing the computation of valid diagnoses. This is why Reiter, for example, explicitly requires a sound and complete theorem prover in his diagnostic algorithm [90].

All in all, in this subsection I contributed with a series of formal properties usually required to implemented diagnostic systems and diagnostic results. Naturally, this is the first step into developing a solution to reason about implemented diagnostic systems for detecting and isolating properties that do not hold in their Models. However, the reader may feel somehow disappointed. After all, the most interesting properties such as the ontological truth of believed system descriptions and observations cannot be proved; and no formal means have been provided to ensure the computation of valid diagnoses. The following subsection focuses on these issues and introduces a series of relations between diagnostic systems' and diagnostic results' properties. These will be used, in the following chapters, to reason about Models.

2.4.2 Relating properties

Implemented diagnostic systems are typically built with one objective in mind: computing valid diagnoses in every situation. If, on one hand, it is hard not to accept this observation, on the other hand and to our knowledge, the necessary conditions to fulfil such objective are missing in the Model-based diagnostic community. Courtesy of the insights provided by Theorem 1, the following theorems address this issue.

Theorem 2. *If SD is an ontologically true believed system description, then for every diagnostic problem $(SD, COMPS, OBS)$ with ontologically true believed system observations:*

1. *every consistency-based model-theoretic diagnosis D_{M-T} is valid.*
2. *some abduction-based model-theoretic diagnoses D_{M-T} may be invalid.*

Proof.

1. Suppose that the consistency-based model-theoretic diagnosis D_{M-T} for the diagnostic problem $(SD, COMPS, OBS)$ is not valid, i.e. $\sigma_T \notin D_{M-T}$. In this case, by Definition 19, $SD \cup OBS \cup \{\sigma_T\}$ is unsatisfiable, thus having no model. From Definition 25, $\{\sigma_T\}$ must have a model \mathcal{M} such that $\exists \mathcal{P}, \mathcal{R} \in \Omega \exists_{\Gamma(\mathcal{M}, \mathcal{P})} (\mathcal{P} \subseteq \mathcal{R}) \wedge (\mathcal{R} \models \Psi)$. Combining all the arguments we get that $\neg(\exists_{\mathcal{M} \in \text{Mod}(SD \cup OBS)} \exists \mathcal{P}, \mathcal{R} \in \Omega \exists_{\Gamma(\mathcal{M}, \mathcal{P})} (\mathcal{P} \subseteq \mathcal{R}) \wedge (\mathcal{R} \models \Psi))$; and either the believed system or the observations are not true.
2. Suppose an ontologically true but incomplete believed system description. Suppose also a believed system output parameter op (for every context) such that, for a given health state σ , for every sentence φ evaluating op and for every observations OBS : $SD \cup OBS \cup \{\sigma\} \not\models \{\varphi\}$ and $SD \cup OBS \cup \{\sigma\} \not\models \{\neg\varphi\}$. In this case, if for a given diagnostic

problem with ontologically true observations and system description, the true health state $\sigma_T = \sigma$ and op is evaluated in a sentence φ of OBS_{exp} , then D_{M-T} is not valid.

■

According to Theorem 2, there can be invalid abduction-based model-theoretic diagnoses resulting from diagnostic problems with ontologically true believed system descriptions and believed system observations. However, we will see in a few paragraphs that some conditions exist to guarantee its validity, namely: the completeness of the believed system description, the observability of the diagnostic problem, and the classical properties of the underlying logic.

Theorem 3. *If A is a sound and complete diagnostic algorithm, then $D_{M-T} = D_{P-T}$ for a given diagnostic problem $(SD, \text{COMPS}, \text{OBS})$.*

Proof.

From Definition 19, σ is a model-theoretic diagnosis hypothesis iff $SD \cup \text{OBS}_{\text{cons}} \cup \{\sigma\}$ is satisfiable and $SD \cup \text{OBS}_{\text{cons}} \cup \{\sigma\} \models \text{OBS}_{\text{exp}}$.

One can make the bridge between model and proof-theory shine by stating that an anti-diagnostic hypothesis is σ such that $SD \cup \text{OBS}_{\text{cons}} \models \{\neg\sigma\}$ or $SD \cup \text{OBS}_{\text{cons}} \cup \{\neg \bigwedge_{o \in \text{OBS}_{\text{exp}}} o\} \models \{\neg\sigma\}$; and stating that every health state σ that is not an anti-diagnostic hypothesis is a diagnostic hypothesis.

Since A is sound and complete, model-theoretic and proof-theoretic anti-diagnostic hypotheses are the same; and since anti-diagnosis and diagnosis are complementary, model-theoretic and proof-theoretic diagnoses are equivalent for a sound and complete diagnostic algorithm.

■

Corollary 4. *If A is a sound and complete diagnostic algorithm and (SD, COMPS) is an ontologically true believed system, then for every diagnostic problem $(SD, \text{COMPS}, \text{OBS})$ with ontologically true observations:*

1. *every consistency-based proof-theoretic diagnosis, D_{P-T} , is valid.*
2. *some abduction-based proof-theoretic diagnoses, D_{P-T} , may be invalid.*

Proof.

Trivial from Theorems 2 and 3.

■

With these three theorems at our hands, some interesting conclusions can be drawn. First of all, inheriting from the conclusions provided in the analysis of Theorem 1 in Subsection 2.2.2, valid consistency-based diagnoses are proved to be a product of correctly implemented diagnostic systems with sound and complete inference rules and ontologically true scientific and mathematical theories and observations. Moreover, as predicted in that same Subsection, from invalid consistency-based diagnoses one can infer the ontological falsehood of believed system descriptions or observations, or the unsoundness or incompleteness of diagnostic algorithms. These results are the actual rational behind Chapter 3. Finally, these three theorems also

tell us that while consistency-based diagnoses can be guaranteed to be valid in some situations, abduction-based diagnoses do not. At this point, it is interesting to prove that, in some cases, these abduction-based diagnoses can retrieve the advantages of their consistency-based counterparts:

Theorem 5. *Let \mathcal{L} be a classical logic. Moreover, let SD be a complete believed system description in \mathcal{L} and let DP be a correctly chosen diagnostic problem based on SD and \mathcal{L} , at least weakly observed. In this case for every health state σ :*

$$SD \cup OBS \cup \{\sigma\} \text{ is satisfiable} \Leftrightarrow \begin{cases} SD \cup OBS_{cons} \cup \{\sigma\} \text{ is satisfiable and} \\ SD \cup OBS_{cons} \cup \{\sigma\} \models_{\mathcal{L}} OBS_{exp} \end{cases}$$

Proof.

- \Leftarrow : If $SD \cup OBS_{cons} \cup \{\sigma\}$ is satisfiable, then it has a model. Since $SD \cup OBS_{cons} \cup \{\sigma\} \models OBS_{exp}$ then every model of $SD \cup OBS_{cons} \cup \{\sigma\}$ is a model of OBS_{exp} (for \mathcal{L} is classic). This being the case, since $OBS_{cons} \cup OBS_{exp} = OBS$, then $SD \cup OBS \cup \{\sigma\}$ has a model, thus being satisfiable.
- \Rightarrow : Since $OBS_{cons} \cup OBS_{exp} = OBS$, $SD \cup OBS_{cons} \cup OBS_{exp} \cup \{\sigma\}$ is satisfiable so one only needs to prove that $SD \cup OBS_{cons} \cup \{\sigma\} \models OBS_{exp}$. Let $\varphi = \bigwedge_{o \in OBS_{exp}} o$. Since SD is a complete believed system description and DP is at least weakly observed, $SD \cup OBS_{cons} \cup \{\sigma\}$ is a complete theory and, this being so, either $SD \cup OBS_{cons} \cup \{\sigma\} \models \{\varphi\}$ or $SD \cup OBS_{cons} \cup \{\sigma\} \models \{\neg\varphi\}$ (for \mathcal{L} is classic). Suppose $SD \cup OBS_{cons} \cup \{\sigma\} \models \{\neg\varphi\}$, which is the same as saying that $SD \cup OBS_{cons} \cup \{\sigma\} \cup \{\varphi\}$ is unsatisfiable. This, however, cannot be the case because $SD \cup OBS_{cons} \cup \{\sigma\} \cup \{\varphi\}$ is satisfiable. Thus, $SD \cup OBS_{cons} \cup \{\sigma\} \models \{\varphi\}$.

■

The presented theorem exposes a bridge between abduction and consistency-based diagnoses provided by the completeness of believed system descriptions, the observability of diagnostic problems and the properties of classical logic. In a nutshell, this theorem tells us that sometimes abduction becomes consistency and vice-versa. The importance of this relation is two-fold. First, consistency-based tools can be used to compute abduction-based explanations in some situations; and vice-versa. Secondly, abduction-based explanation can sometimes inherit from the advantages of the consistency-based one:

Corollary 6. *Let \mathcal{L} be a classical logic. If SD is an ontologically true and complete believed system description in \mathcal{L} and DP is an at least weakly-observed correctly chosen diagnostic problem based on SD and \mathcal{L} , then every model-theoretic consistency-based or abduction-based diagnosis is valid.*

Proof.

Immediate by Theorems 5 and 2.

■

One now has the necessary tools to relate Models' properties with the validity of diagnoses. Nevertheless, a connection is still missing between the properties of Models and the certainty of computed diagnoses:

Theorem 7. *Let SD be a diagnosable believed system description, $(SD, COMPS, OBS)$ be a fully observed correctly-chosen satisfiable diagnostic problem and A be a sound and complete diagnostic algorithm. In this case, all the proof-theoretic diagnoses $D_{P,T}$ are certain.*

Proof.

Immediate from Definitions 32, 33 and 34 and Theorem 3. ■

With all the theorems and corollaries that have been presented in this subsection, as well as with their combinations, one can now relate all the properties introduced in Subsection 2.4.1 and formally understand why some abnormal properties ("holes") of Models, i.e. scientific and mathematical theories, observations and rules of inference, can be so harmful for explanation.

2.5 Originality and Related work

I started this chapter by reaffirming the main academic objective of this dissertation: to provide a tool for reasoning about abnormalities in Models of implemented diagnostic systems and explanation in general. Moreover, I stated that the first step in the path to solving our problem was 1) to build a solid foundation of diagnostic systems themselves, 2) characterise Model-based diagnosis based on such foundations and 3) expose and relate some properties of diagnostic systems and diagnostic results. Thus, my contribution in this chapter appears naturally related with other works focused on these three axes.

Concerning the foundation of diagnostic systems, exposed in Section 2.2, it is a thesis on how scientific and mathematical theories and observations are formed and evolve in observer-participants' minds through information. Deep inside the realm of Philosophy of Science, this thesis is, as far as my knowledge goes, an original contribution drinking from and formalising some concepts informally introduced by Wheeler [114, 113, 115, 116, 117]. First, it uses many of Wheeler's ideas such as the "super-Copernican", the "no question, no answer" or the "No super-turtles! No tower of turtles" principles. Secondly, it formalises information as a model-theoretical notion in a way that fits both scientific realism and empiricism (among other anti-realism positions), and is consistent with the "no question, no answer" principle. When the formal concept of information is then used to explore scientific and mathematical theories, observations and their respective ontological truth, many more ideas, related for instance to the semantic view of theories, become in line with our contribution. The works of Suppes [104, 105], da Costa and French [34], Suppe [102, 103] or van Fraassen [111] present just some of these ideas, where, to my knowledge, the closest we can go to this work is that: a theory is ontologically true if one of its models is [partially] isomorphic to the real-world. However, since the real-world is not even a mathematical structure, these accounts are basically

mathematically flawed and this is where our notion of information comes to our rescue as a necessary condition both for correctly defining the ontological truth of a theory and for proving the original and important Theorem 1.

From the foundation of diagnostic systems, Section 2.3 advanced to a characterisation of Model-based diagnosis nourished by such thesis. As mentioned in that section, most of the exposed concepts relied on the works of Chittaro, Console, de Kleer, Guida, Mackworth, Reiter, Struss, Tasso, Toppano, Torasso and Williams [25, 90, 40, 30, 39, 100]. Nevertheless, my contribution lied behind the different glasses through which Model-based diagnosis was presented. First, the historical view of Models and explanation exposed in Section 2.1 set the stage for a deeper critic on the notion of explanation lying below the classical framework of Model-based diagnosis. Secondly, backed up by the seven principles of Section 2.2, I contributed with an in-depth discussion on the three Models of explanation and how they appear in Model-based diagnosis. This was backed up by a distinction between the real and the cognitive worlds using concepts such as those of system, component, abnormality, function, input, output or replaceability; a distinction which, in my opinion, is a necessary condition for a proper discussion on Model-based diagnosis. Finally, the notions of context of a system and correctly chosen diagnostic problem are also, to my best knowledge, original in the literature.

As for the third axis of the present chapter, it is to my best knowledge an original contribution (backed up by the publication of [13]), providing: 1) a formalisation of many properties in scientific and mathematical theories, observations and rules of inference affecting explanations, and 2) a set of theorems proving how these properties relate with explanans' properties. Besides the works naturally related to this contribution either through the problems mentioned in Section 1.1 or through the two previously discussed axes, I would like to point the works of Struss [100], and Nayak and Levy [79] that relate to this one through the study of property relations between different model abstractions; and the unpublished vision of Mathen, clearly addressing (although in an informal manner) the completeness and ontological truth of scientific and mathematical theories [75].

Chapter 3

Meta-diagnosis

It is difficult to say what truth is, but sometimes it is so easy to recognize a falsehood.

- Albert Einstein

In the introductory chapter many problems of scientific and mathematical theories, observations and rules of inference were exposed. Moreover, it was said that Model-based diagnosis (and explanation in general) needs: 1) a framework for reasoning with abnormal Models using a correct notion of explanation, and 2) a framework for reasoning about abnormal Models for detecting and isolating Model abnormalities. The first point was defined as the main objective of this dissertation. In order to provide a prerequisite enabling the achievement of such goal, the second chapter focused on establishing both a formal and clear definition of a well-founded framework of Model-based diagnosis, and a formal account of abnormalities and of their relations with diagnoses and explanans in general. It is now time to focus on reasoning about abnormalities in the Models of implemented diagnostic systems and explanation in general. In this chapter, I contribute both with a theory and a tool of meta-diagnosis tackling this issue from the academic side. As for the industrial one, it will be the object of the following chapter.

In the first section, Artificial Intelligence is provided with a general logical theory of meta-diagnosis; a theory which enables the detection and isolation of abnormal properties in the Models of implemented diagnostic systems (and explanation in general). This theory has many advantages: first, it enjoys the clear semantics provided by logic; second, it is a general unified theory to reason about any implemented Model-based diagnostic system; and third it can be mapped into a theory of diagnosis, thus inheriting from the arsenal of tools already developed through the years in the diagnostic-world.

With a theory of meta-diagnosis in our swiss-knife, the second section exposes a process for modelling meta-diagnostic problems. Furthermore, it presents two toy-problems, thus offering the reader a flavour of what can be done with the meta-diagnostic framework.

The theoretical view of meta-diagnosis will be complemented with a practical one in the third section; a section devoted to providing Artificial Intelligence with a detailed architectural and functional description of a logical-based meta-diagnosis tool: MEDITO. Exploiting the syntactic equivalence between diagnosis and meta-diagnosis, MEDITO uses diagnostic algorithms from field of Model-based diagnosis. Doing so, it presents empirical proof supporting the meta-diagnosis claim: any diagnostic algorithm can tackle a meta-diagnostic problem.

As for the final section, it will focus on exploring the works in the literature related to ours, and to clearly assessing the originality of our contribution.

3.1 Towards a framework of meta-diagnosis

At the beginning of the introductory chapter, I stated that this dissertation is focused on explanation in the context of why-questions, such as: “why does Tortuga’s car not start?”. Moreover, it was argued that explanation is based on Models, i.e. scientific and mathematical theories, observations and rules of inference; and a series of Model’s and diagnoses’ properties and relations were exposed.

With these tools in our hands, extended questions, such as: “why are the explanans of the “why does Tortuga’s car not start” problem invalid?”; arise in a natural manner. After all, we typically may want to detect, isolate and repair the abnormalities of scientific and mathematical theories, observations and rules of inference, causing abnormalities in the computed explanans. In a nutshell, we can now think of why-questions about why-questions, that is, of meta-explanations or meta-diagnoses ¹. The logical characterisation of these meta-diagnoses is the object of the present section.

3.1.1 On meta-systems, meta-components and other concepts

The formal characterisation of meta-diagnoses starts with the definition of meta-diagnosis as a cognitive-world reasoning process which aims at determining the normal and abnormal meta-components of a meta-system under study. First, the reader is welcome to see the straight connection between this primary account of meta-diagnosis and that of diagnosis in Section 2.3. Secondly, in the same way as the characterisation of diagnoses relied on a clear definition of components and systems, the path into a correct characterisation of meta-diagnoses starts with the notions of meta-components and meta-systems. Let me, as so, begin by addressing the later concept which depends on *meta-parameters* and *meta-goals*.

Definition 36 (Meta-parameter). *A meta-parameter represents a sensed part of reality perceived as a flow of physics to or from an implemented diagnostic system.*

Definition 37 (Meta-Goal). *A meta-goal is a [possibly empty] set of meta-parameters with an admissible value range, and a [possible empty] set of mathematical relations that must hold between them.*

¹Meta-explanation will actually be generalised to cover more than why-questions about why-questions. This, however, will be a subject of Chapter 5; a subject which will be put on standby for now.

Definition 38 (Meta-System). *A meta-system represents a sensed part of reality perceived as a part of physics implementing a diagnostic system and assigned with some meta-goals.*

Note that meta-systems of special importance to us for 1) they are particular systems (cf. Definition 4), 2) they correspond to implemented diagnostic systems which are our object of study, and 3) diagnostic system's and diagnoses' properties (some of the possible meta-parameters in meta-goals) and property relations can be proved. I encourage the reader to retain this remark. As for meta-components, they can be defined as follows:

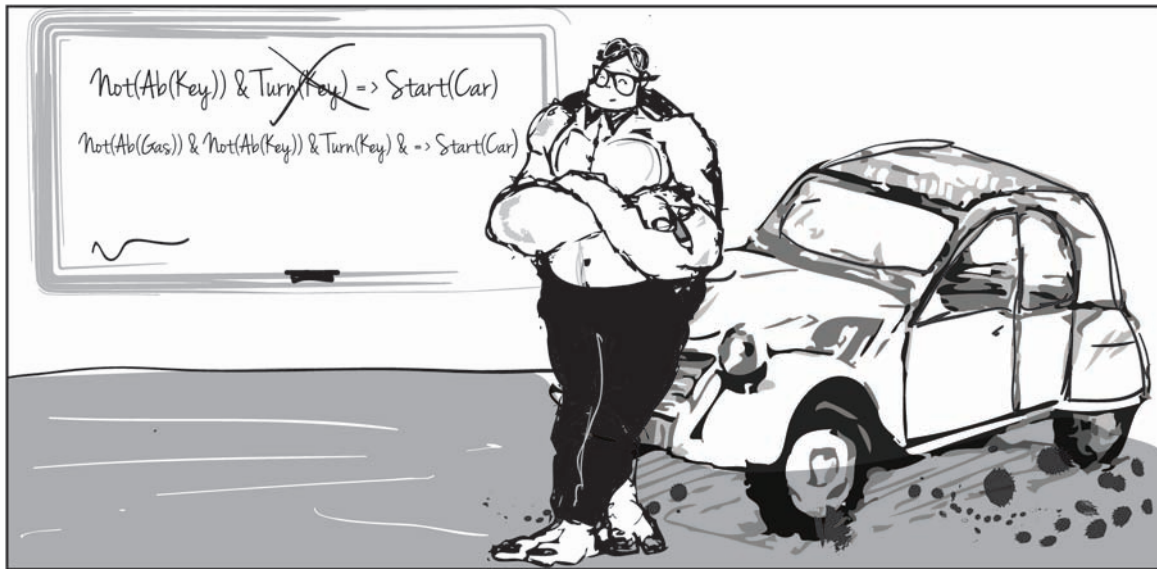
Definition 39 (Meta-component). *A meta-component represents a sensed part of reality perceived (cf. Principle 4) as a replaceable (cf. Definition 6) part of an implemented diagnostic system (cf. Definition 24).*

Naturally, meta-components can be defined at different abstraction levels. Moreover, it is not mandatory for every part (or element) of an implemented diagnostic system to be considered as a meta-component. From this concept, we can go one step further and explore the notion of meta-abnormality.

Definition 40 (Meta-Abnormality). *When assigned with a meta-goal, a meta-component is said to be: normal, if the meta-goal is met; and abnormal otherwise. A meta-goal is met whenever all its underlying implemented diagnostic system's and diagnoses's property values are within range and all its underlying relations hold.*

With the concepts of meta-system, meta-component and meta-abnormality apprehended, the meaning of the introductory sentence: “meta-diagnosis [is] a cognitive-world reasoning process which aims at determining the normal and abnormal meta-components of a meta-system under study”; should now be crystal clear. As for how normal and abnormal meta-components are determined, it is the object of the remainder of this Section.

For now, let me illustrate the notions exposed in the present subsection with the help of Mr. Tortuga and his “car start” situation of Section 1.1. Recall that Tortuga had developed a thesis on how the car starts: “Turning a normal key implies the car start”; a sentence that can be expressed, for example, by the following first-order theory fitting the framework of diagnosis presented in Chapter 2: $\{\neg \text{Ab}(\text{Key}) \wedge \text{Turn}(\text{Key}) \supset \text{Start}(\text{Car})\}$, where *Key* and *Car* are constants (the first one representing a component) and *Turn(.)* and *Start(.)* are unary relations. Moreover, recall that Tortuga once turned the key and his car did not start, i.e. he observed $\text{Turn}(\text{Key}) \wedge \neg \text{Start}(\text{Car})$. Finally, remember that Tortuga first explained (or diagnosed) the fact that the car did not start with an abnormality with the key, only to be proved wrong by Mr. Zapato's conclusion that the car was missing gasoline. In this case, one can view the meta-system representing a sensed part of reality perceived as Tortuga's implemented diagnostic system with the meta-goal of producing valid diagnoses. Moreover, the $\neg \text{Ab}(\text{Key}) \wedge \text{Turn}(\text{Key}) \supset \text{Start}(\text{Car})$ sentence can correspond to a meta-component whose abnormality Tortuga wants to meta-diagnose. Indeed, Tortuga found that this meta-component was abnormal and “replaced” it with the sentence $\neg \text{Ab}(\text{Key}) \wedge \text{Turn}(\text{Key}) \wedge \neg \text{No}(\text{Gasoline}) \supset \text{Start}(\text{Car})$. As stated before, the remainder of this section exposes Tortuga's reasoning process to detect and isolate meta-component abnormalities.



3.1.2 Characterising meta-diagnoses

The notions of meta-system, meta-component and meta-abnormality are the building blocks of meta-diagnoses. With them, it is now possible to introduce a logical framework of Model-based meta-diagnosis. Naturally, as was the case with Model-based diagnosis in Chapter 2, this framework relies on the three Models, i.e. scientific and mathematical theories, observations and rules of inference, and on the concept of explanation.

The first Model: believed meta-systems

Start with the first Model appearing on the form of believed meta-systems, relying on the concepts of meta-system, meta-component, meta-abnormality and perception (cf. Definitions 38, 39 and 40, and Principle 4):

Definition 41 (Believed meta-system). *A believed meta-system is a pair $(M\text{-}SD, M\text{-}COMPS)$ where:*

- *$M\text{-}SD$, the believed meta-system description, is a set of first-order sentences. Semantically, it is a scientific and mathematical theory describing a meta-system and possibly some systems.*
- *$M\text{-}COMPS$, the believed meta-system meta-components, is a finite set of constants. Semantically, each element mc of $M\text{-}COMPS$ represents a meta-component whose abnormality one wants to meta-diagnose.*

To describe the behaviour of believed meta-system meta-components, as well as their interactions, believed meta-system descriptions typically use the meta-abnormality predicate:

Definition 42 (Meta-abnormality predicate). *M-Ab(.)*, the meta-abnormality predicate, is a unary predicate. If $mc \in M\text{-COMPS}$ is a believed meta-system meta-component, then $M\text{-Ab}(mc)$ and $\neg M\text{-Ab}(mc)$ are well-formed-formulas. Semantically, $M\text{-Ab}(.)$ represents the abnormality of a meta-component; while $\neg M\text{-Ab}(.)$ represents its normality.

Such concepts can be illustrated through the Polybox example (cf. Chapter 2):

Example 1 (continued). Suppose that the meta-goal of computing valid diagnoses is assigned to the correctly implemented diagnostic system $(SD, COMPS, OBS, A)$. Moreover, for the sake of the argument assume that OBS is an ontologically true theory and A is a sound and complete diagnostic algorithm. Finally, if each SD -sentence is considered as a meta-component with the meta-goal of being ontologically true, one possibility for $M\text{-SD}$ with $M\text{-COMPS} = \{M_1\text{desc}, M_2\text{desc}, M_3\text{desc}, A_1\text{desc}, A_2\text{desc}\}$ would then be ²:

$$\begin{aligned} \neg M\text{-Ab}(M_1\text{desc}) &\Rightarrow [CP(m_1) \wedge \neg Ab(M_1) \Rightarrow (v(x) = (v(a) + 1) * v(c))] \\ \neg M\text{-Ab}(M_2\text{desc}) &\Rightarrow [CP(m_1) \wedge \neg Ab(M_2) \Rightarrow (v(y) = v(b) * v(d))] \\ \neg M\text{-Ab}(M_3\text{desc}) &\Rightarrow [CP(m_1) \wedge \neg Ab(M_3) \Rightarrow (v(z) = v(c) * v(e))] \\ \neg M\text{-Ab}(A_1\text{desc}) &\Rightarrow [CP(m_1) \wedge \neg Ab(A_1) \Rightarrow (v(f) = v(x) + v(y))] \\ \neg M\text{-Ab}(A_2\text{desc}) &\Rightarrow [CP(m_1) \wedge \neg Ab(A_2) \Rightarrow (v(g) = v(y) + v(z))] \end{aligned}$$

The idea behind such $M\text{-SD}$ is that if a meta-component is not abnormal, then the SD -sentence it represents is ontologically true.

The second Model: meta-observations

From believed meta-systems, one can advance to the notion of meta-observations; used along these to determine the normal and abnormal meta-components. Such notion relies on Principle 4 and on Definitions 4 and 38.

Definition 43 (Meta-observations). The set of meta-observations, $M\text{-OBS}$, is a finite set of first-order sentences. Semantically, meta-observations are observations about a meta-system and possibly some observations the systems underlying such meta-system.

Such concept can be illustrated through a continuation of the Polybox example:

Example 1 (continued). Suppose the true health state is meta-observed, in this example, to be $\neg Ab(M_1) \wedge \neg Ab(M_2) \wedge \neg Ab(M_3) \wedge \neg Ab(A_1) \wedge Ab(A_2)$. In this case, $M\text{-OBS}$ can be:

$$\begin{aligned} &v(a) = 1 \quad v(b) = 2 \quad v(c) = 3 \quad v(d) = 4 \\ &v(e) = 5 \quad v(f) = 11 \quad v(g) = 22 \quad CP(m_1) \\ &\neg Ab(M_1) \wedge \neg Ab(M_2) \wedge \neg Ab(M_3) \wedge \neg Ab(A_1) \wedge Ab(A_2) \end{aligned}$$

²Section 3.2 clarifies the rationale for the meta-system description presented.

The notion of explanation

With a believed meta-system and some meta-observations in our possession, we can advance to the notion of meta-diagnostic problem.

Definition 44 (Meta-diagnostic problem). *A meta-diagnostic problem $M-DP$ is a tuple $(M-SD, M-COMPS, M-OBS)$.*

The solutions of meta-diagnostic problems are meta-diagnoses. These, in turn, depend on the concept of meta-health state.

Definition 45 (Meta-health state). *Let $\Phi \subseteq M-COMPS$. A meta-health state, $\pi(\Phi, M-COMPS \setminus \Phi)$, is the conjunction $[\bigwedge_{mc \in \Phi} M-Ab(mc)] \wedge [\bigwedge_{mc \in (M-COMPS \setminus \Phi)} \neg M-Ab(mc)]$*

Stop for a brief moment and recall some important concepts. In the beginning of the present section it was stated that meta-diagnosis or meta-explanation can be seen as a reasoning process aiming and providing answers to why-questions about why-questions. In a nutshell, meta-diagnosis depends, in its essence, on the idea of explanation. This being so, let me use the idea of explanation provided in the previous chapter, that is, “*explanans* explain some explananda-observations if the latter can be inferred from the former and the context-observations, both at some earlier time, along with some scientific and mathematical theories.” Moreover, and since one has to commit to a logic behind explanation, I will be working with first-order logic with equality. Unsurprisingly, other logics can be chosen, but let me strongly assert that, in some situations (exactly those that are of interest to us), classical logics can be used without harm. This fact will be explored in a few paragraphs. For now, let meta-diagnosis be introduced:

Definition 46 (Meta-diagnosis). *Let $M-DP = (M-SD, M-COMPS, M-OBS)$ be a meta-diagnostic problem to be solved; and $\{M-OBS_{exp}, M-OBS_{cons}\}$ be a partition of $M-OBS$. A meta-diagnosis for $M-DP$ is a set of meta-health states (called, in this context, meta-diagnosis hypotheses) π such that:*

$$M-SD \cup M-OBS_{cons} \cup \{\pi\}$$

is satisfiable, and

$$M-SD \cup M-OBS_{cons} \cup \{\pi\} \models M-OBS_{exp}.$$

Meta-diagnoses are called “consistency-based” when $M-OBS_{exp} = \emptyset$; and are called “abduction-based” otherwise.

3.1.3 Some words on meta-diagnoses

The notion of meta-diagnosis being introduced, it is now time to settle down and explore the implications of such definition.

Syntactic equivalence

The first point to emphasize is the syntactic equivalence between Definitions 19 and 46, i.e. between diagnosis and meta-diagnosis. A series of conclusions follow directly from this fact:

- The notions of partial and kernel meta-diagnosis hypothesis can be directly drawn from Definitions 20 and 21.
- Every diagnostic algorithm can become a meta-diagnostic algorithm if it is sound and complete with respect to the underlying semantics (cf. Definition 35). As so, one does not need to start from scratch into the development of meta-diagnostic algorithms and, for example, the works of [122] or [121] can be used in meta-diagnosis.
- Every approach to handle the complexity problems of diagnosis can be used in meta-diagnosis. Thus, as in the previous point, meta-diagnosis can benefit from the many years of experience gathered in the diagnostic world concerning complexity management.

These conclusions are, in a nutshell, the rational behind this dissertation's lack of a detailed account on meta-diagnostic algorithms and complexity management techniques. Straightforwardly, the third Model behind meta-diagnosis is a gift from diagnosis.

Semantic difference

The discussion on the semantic difference between diagnoses and meta-diagnoses starts with some questions: If why-questions about why-questions are justified, can we also think of why-questions about why-questions about why-questions? If yes, then when should we stop? At meta-meta-diagnosis level or even further?

The answer to all these questions lies behind the semantic difference between diagnosis and meta-diagnosis. Recall that, in the introductory chapter, it was stated that Model-based diagnosis (or explanation in general) is a reasoning process relying on scientific and mathematical theories, on observations and on some rules of inference. Moreover, it was shown that although rules of inference can be proved sound and complete with regards to a certain logic, scientific theories and observations can never be proved ontologically true; for they are forged in reality's moulds. Thus, the fact that the artefact to be diagnosed can be a part of the real-world justifies a meta-diagnosis.

Contrary to Model-based diagnosis, in Model-based meta-diagnosis the artefact to be diagnosed is always perceived as an implemented diagnostic system, a well defined cognitive-world concept. As so, if one hypothesises that meta-systems are based on correct interpretations of well-implemented diagnostic systems, the notion of ontological truth becomes meaningless with regards to interesting believed meta-system descriptions³. This, in turn, justifies the absence of a meta-meta-diagnosis on believed meta-system description abnormalities.

³Although it is true that the general syntax of believed meta-system description allows the inclusion of sentences about any cognitive-world concept representing a view on the real-world; it is here defended that there is a whole series of believed meta-systems that can be built on cognitive-world concepts representing other cognitive-world concepts: the diagnostic system's and diagnosis' properties to be meta-diagnoses. Under the hypothesis that meta-systems are based on correct interpretations or that diagnostic systems are correctly implemented in the real-world, these are precisely the believed meta-system descriptions that interest us.

As for meta-diagnostic algorithms, the results from the previous subsection tell us that they coincide with diagnostic algorithms. Since the rules of inference behind the theorem provers underlying every diagnostic algorithm can be proved sound and complete with respect to a given logic, the meta-meta-diagnosis of meta-diagnostic algorithms' abnormalities can also be avoided.

Concerning meta-observations, these can be separated in two different groups: 1) the ones that coincide with the believed system observations, and 2) the others, called exclusive meta-observations. Concerning the former group, their possible ontological falsehood is taken into account in the meta-diagnosis itself. Thus, no meta-meta-diagnosis is needed for its treatment if one supposes that meta-systems are based on correct interpretations or that diagnostic systems are correctly implemented. Exclusive meta-observations, on the other hand, are the Achilles heel of meta-diagnosis. In fact, while it is true that one can sometimes avoid having observations about systems other than meta-systems in this group; interesting cases, such as the inclusion of diagnoses' validity as a meta-goal, need these observations to be a part of meta-observations; meta-observations that may be, in turn, ontologically false. In a nutshell, the second major hypothesis behind meta-diagnosis is the ontological truth of this group of meta-observations.

With this last conclusion, and before the reader feels that the basis of this dissertation is falling apart, let me affirm that the hypothesis of ontological true exclusive meta-observations:

- is much weaker than the hypothesis of ontologically true believed system descriptions and ontologically true believed system observations on which Model-based diagnosis relies; for it does not carry the problem of induction (cf. Section 2.1.1).
- is perfectly justified in real-world applications such as in implemented Airbus' diagnostic systems, where one is interested in the validity of diagnosis with respect to a given benchmark set by component's suppliers.

Finally, note that the notion of explanation used also affects the need for a meta-meta-diagnosis in two different ways. First, the underlying logic must be monotonic or else a preference criteria on models may exclude potentially wanted solutions. Fortunately, non-monotonic logics are justified when scientific and mathematical theories and observations are not guaranteed to be ontologically true; in which case one needs to "jump to conclusions" and retract them, if necessary, when new observations are gathered. However, courtesy of the meta-equivalent of Corollary 4, the need for no such retraction is guaranteed in the case of ontologically true believed meta-system descriptions and meta-observations, and sound and complete meta-diagnostic algorithms. Since this is exactly the case in meta-diagnosis if one takes the previous discussions into account, then a monotonic logic can be used. Second, and according to the meta-equivalents of Corollary 4 and Theorem 5, meta-diagnoses must either be computed using a consistency-based approach or the believed meta-system description must be complete and the meta-diagnostic problem must be correctly chosen and at least weakly observed. Since the latter condition cannot be assured in general, the further discussions on meta-diagnosis will use a consistency-based approach.

All in all, meta-meta-diagnosis can be avoided if one is sure that the logic behind explanation is monotonic and that no abnormalities affect the Models of the meta-diagnostic

system. This, in turn corresponds to two hypotheses: 1) meta-systems are based on correct interpretations of well-implemented diagnostic systems, and 2) exclusive meta-observations are ontologically true. In this case, meta-diagnosis can be used on its own to detect and isolate abnormalities in Models of implemented diagnostic systems.

3.2 Modelling and solving meta-diagnostic problems

Up until this point, the reader has been provided with a characterisation of meta-diagnosis that, together with the diagnostic systems' and diagnoses' properties and relations provided in Section 2.4, guarantees the detection and isolation of abnormalities in Models of implemented diagnostic systems. In a nutshell, it is now possible to understand how Mr Tortuga determined that the $\neg \text{Ab}(\text{Key}) \wedge \text{Turn}(\text{Key}) \supset \text{Start}(\text{Car})$ sentence was ontologically false. Nevertheless, a discussion on how to model and solve typical practical meta-diagnostic problems is important not only to simplify the approach to these problems, but also to provide the reader with a flavour of what can be done with such framework. This is the object of the present section.

As for the modelling process, although a series of other strategies can be applied, the following is adopted:

1. Identify the meta-system, i.e. the implemented diagnostic system to be meta-diagnosed and its assigned meta-goals.
2. Identify the meta-components and their meta-goals based on the meta-system meta-goals, on diagnostic system's and diagnoses' properties and on their relations, and on the detail level wanted for the solution.
3. Describe meta-components and meta-components' normal/abnormal behaviour on believed meta-systems.
4. Identify meta-observables based on the believed meta-system description.

Concerning the solving process, this section relies on the General Diagnostic Engine (GDE) [40][47]. This was possible because, as discussed in the previous section, every sound and complete diagnostic algorithm can become a meta-diagnostic algorithm (if it is implemented without errors); which is the case of GDE.

3.2.1 The problem of ontologically false believed system description

Imagine we are given the following problem, typically in the V&V process of a believed system description: "In a given meta-system correctly perceived as a well-implemented diagnostic system $S = (\text{SD}, \text{COMPS}, \text{OBS}, \mathcal{A})$, assume OBS are ontologically true believed system observations and \mathcal{A} is a sound and complete diagnostic algorithm. Knowing that all the computed consistency-based diagnoses must be valid, meta-diagnose S using a consistency-based approach.". Suppose an instance of this problem where SD, COMPS and OBS are those of Example 1.

Let us start by following the modelling process:

1. The meta-system is made of:
 - a sensed part of reality correctly perceived as a well-implemented diagnostic system $S = (SD, COMPS, OBS, A)$; where SD , $COMPS$ and OBS are those of Example 1.
 - the meta-goal of computing valid proof-theoretic consistency-based diagnoses.
2. Using Corollary 4 and the problem hypotheses, the meta-components can be chosen to represent every implemented SD -sentence, whose associated meta-goal is their ontological truth.
3. Suppose a sentence A represented by a believed meta-system meta-component mc_1 . Now, either A is ontologically true or mc_1 is abnormal, i.e. $M\text{-}Ab(mc_1) \vee A$. So, the normal behaviour of this sentence would be described, at a believed meta-system level, as: $\neg M\text{-}Ab(mc_1) \Rightarrow A$.
4. From Corollary 4 and the problem hypotheses we get that if every believed meta-component is normal then the $D_{P,T}$ is valid. To assess $D_{P,T}$ validity we need to meta-observe σ_T ⁴ and take into account that $D_{M,T} = D_{P,T}$ in our conditions.

Following such modelling process in our problem instance resulted in the $M\text{-}SD$ and $M\text{-}COMPS$ also given in Example 1. If $M\text{-}OBS$ are also those of that example we get the following meta-diagnosis (from GDE output):

There are 3 kernel meta-diagnosis hypotheses:
 $\{M\text{-}Ab(M_1desc)\} \quad \{M\text{-}Ab(M_2desc)\} \quad \{M\text{-}Ab(A_1desc)\}$

Although not certain, the meta-diagnosis correctly includes the ontological falsehood of the sentence M_1desc .

3.2.2 The problem of diagnosable believed systems and sound and complete diagnostic algorithms

Suppose the M_1desc sentence was corrected using the meta-diagnosis computed in the last subsection; and it can now be assumed - and not proved due to the sixth principle in Section 2.2 - that the believed system description is ontologically true. Imagine some further requirements, typically when testing a new diagnostic algorithm and considering it as a black box, as well as when exploring further properties of the implemented believed system: "Admit the ontological truth of the believed system description and observations, as well as the possible lack of soundness and/or completeness by the diagnostic algorithm. Knowing that all the computed consistency-based diagnoses must be valid and certain, meta-diagnose S using a consistency-based approach.". Suppose an instance of this problem where the SD' (the corrected SD), $COMPS$ and OBS are those of Example 1.

Let us once more follow the modelling process:

⁴As discussed in the previous section, in many real-world applications it is possible to hypothesise the ontological truth of some exclusive meta-observations such as the true health state or a part of it. In the aeronautic domain, for instance, we have access to the true health state since each replaced component is tested after its removal.

1. The meta-system is made of:
 - a sensed part of reality correctly perceived as a well-implemented diagnostic system $S = (SD', COMPS, OBS, A)$; where $COMPS$ and OBS are those of Example 1 and SD' is the corrected SD .
 - the meta-goal of computing valid and certain proof-theoretic consistency-based diagnoses.
2. Using Corollary 4 and Theorem 7, the meta-components can be chosen to represent the implemented diagnostic algorithm and the whole implemented believed system description. Their meta-goals are the soundness and completeness and the diagnosability, respectively.
3. From Corollary 4 and the problem hypotheses we get the normal behaviour of the believed meta-system meta-component associated to the diagnostic algorithm. If Alg is the believed meta-system meta-component associated to A we get: $\neg M\text{-}Ab(Alg) \Rightarrow (\sigma_T \Rightarrow Dis(D_{P-T}))$; where $Dis(\cdot)$ is a function that returns the disjunction of every element in a set. Moreover, using Theorem 7 and the problem hypotheses we get the normal behaviour for the believed meta-system meta-component SD_{comp} : $\neg M\text{-}Ab(Alg) \wedge \neg M\text{-}Ab(SD_{comp}) \Rightarrow (\#D_{P-T} = 1)$
4. From the previous conditions, the meta-observables are the believed system observables extended with the proof-theoretic diagnosis D_{P-T} and the true health state σ_T .

Following such modelling process in our problem instance, a possible $M\text{-}COMPS$ is $\{Alg, SD_{comp}\}$ and a possible $M\text{-}SD$ is the theory made of the following sentences (extended with the appropriate axioms for arithmetic and so on):

$$\begin{aligned} &\neg M\text{-}Ab(Alg) \Rightarrow (\sigma_T \Rightarrow Dis(D_{P-T})) \\ &\neg M\text{-}Ab(Alg) \wedge \neg M\text{-}Ab(SD_{comp}) \Rightarrow (\#D_{P-T} = 1) \end{aligned}$$

Suppose the meta-observations: $\sigma_T = \neg Ab(M_1) \wedge \neg Ab(M_2) \wedge \neg Ab(M_3) \wedge \neg Ab(A_1) \wedge Ab(A_2)$ and the proof-theoretic diagnosis D_{P-T} is $\{\neg Ab(M_1) \wedge \neg Ab(M_2) \wedge \neg Ab(M_3) \wedge \neg Ab(A_1) \wedge \neg Ab(A_2)\}$.

We get the following kernel meta-diagnosis hypothesis (from GDE output): $\{M\text{-}Ab(Alg)\}$.

3.3 MEDITO: a logic-based meta-diagnosis tool

In the previous sections, the reader has been provided with a meta-diagnosis theory that, although far from a practical application, is a necessary condition for building one. It is now time to introduce MEDITO: a logic-based MEta-DIagnostic TOol aiming at computing the implemented diagnostic systems's normal and abnormal meta-components.

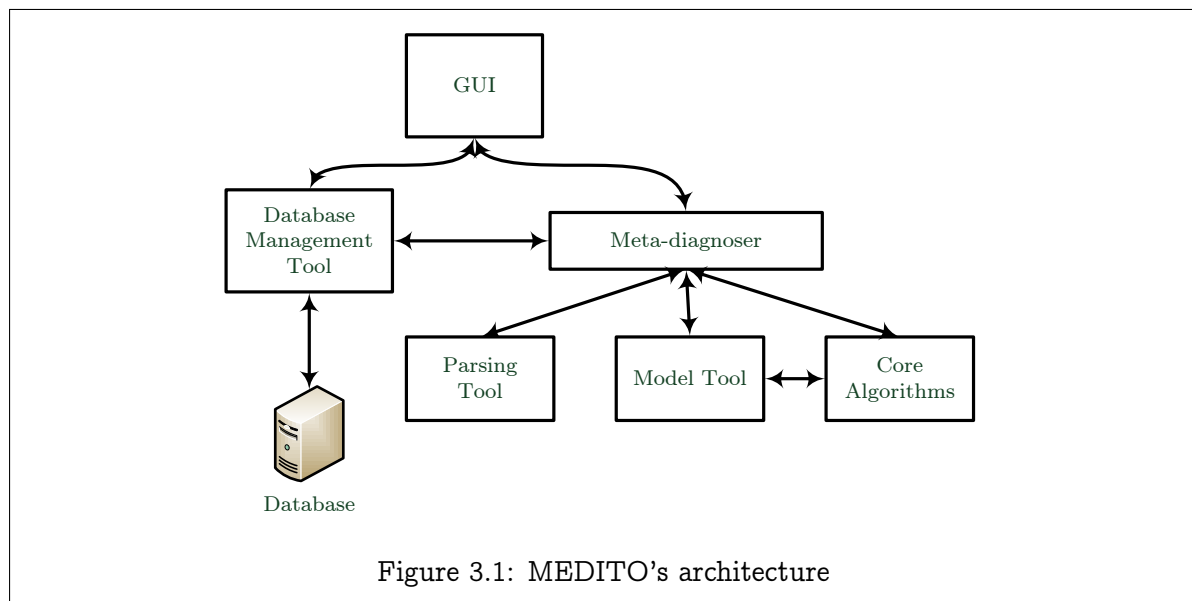
As for the computational implementation, MEDITO relies on the fact that every meta-diagnostic problem can be seen as a diagnostic one. This way, MEDITO uses Zhao and Ouyang [122] [121] diagnostic algorithms for determining all kernel meta-diagnosis hypothesis for a given meta-diagnostic problem. Since these algorithms require a test for determining the consistency of a logical theory, MEDITO also implements a Constraint Satisfaction Problem

(CSP) solver, *CHOCO* [107]. Note that the choice of *CHOCO* (as a CSP solver) and Zhao and Ouyang algorithms (as diagnostic algorithms) was one among the many possible choices of diagnostic algorithms and CSP solvers available.

With respect to *MEDITO* limitations, due to *CHOCO*'s language constraints, *MEDITO* can only accept those implemented diagnostic systems where *SD* and *OBS* are finite first-order theories; for determining the ontological truth of implemented sentences requires *M-SD* to use a language at least as expressive as the one in *SD* and *OBS*, and *CHOCO* does not handle those languages that are more expressive than first-order language. Note that this limitation is linked to my implementation choice and it is not a restriction of the theory of meta-diagnosis exposed in the previous section. In a nutshell, other choices of CSP solvers may overcome this handicap; a handicap that does not impact the general idea behind the present section.

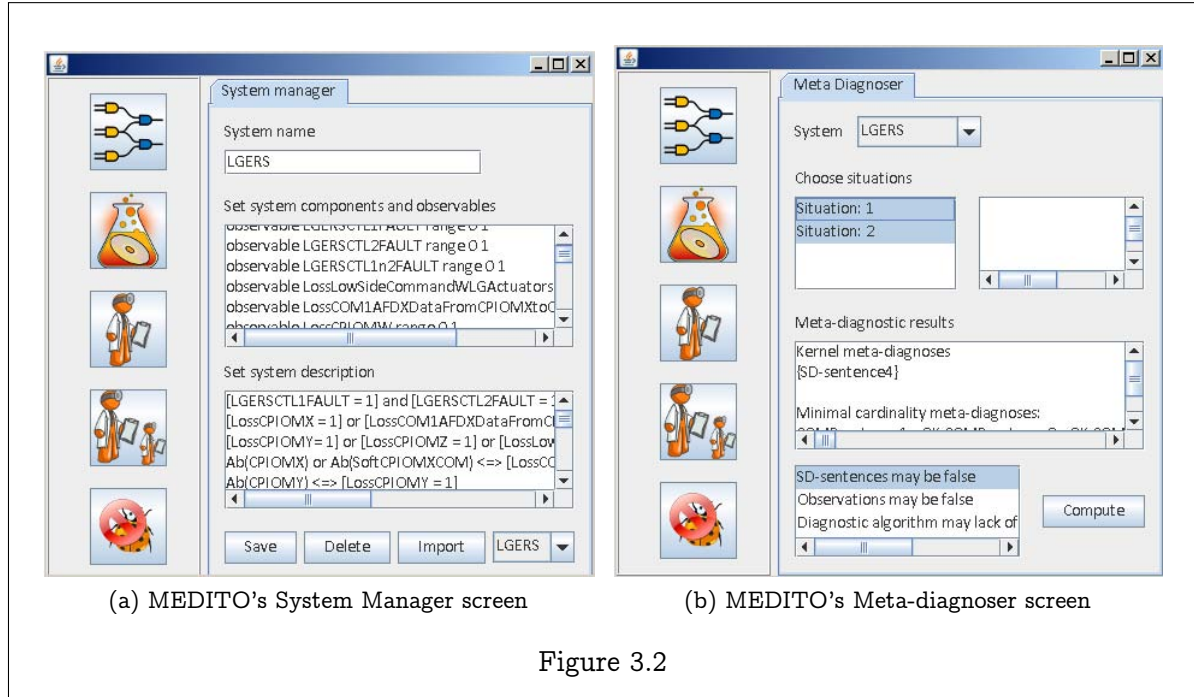
In the two subsections that follow, I will focus on: 1) providing a general view on *MEDITO*'s architecture, 2) giving a detailed view on *MEDITO*'s modules, and 3) using *MEDITO* to meta-diagnose the Polybox problem.

3.3.1 *MEDITO*'s architecture



As depicted in Figure 3.1, *MEDITO* is composed of six interacting modules, where:

- the *GUI* module provides a Graphical User Interface for handling user inputs such as adding, editing or removing believed systems, observations and meta-observations; as well as for displaying the meta-diagnoses,
- the *Database Management Tool* module is responsible for managing *MEDITO*'s SQL database containing believed systems, observations and meta-observations,
- the *Meta-diagnoser* module manages the computation of meta-diagnoses based on a choice of meta-components and test-cases (pairs of observations and meta-observations),



- the *Parsing Tool* module transforms user-input text into CHOCO Java objects,
- the *Model Tool* module creates a CSP model in CHOCO and checks for the consistency between such model and a given CHOCO constraint, and
- the *Core Algorithms* module is responsible for implementing Zhao and Ouyang algorithms [122] [121] for computing all hitting sets and minimal conflict sets of a theory.

3.3.2 A detailed view on MEDITO's modules

Following the architectural overview of MEDITO in the preceding subsection, we provide, hereafter, a detailed view of each MEDITO module.

MEDITO's GUI module

MEDITO's Graphical User Interface is divided in three main screens. In MEDITO's *System manager* screen, depicted in Figure 3.2a, the user is able to create, change and delete a believed system. These are represented by two text blocs, the first one containing the believed system components' and observables' declarations and the second one the believed system description.

In the first text bloc, text lines of the form: “component c”, and “observable o range a b”; are used to declare, respectively, a believed system component c and a believed system observable o ranging from value a to b. As for the second text bloc, a believed system description is declared through a series of text lines; each of them representing a SD-sentence in the Skolem normal form and with all the universal quantifiers omitted.

Moving to MEDITO's *Test Cases manager* screen, it is where the user is able to create, change and delete a series of test-cases, i.e. pairs of observations and meta-observations, for each believed system. Observations and meta-observations are represented by two text blocs containing text lines representing OBS and M-OBS sentences.

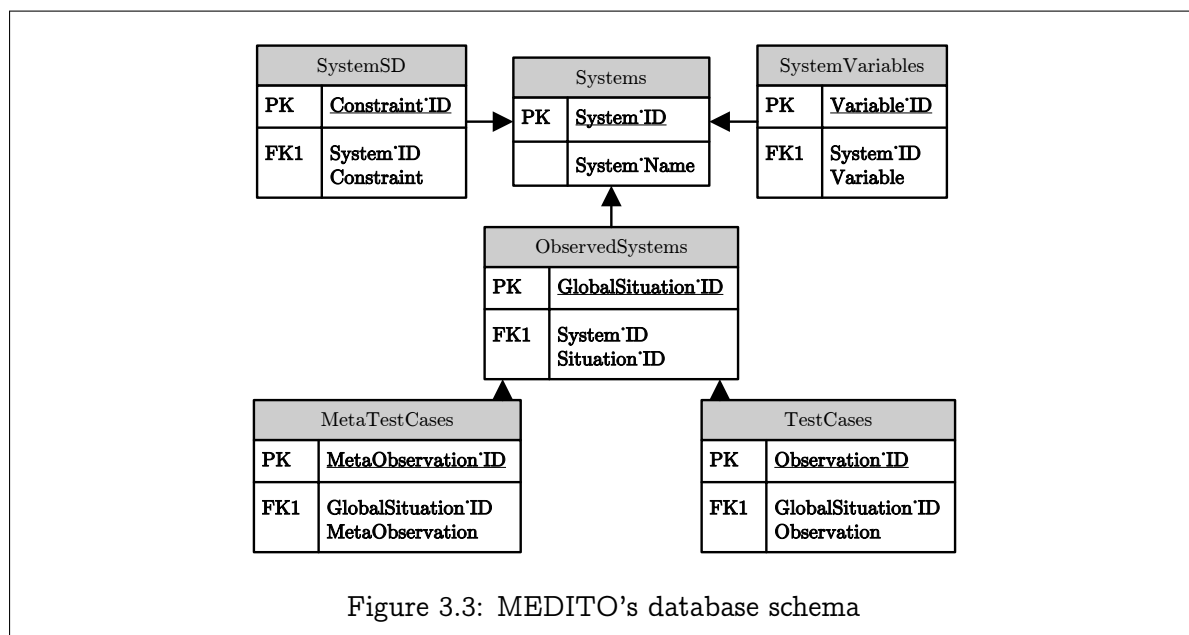
Finally, MEDITO's *Meta Diagnoser* screen, (cf. Figure 3.2b), enables the user to choose a believed system and a series of test-cases to compute meta-diagnoses, according to some preferences on meta-components. In MEDITO's current version, one can choose:

- every SD-sentence to be a meta-component whose property of truth may be lacking,
- every OBS-sentence to be a meta-component whose property of truth may be lacking, and/or
- the diagnostic algorithm to be a meta-component whose properties of soundness and/or completeness may be lacking.

Finally, according to users choices, MEDITO automatically builds a believed meta-system description to fit the user needs and displays the kernel meta-diagnoses this same screen.

MEDITO's Database Management Tool module and database schema

The *Database Management Tool* (DMT) module enables the user to store, change and delete believed systems and test-cases in a SQL database; and provides the *Meta-diagnoser* module access to such database in order to compute meta-diagnoses. Figure 3.3 illustrates MEDITO's SQL database schema, where: "PK" refers to a primary key, "FK" refer to a foreign key, the word "system" refers to a believed system, the word "variable" refers to an observable with some value ranges or to a component, and the word "situation" refers to a set of believed system observations and meta-observations.



MEDITO's Meta-diagnoser module

The *Meta-diagnoser* module manages the computation of meta-diagnoses by making use of the *Parsing Tool* (PT), *Model Tool* (MT) and *Core Algorithms* (CA) modules.

Algorithm 1 Algorithm implemented in the *Meta-diagnoser* module to compute kernel meta-diagnoses

Require: A believed system name, `systemName`

Require: A test-case identification, `situationID`

Require: Meta-comps. choice, `{sdOp,obsOp,algOp}`

{Step1: Get observables (Ob), COMPS, OBS, M-OBS from DB, parse user-text and initialize variables}

`mdPbText` \leftarrow DMT.imp(systemName,situationID)

`pb` \leftarrow PT.parse(mdPbText)

`pb.mOb` \leftarrow `pb.Ob`

`pb.mSD` \leftarrow \emptyset

{Step2: Create M-COMPS, M-SD and M-OBS based on user meta-component choices}

if `sdOp` = TRUE then

`pb.mCOMPS` \leftarrow createMCOMPS(`pb.SD`)

`pb.mSD` \leftarrow `pb.mSD` \cup transfToMeta(`pb.SD`)

`pb.mOb` \leftarrow `pb.COMPS` \cup `pb.mOb`

end if

if `obsOp` = TRUE then

`pb.mCOMPS` \leftarrow `pb.mCOMPS` \cup createMCOMPS(`pb.OBS`)

`pb.mSD` \leftarrow `pb.mSD` \cup transfToMeta(`pb.OBS`)

`pb.mOb` \leftarrow `pb.COMPS` \cup `pb.mOb`

else

`pb.mOb` \leftarrow `pb.mOb` \cup `pb.OBS`

end if

if `algOp` = TRUE then

 (...)

end if

{Step3: Solve the meta-diagnostic problem}

`CSPmod` \leftarrow MT.createModel(`pb`)

(`mDiag`,`mDiagMinCard`) \leftarrow CA.metaDiag(`CSPmod`,`pb.mCOMPS`)

return (`mDiag`,`mDiagMinCard`)

Algorithm 1 is implemented in the *Meta-diagnoser* module. Intuitively, this algorithm's:

Step 1: serves the purpose of importing a meta-diagnostic problem using *Database Management Tool* module's import function and using the *Parsing Tool* module to: 1) transform user-input-text components and observables declarations into CHOCO variables; and 2) transform system descriptions, observations and meta-observations into CHOCO constraints.

Step 2: serves the purpose of building meta-components and a meta-system description, by taking user choices of meta-components into account.

First, function `createMCOMPS(.)` associates a meta-component to each input-theory-sentence. As for function `transfToMeta(.)`, it transforms every SD-sentences and/or OBS-sentences stating “A” into meta-SD-sentences: $\neg M\text{-}Ab(mc) \Rightarrow A$; where `mc` is the meta-component associated to the sentence. This is as so because if a sentence stating “A” is not guaranteed to be ontologically true, then either “A” is ontologically true or the meta-component `mc` associated to such sentence is abnormal, i.e. $\neg M\text{-}Ab(mc) \Rightarrow A$.

Finally, if either the SD or OBS sentences can be ontologically false, the real-system health state is added as a possible meta-observable as a result of Theorem 2; and if the observations are assumed to be ontologically true, then they are added directly as meta-observations. As for the diagnostic algorithm, if it can be ontologically false then the computed proof-theoretic diagnoses are added as possible meta-observables as a result of Corollary 4.

Step 3: uses the *Model Tool* and *Core Algorithms* modules, respectively, to 1) create a CHOCO CSP model where the decision variables are the meta-components (through the function `createModel(.)`), and 2) compute, based on such model, all the kernel meta-diagnoses and minimal cardinality diagnoses for the meta-diagnostic problem (through the function `metaDiagnose(.)`).

MEDITO’s Parsing Tool module

MEDITO’s *Parsing Tool* module implements the parsing of user-input-text fields into CHOCO variables and constraints. This is done according to the syntax and semantics described in Subsubsection 3.3.2.

MEDITO’s Model Tool module

The *Model Tool* module in MEDITO encodes two simple functionalities. First, it enables the construction of a CHOCO model based on a series of CHOCO variables and constraints. Second, it enables one to test the consistency between a CHOCO model and a constraint through CHOCO’s exception in the “`propagate`” method; a method that propagates constraints until a fixed point is achieved and a set of solutions is found or a contradiction is detected.

MEDITO’s Core Algorithms module

The last module to be described is MEDITO’s *Core Algorithms* module. By making use of the syntactic equivalence between diagnosis and meta-diagnosis (cf. Section 3.1) this module implements Zhao and Ouyang diagnostic algorithms and applies them in a meta-diagnosis context to compute kernel meta-diagnoses. Moreover, it also transforms the problem into a min-CSP problem and solves it with CHOCO in order to obtain all the minimal cardinality meta-diagnoses. Note, once again, that this was one of the possible choices of algorithms.

In a more detailed manner:

- Zhao and Outang SE-tree based Minimal Hitting Set computation algorithm [122] is implemented in MEDITO's *Core Algorithms* module "MetaDiag" function to compute the kernel meta-diagnoses for a given meta-diagnostic problem, based on all Minimal Conflict Sets for that same problem.
- In order to compute all the Minimal Conflict Sets for the meta-diagnostic problem, MEDITO's *Core Algorithms* module implements Zhao and Ouyang SE-tree based Minimal Conflict Set computation algorithm detailed in [121]. To do so, the algorithm takes as input the CHOCO model generated by the *Model Tool* module and representing the diagnostic problem to be solved, as well as the set of meta-components in the meta-diagnostic problem.
- Finally, Zhao and Ouyang algorithm for computing all Minimal Conflict Sets needs an underlying consistency checker to verify if a set is a conflict set. As so, MEDITO's *Core Algorithms* module transforms the set to be verified into a CHOCO conjunction constraint and makes use of *Model Tool* module's consistency check function to test the consistency between that constraint and the CHOCO model representing the meta-diagnostic problem.

As for the minimal cardinality meta-diagnoses, these are obtained by adding a constraint on the number of abnormal meta-components to the CHOCO CSP model representing the meta-diagnostic problem; and solving the resulting new CSP model by minimizing such constraint.

3.3.3 Meta-diagnosing with MEDITO

Before concluding this chapter, the reader is provided with a demonstration of MEDITO's results meta-diagnosing the Polybox example, illustrated in Figures 3.4a, 3.4b and 3.4a.

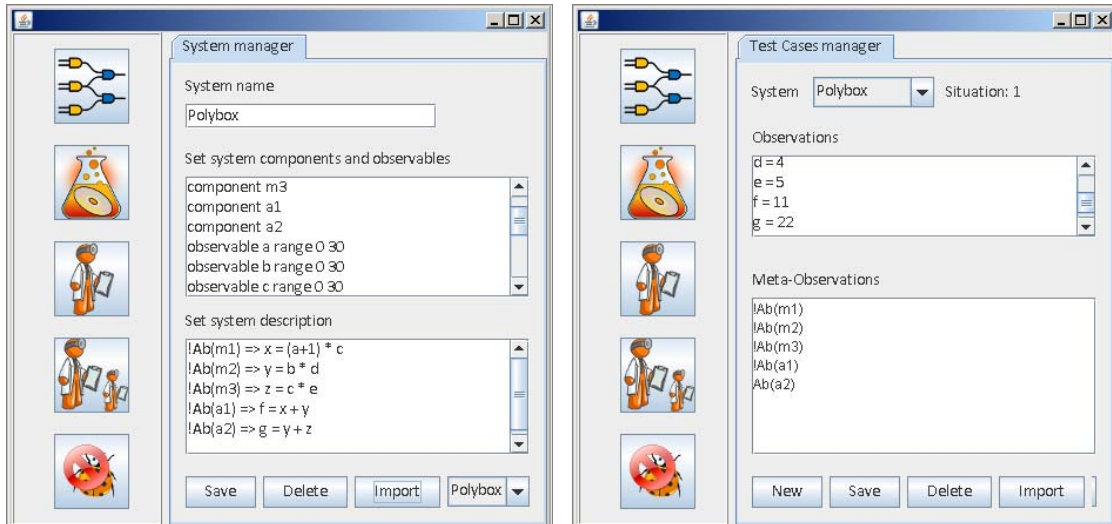
By obtaining the same meta-diagnoses as the ones predicted in Section 3.2, an illustration has been provided of meta-diagnosis claim that any sound and complete diagnostic algorithm can be used to solve a meta-diagnostic problem.

Complexity-wise, this tool inherits from the properties of Zhao and Ouyang diagnostic algorithms: it has a worst-case complexity of $O(2^n)$ and a consequent inability to handle large amounts of meta-components (as it is the case with every diagnostic algorithm aiming at computing every diagnosis we are aware of). Moreover, MEDITO's performance is proportional to the time needed by CHOCO to execute the consistency check of a CSP model. This being said, MEDITO computed the solution of the Polybox example in 142 milliseconds ⁵.

3.4 Originality and Related work

The objective of the present chapter was to offer Artificial Intelligence both a theory and a tool for reasoning about abnormalities in Models of implemented diagnostic systems, i.e. for meta-diagnosis or meta-explanation in general. As far as my knowledge goes an original contribution

⁵Intel Core 2 Duo CPU at 2.4GHz and 2Gb of RAM



(a) System Manager screen for Polybox example (b) Test Cases Manager screen for Polybox example



(c) Meta-diagnoser screen for Polybox example

Figure 3.4

- a statement supported by the publication of [14] and [15] - it was structured in three Sections where 1) a logical characterisation of meta-diagnosis was developed, 2) a modelling process was proposed and applied in two toy-problems, and 3) the general architecture and detailed modular specification of a meta-diagnosis tool was provided.

Concerning the logical characterisation of meta-diagnosis and the proposed modelling process, exposed in Sections 3.1 and 3.2, they provide a framework for reasoning about abnormalities in Models of implemented diagnostic systems. With respect to the works related to this one, let me start by stating that in the Model-based diagnosis community one can find some literature that recognise the existence of abnormalities in implemented diagnostic systems, even if very locally in the whole spectrum of possible abnormalities. Examples of such recognition are Davis and Hamscher's, statement in [36]: "a model is never completely correct"; Console, Dupré and Torasso's statement in [27]: "[in some cases] a complete model is either not available or intractable"; or Struss's work on abstractions and simplifications of models in [100]. As for managing abnormalities, few attempts have been made. Exceptions are, for instance, the management of incomplete believed systems in [27] and [120] or the management of uncertain observations in [71]. Furthermore, to our best knowledge, in the Model-based diagnostic community there has been little work done on detecting and isolating abnormalities in implemented diagnostic systems; despite the interest and ubiquity of the problem. In fact, the closer we can get to such results is the work of Yeung and Kwong in [120] where the authors not only focus on fault detection and isolation but also attempt to learn what must be changed to repair the believed system incompleteness. Finally, even in other communities such as in Modelling and Simulation and Systems Engineering, no works exist, to our knowledge, on the subject of meta-diagnosis. Once again, the further we can get are the very informal accounts of, for example, Naylor and Finger [80], Weld [112] or Sargent [93]. All in all, our theory of meta-diagnosis provides Model-based diagnosis with a uniform framework for dealing with possibly abnormal implemented diagnostic system. Moreover, meta-diagnosis can be used to isolate abnormalities in implemented diagnostic systems and select the best approach to manage them. For example, if a meta-diagnosis is the incompleteness of a believed system, then one can deal with such incompleteness by using Console, Dupré and Torasso's approach. Finally, note that the exposed concept of meta-diagnosis has no relation to the homonymous one defined by de Kleer in [38].

The third section on the present chapter was built around a general architecture and detailed modular specification of a logic-based meta-diagnosis tool: MEDITO; a tool providing empirical proof supporting meta-diagnosis claim that any diagnostic algorithm can be used to solve a meta-diagnostic problem. This proof will become even clearer in Chapter 4, where MEDITO will be used to meta-diagnose an Airbus A380 LGERS believed system description with very positive practical results. Finally, being the characterisation of meta-diagnosis an original contribution of this dissertation, MEDITO naturally inherits from this originality.

Chapter 4

The Airbus world

I believe in everything until it's disproved. So I believe in fairies, the myths, dragons. It all exists, even if it's in your mind. Who's to say that dreams and nightmares are not as real as the here and now?

- John Lennon

The introductory chapter announced a series of academic and industrial objectives that involved, by and large, reasoning about abnormalities in Models of implemented diagnostic systems. So far I have concentrated mostly on the former and accomplished the objectives of exposing a clear framework of diagnosis, defining and relating properties of diagnoses and diagnostic systems, and providing a theory and tool of meta-diagnosis for reasoning about abnormalities in Models of implemented diagnostic systems.

It is now time to focus on the industrial problems and objectives of this dissertation. In order to dive deep into this subject, let me remind the reader of what has been stated in the first chapter of this book concerning implemented Airbus diagnostic systems:

- The monitoring functions act as local observers providing discrepancies between a subsystem's normal and current behaviour (observations) to the Built-In Test Equipments and the Flight Warning System.
- The Built-In Test Equipments act as local agents performing Model-based diagnosis on some subsystems. They use their own algorithms (rules of inference), knowledge bases (scientific and mathematical theories) and the discrepancies sent by the monitoring functions (explananda-observations), among other observations, to produce Maintenance Messages (local diagnoses).
- The Centralized Maintenance System acts as a global agent performing Model-based diagnosis on the whole aircraft. It uses its own algorithms (rules of inference), knowledge bases (scientific and mathematical theories), Maintenance Messages (context-observations) and Aircraft Effects and Maintenance Effects (explananda-observations), among other observations, to produce a Post-Flight Report (aircraft-level automatic diagnosis).

- The Troubleshooting Manual procedures, through a series of manual isolation steps, enrich the contents of Maintenance Messages and, by transitivity, of Post-Flight reports.

With the material of the previous chapters, it is now possible to formally understand that local (subsystem-level) and global (aircraft-level) implemented diagnostic systems' Models can present a series of abnormalities affecting the properties of the computed diagnoses. This is the actual rationale behind the industrial objectives of this dissertation. As so, in this chapter, Airbus is provided with:

1. a common framework of diagnosis for every studied Airbus aircraft program - A380, A400M, A350 and research - enabling the comparison of diagnostic methods and results. Such framework will be done by thoroughly analysing the semantics of the results computed by the local and global diagnostic systems; and mapping them into the framework of Model-based diagnosis presented in Chapter 2.
2. a method and tool, based on the common framework of diagnosis, for checking local (subsystem-level) consistency between Maintenance Messages and Troubleshooting Manual procedures. Moreover, the developed tool, CONCKER (CONsistency ChecKER), be used to analyse some real-world Airbus data.
3. a method and tool, also based on the common framework of diagnosis and on the theoretical tools of Chapter 2, for validating and comparing configurable Built-In Test Equipments/Centralized Maintenance System algorithms and knowledge bases and Troubleshooting Manual procedures performances using heterogeneous real-world multi-program data. Once again, this Diagnosis Performance Assessment System, DPAT, will be used to analyse some Airbus real-world data.
4. a method and tool to detect and to isolate abnormalities in Models of the Centralised Maintenance System, i.e. observations, knowledge bases and algorithms. By making use of the material provided in Chapters 2 and 3, this industrial problem will be transformed into a meta-diagnostic problem that fits our theory of meta-diagnosis. As so, MEDITO will then be used as a tool to solve a real-world problem.

Each of these contributions will be exposed in a section of its own and according to the order above. As for the local-level equivalent of the fourth contribution, since it is not an Airbus responsibility, it is out of this book's scope. Nevertheless, I will try to convince the reader that transposing our fourth contribution to a local-level is far from being a daunting task.

As for the proof of this contribution's originality, it will be provided through a clear comparison between this work and other related ones in the literature. Such will be exposed, as usual, in the last section.

As a side-note, the reader is alerted for the fact that, for confidentiality issues, some names and data have been altered in the present chapter. However, this has absolutely no impact on the results presented.

4.1 Towards a common framework of diagnosis at Airbus

A common framework of diagnosis is essential, at Airbus, to enable the comparison between different aircraft programs' diagnostic methods and results. Hence, I will start by focusing on establishing such framework, both at subsystem and at aircraft-level; a framework that has been revised and accepted by Airbus' diagnostic community [11, 10].

4.1.1 Defining a common vocabulary of diagnosis

In my opinion, the first step into establishing a common framework of diagnosis at Airbus is the definition of a common vocabulary enabling the communication between engineers, experts and researchers working with implemented diagnostic systems of the different aircraft programs. This will be the object of the present subsection which is based on the academical works of Chittaro, Guida, Tasso and Toppano [25], Ribot, Pencolé and Combacau [91], and Isermann and Ballé [65]; and on the Airbus technical report of Cheriere [24]. Start with the notions of *Line Replaceable Unit* and *Line Replaceable Unit mode*.

Definition 47 (Line Replaceable Unit). *A Line Replaceable Unit (LRU) is a component (cf. Definition 7) designed to be easily replaceable during line maintenance activities. These include software, hardware and wirings.*

Definition 48 (Line Replaceable Unit mode). *A Line Replaceable Unit mode is a component that is part of a LRU.*

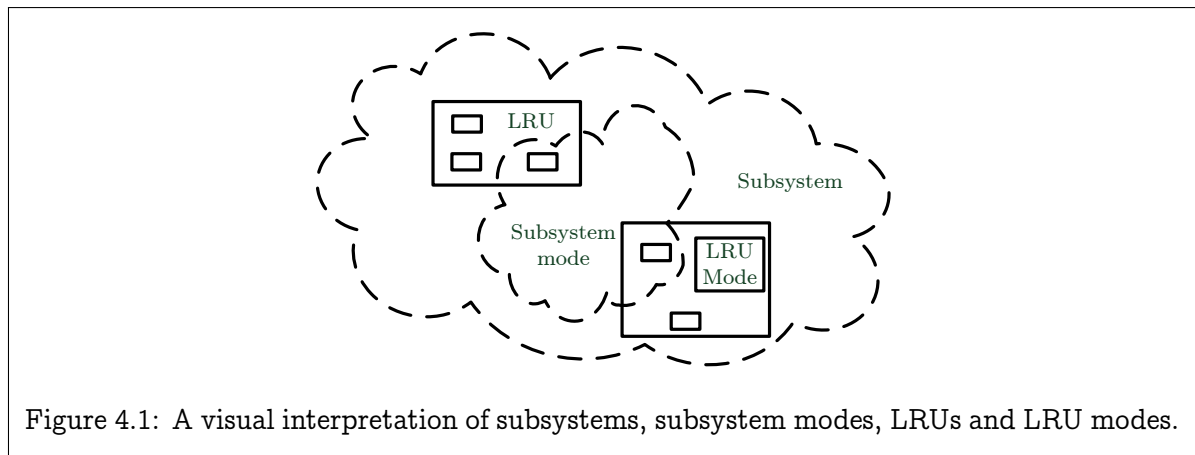


Figure 4.1: A visual interpretation of subsystems, subsystem modes, LRUs and LRU modes.

The distinction between LRUs and LRU modes is extremely important. In fact, while LRUs are the atoms of maintenance from the viewpoint of line mechanics; LRU modes are (some of) the atoms of maintenance from Airbus diagnostic systems' perspective. This perspective also brings with it the notions of *subsystem* and *subsystem mode*, relying on the concept of system (cf. Definitions 4).

Definition 49 (Subsystem). *A subsystem is a system whose perceived part of physics is a set of LRUs and components other than LRU modes.*

Definition 50 (Subsystem mode). *A subsystem mode is a system that is part of a subsystem.*

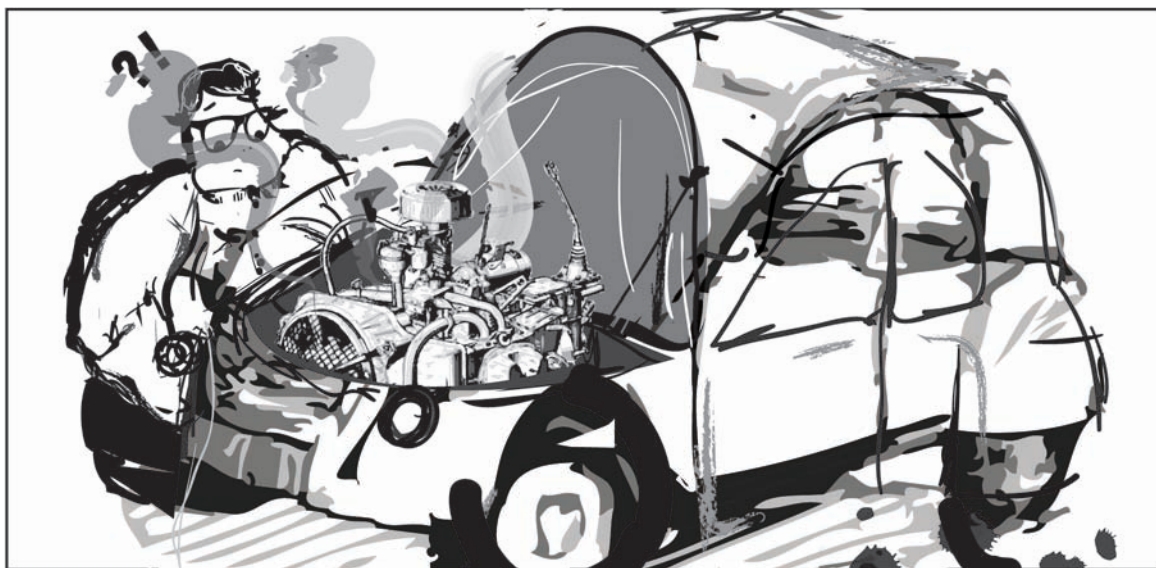
The notions of subsystem and subsystem mode, whose possible visual interpretation is given in Figure 4.1, will prove to be interesting in the following developments. For now, note that LRU modes are components and, as so, they may be abnormal (cf. Definition 8). At Airbus, depending on the LRU mode, these abnormalities take different names. Start with the notion of *LRU mode fault*, relying on the concept of perception (cf. Principle 4):

Definition 51 (LRU mode fault). *A LRU mode fault is a sensed part of reality perceived as an unpermitted deviation in a property of a LRU mode. The set of LRU modes that may be faulty will be noted **FAULT**.*

As stated above, a LRU mode fault is a possible abnormality. As for abnormalities - not just LRU mode faults - they can cause a subsystem mode goal (cf. Definition 3) not to be achieved. This is where the concept of *failure* appears.

Definition 52 (Failure). *A failure is the inability of a subsystem mode to meet a goal. The set of subsystem modes that may fail will be noted **FAIL**.*

At this point, it is interesting to illustrate the concepts of LRU mode fault and failure as well as their connections. Suppose there is a short-circuit in the battery of Mr Tortuga's car. In this case, the battery is said to be faulty. Moreover, due to this fault, suppose the battery is unable to provide a stable electrical current to other car components. In this case, a failure is said to occur. Hence, faults can lead to failures. As for failures in a subsystem mode, they can lead to failures in other subsystem modes (called, here after, inter-subsystem failures), as the inability to provide stable electrical current may lead to the inability of the car's electronic control unit to acquire data; failures can lead to faults, as the inability to provide stable electrical current can lead to a current spike which may lead to damages in some equipments; and faults may lead to faults, as a damaged battery can lead to a spilling of acid over another component, thus damaging it.



While failures can be caused by LRU mode faults, there are other possible abnormalities that may cause a given failure: *LRU regular inoperative modes* and *non-LRU non-faulty abnormal conditions*; the other possible abnormalities considered at Airbus.

Definition 53 (LRU Regular inoperative mode). *A LRU Regular Inoperative Mode (RIM) is a sensed part of reality perceived as a permitted working LRU mode potentially causing a failure. LRU modes that may be regular inoperative will be noted RIM.*

Definition 54 (Non-LRU non-faulty abnormal condition). *A non-LRU Non-Faulty Abnormal Condition (NFAC) is a sensed part of reality perceived as an unpermitted deviation in a property of a component other than a LRU, potentially causing a failure. The set of these components will be noted NFAC.*

The notion of RIM can be exemplified by a LRU reset causing a temporary failure. As for that of NFAC, ice present in the fuel tank is an example of a non-LRU non-faulty abnormal condition that may cause a series of failures in the fuel valves. With Faults, RIMs, NFACs and failures exposed, one is now able to focus on the concept of Virtual Line Replaceable Unit.

Definition 55 (Virtual Line Replaceable Unit). *A Virtual Line Replaceable Unit (VLRU) is a A380/A400M/A350-specific concept representing either:*

1. *a single non-LRU component, possibly under a non-faulty abnormal condition,*
2. *a single LRU mode that may be regular inoperative, or*
3. *a set of LRU modes that may be faulty [5].*

The set of virtual line replaceable units will be noted VLRU. Finally, the function Covered(.) returns the elements of NFAC, RIM or FAULT represented by a VLRU.

All the concepts presented above are grouped in the notions of Element, a concept particular to the A380, A400M and A350 aircraft programs [5]; and Accusable Object, a concept particular to the research program [24].

Definition 56 (Element). *An element, noted e , is either a member of FAULT, a member of VLRU or a special member of NFAC called power supply accusation (PSA).*

Definition 57 (Accusable Object). *An accusable object, noted o , is either a member of FAULT, a member of RIM, a member of NFAC or a member of FAIL.*

The abnormality of Elements and Accusable Objects are, respectively, what the A380/A400M/A350 and the research programs diagnostic systems judge as potential causes of all the failures in the aircraft. This is why such concepts are introduced and also why the reader is encouraged to retain them. Moreover, note that the word “Element” and the expression “Accusable Object” (and plurals), will always be capitalised when used in this context.

At this point, let me stop for a brief moment to regain a high-level vision. As the reader has been told numerous times, explanation (and Model-based diagnosis in particular) is a reasoning process where *explanans* explain some explananda-observations if the latter can

be inferred from the former and some context-observations, both at some earlier time, along with some scientific and mathematical theories representing the artefact to be diagnosed. So far, I have concentrated on abnormalities, the explanans as computed by implemented Airbus diagnostic systems. It is now time to focus on observations at Airbus and introduce the concepts of *system symptom*, *Aircraft Effect*, *Maintenance Effect*, *Global Effect* and *context observation*. These rely, naturally, on Principle 4.

Definition 58 (System symptom). *A System Symptom is an observation indicating a local-level (subsystem-level) failure to be explained, usually by a Built-In Test Equipment* ¹.

Definition 59 (Aircraft Effect). *An Aircraft Effect is an observation of a global-level (aircraft-level) failure with an operational impact, that has to be explained by the maintenance function of the Centralised Maintenance System.*

Definition 60 (Maintenance Effect). *A Maintenance Effect is the observation of a global-level (aircraft-level) failure with a maintenance impact, that has to be explained by the maintenance function of the Centralised Maintenance System.*

Definition 61 (Global Effect). *A Global Effect is either an Aircraft Effect or a Maintenance Effect. The unary function GEPe(.) returns the set of Elements/Accusable Objects whose abnormality may cause a given Global Effect.*

Definition 62 (Context observation). *A Context Observation is a local or global-level observation, other than aircraft or Maintenance Effects, sent to the implemented diagnostic system.*

Concerning System Symptoms, in the aircraft programs A380, A400M and A350, they correspond to “confirmed events” as defined in [4]; and in the research program they correspond to symptoms [24]. An example of a System Symptom is the Loss of COM1 AFDX data from CPIOM X to CPIOM W. Concerning Aircraft Effects, they correspond to warnings, dispatch advisory messages, flags, advisories and crew observations. An example is the L/G CTL SYS 1 + 2 FAULT warning. Finally, context observations do not have a general name at Airbus. An example of context observation is the flight phase.

Before ending this subsection, the concepts involved with the explanation of System Symptoms will be introduced in the form of *Maintenance Messages* and *Troubleshooting Manual procedures*.

Definition 63 (Maintenance Message). *A Maintenance Message, noted mm, is a local diagnosis sent by a Built-In Test Equipment to the Centralised Maintenance System. For every Airbus aircraft program, a Maintenance Message contains:*

- *a Fault Message Code (FMC) identifying the Maintenance Message and mapping it to a set of System Symptoms. SS(.) is a unary function returning the set of System Symptoms associated to a given Maintenance Message.*

¹If a system symptom indicates a subsystem-level failure that can never lead to a Maintenance or Aircraft Effect, then it must not be explained by a Built-In Test Equipment. These cases, however, will not be treated in this dissertation.

- a *BITE identifier*, indicating the Built-In Test Equipment that computed the Maintenance Message. *BITE(.)* is a unary function returning the identifier of the Built-In Test Equipment that computed a given Maintenance Message.
- an *explanation*, that is, a boolean combination of AB-literals over either Elements, in the A380/A400M/A350 case, or Accusable Objects, in the research-program case. *E(.)* is a unary function returning the explanation of a given Maintenance Message. *PCMM(.)* is a unary function returning the Elements/Accusable Objects present in the explanation of a Maintenance Message.
- a *Task Reference*, pointing to a single Troubleshooting Manual procedure (cf. Definition 64). *Task(.)* is a unary function returning the task reference of a given Maintenance Message.

Definition 64 (Troubleshooting Manual procedure). A Troubleshooting Manual procedure, noted *tsmp*, is a task pointed by a Maintenance Message enabling the confirmation and isolation of the abnormalities causing a given failure. For every Airbus aircraft program, a Troubleshooting Manual procedure contains:

- A Task Reference, identifying the TSM procedure.
- Some FMC-references, i.e. a set of Fault Message Codes of the Maintenance Messages pointing to the TSM procedure. *MMRef(.)* is a unary function returning the FMC of these Maintenance Messages for a given TSM procedure.
- A set of possible causes, i.e. a set of elements from *FAULT* and *NFAC* potentially verified in the isolation phase and replaced or restored in a TSM procedure. *PCTSM(.)* is a unary function returning the set of possible causes for a given TSM procedure.
- Some BITE Tests for isolation, the Built-In Test Equipment's tests that need to be done in the manual isolation step. *Tests(.)* is a unary function returning the set of tests associated to a given TSM procedure. *TestOwner(.)* is a unary function returning the BITE identifier for a given test.

Finally, at Airbus each Built-In Test Equipment is in charge of a given exclusive set of Elements (in the A380/A400M/A350 case) or Accusable Objects (in the research-program case). This set is called the BITE Perimeter. The unary function *Per(.)* returns the perimeter of a given BITE; while the unary function *Owner(.)* returns the BITE whose Perimeter a given Element/Accusable Object or test belongs to.

4.1.2 Defining a common subsystem-level diagnostic framework

Up to this point nothing has been done except the establishment of a common vocabulary of diagnosis. This vocabulary enables the attachment of every Airbus aircraft program concept to ones introduced in the remainder of this section. It is now time to focus on local-level diagnostic systems and contribute with: 1) a common framework of subsystem-level diagnosis, and 2) a method to transpose every aircraft-program-specific subsystem-level diagnosis into our common framework.

Before tackling these subjects, the the reader is provided with the exact notion of what local-level diagnoses are considered to be. Recall that, in every aircraft program:

- the Built-In Test Equipments perform Model-based diagnosis on some subsystems, hence producing Maintenance Messages (local diagnoses).
- associated to every Maintenance Message there is a Troubleshooting Manual procedure enriching its contents.

As so, two types of local-level diagnoses can be considered: the MM-related local diagnoses and the TSM-related local diagnoses. While both are explanations of System Symptoms, MM-related and TSM-related local diagnoses may differ, since: 1) the teams in charge of the implemented diagnostic system computing MM-related local diagnoses and the ones in charge of producing TSM procedures are not the same, and 2) no exhaustive process guarantees that both explanations are consistent. The reader is encouraged to retain this fact.

The subsystem-level common framework of diagnosis

The common framework of subsystem-level diagnosis arises by instantiating the diagnostic framework of Chapter 2 to Airbus local-level diagnostic systems (cf. Definitions 19 and 16).

Definition 65 (System symptom diagnosis). *A System Symptom diagnosis is a diagnosis for the diagnostic problem $(SD, COMPS, OBS)$ with:*

- OBS , the set of System Symptoms and context observations observed by the BITE.
- $OBS_{exp} \subseteq OBS$, the set of System Symptoms to be diagnosed.
- $OBS_{cons} = OBS \setminus OBS_{exp}$.
- SD , the believed system description representing the subsystem to be diagnosed. At Airbus it corresponds to Built-In Test Equipments knowledge bases.
- $COMPS$, the set of components whose abnormality is to be diagnosed in the subsystem. At Airbus these correspond to every element of $FAULT \cup RIM \cup NFAC \cup FAIL$ that may lead to an observable System Symptom.

From A380/A400M/A350 local-level diagnoses to the common framework

As stated in the beginning of the present subsection, two types of local diagnoses can be considered at Airbus, namely the MM-related and TSM-related ones. Moreover, it was also exposed that both types of local diagnoses are explanations of System Symptoms. As for the objective of the present subsection, it is to map A380, A400M and A350 MM-related and TSM-related local diagnoses into the subsystem-level common framework of diagnosis above. To achieve such objective, a thorough analysis of these aircraft programs local diagnoses' semantics is needed.

MM-related local diagnoses

Concerning A380/A400/A350 MM-related local diagnoses, to analyse them one has to focus on A380/A400M/A350 Maintenance Messages and on their particularities². Hence:

²I will focus on nothing but Standard B Maintenance Messages as described in [4], for Standard A ones are mapped, in the A380, A400M and A350 aircraft programs, into these ones before being treated by the

- The explanation in A380/A400M/A350 Maintenance Messages is a boolean combination of abnormalities of a fixed number of Elements (from one to eight depending on the elements to be accused).
- The A380/A400M/A350 Maintenance Messages contain a priority report, indicating the internal or external property of the MM³. A MM is said to be internal iff all the Elements in the explanation belong to the emitter BITE perimeter.

As agreed in [10], the semantics behind an A380/A400M/A350 Maintenance Message explanation depend on its priority report. As so, intuitively:

- for every internal (high priority) MM, the abnormality of the non-VLRU Elements directly present or represented by a VLRU in the MM explanation is seen as a possible cause of the set of System Symptoms associated to the MM.
- for every external (low priority) MM, the abnormality of the MM-explanation Elements that belong to the perimeter of the emitter BITE is seen as above. As for those MM-explanation Elements that belong to a different BITE perimeter, the abnormality of every non-VLRU Element directly present or represented by a VLRU in that BITE perimeter is seen as a possible cause of the System Symptoms associated to the MM.

These remarks enable the formal mapping of A380/A400/A350 MM-related local diagnoses into the subsystem-level common framework of diagnosis.

Definition 66 (A380/A400M/A350 MM as System Symptom diagnoses). *Let mm be a A380/A400M/A350 Maintenance Message and let $e_i \in PCMM(mm)$. Moreover, let E' be the transformation of $E(mm)$ with:*

- $Ab(e_i) \mapsto \bigvee_{e' \in Per(Owner(e_i))} Ab(e')$, if $e_i \notin Per(BITE(mm))$,

and let E'' be the transformation of E' with

- $Ab(e_i) \mapsto \bigvee_{e' \in Covered(e_i)} Ab(e')$, if $e_i \in VLRU$.

A System Symptom diagnosis for $SS(mm)$ is the set of health states covered (cf. Subsection 2.3.2) by E'' .

TSM-related local diagnoses

As for A380/A400/A350 TSM-related local diagnoses, its analysis depends on TSM procedures. Before going any further, I encourage the reader to note that each MM points to a TSM procedure, while the opposite is not true. This means that an MM-explanation should always entail a TSM procedure explanation; a fact that will be exploited in Section 4.2. Facing this information, the semantics behind a TSM procedure can be defined. As agreed in [10]:

- For every possible cause directly present in a TSM procedure or represented by a VLRU in the perimeter of a BITE in the BITE Tests for isolation, its abnormality is seen as a possible cause of at least one set of symptoms associated to the FMC referenced in the TSM procedure.

Centralised Maintenance System.

³The Internal/external priority in standard B Maintenance Messages corresponds to the high/low priority field in standard A ones.

- Every LRU Regular Inoperative Mode present in a MM-related local diagnosis associated to a FMC referenced in the TSM procedure is seen as a possible cause of at least one set of symptoms associated to the FMC referenced in the TSM procedure.

It is now possible to formally map A380/A400/A350 TSM-related local diagnoses into our subsystem-level common framework of diagnosis.

Definition 67 (A380/A400M/A350 TSM procedures as System Symptom diagnoses). *Let $tsmp$ be a A380/A400M/A350 TSM procedure and $R = \bigcup_{mm \in MMRef(tsmp)} RIM \cap PCMM(mm)$. A System Symptom diagnosis for $\bigvee_{mm \in MMRef(tsmp)} \bigwedge_{x \in SS(mm)} x$ is the set of health states covered by:*

$$\bigvee_{e \in PCTSM(tsmp)} Ab(e) \vee \bigvee_{r \in R} Ab(r) \vee \bigvee_{t_j \in Tests(tsmp)} \bigvee_{e' \in Per(TestOwner(t_j))} Ab(e's).$$

From research-program local-level diagnoses to the common framework

Once we have defined the view of A380/A400M/A350 MM-related and TSM-related local diagnoses as System Symptom diagnoses, it is time to focus on the research-program counterpart.

MM-related local diagnoses

In order to study the syntax and semantics of research-program MM-related local diagnoses, let me follow the same strategy as before and analyse the particularities of research-program Maintenance Messages (also called Health Data) [24]. Straightforwardly:

- in the research-program case, Maintenance Messages' explanations are about Accusable Objects instead of Elements. This being so, it is possible for failures to be a part of MM-explanations.
- the Maintenance Message explanation is the research-program case (also called Sufficient Diagnosis) is a kernel diagnosis (cf. Definition 21) for the set of System Symptoms associated to the FMC.

Being research-program MM-related explanations already connected to the framework of Chapter 2, mapping them into our subsystem-level common framework of diagnosis becomes a trivial task.

Definition 68 (Research-program MM as System Symptom diagnoses). *Let mm be a research-program Maintenance Message. A System Symptom diagnosis for $SS(mm)$ is the set of health states covered by $E(mm)$.*

The reader is encouraged to compare Definitions 66 and 68 and verify a two very important aspects. First, contrary to research-program local diagnoses, the restrained syntax of A380/A400M/A350 MM-related local diagnoses makes the usage of VLRUs for representing sets of Elements mandatory. Moreover, the fact that failures can be a part of research-program MM-related diagnoses makes it possible to logically establish a cascade of Maintenance Messages; for some MM System Symptoms may be related to a failure that is a part of another MM local diagnosis. This is impossible in the A380/A400M and A350 cases and is one of the most important rationales behind the research-program diagnostic concepts.

TSM-related local diagnoses

Finally, concerning research-program TSM-related local diagnoses, they are almost equivalent to the A380/A400M/A350 TSM-related local diagnoses if one takes into account the changes introduced by the suppression of VLRUs and the introduction of failures. This being so:

Definition 69 (Research-program TSM procedures as System Symptom diagnoses). *Let $tsmp$ be a research-program TSM procedure, let $R = \bigcup_{mm \in MMRef(tsmp)} RIM \cap PCMM(mm)$ and let $Func(.)$ be a mathematical function returning the aircraft function whose failure is associated to a given BITE test. A System Symptom diagnosis for $\bigvee_{mm \in MMRef(tsmp)} \bigwedge_{x \in SS(mm)} x$ is the set of health states covered by:*

$$\bigvee_{e \in PCTSM(tsmp)} Ab(e) \vee \bigvee_{r \in R} Ab(r) \vee \bigvee_{t_j \in Tests(tsmp)} Ab(Func(t_j)).$$

4.1.3 Defining a common aircraft-level diagnostic framework

In the previous subsection I provided Airbus with both a common framework of subsystem-level diagnosis, and a method to transpose every aircraft-program-specific subsystem-level diagnosis into that common framework. It is now time to focus on their aircraft-level equivalent. In order to do so, recall that, in every aircraft program:

- the Centralized Maintenance System performs Model-based diagnosis on the whole aircraft, thus producing a Post-Flight Report (aircraft-level automatic diagnosis).
- the Troubleshooting Manual procedures enrich the contents of Maintenance Messages and, by transitivity, of Post-Flight reports.

The aircraft-level common framework of diagnosis

The common framework of aircraft-level diagnosis arises in the same way as the subsystem-level one did, that is, by trivially instantiating the diagnostic framework of Chapter 2 to Airbus aircraft-level diagnostic systems (cf. Definitions 19 and 16).

Definition 70 (Global Effect diagnosis). *A Global Effect diagnosis is a diagnosis for the diagnostic problem $(SD, COMPS, OBS)$ with:*

- OBS , the set of Global Effects and context observations observed by the Centralised Maintenance System.
- $OBS_{exp} \subseteq OBS$, the set of Global Effects to be diagnosed.
- $OBS_{cons} = OBS \setminus OBS_{exp}$.
- SD , the believed system description representing the aircraft for diagnostic purposes. At Airbus these are Centralised Maintenance System knowledge bases.
- $COMPS$, the set of components whose abnormality is to be diagnosed in the aircraft. At Airbus these correspond to every element of $FAULT \cup RIM \cup NFAC$ that may lead to an observable Global Effect.

From A380/A400M/A350 aircraft-level diagnoses to the common framework

As in the previous subsection, a mapping between the A380/A400M/A350 global diagnoses and our framework of aircraft-level diagnosis has to be developed. In order to do so, one must understand the semantics behind a Post-Flight Report, i.e. the report computed by the Centralised Maintenance System:

- Syntax-wise, A380 and A400M Post-Flight Reports consist of a set of PFR items. These contain a series of Global Effects related to the PFR item, as well as a set of Fault Cases. As for Fault Cases, each of them contains a set of received Maintenance Messages with one and only one of those tagged “Fault Origin”. As far as the semantics goes, each PFR item represents a maintenance action necessary for every Global Effect present in it to disappear. As for the maintenance action, it is selected among the TSM procedure pointed by Fault Origins in each Fault Case.
- With respect to its syntax, A350 Post-Flight Reports consist of a set of PFR global items, each of which containing a set of related Global Effects and a boolean combination of Fault Cases. As for Fault Cases, they are defined as above. Moreover, contrary to the A380 and A400M cases, the Global Effects are exclusive of each PFR global item. Semantics-wise, each global PFR item represents a boolean combination of maintenance actions necessary and sufficient for every Global Effect in the global PFR item to disappear. As before, each maintenance action is represented by a TSM procedure pointed by the Fault Origin of each Fault Case.

All in all, in both the A380/A400M and the A350 cases, one can see that every Maintenance Message tagged as Fault Origin was effectively received, meaning that its underlying System Symptoms and associated failure were registered. As so, the explanation of each Global Effect is the conjunction of each Maintenance Message explanation interpreted at an aircraft level. Using the same arguments as those of Subsection 4.1.2, we are once more confronted to two types of aircraft-level diagnoses: the MM-related global diagnoses and the TSM-related global diagnoses; inheriting the possible difference between each other from the local differences between Maintenance Messages’ and Troubleshooting Procedures’ local diagnoses.

MM-related aircraft-level diagnoses

Concerning MM-related aircraft-level diagnoses, at Airbus it was agreed that:

- for every Maintenance Message, the abnormality of the non-VLRU Elements directly present or represented by a VLRU in the MM explanation is seen as a possible cause of the Global Effect associated to the MM.
- for every non-VLRU Elements absent in every MM explanation associated to the Global Effect, its abnormality is considered not to be a possible cause of the Global Effect.

As so:

Definition 71 (A380/A400M/A350 MM-related PFR as Global Effect diagnoses). *Let PFR be an A380/A400M/A350 Post-Flight Report with a given Global Effect GE. Moreover, let ORIG be the set of Fault Origin Maintenance Messages in every PFR item or global PFR*

item where GE is present. Furthermore, let $NC = GEP_{\text{Per}}(GE) \setminus \bigcup_{mm \in \text{ORIG}} PCMM(mm)$. Finally, let E'_x be the transformation of $E(mm_x)$ with:

- $Ab(e_i) \mapsto \bigvee_{e' \in \text{Covered}(e_i)} Ab(e')$, if $e_i \in \text{VLRU}$.

A Global Effect diagnosis for GE is the set of health states covered by:

$$\bigwedge_{mm_x \in \text{ORIG}} E'_x \wedge \bigwedge_{e' \in NC} \neg Ab(e')$$

TSM-related aircraft-level diagnoses

Concerning TSM-related aircraft-level diagnoses, at Airbus it was agreed that:

- for every possible cause in a TSM procedure, its abnormality is seen as a possible cause of the Global Effect associated to that TSM procedure (through a Maintenance Message).
- every LRU Regular Inoperative Mode present in a MM explanation associated to a FMC referenced in the TSM procedure is seen as a possible cause of the Global Effect associated to that TSM procedure (by transitivity through a Maintenance Message)
- for every possible cause absent in every TSM procedure associated to a Global Effect (by transitivity through a Maintenance Message), its abnormality is considered not to be a possible cause of the Global Effect.

As so:

Definition 72 (A380/A400M/A350 TSM-related PFR as Global Effect diagnoses). *Let PFR be an A380/A400M/A350 Post-Flight Report with a given Global Effect GE . Moreover, let ORIG be the set of Fault Origin Maintenance Messages in every PFR item or global PFR item where GE is present; and TSMORIG be the set of Troubleshooting Manual procedures pointed by them. Furthermore, let $PC = \bigcup_{tsmp \in \text{TSMORIG}} PCTSM(tsmp)$ and let $R = \bigcup_{mm \in \text{ORIG}} RIM \cap PCMM(mm)$. Finally, let $NPC = GEP_{\text{Per}}(GE) \setminus PC \cup R$. A Global Effect diagnosis for GE is the set of health states covered by:*

$$[\bigwedge_{tsmp \in \text{TSMORIG}} \bigvee_{e \in PC} Ab(e) \vee \bigvee_{r \in R} Ab(r)] \wedge \bigwedge_{e \in NPC} \neg Ab(e)$$

From research-program aircraft-level diagnoses to the common framework

Finally, it is time to establish a mapping between research-program global diagnoses and our framework of aircraft-level diagnosis. Contrary to the A380/A400M/A350 case, these global diagnoses are not simply a boolean combination of Maintenance Messages. As so, if on the one hand one can define MM-related global diagnoses, the TSM-related ones cannot be defined; for as far as our knowledge goes there is still no method defined for selecting Troubleshooting Procedures in the research-program. Concerning MM-related global diagnoses:

Definition 73 (Research-program MM-related PFR as Global Effect diagnoses). *Let PFR be a research-program Post-Flight Report with a given Global Effect GE . Moreover, let $E_{GE} = f(Ab(o_1), \dots, Ab(o_n))$ be a kernel diagnosis in the research-program Post-Flight Report explaining GE , where each o_i is an Accusable Object. A Global Effect diagnosis for GE is the set of health states covered by:*

$$E_{GE} \wedge \bigwedge_{o \in GEP_{\text{Per}}(GE) \setminus \{o_1, \dots, o_n\}}$$

4.2 Checking for local consistency with CONCKER

The first section of the present chapter defined a common framework of diagnosis at Airbus; a framework which enables the comparison of different aircraft programs' diagnostic methods and results. With such framework, I now have all the necessary means to address one of the major industrial objectives: to define a method and specify and develop a tool to check for the consistency between Troubleshooting Manual procedures and Maintenance Messages.

4.2.1 Improving local-diagnoses' consistency at Airbus

From the numerous insights provided by the formalisation of Airbus diagnostic-world in the previous section, the first that I will explore is the existence of two types of local-diagnoses: the MM-related and the TSM-related ones.

First of all, note that theory tells us that every Maintenance Message should point to a single Troubleshooting Manual procedure; and that the latter can be pointed by several Maintenance Messages. Hence, for a given TSM procedure, its TSM-related local-diagnosis should cover all the MM-related local diagnoses of all the System Symptoms in every single Maintenance Message that points at it. In a nutshell, with the framework of the previous section one can assert a mathematical condition that must hold between Maintenance Messages and TSM procedures: if MM is the set of all the potentially emitted Maintenance Messages for a given aircraft, $SSDMM(.)$ is a function returning the System Symptom diagnosis associated to a Maintenance Message and $SSDTSMP(.)$ is the TSM counterpart of $SSDMM(.)$, then:

$$\begin{aligned} & \forall_{mm \in MM} \text{Task}(mm) \neq \emptyset, \text{ and} \\ & \forall_{mm \in MM} SSDMM(mm) \subseteq SSDTSMP(\text{Task}(mm)) \end{aligned}$$

If, on one hand, theory established a formal relation between MM-related and TSM-related local diagnoses; things can be somehow different in practice for the A380, A400M and A350 aircraft programs. In fact, line mechanics can be confronted with situations where:

1. a Maintenance Message points to no TSM procedure, and
2. a MM-related local-diagnosis is not covered by its associated TSM-related local-diagnosis.

Such situations could happen because, as stated in the previous section: 1) the teams in charge of the implemented diagnostic system computing MM-related local diagnoses and the ones in charge of producing TSM procedures are not the same, and 2) the process guaranteeing the consistency between both explanations is not perfect.

Noting that the first condition is an organisation constraint at Airbus, this dissertation contributes with a CONSistency CheckER method and tool, CONCKER, that exhaustively checks for the validity of the condition mentioned above for every Maintenance Message in Airbus' A380, A400M and A350 aircraft programs.

In the three subsections that follow, I will focus on: 1) providing a general view on CONCKER's architecture, 2) giving a detailed view on CONCKER's modules, and 3) using CONCKER against some real-world data. Finally, I would like to state that CONCKER was built

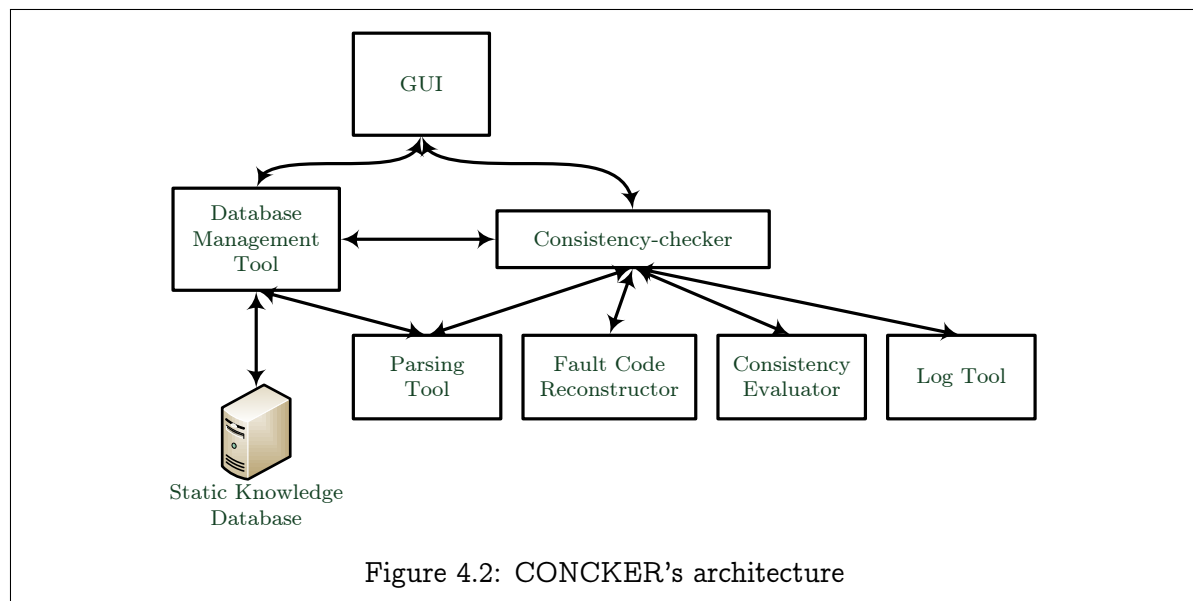
with the help of Boulaabi [20], Stella [99] and Oh [83], three Airbus interns; and that CONCKER is effectively used today, at Airbus, to tackle the local consistency checking problem.

4.2.2 CONCKER's architecture

Start by focusing on CONCKER's architecture.

As depicted in Figure 4.2, CONCKER is composed of seven interacting modules, where:

- the *GUI* module provides a Graphical User Interface for handling user inputs such as adding, editing or removing Tuning files, DK2 customisation files (VLRU coverage definition), Troubleshooting Manual files, Filter files and customised lists of Maintenance Messages; as well as for launching the consistency checks and displaying the results.
- the *Database Management Tool* module is responsible for managing and querying CONCKER's H2 database engine,
- the *Consistency-checker* module manages the customisable computation of inconsistencies between a set of Maintenance Messages and one or many Troubleshooting Procedures,
- the *Parsing Tool* module transforms the raw input data (eg. the text in TSM procedures) into H2 database engine objects or java objects.
- the *Fault Code Reconstructor* builds the MM-explanation of every Maintenance Message either in the Tuning file or in the user inputs. It uses a subsystem-level interpretation of MM-explanation according to the common framework of subsystem-level diagnosis presented in Subsection 4.1.2.
- the *Consistency Evaluator* module is responsible for computing all the inconsistencies between a set of Maintenance Messages and a set of Troubleshooting Manual procedures according to the framework of Subsection 4.1.2.
- the *Log Tool* module logs the inconsistencies found.



4.2.3 A detailed view on CONCKER's modules

After exposing the architecture of CONCKER, it is time to focus on each of its modules and provide the reader with a detailed account of such tool. Note that the *Parsing Tool* and *Log Tool* modules will not be analysed for no conceptual value would be added.

CONCKER's GUI module

Concerning CONCKER's Graphical User Interface, it is divided in a series of screens enabling: 1) the injection, edition and removal of data into and from CONCKER's database, 2) the configuration of the consistency analysis, and 3) the display of CONCKER's results.

In CONCKER's *Data Input* screen, depicted in Figure 4.3, the user is able to select the Tuning files and Troubleshooting Manual files to be stored in the static knowledge database; and the Filter files and MM-input files in the consistency-check phase. I will come back to a detailed view on these files in a few paragraphs.

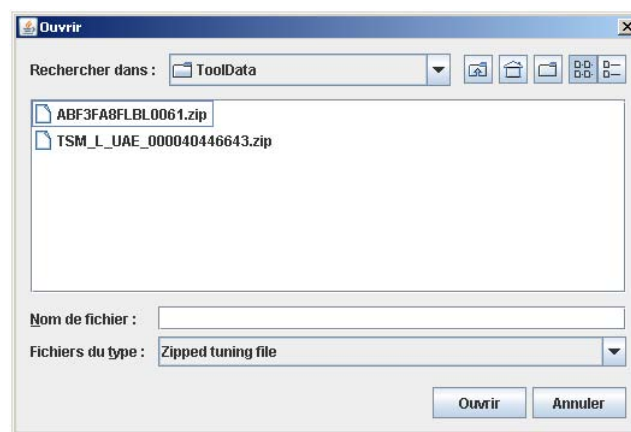
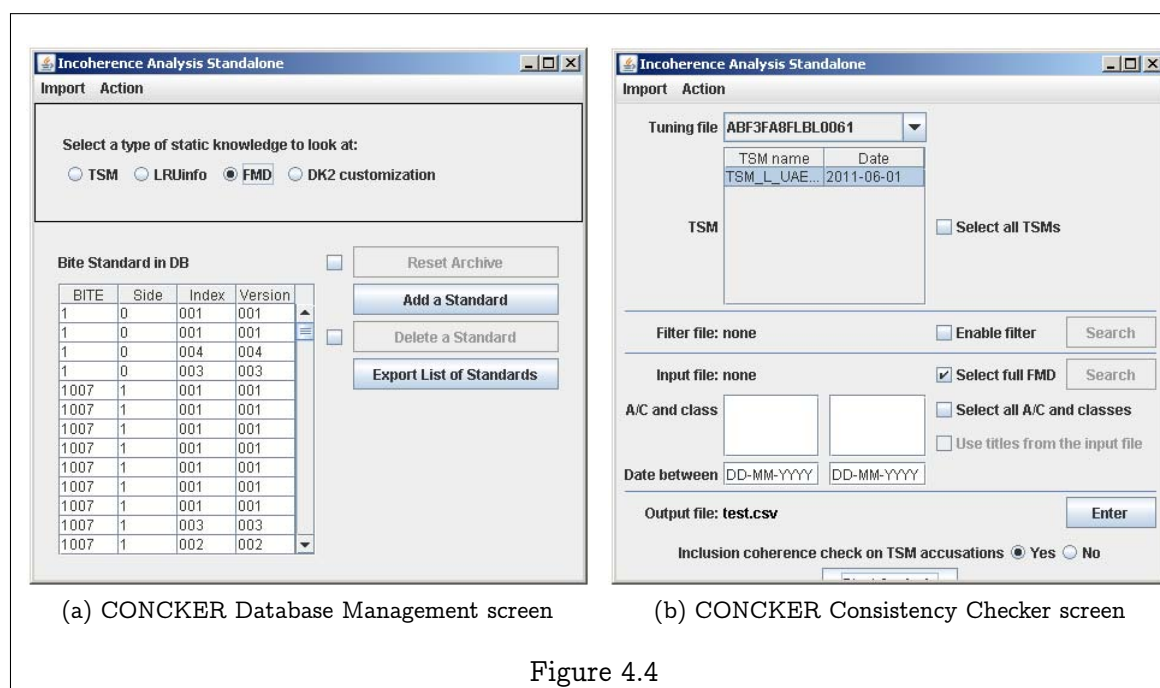


Figure 4.3: CONCKER Data Input screen

With respect to CONCKER's *Database Management* screen, it enables the user to manage the Troubleshooting Manuals, Fault Message Descriptions and LRU-info standards. Moreover, it provides the user with the possibility of defining the elements covered by the Virtual Line Replaceable Units. Figure 4.4a illustrates such screen.

From the Database Management screen, let us move to the *Consistency Checker* one. This screen allows the user to customise the consistency checking and select the Maintenance Messages and Troubleshooting Manuals to be checked. As depicted in Figure 4.4b, the user must select a Tuning file enabling CONCKER to acquire an aircraft configuration of elements. Second, at least one Troubleshooting Manual has to be selected for consistency checking. Third, the user is allowed to select a Filter file with a list of Fault Message Codes corresponding to Maintenance Messages that shall not be tested for consistency. Moreover, the choice of the set of Maintenance Messages for consistency checking has to be made. Hence, the user has the



possibility of either selecting all the Maintenance Messages in the Fault Message Description of the selected Tuning file; or selecting a customisable set of Maintenance Messages with an option for filtering aircraft, MM classes and dates. This allows real-world data to be analysed at Airbus with a minimal user effort. At last, the user may select between a full or partial consistency check. I will come back to this point later in this section.

After the computation of the consistency-check results, these are stored in a .csv file and automatically displayed to the user as depicted in Figure 4.6.

CONCKER's Database Management Tool module and database schema

With respect to the *Database Management Tool* (DMT) module, it is responsible for managing the interaction between CONCKER's modules and its H2 static knowledge database, depicted in Figure 4.5.

As the reader may remark, the schema of CONCKER's static knowledge database standardises some of the data contained in Maintenance Messages and Troubleshooting Manual procedures as mentioned in Section 4.1. This being so, the first major function of the *Database Management Tool* module is to parse the Tuning files and Troubleshooting Manual files to fill this database. This is done with the help of the *Parsing Tool* module. Moreover, this module manages the database and answers to queries from other CONCKER modules.

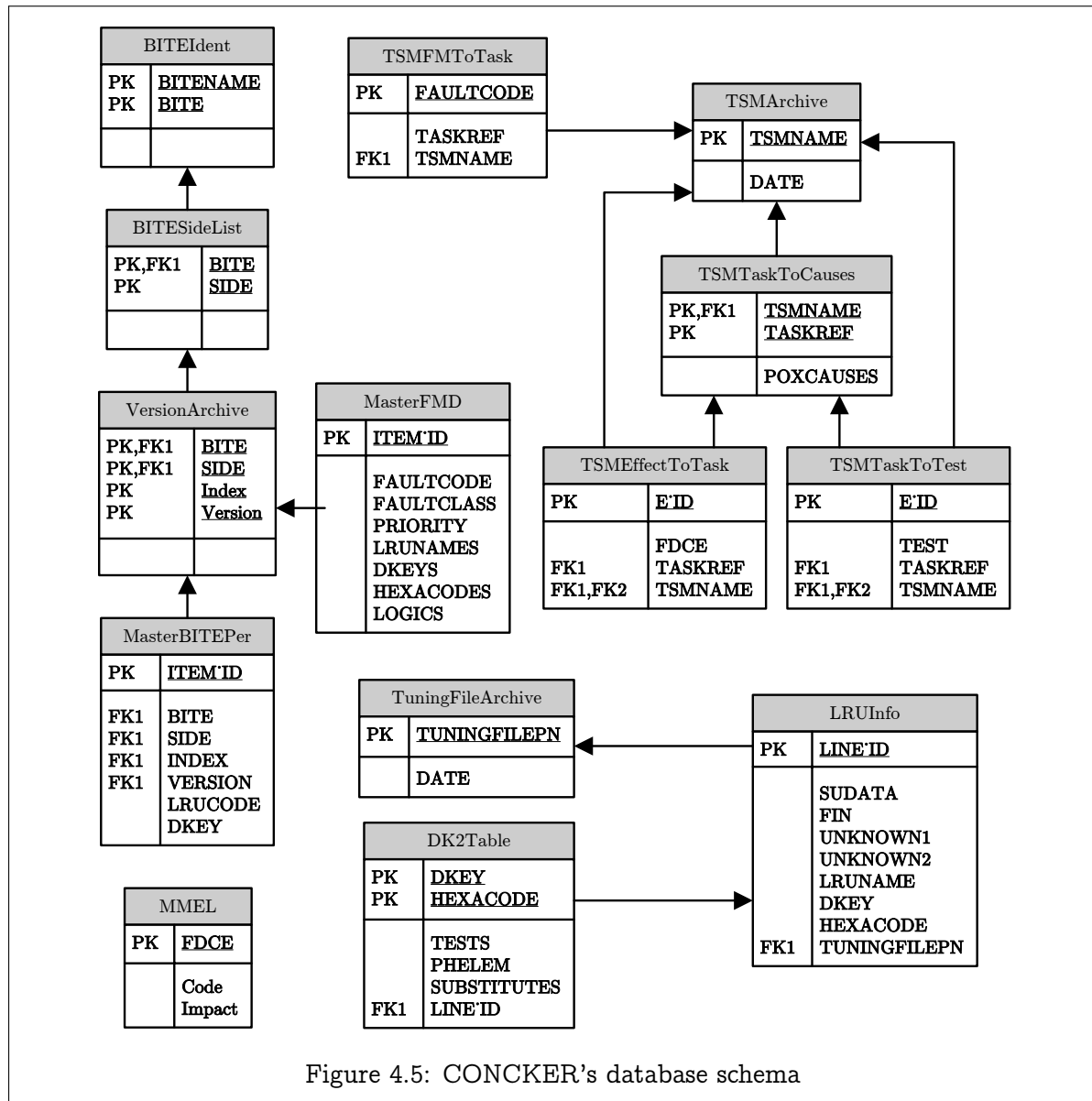


Figure 4.5: CONCKER's database schema

CONCKER's Consistency-checker module

From CONCKER's database and DMT module, it is now time to focus on the consistency analysis. The *consistency-checker* module is responsible for computing the inconsistencies between a given set of Maintenance Messages and Aircraft Effects, and a set of Troubleshooting Manual procedures. This is done with the help of the *Parsing Tool* (PT), *Fault Code Reconstructor* (FCR), *Consistency Evaluator* (CE) and *Log Tool* (LT) modules.

Require: A Tuning File part-number, `tuningFilePN`.

Require: A set of Troubleshooting Manual names, `TSMNames`.

Require: A Filter file path, `filterFilePath`.

Require: A MM-AE-input-file path, `inputFilePath`.

Require: A set of analysis options, $\{\text{filterOp}, \text{fullFMDOp}, \text{allACOp}, \text{useInputMMExpOp}, \text{datesOp}, \text{ACAndClassesOp}, \text{inclusionAnalysisOp}\}$.

$\text{MM} \leftarrow \emptyset$; $\text{AE} \leftarrow \emptyset$; $\text{TSMP} \leftarrow \emptyset$; $\text{filteredMM} \leftarrow \emptyset$; $\text{incResults} \leftarrow \emptyset$

{Step1: Get the list of Maintenance Messages (MM) and Aircraft Effects (AE) from DB or user inputs. }

if `fullFMDOp = True` then

$\{\text{MM}, \text{AE}\} \leftarrow \text{DMT.impMMAE}(\text{tuningFilePN})$

else

$\{\text{MM}, \text{AE}\} \leftarrow \text{PT.parseInputMMAE}(\text{inputFilePath}, \text{allACOp}, \text{datesOp}, \text{ACAndClassesOp})$

end if

$\text{MM} \leftarrow \text{FCR.rebuildExp}(\text{MM}, \text{tuningFilePN}, \text{useInputMMExpOp}, \text{inputFilePath})$

{Step2: Filter the MM according to the Filter option (`filterOp`) and the Filter file (`filterFilePath`). }

if `filterOp = True` then

$\text{filteredMM} = \text{PT.parseFilterMM}(\text{filterFilePath})$

$\text{MM} \leftarrow \text{removeMM}(\text{MM}, \text{filteredMM})$

end if

{Step3: Check for inconsistencies according to the Inclusion option (`inclusionAnalysisOp`) and log them. }

$\{\text{incResults}, \text{TSMP}\} \leftarrow \text{CE.evalNotFound}(\text{MM}, \text{AE}, \text{TSMNames})$

if `inclusionAnalysisOp = True` then

$\text{incResults} \leftarrow \text{incResults} \cup \text{CE.evalInclusion}(\text{MM}, \text{AE}, \text{TSMP})$

end if

$\text{LT.log}(\text{incResults})$

Algorithm 2 is implemented in CONCKER's *Consistency-checker* module. Intuitively:

Step 1: serves the purpose of importing a set of Maintenance Messages Fault Code and Aircraft Effects FDCE codes either from CONCKER's static knowledge database (using the DMT module), or from an input file provided by the user (using the PT module). In the former case, according to the `tuningFilePN`, CONCKER queries the MMEL table to gather Aircraft Effects' FDCE identification and the MasterFMD table to gather Maintenance Messages' Fault Message Codes (cf. Figure 4.5). As for the latter case, CONCKER parses the `inputFilePath` respecting the `allAircraftOp`, `datesOp` and `aircraftAndClassesOp` options to gather the same data as in the former case. Finally, CONCKER's `rebuildExp` function in the FCR module builds

the MM-explanation of all the Maintenance Messages in MM using the common framework of subsystem-level diagnosis (cf. Subsection 4.1.2). This is done either using the data in the static knowledge database or in the user input files, according to `useInputMMExplanationOP`.

Step 2: serves the purpose of filtering the MM set according to the Fault Message Codes specified in the `filterFilePath` file.

Step3: serves the purpose of computing the inconsistencies between the sets MM and AE, and the Troubleshooting Manual procedures mapped by the `TSMNames`. First, the MM and AE for whom no TSM procedure is identified is computed by the `evalNotFound` function in the CE module. Moreover, the TSM procedures found are stocked in `TSMP`. Then, if the `inclusionAnalysisOp` option is selected, CONCKER identifies those MM whose explanation is not consistent with the associated TSM procedure explanation. This is done using the `evalInclusion` function in the CE module. Finally, inconsistencies are logged using the `log` function in the *Log Tool* module.

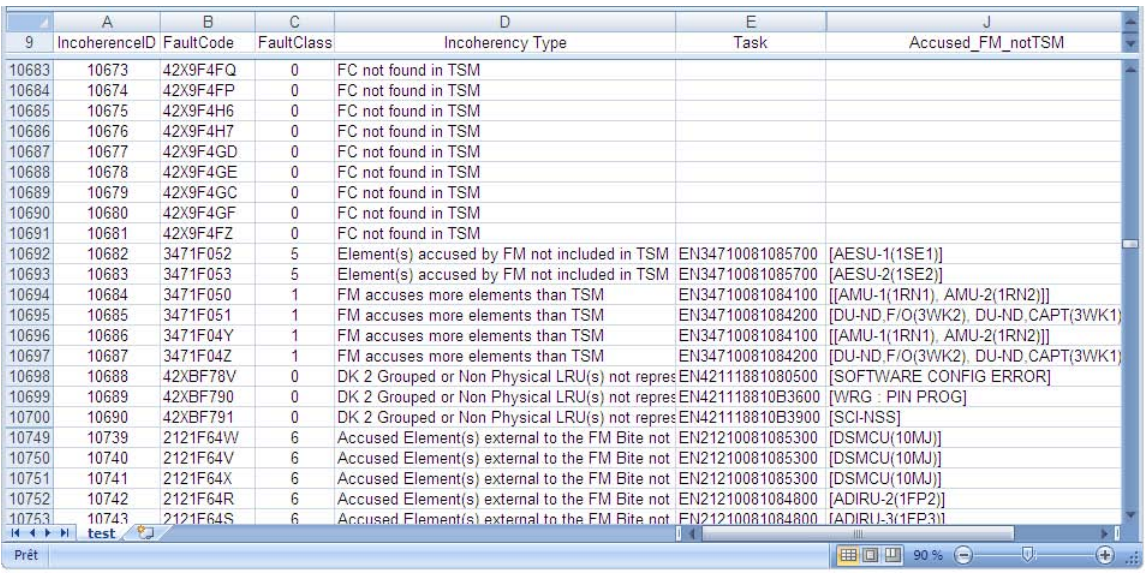
CONCKER's Fault Code Reconstructor module

The *Fault Code Reconstructor* module is responsible for building a first-order sentence representing a MM-explanation interpreted according to the common framework of subsystem-level diagnosis exposed in Subsection 4.1.2. To do so, this module uses either the data in the `BITEIdent`, `BITESideList`, `VersionArchive`, `MasterFMD`, `MasterBITEPer`, `TuningFileArchive`, `LRUInfo` and `DK2Table` tables of CONCKER's static knowledge database; or the equivalent data contained in the user input file.

CONCKER's Consistency Evaluator module

CONCKER's heart is the *Consistency Evaluator* module and its two main functions:

- The `evalNotFound` function is responsible for 1) gathering all the Fault Message Codes in MM and all the FDCE codes in AE, 2) verifying through SQL queries and according to the Troubleshooting Manual procedures in `TSMNames`, if these codes have a counterpart in the `TSMFMtofTask` and `TSMEffectToTask` tables in CONCKER's specific knowledge database (according to the first condition provided in Subsection 4.2.1, i.e.: $\forall_{mm \in MM} \neq \emptyset$, and 3) returning the identification of all the TSM procedures found, as well as all the elements of the sets MM and AE without an associated TSM procedure.
- The `evalInclusion` function is responsible for determining all the Maintenance Messages in MM whose MM-related local-diagnosis is inconsistent with the associated TSM-related diagnosis according to the condition provided in Subsection 4.2.1, i.e.: $\forall_{mm \in MM} SS-DMM(mm) \subseteq SSDTSMP(Task(mm))$. Since the analysis of Definitions 66 and 67 tells us that: 1) there are no $\neg Ab(.)$ in either mapped System Symptom diagnosis, and 2) the TSM-related local-diagnoses are a disjunction of abnormalities; then the consistency analysis becomes an inclusion check of Elements, implemented through SQL queries.



9	A	B	C	D	E	J
	IncoherenceID	FaultCode	FaultClass	Incoherency Type	Task	Accused_FM_notTSM
10683	10673	42X9F4FQ	0	FC not found in TSM		
10684	10674	42X9F4FP	0	FC not found in TSM		
10685	10675	42X9F4H6	0	FC not found in TSM		
10686	10676	42X9F4H7	0	FC not found in TSM		
10687	10677	42X9F4GD	0	FC not found in TSM		
10688	10678	42X9F4GE	0	FC not found in TSM		
10689	10679	42X9F4GC	0	FC not found in TSM		
10690	10680	42X9F4GF	0	FC not found in TSM		
10691	10681	42X9F4FZ	0	FC not found in TSM		
10692	10682	3471F052	5	Element(s) accused by FM not included in TSM	EN34710081085700	[AESU-1(1SE1)]
10693	10683	3471F053	5	Element(s) accused by FM not included in TSM	EN34710081085700	[AESU-2(1SE2)]
10694	10684	3471F050	1	FM accuses more elements than TSM	EN34710081084100	[[AMU-1(1RN1), AMU-2(1RN2)]]
10695	10685	3471F051	1	FM accuses more elements than TSM	EN34710081084200	[DU-ND F/O(3WK2), DU-ND CAPT(3WK1)]
10696	10686	3471F04Y	1	FM accuses more elements than TSM	EN34710081084100	[[AMU-1(1RN1), AMU-2(1RN2)]]
10697	10687	3471F04Z	1	FM accuses more elements than TSM	EN34710081084200	[DU-ND F/O(3WK2), DU-ND CAPT(3WK1)]
10698	10688	42XBF78V	0	DK 2 Grouped or Non Physical LRU(s) not repres	EN42111881080500	[SOFTWARE CONFIG ERROR]
10699	10689	42XBF790	0	DK 2 Grouped or Non Physical LRU(s) not repres	EN42111881083600	[WRG : PIN PROG]
10700	10690	42XBF791	0	DK 2 Grouped or Non Physical LRU(s) not repres	EN42111881083900	[SCI-NSS]
10749	10739	2121F64V	6	Accused Element(s) external to the FM Bite not	EN21210081085300	[DSMCU(10MJ)]
10750	10740	2121F64V	6	Accused Element(s) external to the FM Bite not	EN21210081085300	[DSMCU(10MJ)]
10751	10741	2121F64X	6	Accused Element(s) external to the FM Bite not	EN21210081085300	[DSMCU(10MJ)]
10752	10742	2121F64R	6	Accused Element(s) external to the FM Bite not	EN21210081084800	[ADIRU-2(1FP2)]
10753	10743	2121F64S	6	Accused Element(s) external to the FM Bite not	EN21210081084800	[ADIRU-3(1FP3)]

Figure 4.6: CONCKER results

4.2.4 Checking for local consistency with CONCKER

Up to this point I have provided the reader with a description CONCKER in terms of its architecture and modular specification. This tool, implementing a method for verifying the consistency between local diagnoses at Airbus, is already used at Airbus for such purposes. As so, it is now time to provide empirical proof supporting my statements by demonstrating CONCKER's results against real-world data. Before doing so, however, let me remind you that for confidentiality issues, some names and data have been altered. This has been done in a way that does not impact the results presented.

The first step into using CONCKER is to populate its static knowledge-base, using the input menu functions, with some Tuning files and Troubleshooting Manual files. These, gathered directly from the same sources as the airline ones, constitute the basis of our study. Figure 4.4a depicts the state of the Static Knowledge screen after the insertion of such files.

The input of Tuning files and Troubleshooting Manual files is followed by the configuration of the consistency analysis in CONCKER. As depicted in Figure 4.4b, in this demonstration I have chosen to analyse the consistency of the full list of Maintenance Messages in the Tuning file with the selected part-number; against the a single Troubleshooting Manual. Moreover, I have decided to perform a complete analysis.

As illustrated in Figure 4.6, there are many inconsistencies found; some of which were corrected after this analysis, thus showing its importance.

4.3 Assessing performance with DPAT

In the beginning of the present chapter, I stated that four objectives were to be achieved. With two of them already tackled, it is now time to focus on another one: the definition of a method and specification of a tool to automatically validate and compare different configurable Built-In Test Equipments/Centralized Maintenance System algorithms and knowledge bases and Troubleshooting Manual procedures performances using heterogeneous real-world multi-program; and introduce the reader to the Diagnosis Performance Assessment Tool (DPAT).

4.3.1 Improving Airbus diagnostic performance assessment

The arrival of the A380 aircraft program and its severely interconnected complex subsystems, in the beginning of the present millennium, brought up a series of problems to implemented Airbus' diagnostic systems. While, until then, experts of a given subsystem could forge believed system descriptions that could easily be put together by them, this is no more the case and Centralised Maintenance System's knowledge bases are much more error-prone. Moreover, with maintenance becoming a very important factor distinguishing aircraft manufacturers, Airbus strives to produce better implemented diagnostic systems that quickly become mature. Therefore, the problem of automatically validating and comparing the performance of different solutions for implemented diagnostic systems - even before detecting and isolating abnormalities in the Models of certain solutions - is becoming a fundamental question at Airbus; a question whose absence of answer disables reuses, retrofits and proofs of added value. Unfortunately, and despite the importance of this problem, finding a solution to it is far from being a simple task for:

1. different diagnostic systems from heterogeneous aircraft programs demand specific input syntaxes and semantics.
2. different diagnostic systems from heterogeneous aircraft programs produce outputs in specific syntaxes and semantics.
3. diagnostic systems are not [sufficiently] formalised, let alone in the framework of Chapter 2. As so, properties such as the soundness and completeness of their diagnostic algorithms cannot be proved.
4. Airbus diagnostic system's common performance criteria are not optimal.

Straightforwardly, the common framework of diagnosis provided in the first section addresses the first two items of the list above for: 1) every specific aircraft-level diagnosis can be seen as a standard Global Effect diagnosis (cf. Definition 70), and 2) every specific subsystem-level diagnosis can be seen as a System Symptom diagnosis (cf. Definition 65).

Concerning the last two items, things are somehow more complicated. In fact, the absence of a high degree of formalisation leaves no choice but to consider implemented Airbus' diagnostic systems as grey boxes whose direct performance analysis is a daunting task ⁴. Fortunately, the combined results of Chapter 2 and of Section 4.1 come to our help. In fact, if 1)

⁴Note, however, that the "white part" can even be meta-diagnosed as we will see in the following section.

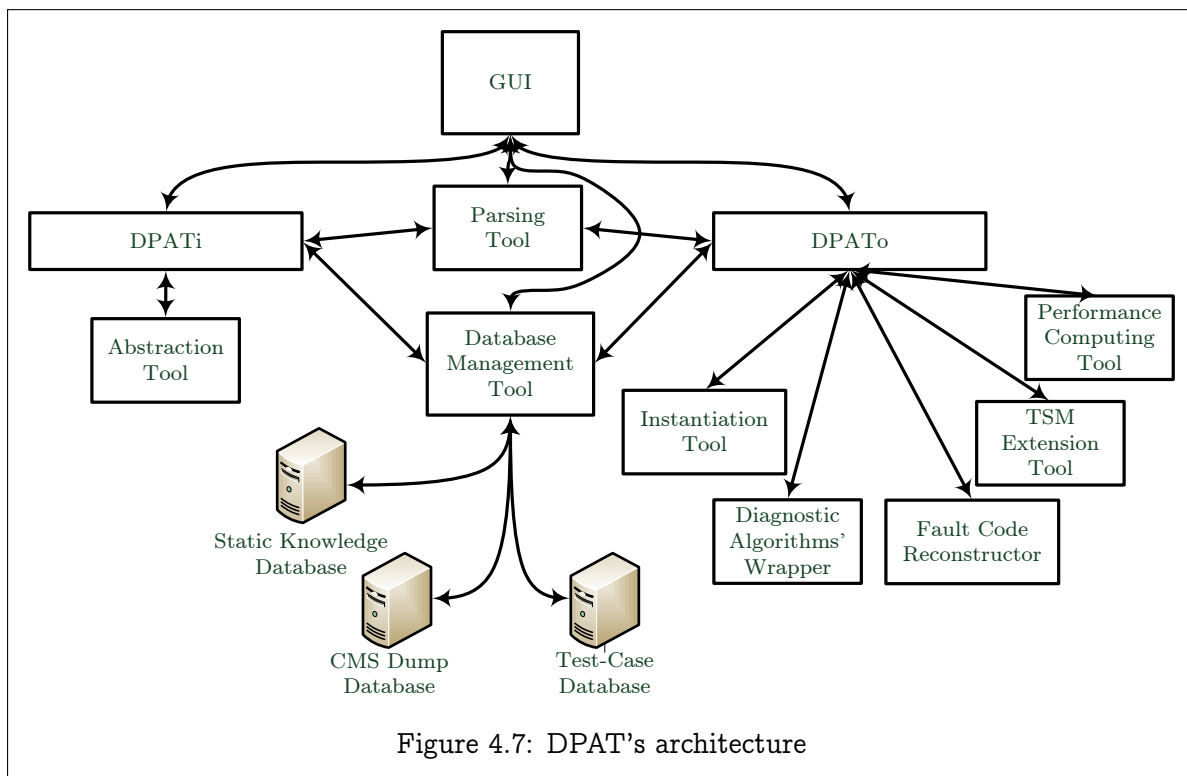
Global Effect diagnoses (cf. Definition 70) are diagnoses (cf. Definition 19), 2) diagnoses have a series of theoretically well defined properties whose absence one may consider abnormal (cf. Definitions 26 and 27), and 3) the properties of diagnoses are directly related to the properties of diagnostic systems' Models (cf. Subsection 2.4.2); then one can infer the "quality" of implemented Airbus diagnostic systems by analysing the properties of their computed specific diagnoses. This is how the problem of automatically validating and comparing different configurable Built-In Test Equipments/Centralized Maintenance System algorithms and knowledge bases and Troubleshooting Manual procedures performances using heterogeneous real-world multi-program will be tackled; and how DPAT (Diagnosis Performance Assessment Tool) will appear to the world.

In order to describe exactly how DPAT works, recall that the Centralized Maintenance System uses its own algorithms (rules of inference), knowledge bases (scientific and mathematical theories), Maintenance Messages (context-observations) and Global Effects (explananda-observations), among other observations, to produce a Post-Flight Report (aircraft-level automatic diagnosis). This being so, the term:

- "dynamic inputs" will refer to all the observations received by the Centralized Maintenance System, i.e. Maintenance Messages, Aircraft Effects and so on.
- "static inputs" will refer to all the Centralized Maintenance System scientific and mathematical theories, i.e. Tuning Files, Troubleshooting Manual files, and so on.
- "CMS dump" will refer to the CMS internal logs containing, among others, 1) all the dynamic inputs received during a given number of flights, and 2) the configuration of the CMS static inputs at each time.
- "test-case" will refer to: 1) a set of dynamic inputs emitted during a test situation in Airbus' common framework of diagnosis and/or the automatic diagnosis produced by the CMS algorithm used, 2) for each Global Effect that is part of the dynamic inputs, the [partial] conjunction of abnormalities that caused such effect, 3) an identification of the test situation from which dynamic inputs were retrieved, and 4) a set static inputs.

In a nutshell, DPAT's main functions enable: 1) the generation of test-cases using some aircraft-program-specific dynamic inputs (gathered from real flight data) and the [partial] conjunction of abnormalities that caused each Global Effect (gathered from mechanics' reports), 2) the simulation of test-cases using every Centralised Maintenance System algorithm and knowledge base (gathered from Airbus' software repository), and 3) the computation and comparison of a set of performance criteria over each simulation result. DPAT uses the framework of Section 4.1 for the generation of standard test-cases, for the translation of test-cases into every CMS algorithm specific input and for the translation of Post-Flight Reports into diagnoses, according to its definition in Chapter 2. This enables the computation of some standard performance criteria, whose exact definition will be given in a few pages.

The remainder of this section will be devoted to present DPAT's architecture and detailed modular view; as well as to using DPAT to determine the performance of some implemented diagnostic systems using real-world data. As it was the case with CONCKER, DPAT was built with the help of Boulaabi [20], Stella [99] and Oh [83]; three Airbus interns.



4.3.2 DPAT's architecture

Exactly in the same way as I presented MEDITO and CONCKER, let me start by focusing on DPAT's architecture.

As depicted in Figure 4.7, DPAT is composed of twelve interacting modules, where:

- the *GUI* module provides a Graphical User Interface for handling user inputs and launching performance computation.
- the *Database Management Tool* module is responsible for managing and querying DPAT's H2 and Oracle databases.
- the *Parsing Tool* module converts all the raw input data into Oracle, H2 Database Engine or java objects; and converts all the computed Post-Flight Reports into the common framework of aircraft-level diagnosis (cf. Section 4.1).
- the *DPATi* module is responsible for creating DPAT test-cases.
- the *Abstraction Tool* module abstracts the aircraft-program-specific dynamic inputs so they can be exploited by every Airbus diagnostic algorithm.
- the *DPATo* module is responsible for computing and comparing the performance of every aircraft-level diagnosis generated by a set of diagnostic algorithms and test-cases.
- the *Instantiation Tool* module is the dual of the *Abstraction Tool* module. It instantiates a test-case into a set of aircraft-program-specific dynamic inputs adapted to a diagnostic algorithm to be tested.
- the *Diagnostic Algorithms' Wrapper* is responsible for launching a set of selected diagnostic algorithms seen as black boxes.

- the *Fault Code Reconstructor* builds the MM-explanation of every Maintenance Message in the computed and parsed Post-Flight Report. It uses an aircraft-level interpretation of MM-explanation according to the common framework of aircraft-level diagnosis presented in Subsection 4.1.3.
- the *TSM Extension Tool* module builds the TSM-explanation associated to every Maintenance Message in the computed and parsed Post-Flight Report. It uses an aircraft-level interpretation of TSM-explanation according to the common framework of aircraft-level diagnosis presented in Subsection 4.1.3.
- the *Performance Computing* module computes aircraft-level diagnoses' performance.

This being so, DPAT can be separated in four big blocks, with three of them directly visible in Figure 4.7:

1. the “Utilities” block, where the databases are managed, all the parsing is done and the User Interface is managed thanks to the *Parsing Tool*, *Database Management Tool* and *GUI* modules.
2. the “Specific-knowledge Generation” block, where the knowledge base of the Diagnosis Performance Assessment Tool is built offline.
3. the “Test-case Generation” block, where test-cases are built thanks to the *DPAT_i* and *Abstraction Tool* modules, and
4. the “Performance Assessment” block, where the performance indicators are computed and grouped into performance reports thanks to the *DPAT_o*, *Instantiation Tool*, *Diagnostic Algorithms' Wrapper*, *Fault Code Reconstructor*, *TSM Extension Tool* and *Performance Computing Tool* modules.

Hereafter I provide a detailed description of each block followed by the usage of DPAT with some real-world data.

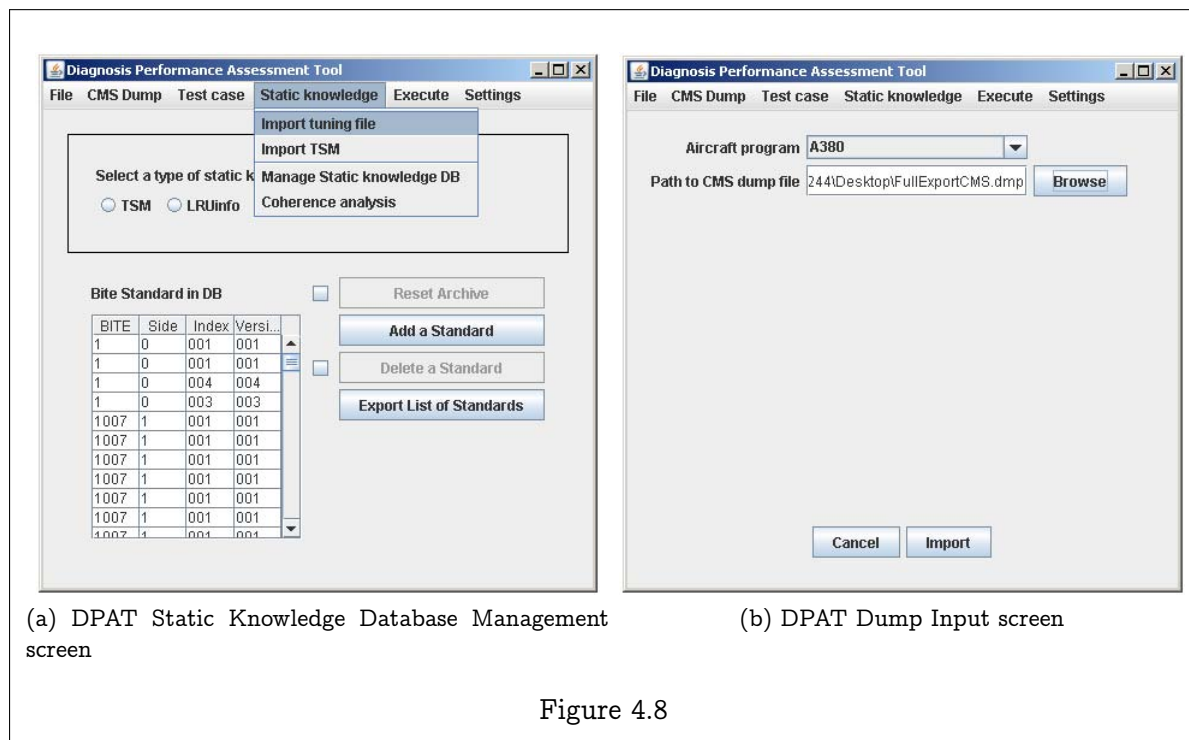
4.3.3 “Utilities” block

As stated above, DPAT's utilities block is concerned with the management of databases, of the graphical user interface and the parsing of numerous DPAT files. As with CONCKER, the *Parsing Tool* module will not be analysed for no conceptual value would be added.

DPAT's GUI module

DPAT's Graphical User Interface is divided in numerous screens enabling: 1) the injection, edition and removal of data into and from DPAT's databases, 2) the computation of all the performance criteria needed, 3) the configuration of the consistency analysis as in CONCKER, and 4) the display of DPAT's results.

First, as shown in Figure 4.8a CONCKER's *Data Input*, *Database Management* and *Consistency Checker* screens are replicated in DPAT; for DPAT's static knowledge database schema and executable functions are the same as CONCKER's. With respect to DPAT's



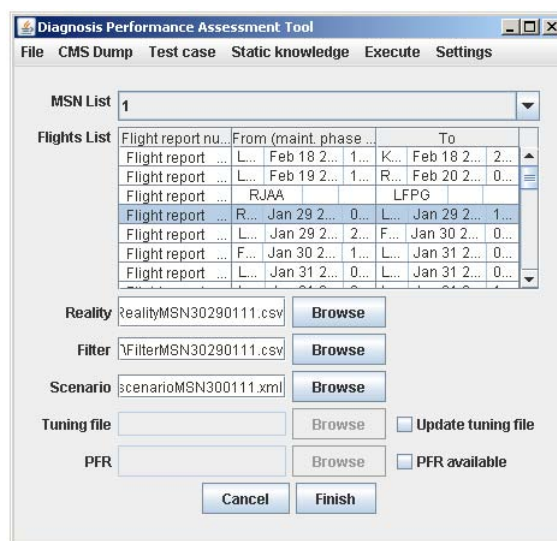
original screens, the first one is the *Dump Input* screen (cf. Figure 4.8b), enabling the user to introduce a CMS dump from any aircraft program into DPAT's CMS Dump database.

From these screens, let us move to the *Test-Case Input* screen (cf. Figure 4.9), enabling the user to 1) define a test-case based on all the flights in all the CMS dumps, and 2) inject the selected test-case in DPAT's Test-Case database. Naturally, a *Test-Case Removal* screen also exists, enabling the user to remove test-cases from the homonymous database.

With the screens enabling the injection, edition and removal of data into and from DPAT's databases, it is time to focus on the data usage. DPAT's execution menu takes the user directly to the *Test-Case Selection* screen, depicted in Figure 4.10a. Using this screen, the user is able to 1) select a set of test-cases from DPAT's Test-Case database, 2) define if the original Post-Flight Report sent by the on-board CMS shall also be used for performance comparison (if available), and 3) define if the TSM-extension step shall be performed; among others. Note that, in this version of DPAT, the diagnostic algorithms to be compared are hard-coded.

In the cases where the user determines that a TSM-extension shall be performed, she is presented with the *TSM Extension* screen, depicted in Figure 4.10b. In this screen, the user is able to select a series of TSMs as well as the types of extension to be performed ⁵. Finally, the user is presented with the results computed by DPAT in the *Performance Comparison* screen. This screen, depicted in Figure 4.15, will be presented further in this dissertation for comprehension purposes.

⁵The user may select if the TSM-related explanation shall replace or be concatenated to the MM-related explanation.

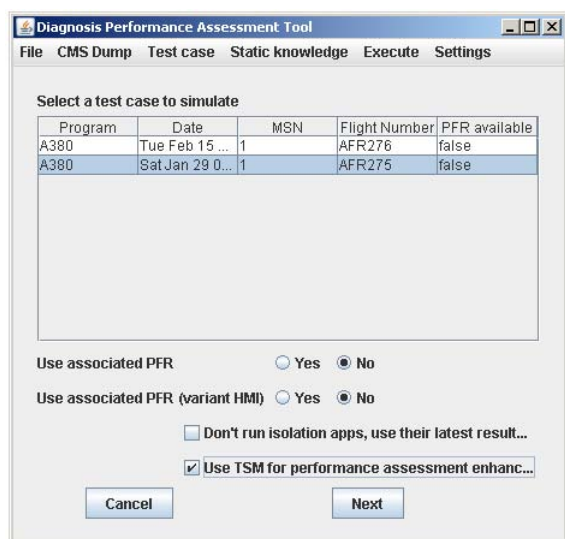


The screenshot shows the 'Diagnosis Performance Assessment Tool' window. The 'MSN List' dropdown is set to '1'. The 'Flights List' table contains the following data:

Flight report nu...	From (maint. phase ...	To
Flight report ...	L... Feb 18 2... 1...	K... Feb 18 2... 2...
Flight report ...	L... Feb 19 2... 1...	R... Feb 20 2... 0...
Flight report ...	RJAA	LFPG
Flight report ...	R... Jan 29 2... 0...	L... Jan 29 2... 1...
Flight report ...	L... Jan 29 2... 2...	F... Jan 30 2... 0...
Flight report ...	F... Jan 30 2... 1...	L... Jan 31 2... 0...
Flight report ...	L... Jan 31 2... 0...	L... Jan 31 2... 0...

Below the table, there are input fields for 'Reality' (RealityMSN30290111.csv), 'Filter' (FilterMSN30290111.csv), and 'Scenario' (scenarioMSN300111.xml), each with a 'Browse' button. There are also fields for 'Tuning file' and 'PFR', each with a 'Browse' button and a checkbox for 'Update tuning file' and 'PFR available' respectively. 'Cancel' and 'Finish' buttons are at the bottom.

Figure 4.9: DPAT Test-Case Input screen

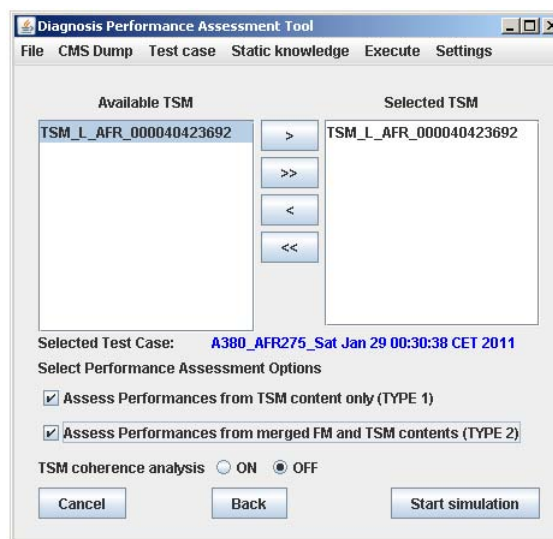


The screenshot shows the 'Diagnosis Performance Assessment Tool' window. The 'Select a test case to simulate' section contains a table with the following data:

Program	Date	MSN	Flight Number	PFR available
A380	Tue Feb 15 ...	1	AFR276	false
A380	Sat Jan 29 0...	1	AFR275	false

Below the table, there are radio buttons for 'Use associated PFR' (Yes/No) and 'Use associated PFR (variant HMI)' (Yes/No). There are also checkboxes for 'Don't run isolation apps, use their latest result...' and 'Use TSM for performance assessment enhanc...'. 'Cancel' and 'Next' buttons are at the bottom.

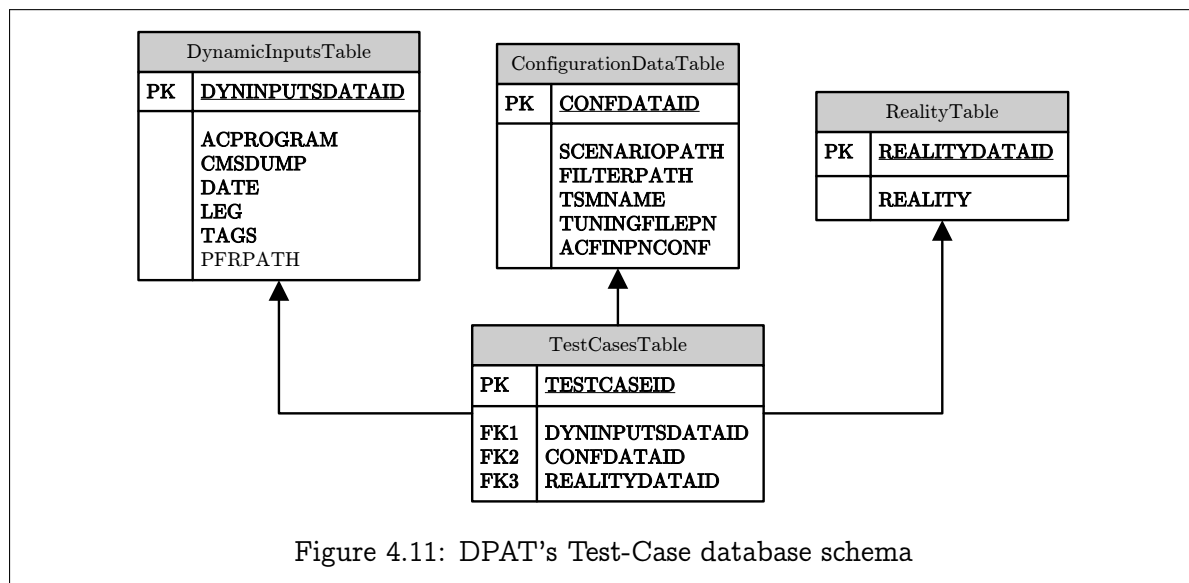
(a) DPAT Test-Case Selection screen



The screenshot shows the 'Diagnosis Performance Assessment Tool' window. The 'Available TSM' list contains 'TSM_L_AFR_000040423692'. The 'Selected TSM' list also contains 'TSM_L_AFR_000040423692'. Below the lists, the 'Selected Test Case' is 'A380_AFR275_Sat Jan 29 00:30:38 CET 2011'. There are checkboxes for 'Assess Performances from TSM content only (TYPE 1)' and 'Assess Performances from merged FM and TSM contents (TYPE 2)'. There are radio buttons for 'TSM coherence analysis' (ON/OFF). 'Cancel', 'Back', and 'Start simulation' buttons are at the bottom.

(b) DPAT TSM Extension screen

Figure 4.10



DPAT's Database Management Tool module and database schemas

With respect to the *Database Management Tool* (DMT) module, it is responsible for managing the interaction between DPAT's modules and its CMS Dump, Test-Cases and Static Knowledge databases. Concerning this last database, its schema is exactly the same as that of CONCKER's database; for DPAT needs exactly the same data as CONCKER for achieving its goal. We remind the reader that such database schema is depicted in Figure 4.5. As for the CMS Dump databases, there is a database schema for each aircraft program; schemas that will not be presented in this book for confidentiality purposes. Finally, the Test-Case database schema is presented in Figure 4.11.

Concerning such database, note that it standardises the notion of test-case presented before. Moreover, I want to attract the reader's attention to the "TAGS" field in the "DynamicInputsTable"; as it handles the tags produced by the abstraction of aircraft-program-specific dynamic inputs in the "Specific-knowledge Generation" block presented hereafter.

4.3.4 "Specific-knowledge Generation" block

As stated in the beginning of the present section, DPAT aims at automatically validating and comparing different configurable Built-In Test Equipments/Centralized Maintenance System algorithms and knowledge bases and Troubleshooting Manual procedures' performances. From the many technical problems that appear when one tries to achieve such goal, I hereafter treat the first one: since each CMS algorithm can, in general, be designed to handle observations and scientific and mathematical theories with different syntaxes and semantics; DPAT needs some means to convert any aircraft-program-specific data so it can be injectable into any CMS diagnostic algorithm. This is done with the help of both the common framework of subsystem-level diagnosis (cf. Subsection 4.1.2) and the "specific-knowledge table" presented in a few paragraphs.

The DPAT block to generate the specific-knowledge table is an offline manual module divided in four steps to be followed in a linear manner:

1. Specific knowledge identification,
2. Specific knowledge common tagging,
3. Specific knowledge table initialization, and
4. Common tag cross-identification.

Step 1: Specific knowledge identification

As stated above, the first step when building a specific-knowledge table is to identify specific knowledge in each diagnostic algorithm to be analysed in DPAT; where specific knowledge is considered to be any information the diagnostic algorithm needs to retrieve in its observations and scientific and mathematical theories.

To illustrate this concept, suppose a diagnostic algorithm evaluating a condition such as: if (Emitter BITE = ID) then In this case, Emitter BITE = ID is an example of specific knowledge.

Step 2: Specific knowledge common tagging

With every diagnostic algorithm's specific knowledge identified, the second step into the construction of the specific-knowledge table is to determine a complete base of common tags transversal to every diagnostic algorithm. As for common tags, they are defined as cross-algorithm functional abstractions of specific knowledge. Accordingly, every specific knowledge identified in step 1 has to be associated to a common tag; and two pieces of information in two diagnostic algorithms that are functionally equivalent shall share the same tag.

An example of a common tag is Emitter BITE = LGERS BITE, which could be mapped in two different diagnostic algorithms, for instance, as Emitter BITE = 10 and Emitter BITE = 98.

Step 3: Specific knowledge table initialization

After the association of common tags with the specific knowledge previously identified, the third step to generate a specific-knowledge table is to gather these data in a "specific-knowledge table" (cf. Figure 4.12), associating common tags, diagnostic algorithms and specific knowledge, and containing:

- a "common tag" column, where common tags are gathered one per row.
- one "diagnostic algorithm x from a/c program p" column for each diagnostic algorithm present in DPAT; where the image (specific knowledge associated) of each common tag for a given diagnostic algorithm is introduced.
- a "specific for" column, with an indication of what diagnostic algorithms use each common tag as specific knowledge.

Common Tag	CMS Alg. X prog. p	...	CMS Alg. Y prog. q	Specific for
A	A ID in X			X
B	B ID in X		B ID in Y	X , Y
C			C ID in Y	Y

Figure 4.12: Specific knowledge table after Step 3

Step 4: Common tag cross-identification

At this point, the “specific-knowledge table” is only partly filled; having blank cells corresponding to diagnostic algorithms where a given common tag is not used. As so, the last step in the “Specific-knowledge Generation” block consists in filling those cells with: 1) The knowledge corresponding to the common tag, if there is a correspondence, or 2) “Not defined”, otherwise.

4.3.5 “Test-case Generation” block

The “Test-case Generation” block serves the purpose of building a multi-program test-case database necessary for DPAT to overcome the problem of using the dynamic inputs specific to a given aircraft program with a diagnostic algorithm of another aircraft program.

Relying on the previously generated specific-knowledge table, the DPAT block to generate test-cases is divided in the following three steps, linearly implemented in the *DPAT_i* module:

1. Test-case data selection,
2. Dynamic inputs abstraction, and
3. Test-case and test-data injection in the databases.

Step 1: Test-case gathering

The first step of the test-case generation block consists in selecting the data gathered from a given aircraft to be a part of the test-case. Note that if no dynamic inputs are present in a given test-case, i.e. automatic diagnosis based test-case, the following step 2 is to be skipped.

Step 2: Dynamic inputs abstraction

Test-cases contain, as defined above, a set of dynamic inputs specific for a given program. In order to produce a multi-program test-case database, these inputs have to be functionally abstracted using the specific-knowledge table generated through the “Specific-knowledge Generation” block and the *Abstraction Tool* module. This module abstracts a set of dynamic inputs collected from a given program *p* according to Algorithm 3.


```

for  $i = 1$  to  $\text{size}(\text{dynInputs})$  do
   $\text{element} \leftarrow \text{dynInputs}[i]$ 
  for  $j = 1$  to  $\text{size}(\text{attributes}(\text{element}))$  do
     $\text{attribute} \leftarrow \text{attributes}(\text{element})[j]$ 
    for  $k = 1$  to  $\text{height}(\text{specificKnowledgeTable})$  do
       $\text{row} \leftarrow \text{specificKnowledgeTable}[k]$ 
      if  $\text{matches}(\text{getAircraftProgramColumn}(\text{row}, p), \text{value}(\text{attribute}))$  then
         $\text{tag}(\text{attribute}, \text{getCommonTag}(\text{row}))$ 
      end if
    end for
  end for
end for

```

Note that the $\text{tag}(\cdot)$ function is responsible for associating a common tag to every dynamic input attribute identified in the specific-knowledge table.

Step 3: Test-case injection

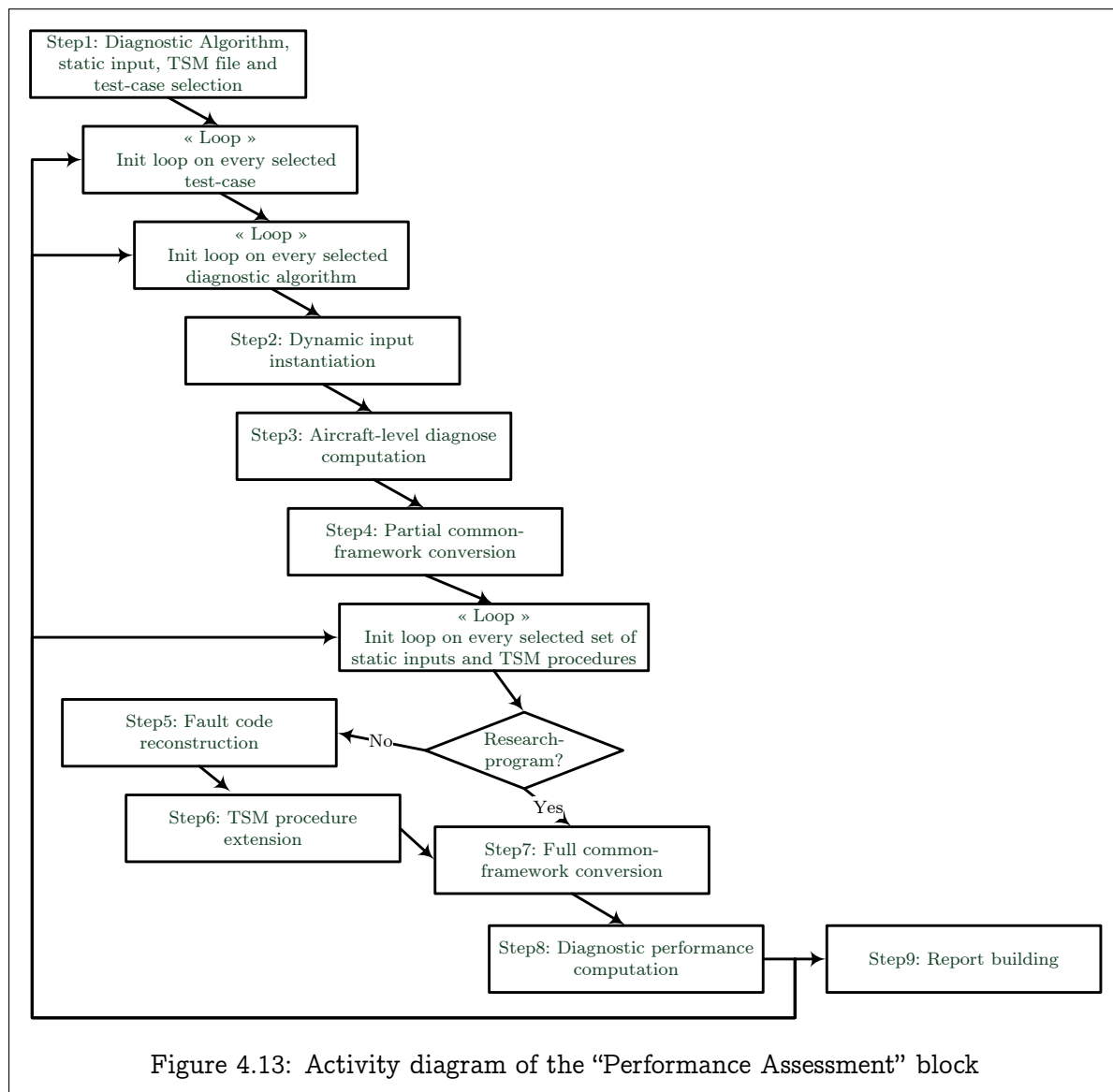
Following step 2, test-cases with abstract dynamic inputs are to be injected in the test-case database using the *Database Management Tool* module.

4.3.6 “Performance Assessment” block

Once a specific-knowledge table has been created using the block described in Subsection 4.3.4, and a database of test-cases has been created using the block described in Subsection 4.3.5; it is now time to focus on the “Performance Assessment” block, implemented in the *DPATo* module through the following steps:

1. Diagnostic algorithm, static input, TSM procedure and test-case selection,
2. Dynamic input instantiation,
3. Aircraft-level diagnose computation,
4. Partial common-framework conversion,
5. Fault code reconstruction,
6. Troubleshooting procedure extension,
7. Full common-framework conversion,
8. Diagnosis performance computation, and
9. Report building.

These steps are followed according to diagram 4.13.



Step 1: Inputs and options selection

The first step of the “Performance Assessment” block is the selection, through the *GUI* module, of the CMS algorithms, CMS knowledge bases and TSM procedures one wants to compare; as well as the test case one wants to use for the performance assessment.

This being so, DPAT users select:

1. Zero or more CMS algorithms (to be compared),
2. Zero or more sets of static inputs per CMS algorithm chosen (to be compared),
3. Zero or more TSM procedures (to be compared), and
4. One or more test-cases from the test-case database.

Note that if no dynamic inputs are present in a given test-case, i.e. automatic diagnosis based test-case, the following steps 2 and 3 are to be skipped.

Step 2: Dynamic inputs instantiation

With CMS algorithms, static inputs, TSM procedures and test-cases selected, the second step in DPAT “Performance Assessment” block is the instantiation of test-cases dynamic inputs using the *Instantiation Tool* module. Straightforwardly, for every selected diagnostic algorithm of a given program q , dynamic inputs instantiation goes according to Algorithm 4.

```

crossReferenceFile  $\leftarrow \emptyset$ 
for  $i = 1$  to  $\text{size}(\text{dynInputs})$  do
  element  $\leftarrow \text{dynInputs}[i]$ 
  for  $j = 1$  to  $\text{size}(\text{attributes}(\text{element}))$  do
    attribute  $\leftarrow \text{attributes}(\text{element})[j]$ 
    if  $\text{isTagged}(\text{attribute})$  then
      tag  $\leftarrow \text{getCommonTag}(\text{attribute})$ 
      for  $k = 1$  to  $\text{height}(\text{specificKnowledgeTable})$  do
        row  $\leftarrow \text{specificKnowledgeTable}[k]$ 
        if  $\text{matches}(\text{getCommonTag}(\text{row}), \text{tag})$  then
          if  $\text{matches}(\text{getSpecificFor}(\text{row}), q)$  then
            tempDyn  $\leftarrow \text{value}(\text{attribute})$ 
            tempSub  $\leftarrow \text{select}(\text{getAircraftProgram Column}(\text{row}, q))$ 
             $\text{value}(\text{attribute}) \leftarrow \text{tempSub}$ 
             $\text{store}(\text{crossReferenceFile}, \text{tag}, \text{tempDyn}, \text{tempSub})$ 
          end if
        end if
      end for
    end if
  end for
end if
end for
end for

```

By and large, the objective of the algorithm is to convert dynamic inputs from its origin program X to the framework of a CMS algorithm from a program Y ; and tracing all the data conversion in a temporary file (*crossReferenceFile*).

Step 3: Aircraft-level diagnoses computation

The automatic diagnoses computation step consists in running the CMS algorithms selected in Step 1 as black boxes configured with the static inputs selected in that same step, and having as inputs the instantiated dynamics inputs from Step 2. This is done using the *Diagnostic Algorithms' Wrapper* module.

Step 4: Partial common-framework conversion

Following the computation of aircraft-level diagnoses specific to each diagnostic algorithm, the Partial common-framework conversion step uses the *Parsing Tool* module to gather: a) every Global Effect and the associated Fault Origin Maintenance Messages from the Post-Flight Report, in the A380/A400M/A350 case; or b) every Global Effect and the associated kernel diagnosis from the Post-Flight Report, in the research-program case.

Step 5: Fault code reconstruction

As for DPAT's fault code reconstruction step, it uses the *Fault Code Reconstructor* module for building a first-order sentence representing a MM-explanation interpreted according to the common framework of aircraft-level diagnosis exposed in Subsection 4.1.3. Being an aircraft-level equivalent of CONCKER's homonymous module, it uses either the data in the BITEIdent, BITESideList, VersionArchive, MasterFMD, MasterBITEPer, TuningFileArchive, LRUInfo and DK2Table tables of DPAT's static knowledge database; or the equivalent data contained in the Post-Flight Report. Finally, this step is not executed in the research-program case for its computed diagnoses are already as required.

Step 6: Troubleshooting Manual procedures extension

Since every Maintenance Message points ⁶ to a given TSM procedure, each of which containing a TSM-explanation; one may extend an MM-explanation with the associated TSM-explanation. This is the objective of the TSM procedures extension step relying on the *TSM Extension Tool* module. This module, in turn, makes use of the framework of Subsection 4.1.3 and the *TSMArchive*, *TSMTaskToCauses*, *TSMFMDToTask*, *TSMTaskToTest*, TuningFileArchive, LRUInfo and DK2Table tables of DPAT's static knowledge database to extend MM-explanations with the associated TSM-explanations. Finally, this step is not executed in the research-program case.

Step 7: Full common-framework conversion

The objective of the "Performance Assessment" block's sixth step is to complete the mapping of aircraft-program-specific Post-Flight Reports into the common framework of aircraft-level diagnosis. This is done by using the *Parsing Tool* module and the results of Subsection 4.1.3 to produce a Global Effect diagnosis (cf. Definition 70). Finally, using the *crossReferenceFile*, all the data is converted back to the dynamic inputs' initial format.

⁶While, in theory, every Maintenance Message is associated to a Troubleshooting Manual procedure; as shown in Section 4.2 this is not always true in practice. As so, CONCKER must be used before DPAT if one wants to follow the TSM-procedure-extension step.

Step 8: Diagnosis performance computation

Regardless of the selected diagnostic algorithms, static inputs or TSM procedures, at this point one has a Global Effect diagnosis (cf. Definition 70) at her disposal. It is now time to focus on computing the performance of such diagnosis with the help of the *Performance Computing Tool* module. Note that the results that follow were agreed at Airbus in [12] and [60].

Before going any further, let me state that the performance criteria presented hereafter depend on the true health state. This, in turn, is recovered by transforming the Reality input file into a conjunction of AB-literals.

The *Performance Computing Tool* module implements a series of performance criteria based on the results of Chapter 2 and of the present chapter. The first two criteria implemented are exactly the degrees of validity (cf. Definition 26) and uncertainty (cf. Definition 27) already introduced in Chapter 2. Besides these, one more criterium is also implemented by the present module: the diagnostic distance; firstly introduced in the Airbus technical report [9].

As far as the diagnostic distance goes, it aims at providing some means to evaluate the quality of diagnoses with a single criteria. Even if this may seem an essential problem of a diagnostic system conceiver, literature lacks of answers to such challenge. Krysanter and Nyberg [68] try to compute both coverage and false coverage probabilities, Vachtsevanos, Lewis, Roemer, Hess and Wu [109] provide several metrics for implemented diagnostic systems mainly based on false positives, false negatives, time delays and percent isolation to one LRU, Baldi, Brunak, Chauvin, Andersen and Nielsen [8] give an overview of the existing methods to compute the accuracy of prediction algorithms for classification; however, although these methods for evaluating the performance of diagnoses are adequate in many situations, they do not provide the desired results for Airbus, simply because they fail to reach our goals in situations where multiple faults are present and the diagnosis covers many partially valid hypotheses.

Intuitively, the diagnostic distance is a fusion of the degrees of validity and uncertainty along an importance scale. Accordingly, the diagnostic distance between a diagnosis and its associated true health state is zero if and only if the diagnosis is valid and certain. Moreover, the diagnostic distance decreases with the degree of validity and increases with the degree of uncertainty. Furthermore, if two diagnoses have the same degree of validity, the one with the lower degree of uncertainty will be closer to the true health state. Finally, if two diagnoses have different degrees of validities, the one with the higher one will be closest to the true health state. Formally:

Definition 74 (Diagnostic Distance). *Let σ_T the true health state, let $\#(.)$ be the cardinality function and let $\text{Hamm}(a,b)$ the Hamming distance between a and b [52]. Moreover, let $D = \{d_1, \dots, d_n\}$ be a diagnosis such that:*

$$\forall_{i,j \in \mathbb{N}^2} i < j \Rightarrow \text{Hamm}(d_i, \sigma_T) \leq \text{Hamm}(d_j, \sigma_T)$$

Finally:

$$\bullet \lambda \stackrel{\text{def}}{=} 2^{\#(\text{COMPS})}.$$

- $\alpha \stackrel{def}{=} \#(COMPS)$.
- $\beta_i \stackrel{def}{=} Ham(m(d_i, \sigma_T))$.
- $\gamma \stackrel{def}{=} \#(D)$.
- $T = \gamma - 1$ iff $\beta_n = 0$, and $T = \gamma$ otherwise.
- $\zeta = 1$ iff $\beta_1 = 0$, and $\zeta = 0$ otherwise.

In this case, the diagnostic distance between D and σ_T , noted $DD(D, \sigma_T)$ is:

$$\left[1 - \left(\frac{\beta_1 - 1}{\alpha} - \frac{\lambda - \gamma}{\alpha \cdot \lambda} - \sum_{i=2}^T \frac{\beta_i - 1}{\alpha^i \cdot \lambda} - \frac{1}{\alpha^{\gamma \cdot \lambda \cdot (\gamma - T + 1)}} \right) \cdot (1 - \zeta) \right]$$

A full proof that the diagnostic distance is a distance varying in $[0, 1] \in \mathbb{R}$ is given in [9].

All in all:

- from invalid MM-related/TSM-related global-level diagnoses, DPAT can infer that either the BITE reports, CMS algorithm, the BITE/CMS knowledge bases or the TSM procedures are not ontologically true,
- from valid but uncertain automatic/end-to-end global-level diagnostic results, DPAT can infer that either the CMS algorithm, the BITE/CMS knowledge bases or the TSM procedures are not diagnosable,
- from the diagnostic distance of MM-related/TSM-related global-level diagnoses, DPAT can infer the overall best end-to-end diagnostic chain from monitoring to aircraft-level diagnosis.
- by comparing automatic and end-to-end computed diagnostic distances DPAT can detect errors in the TSM procedures, and so on.

Finally, concerning the isolation of abnormalities following their detection, this will be the object of Section 4.4.

Step 9: Report building

The report building step is responsible for: 1) the fusion of the degree of validity, degree of certainty and diagnostic distance computed using the selected combinations of test-cases, diagnostic algorithms, static inputs and TSM procedures; and 2) the logging of such results in a single performance report on the format of Figure 4.14.

	CMS Alg. X Program p with static input i and TSM j	...	CMS Alg. Y program q with static input k and TSM l
Fusion Precision			
Fusion Validity			
Fusion Diag. Distance			

Figure 4.14: Layout of a diagnosis comparison performance report

The fusion step is a weighted average of the performance criteria for every test case.

4.3.7 Assessing performance with DPAT

Since the beginning of this section and up to this point, I have provided the reader with a full description of DPAT's architecture and blocks. It is now time to use DPAT against real-world data to compare different configurable Built-In Test Equipments/Centralized Maintenance System algorithms and knowledge bases and Troubleshooting Manual procedures performances. The demonstration that follows has the objective of providing empirical evidence that DPAT can be used in practice to tackle our initial problems. Before doing so, I would like to state that only three Airbus diagnostic algorithms have been used in the present version of DPAT: the diagGraph algorithm, the CVT algorithm and the TSFT algorithm corresponding to the main algorithms in the research, A350 and A380 aircraft programs. However, in the following example the diagGraph results will be omitted.

The usage of DPAT starts with the insertion of data into its Static Knowledge database. This is done thanks to the Static Knowledge Database Management screen depicted in Figure 4.8a. For this demonstration three tuning files and one set of TSM procedures were added.

Following the population of the Static Knowledge database, let us move to populating the CMS Dump databases. This is done thanks to the *Dump Input* screen of Figure 4.8b. For this demonstration a single A380 CMS dump was provided as input.

Once the CMS Dump databases are populated, one is able to create test-cases by selecting, in the *Test-Case Input* screen (cf. Figure 4.9):

- a set of dynamic inputs corresponding to a flight among all the ones contained in the CMS Dump databases,
- a set of static inputs to be tested corresponding to a Tuning file ⁷, a Filter file and a Scenario file,
- an optional PFR file corresponding to an aircraft-level diagnosis already computed for such dynamic inputs (eg. the original PFR for the selected flight), and
- a Reality file containing the abnormalities that actually caused each Global Effect in the dynamic inputs.

With all the inputs provided, let us move to computing and comparing the performance of CVT and TSFT against one of the inserted test-cases. As depicted in Figure 4.10a, in this demonstration I have selected, in DPAT's *Test-Case Selection* screen, a test-case with an associated flight number. Moreover, I have chosen to perform a TSM extension as depicted in Figure 4.10b.

Before jumping to the results obtained, let me provide the reader with some information about the selected test-case. First, among the Global Effects present in the flight chosen, we will be comparing the performance of CVT and TSFT diagnosing the CDS CAPT+F/O EFIS CTL PNLs FAULT Aircraft Effect. Moreover, the fault of the AESU-1 was the real cause behind the failure that originated this effect.

⁷Note that if no Tuning file is selected, the Tuning file corresponding to the one effectively used in the selected flight will be automatically selected.

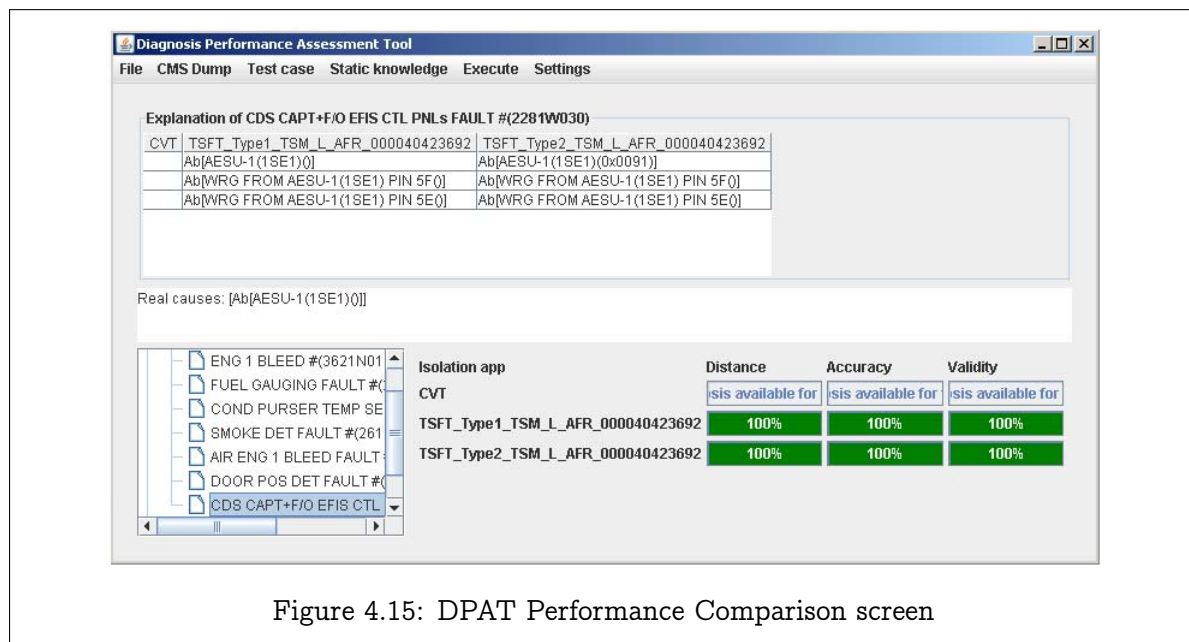


Figure 4.15: DPAT Performance Comparison screen

As illustrated in the upper part of Figure 4.15, CVT failed to find an explanation for this Global Effect. As for TSFT, both TSM-related aircraft-level diagnoses for the Global Effect CDS CAPT+F/O EFIS CTL PNLs FAULT were that either the abnormality of the LRU AESU-1, the abnormality of the wiring connected to the AESU-1 pin 5F or the abnormality of the wiring connected to the AESU-1 pin 5E explained this Global Effect. The application of the performance criteria exposed in the previous subsection explains the obtained Distance, Accuracy and Validity results. However, two comments are important at this point. First, the Accuracy indicator illustrated in Figure 4.15 corresponds to the degree of certainty, i.e. 1 - degree of uncertainty, of each aircraft-level diagnosis. Secondly, one may have been expecting the accuracy and diagnostic distance to be slightly less than 100% since, after all, the computed diagnoses were uncertain. This was in fact the case. However, due to the number of components in the perimeter of the Global Effect, both the degree of uncertainty and the diagnostic distance were very close to zero and were rounded in by DPAT's GUI.

4.4 Meta-diagnosing Airbus Models with MEDITO

In the previous sections I have accomplished the objectives of providing Airbus with a common framework of diagnosis and with some methods and tools for checking local (subsystem-level) consistency between Maintenance Messages and Troubleshooting Manual procedures and for validating and comparing configurable Built-In Test Equipments/Centralized Maintenance System algorithms and knowledge bases and Troubleshooting Manual procedures performances using heterogeneous real-world multi-program data.

It is now time for me to concentrate on the last objective of this chapter, that is, on the detection and isolation of Model abnormalities in the Centralised Maintenance System

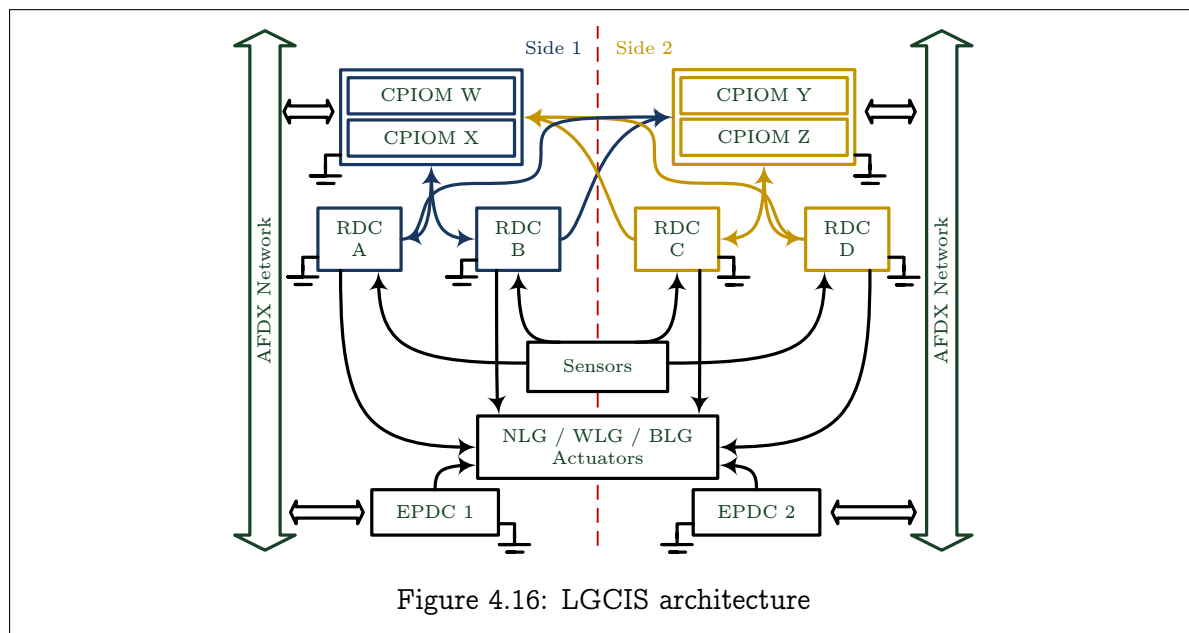
observations, knowledge bases and algorithms. If on the introductory chapter achieving this objective seemed a daunting task, this is far from being true at this point. In fact, since the results of Chapter 3 offer us a method and tool to detect and to isolate Model abnormalities in implemented diagnostic systems, provided they are defined in the framework of Chapter 2; we only need to represent Centralised Maintenance System observations, knowledge bases and algorithms in such framework. Courtesy of Section 4.1 this is now an easier task.

In the remainder of this section I will illustrate how this representation can be done through an example. Hence, I will focus on the problem of detecting and isolating abnormalities in the Centralised Maintenance System description of a Landing Gear Extension and Retraction System (LGERS); while assuming that Centralised Maintenance System's believed system observations are ontologically true and its diagnostic algorithms are sound and complete. Once more, some data in the present section has been altered for confidentiality issues.

4.4.1 An Airbus landing gear extension and retraction system

The purpose of the Airbus A380 Landing Gear Extension and Retraction System (LGERS) is to provide controlled Landing Gear extension and retraction and door opening/closing such that the aircraft can perform a normal landing and cruise flight. Structurally speaking, an A380 landing gear consists of: a retracting Nose Landing Gear (NLG), two retracting Wing Landing Gears (WLG), and two retracting Body Landing Gears (BLG). In terms of the associated actuators, all the landing gears (NLG, WLG and BLG) have retract, unlock and gear uplock actuators; and the associated doors are, for the most part, hydraulically operated through some hydraulic actuators, being the remaining doors mechanically operated.

As for the Landing Gear Control [and Indication] System (LGCIS), it serves the primary purpose of detecting and controlling the position of the landing gears, doors and uplocks. Moreover, its simplified general architecture is depicted in Figure 4.16, where component real names and some details have been changed/removed for confidentiality issues.



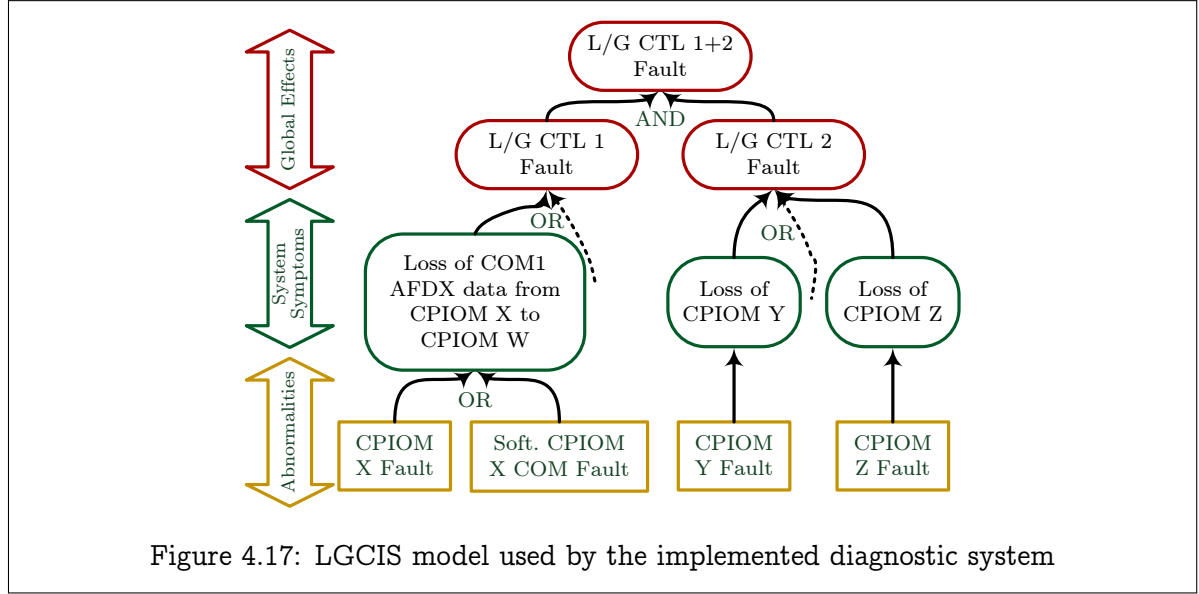
First of all, one shall note that LGCIS is decomposed, for safety purposes, into two redundant sides executing, by and large, the same functions in alternate active/standby modes.

In terms of sensor data, each LGCIS side has its own sensor inputs which are transmitted to the Core Processing Input and Output Modules (CPIOM) - that can be seen as aircraft generic computers - through some Remote Data Concentrators (RDC) by means of Arinc 429 data links.

Using such sensor data and the control commands provided by the landing gear control lever (not represented in Figure 4.16), CPIOM W (resp. Y) LGERS COM HIGH software computes and sends control signals via the AFDX network to the Electrical Power Distribution Centre 1 (resp. 2) controlling the actuators; and CPIOM X (resp. Z) LGERS COM LOW software computes and sends control signals through RDCs A and B (resp. C and D) to the actuators.

4.4.2 An Airbus LGERS believed system

From the brief description of an Airbus A380 LGERS given in the previous subsection, let us move to its representation in Airbus' diagnostic systems. Relying on each system description, such as the LGERS one provided in Subsection 4.4.1, and on BITE/FWS isolation capabilities, Airbus engineers build a CMS aircraft believed system. Suppose that a part of such believed system is depicted in Figure 4.17, where: 1) continuous arrows symbolize material implications, 2) dashed arrows symbolize unrepresented nodes present in the believed system, and 3) there is an underlying hypothesis that only the predecessors of a node imply it (completeness hypothesis).



So, for instance, the nodes L/G CTL 1 Fault, L/G CTL 2 Fault and L/G CTL 1+2 Fault as well as their edge relations are involved in the first-order sentence: $[L/G \text{ CTL } 1 \text{ Fault} = \top] \wedge [L/G \text{ CTL } 2 \text{ Fault} = \top] \Leftrightarrow [L/G \text{ CTL } 1+2 \text{ Fault} = \top]$; where the reader may notice the equivalence relation - instead of a material implication - due to the completeness hypothesis ⁸.

Finally, to provide the reader with some intuition on such model, we can see, for instance, that engineers expect LGCIS side 2 to be lost after a CPIOM Y fault, since as described before the LGERS COM HIGH software in this CPIOM computes non-redundant side 2 control inputs.

4.4.3 Meta-diagnosing an Airbus LGERS believed system with MEDITO

As mentioned before, reality is sometimes different from what one expects and engineers continuously struggle to produce ontologically true believed systems - a necessary (although insufficient) condition to assure correct component replacements (cf. Theorem 2). In this subsection, we provide an example of how to use MEDITO to automatically detect and isolate ontologically false sentences in LGERS SD.

Using MEDITO's *System manager* screen, the first thing to do is to provide MEDITO with the LGERS believed system one wants to meta-diagnose such as the one depicted in Figure 4.17 (cf. Figure 3.2a) ⁹.

Once this is done, observations and meta-observations are added in MEDITO's *Test-Cases manager* screen. In this example, a test-case was added, by using some [not] received FWS warnings: $OBS = \{LGERSCTL1+2FAULT=\perp, LGERSCTL1FAULT=\top, LGERSCTL2FAULT=\perp\}$; and some mechanic feedback on real-world system unit faults: $M-OBS = \{Ab(CPIOMW), \neg Ab(CPIOMX), \neg Ab(CPIOMZ), \neg Ab(SoftCPIOMXCOM)\}$.

⁸The symbol \top represents logical true; and the symbol \perp represents logical false.

⁹For space constraints it is impossible to present the reader with the full LGERS believed system.

Finally, in the *Meta Diagnoser* screen, the option “SD-sentences may be false” is selected.

In this example, MEDITO computed three kernel diagnoses for a fifteen-sentences SD in 29689 milliseconds; as well as three minimal cardinality diagnoses in 63 miliseconds¹⁰. The computed kernel meta-diagnoses were the following:

$$\{\text{SD-sentence2}\}, \{\text{SD-sentence8}\}, \{\text{SD-sentence4}\}$$

where:

$$\begin{aligned} \text{SD-sentence2:} \quad & [\text{LossCPIOMX}=\top] \vee [\text{LossCOM1AFDXDataFromXtoW}=\top] \\ & \vee \dots \Leftrightarrow [\text{LGERCTL1FAULT}=\top] \\ \text{SD-sentence4:} \quad & \text{Ab}(\text{CPIOMX}) \vee \text{Ab}(\text{SoftCPIOMXCOM}) \\ & \Leftrightarrow [\text{LossCOM1AFDXDataFromXtoW}=\top] \\ \text{SD-sentence8:} \quad & \text{Ab}(\text{CPIOMX}) \Leftrightarrow [\text{LossCPIOMX}=\top] \end{aligned}$$

Based on these results, one can see that either SD-sentence2, SD-sentence4 or SD-sentence8 are ontologically false. Since meta-diagnosis can rely on a monotonic logic under some hypotheses (those that we have been using), as discussed in Section 3.1.3, and since our meta-diagnosis engine is considered sound and complete with respect to the underlying logic, adding new meta-observations can only reduce the set of meta-diagnoses.

As so, the provided test-case was extended by adding some received BITE information to the set OBS: $\text{LossCOM1AFDXDataFromXtoW}=\perp$. As depicted in Figure 3.2b, the computed kernel meta-diagnoses were:

$$\{\text{SD-sentence4}\}$$

The reason why $\{\text{SD-sentence4}\}$ was ontologically false was later found: the abnormality of CPIOM W can also lead to the loss of COM 1 AFDX data from CPIOM X to CPIOM W; since the later CPIOM cannot acknowledge receiving the data. This was confirmed by changing SD-sentence4 to $\text{Ab}(\text{CPIOMX}) \vee \text{Ab}(\text{SoftCPIOMXCOM}) \vee \text{Ab}(\text{CPIOMW}) \Leftrightarrow [\text{LossCOM1AFDXDataFromXtoW}=\top]$; and noticing the resulting empty set of kernel meta-diagnoses.

4.5 Originality and related work

I started this chapter with the ambition of providing Airbus with: 1) a common framework of diagnosis for every studied Airbus aircraft program - A380, A400M, A350 and research - enabling the comparison of diagnostic methods and results, 2) a method and tool for checking local (subsystem-level) consistency between Maintenance Messages and Troubleshooting Manual procedures, 3) a method and tool for validating and comparing configurable Built-In Test Equipments/Centralized Maintenance System algorithms and knowledge bases and Troubleshooting Manual procedures performances using heterogeneous real-world multi-program

¹⁰Intel Core 2 Duo CPU at 2.4GHz and 2Gb of RAM

data, and 4) a method and tool to detect and to isolate Model abnormalities in the Centralised Maintenance System observations, knowledge bases and algorithms. I can now state that my goal was achieved, thus answering to all the industrial objectives presented in the beginning of this book.

Concerning the common framework of diagnosis, exposed in Section 4.1, it is to my knowledge an original work at Airbus; a fact supported by [11] and [10]. Moreover, it is a very important contribution to the Airbus diagnostic world. First, it clearly defines a bridge between the academic and the industrial world with formal vision on the expected local-level and aircraft-level Airbus diagnoses. Second, it provides a language enabling the common understanding between every diagnostic system conceiver at Airbus. Finally, it is a central piece enabling the other industrial contributions in this chapter.

From the introduction of a common framework of diagnosis we advanced, in Section 4.2, to tackle the problem of checking for local (subsystem-level) consistency between Maintenance Messages and Troubleshooting Manual procedures with CONCKER and its underlying method. Straightforwardly, I will not discuss the originality of this industrial contribution outside the Airbus scope, simply because it is essentially an answer to an Airbus-exclusive problem. As for the extent of this contribution, CONCKER (as far as my knowledge goes an original work at Airbus) is already being used by the Airbus departments responsible for producing Troubleshooting Manuals both for: 1) checking the consistency between a whole new batch of TSM procedures, and 2) establishing a priority among the TSM procedures to change when multiple inconsistencies are found for a given batch. This being so, CONCKER has the potential of saving airlines from TSM-procedure-error-generated delays and the correspondent economical and financial losses.

In the third section we introduced the reader to DPAT, and showed how to tackle the problem of automatically validating and comparing different configurable Built-In Test Equipments/Centralized Maintenance System algorithms and knowledge bases and Troubleshooting Manual procedures performances using heterogeneous real-world multi-program. As far as the whole DPAT goes, literature lacks of material in its domain; with the exception being the works of Kurtoglu, Narasimhan, Poll, Garcia, Kuhn, de Kleer, van Gemund and Feldman in [70] [69]. According to its authors, the objective of such works is to:

1. “define a standard representation format for the system description [the model of the system], sensor data, and diagnosis result”,
2. “develop a software run-time architecture that can run specific scenarios from actual system, simulation, or other data sources such as files (individually or as a batch), execute Diagnostic Algorithms, send scenario data to the DA at appropriate time steps, and archive the diagnostic results from the DA.”, and
3. “Define a set of metrics to be computed based on the comparison of the actual scenario and diagnosis results from the diagnostic algorithm”.

Despite being close to solving our technical problem described above, the works of Kurtoglu et al. do not address certain problems:

1. Contrary to our technical problem hypotheses, the CMS/BITE knowledge bases are considered to be ontologically true, complete and known *a priori*.

2. CMS algorithms have to be built to match the underlying framework and cannot be seen as a black-box by the system user. Consequently, no methods for transforming dynamic and static data from its original framework and the CMS algorithm framework are provided.
3. CMS algorithms are supposed to output diagnoses in a standardized format with limited underlying semantics. Consequently, no methods for transforming diagnoses into a common framework are provided.
4. TSM procedures are not addressed in Kurtoglu et al. framework. As so, no methods are provided to assess TSM procedures performance and ensure they are consistent with BITE/CMS knowledge bases.
5. The framework provided cannot handle semi-distributed diagnostic architectures such as the ones in Airbus or Boeing implemented diagnostic systems.

Moreover, concerning the performance metrics implemented in DPAT, they are an original contribution establishing a return on the algorithms, observations and scientific and mathematical theories through the results of Section 2.4. All in all, DPAT adds value both for airlines and for Airbus by enabling the improvement of diagnostic systems and diagnoses not only in the design and development phases of the aircraft life-cycle, but also in the post entry-in-service phase where new tests cases based on in-service observed real data can be used. This implies, among others: 1) savings for Airbus in the path towards implemented diagnostic systems' maturity, and 2) less potential flight delays and cancellations for airlines.

As for the problem of detecting and isolating abnormalities in Models of implemented Airbus diagnostic systems, it was tackled in Section 4.4 thanks to MEDITO. Accordingly, the originality of this contribution is strictly related to that of Section 3.3; with the difference being made through the application of MEDITO to an industrial problem.

Finally, the most important contribution of this whole chapter was providing empirical evidence that the academic and industrial worlds benefit from each others' contributions. In a nutshell, CONCKER, DPAT and MEDITO are three examples of Airbus tools that drink directly from the academic-world fountain.

Chapter 5

Generalising meta-systems

It ain't what you don't know that gets you in trouble. It's what you know for sure that just ain't so.

- Mark Twain

With the three previous chapters, Artificial Intelligence has now some means to reason about abnormalities in Models of implemented diagnostic systems and explanation is general; and Airbus is finally provided with CONCKER, DPAT and MEDITO, three tools with underlying methods addressing our industrial objectives. When all is said and done, it is time to focus once more on meta-diagnosis and explore some questions about meta-systems and their scope.

Before doing so, however, I would like to focus on logic; a formalism used since the beginning of this book for dissecting diagnosis and meta-diagnosis both in the academic and in the industrial worlds. The use of logic is sometimes a subject of numerous critics, mainly concerning, among others, 1) its lack of pragmatism motivated by an excessive concern with form, and 2) its lack of efficiency shown by the time-complexity issues of most logic-based solutions to Artificial Intelligence problems [81][16]. Although there is a hint of truth behind such points, note that logic provides a precise syntax and clear semantics for representing and reasoning, not content; and is exactly this content what defines the complexity of the underlying problems. As so, what some may call an excessive concern with form is exactly what enables logical approaches to clearly expose the premisses behind a given problem; premisses that are usually implicit in non-logical approaches solving such problems in an efficient manner.

Accordingly, and despite my preference in this dissertation for logic as a formalism for Model-based diagnosis and meta-diagnosis, I am aware that not every Model-based diagnostic approach is based on the roots of Artificial Intelligence, let alone logic (eg. Analytical Redundancy Relations' approaches to Model-based diagnosis which are based on control theory). Therefore, if in Chapter 3 it was claimed that meta-diagnosis was a general theory enabling the detection and isolation of abnormal properties in Models of implemented diagnostic systems

(and explanation in general); it is time to pay the price for such assertion and provide the reader with evidence supporting it. This will be done in the first section, by meta-diagnosing a non-logic-based meta-system: a diagnostic system based on Analytical Redundancy Relations (ARR), an important approach in the *FDI* Model-based diagnostic community.

The general character of meta-diagnosis is not restrained to the formalism on which meta-system are based. As the reader may recall, in the introductory chapter it was stated that, presenting a more than sufficient reason for one to embrace the study of Models, explanation was certainly far from being the only motivation; for Models play a central role in many other fields such as prediction, planning and learning. It is time to dive deep into this subject and provide evidence that meta-systems are not limited to implemented diagnostic systems; and meta-diagnosis can be used to detect and isolate abnormalities in Models of prediction, planning or learning systems. Hence, the second section of the present chapter will be devoted to meta-diagnosing an implemented abnormality prediction system.

As for the third section, it will be concerned with discussing the originality of the contributions provided in this chapter and establishing a relation with other works in literature.

5.1 Meta-diagnosing non-logic-based diagnostic systems

As stated above, the objective of the present section is to provide empirical evidence supporting meta-diagnosis claim of being a general theory for reasoning about Models in implemented diagnostic systems, these being logic-based or not. In order to do so, I will focus on meta-diagnosing implemented diagnostic systems built-up using the Analytical Redundancy Relation (ARR) approach from the *FDI* Model-based diagnostic community.

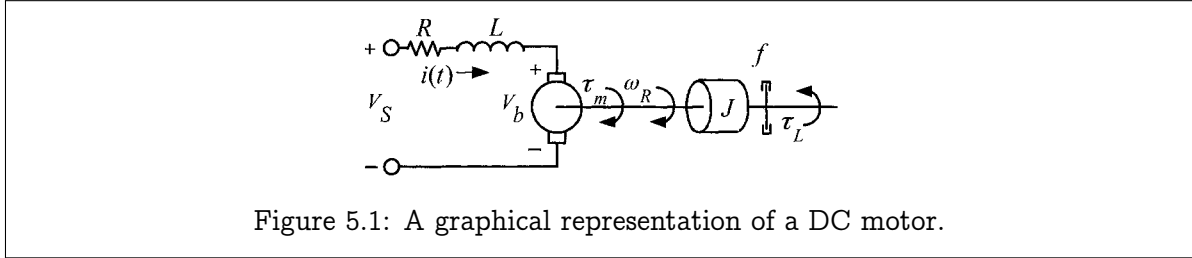
As for the organisation, in Subsection 5.1.1 the reader will be provided with an example of how an ARR-based scientific and mathematical theory can represent a DC motor, diagnosis-wise. Then, thanks to some observations, the appearance of invalid diagnoses in this approach will be illustrated, and the need for meta-diagnosis in ARR-based implemented diagnostic systems will find a motivation. As for Subsection 5.1.2, it will expose a *Bridge* (cf. [32] and [31]) between the *FDI* ARR-approach and the DX consistency-based logical approach; a *Bridge* that will be used to map our ARR-based diagnostic problem into the framework of Section 2.3. Such mapping enables the application, in Subsection 5.1.3, of our theory of meta-diagnosis introduced in Chapter 3. Finally, one will return to the *FDI* community's world and have such results explained and generalised. This will be the subject of Section 5.1.4.

5.1.1 When Models hurt ARR-based MBD: a DC motor trial

The present subsection is devoted to illustrating how abnormalities in Models of ARR-based implemented diagnostic systems can affect the computed diagnoses, thus providing a basis example for the remainder of this section. In order to do so, I will start by building an ARR-based scientific and mathematical theory representing a DC motor, and then move to exposing how a diagnostic system based on this theory fails to compute a valid diagnosis.

Modelling a DC motor

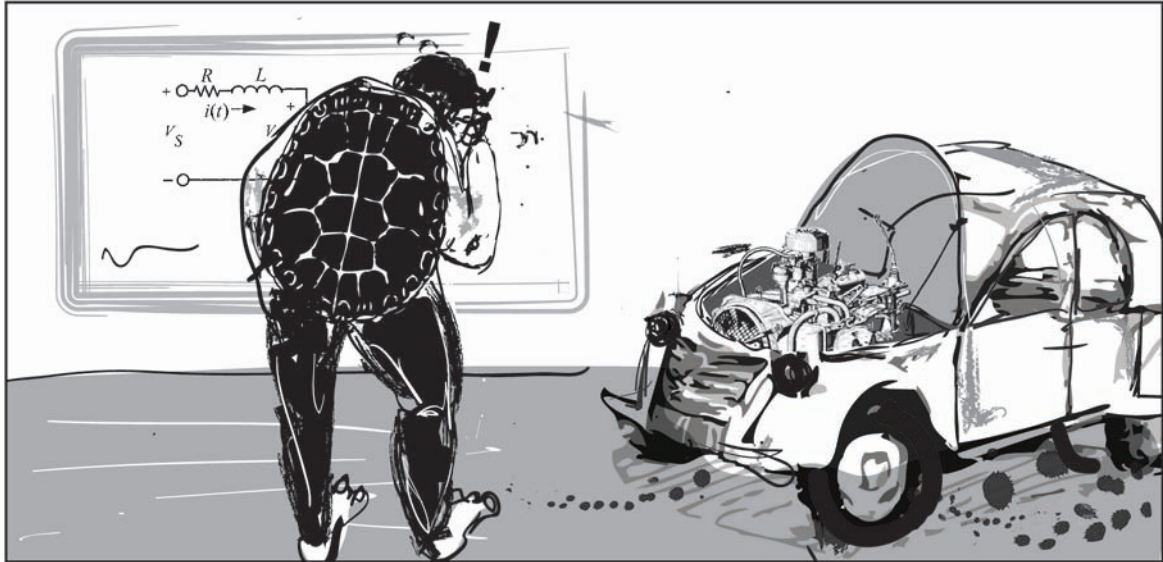
Imagine that, after a series of experiences with his new car, Mr Tortuga decided to spend some nights building a diagnostic system for the DC Motor start-engine. Suppose, for the sake of the argument, that he decided to use an FDI analytical redundancy approach. Assume Tortuga perceives the motor as depicted in Figure 5.1.



Straightforwardly, from the electrical part of this circuit he can extract the equations:

$$\begin{aligned} V_L &= V_S - R \cdot i - K_v \cdot \omega_R \\ \frac{d(i)}{d(t)} &= -\frac{R}{L} \cdot i - \frac{K_v}{L} \cdot \omega_R + \frac{V_S}{L} \end{aligned} \quad (5.1)$$

where R is the resistance of the electrical wiring of the rotor loop, L is its inductance, V_L is the voltage across L , V_S is the drive voltage, i is value of the current in the rotor loop, ω_R is the shaft angular velocity and K_v is the velocity constant determined by the flux density, the reluctance of the core, and the number of turns of the armature winding. Moreover, admit R and L are modelled as constants.



As for the mechanical part of the circuit, suppose it is represented by the following equation:

$$\frac{d(\omega_R)}{d(t)} = \frac{K_T}{J} \cdot i - \frac{f}{J} \cdot \omega_R - \frac{\tau_L}{J} \quad (5.2)$$

where J is the rotor moment of inertia, f is the coefficient of viscous friction, τ_L is the load and K_T is the torque constant determined by the flux density, the reluctance of the core, and the number of turns of the armature winding. Again, suppose J and f are modelled as constants.

Moreover, imagine there is a set of sensors placed for diagnosis purposes so that the drive voltage V_S , the voltage across L , the angular velocity of the shaft ω_R and the current i can be measured. Furthermore, admit that Tortuga predicts possible faults in the rotor causing changes in the resistance and inductance of the electrical wiring in the rotor loop, in the rotor moment of inertia, in the coefficient of viscous friction or in the torque constant/velocity constant, resp. f_R , f_L , f_J , f_f , and f_K .

Hypothesizing no electromagnetic losses entails that electrical and magnetic powers equal each other and $K_v = K_T = K$. From all this information, the following state-space representation can be built:

$$\begin{pmatrix} \dot{i} \\ \dot{\omega}_R \end{pmatrix} = \begin{pmatrix} -\frac{R}{L} & -\frac{K}{L} \\ \frac{K}{J} & -\frac{f}{J} \end{pmatrix} \cdot \begin{pmatrix} i \\ \omega_R \end{pmatrix} + \begin{pmatrix} \frac{1}{L} & 0 \\ 0 & -\frac{1}{J} \end{pmatrix} \cdot \begin{pmatrix} V_S \\ \tau_L \end{pmatrix}$$

$$\begin{pmatrix} i_{obs} \\ \omega_{R_{obs}} \\ V_{L_{obs}} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ -R & -K \end{pmatrix} \cdot \begin{pmatrix} i \\ \omega_R \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} V_S \\ \tau_L \end{pmatrix}$$

Suppose also that $R = 1\Omega$, $L = 0.5H$, $K = 0.01N.m/A$, $J = 0.01kg.m^2/s^2$, $f = 0.1N.m.s$ and $\tau_L = 0$.

The next step into building a diagnostic system using an analytical redundancy approach is the computation of the Analytical Redundancy Relations (ARR) of interest. Imagine that Tortuga determined the following ARR's, where the first three correspond to a motor with a positive drive voltage and the fourth one corresponds to a motor in an open circuit:

ARR1 r_1 where $r_1 = V_{L_{obs}} - [V_S - R \cdot i(t) - K \cdot \omega_R(t)]$

ARR2 r_2 where $r_2 = i_{obs} - i(t)$

ARR3 r_3 where $r_3 = \omega_{R_{obs}} - \omega_R(t)$

ARR4 r_4 where $r_4 = \omega_{R_{obs}} - \omega_R(t)$

As for their structures, ARR1 has a structure $\{f_L, f_R, f_K\}$, ARR2 has a structure $\{f_L, f_R, f_K, f_J, f_f\}$, ARR3 has a structure $\{f_L, f_R, f_K, f_J, f_f\}$ and ARR4 has a structure $\{f_K, f_J, f_f\}$

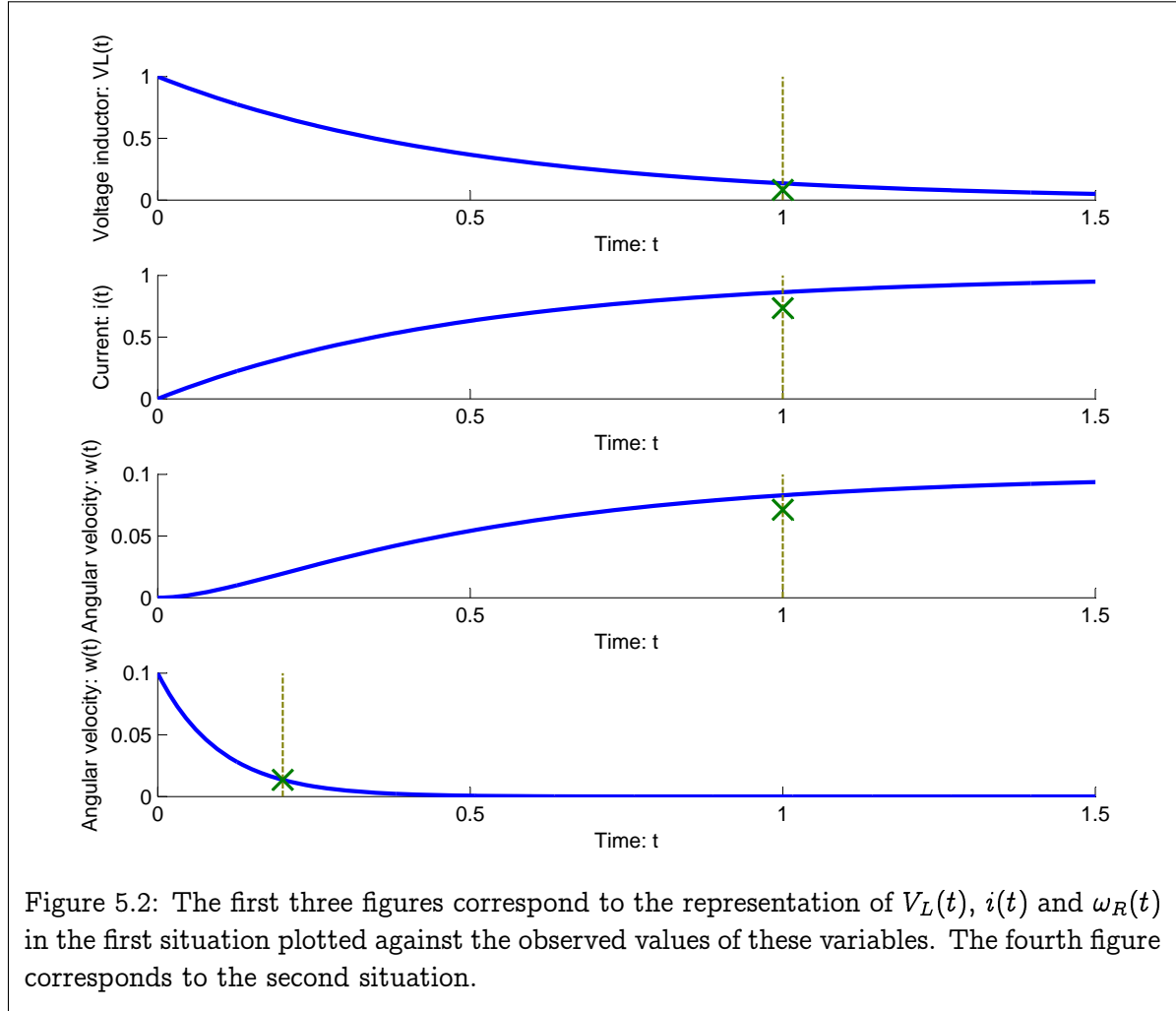
In this case, when rejecting both the exoneration and the no-compensation assumptions [32] the fault signature matrix was determined to be:

	f_L	f_R	f_K	f_J	f_f
ARR1	x	x	x	0	0
ARR2	x	x	x	x	x
ARR3	x	x	x	x	x
ARR4	0	0	x	x	x

This is the representation that will be used for diagnosis.

Diagnosing the DC motor

All the necessary elements to tackle the problem are now gathered. Suppose that, one fine day, the DC motor was started with a drive voltage of 1V. Then, 1s after its start, V_L , i and ω_R were measured to be, respectively, 0.0822V, 0.7348A and 0.0713rad/s. Some minutes later, admit the DC motor was put in an open circuit and that 0.2s after this, ω_R was measured to be 0.0135rad/s. Figure 5.2 illustrates these four measurements (crosses) and depicts the predicted values of those same variables.



Based on such observations, the vector of residuals was computed to be $[1,1,1,0]^T$. Note that, for the sake of simplicity, “measurement noise”, “model uncertainties” and so on were not taken into account. Nevertheless, this has absolutely no impact on the generality of the obtained results; for I could easily have rendered the present example slightly more complex through the introduction of such factors in the representation used for diagnosis. Going back to the DC motor, the consequent diagnosis of the computed vector of residuals, assuming the existence of no faults or repairs between observations, is: $\{f_L\}$, $\{f_R\}$, $\{f_K\}$ or any multiple fault

resulting as a combination of at least these faults.

Finally, suppose that when Tortuga returned back to his house in order to repair his vehicle he checked for these three faults, but everything was found to be normal. He is now facing a meta-diagnosis problem that needs to be solved: why does the diagnostic system determine some faults to be present while his own perception is that every component is normal?

5.1.2 From an FDI approach to a DX one: transforming the problem

A precondition for computing meta-diagnoses (cf. Chapter 3) is having a problem formalised in the framework of diagnosis provided in Chapter 2, a framework usually referred to, in the literature, as a DX logical framework. As so, the present subsection is devoted to transforming the FDI ARR-based framework of the previous section into a DX logical one. This will be done with the help of *Bridge* (cf. [32] and [31]).

According to *Bridge*, the conversion of a diagnostic system in an ARR-based framework into a logic-based one consists in transforming ARR's and measurements into believed systems and observations. First, the set COMPS may be defined as $\{L, R, K, J, f\}$, semantically representing the inductor, resistor, rotor moment of inertia, torque/velocity constant and coefficient of viscous friction "components". Moreover, SD can be designed by adapting the structures and the residuals in the analytical redundancy relations. The following sentences, extended with the appropriate axioms for arithmetic, the equations describing i and ω_R and so on, determine SD.

$$\begin{aligned} S_1: & \text{Closed}(\text{circuit}) \wedge \neg \text{Ab}(L) \wedge \neg \text{Ab}(R) \wedge \neg \text{Ab}(K) \Rightarrow (V_{L_{obs}} - [V_s - R \cdot i - K \cdot \omega_R] = 0) \\ S_2: & \text{Closed}(\text{circuit}) \wedge \neg \text{Ab}(L) \wedge \neg \text{Ab}(R) \wedge \neg \text{Ab}(K) \wedge \neg \text{Ab}(J) \wedge \neg \text{Ab}(f) \Rightarrow (i_{obs} - i = 0) \\ S_3: & \text{Closed}(\text{circuit}) \wedge \neg \text{Ab}(L) \wedge \neg \text{Ab}(R) \wedge \neg \text{Ab}(K) \wedge \neg \text{Ab}(J) \wedge \neg \text{Ab}(f) \Rightarrow (\omega_{R_{obs}} - \omega_R = 0) \\ S_4: & \text{Open}(\text{circuit}) \wedge \neg \text{Ab}(K) \wedge \neg \text{Ab}(J) \wedge \neg \text{Ab}(F) \Rightarrow (\omega_{R_{obs}} - \omega_R = 0) \end{aligned}$$

The reader may notice that sentence S_1 corresponds to *ARR1*, and so on. This will be exploited later in this section. As for the observations, they are two sets associated to the two situations mentioned before:

$$\begin{aligned} \text{OBS}_1 &= \{\text{Closed}(\text{circuit}), t = 1, V_s = 1, V_{L_{obs}} = 0.0822, i_{obs} = 0.7348, \omega_{R_{obs}} = 0.0713\} \\ \text{OBS}_2 &= \{\text{Open}(\text{circuit}), t = 0.2, \omega_{R_{obs}} = 0.0135\} \end{aligned}$$

Kernel consistency-based diagnoses can then be computed for both the diagnostic problems $\text{DP}_1 = (\text{SD}, \text{COMPS}, \text{OBS}_1)$ and $\text{DP}_2 = (\text{SD}, \text{COMPS}, \text{OBS}_2)$, using the same assumptions as in the previous section, i.e. rejecting both the exoneration and no-compensation hypotheses (a default in consistency-based approaches). These are $\text{KD}_1 = \{\{\text{Ab}(L)\}, \{\text{Ab}(R)\}, \{\text{Ab}(K)\}\}$ and

$KD_2 = \emptyset$. Finally, assuming that no faults or repairs occurred between both situations, the final consistency-based kernel diagnosis is: $KD = \{\{Ab(L)\}, \{Ab(R)\}, \{Ab(K)\}\}$; the exact same result as the one obtained in FDI's analytical redundancy approach. This comes with no surprise since as affirmed in [32], when no fault models are present - and they never are in the present FDI analytical redundancy approach - and when "releasing the exoneration and the no-compensation assumptions", then "FDI and DX views agree on diagnoses".

5.1.3 Meta-diagnosis

At this point, let us stop for a moment and re-gain a high-level vision of this section. I started this section by stating that the theory of meta-diagnosis introduced in Chapter 3 could be used to reason about abnormalities in Models of non-logic-based implemented diagnostic systems. Then, I motivated the need for meta-diagnosis in the FDI community. This was illustrated with a DC motor example in Subsection 5.1.1. When this stage was reached, I stated that a precondition for computing meta-diagnoses using the framework of Chapter 3 was having a problem formalised in a DX logical framework. As so, the previous section was devoted to transforming the FDI analytical-redundancy-based depart framework into a DX consistency-based logical one. It is finally the time to meta-diagnose our DC motor diagnostic system. In order to do so, let me recall the modelling process explained in Section 3.2:

1. Identify the meta-system, i.e. the diagnostic system to be meta-diagnosed and its assigned meta-goals.
2. Identify the meta-components and their meta-goals based on the meta-system meta-goals, on diagnostic system's and diagnoses' properties and on their relations, and on the detail level wanted for the solution.
3. Describe meta-components and meta-components' normal/abnormal behaviour on believed meta-systems.
4. Identify meta-observables based on the believed meta-system description.


In Tortuga's DC motor case, one is looking for the reason behind the invalid diagnoses computed. Moreover, for the sake of the argument, imagine that Tortuga assumes the implemented diagnostic algorithm computing diagnoses is sound and complete and that the implemented believed system observations are ontologically true. Hence, he is confronted with the typical problem of Subsection 3.2.1, i.e. with "the problem of ontologically false believed system descriptions". This being so:

1. The meta-system is made of:
 - the implemented diagnostic system $S = (SD, COMPS, OBS, A)$; where SD , $COMPS$ and OBS are those of the previous subsection.
 - the meta-goal of computing valid proof-theoretic consistency-based diagnoses.
2. Using Corollary 4 and the problem hypotheses, the meta-components can be chosen to represent every implemented SD -sentence, whose meta-goal is their ontological truth.
3. Every SD -sentence s represented by a meta-component S is modelled, at a meta-system level as: $\neg M\text{-}Ab(S) \Rightarrow s$.

4. From Corollary 4 and the problem hypotheses we get that if every meta-component is normal then the D_{P-T} is valid. To assess D_{P-T} validity we need to meta-observe σ_T and take into account that $D_{M-T} = D_{P-T}$ in our conditions.

In a nutshell, meta-components can be chosen as $M-COMPS = \{S_1, S_2, S_3, S_4\}$; and the following meta-system description, $M-SD$, can be built:

$$\begin{aligned} \neg M-Ab(S_1) &\Rightarrow [Closed(circuit) \wedge \neg Ab(L) \wedge \neg Ab(R) \wedge \\ &\quad \wedge \neg Ab(K) \Rightarrow (V_{L_{obs}} - [V_s - R \cdot i - K \cdot \omega_R] = 0)] \\ \neg M-Ab(S_2) &\Rightarrow [Closed(circuit) \wedge \neg Ab(L) \wedge \neg Ab(R) \wedge \\ &\quad \wedge \neg Ab(K) \wedge \neg Ab(J) \wedge \neg Ab(f) \Rightarrow (i_{obs} - i = 0)] \\ \neg M-Ab(S_3) &\Rightarrow [Closed(circuit) \wedge \neg Ab(L) \wedge \neg Ab(R) \wedge \\ &\quad \wedge \neg Ab(K) \wedge \neg Ab(J) \wedge \neg Ab(f) \Rightarrow (\omega_{R_{obs}} - \omega_R = 0)] \\ \neg M-Ab(S_4) &\Rightarrow [Open(circuit) \wedge \neg Ab(K) \wedge \neg Ab(J) \wedge \\ &\quad \wedge \neg Ab(f) \Rightarrow (\omega_{R_{obs}} - \omega_R = 0)] \end{aligned}$$

extended with the appropriate axioms for arithmetic, the equations describing $i(t)$ and $\omega_R(t)$ and so on. As for meta-observations, they contain all the observations at system-level. Moreover, **this theory is extended by taking into account the unobserved abnormality** in the three components analysed by Tortuga: L, R and K. As so, 

$$\begin{aligned} M-OBS_1 &= \{\neg Ab(L), \neg Ab(R), \neg Ab(K), Closed(circuit), t = 1, V_s = 1, V_{L_{obs}} = 0.0822, \\ &\quad i_{obs} = 0.7348, \omega_{R_{obs}} = 0.0713\} \\ M-OBS_2 &= \{\neg Ab(L), \neg Ab(R), \neg Ab(K), Open(circuit), t = 0.2, \omega_{R_{obs}} = 0.0135\} \end{aligned}$$

With a meta-system and some meta-observations the consistency-based kernel meta-diagnoses for both meta-diagnostic problems $M-DP_1 = (M-SD, M-COMPS, M-OBS_1)$ and $M-DP_2 = (M-SD, M-COMPS, M-OBS_2)$ can now be computed. These are $M-KD_1 = \{M-Ab(S_1)\}$ and $M-KD_2 = \emptyset$. Finally, since it is assumed that no changes in the Model occurred between both situations, then the final consistency-based kernel meta-diagnosis is: $M-KD = \{M-Ab(S_1)\}$. In a nutshell, the meta-diagnosis tells us that the first sentence in the believed system description must be corrected.

5.1.4 Back to an FDI approach

In the previous subsection, a consistency-based kernel meta-diagnosis was computed. However, what does that mean in terms of the elements in FDI's analytical redundancy framework: ARR's and measurements?

The answer to such question relies on the fact that, under some assumptions, believed system description sentences are perfectly equivalent to ARR's residuals and their structure. Recall, once again, the words of [32]: "Releasing the exoneration and the no-compensation

assumptions (...) FDI and DX views agree on diagnoses". As also stated in that work, this is true whenever one is not interested in fault models; for the present analytical redundancy approach cannot handle these objects. Since these were exactly the assumptions made all along the present section (and these assumptions can always be made when one wants to meta-diagnose FDI analytical redundancy approach Models) then the result from the previous section tells us that ARR1 needs to be repaired, either in terms of its structure or in terms of its residual definition.

Using this result, Mr Tortuga inspected the residual in ARR1, i.e. $r_1 = V_{L_{obs}} - [V_s - R \cdot i(t) - K \cdot \omega_R(t)]$ and, with the help of Mr Zapato's knowledge about cars, he found out the problem. In fact, since he worked during the night, the DC motor diagnostic system was always tested in low temperatures. Unfortunately, when he used the car in a hot sunny day when high temperatures were registered, the resistance of the resistor changed. This happened because, being made of iron ¹, the resistor has an electrical resistance temperature coefficient of, by and large, 0.0062K^{-1} . When changing the residual equation of ARR1 to $r_1 = V_{L_{obs}} - [V_s - R_0 \cdot (1 + \alpha \cdot (T - T_0)) \cdot i(t) - K \cdot \omega_R(t)]$ with $R_0 = 1\Omega$, the test temperature $T_0 = 5\text{C}$ and the temperature registered in the region where the drive took place $T = 45\text{C}$; Tortuga verified that, in fact, the implemented diagnostic system provided the correct answers. The Model was then assumed as being correct once again. Figure 5.3 extends Figure 5.2 with the new Model to provide a visual representation of what has been stated.

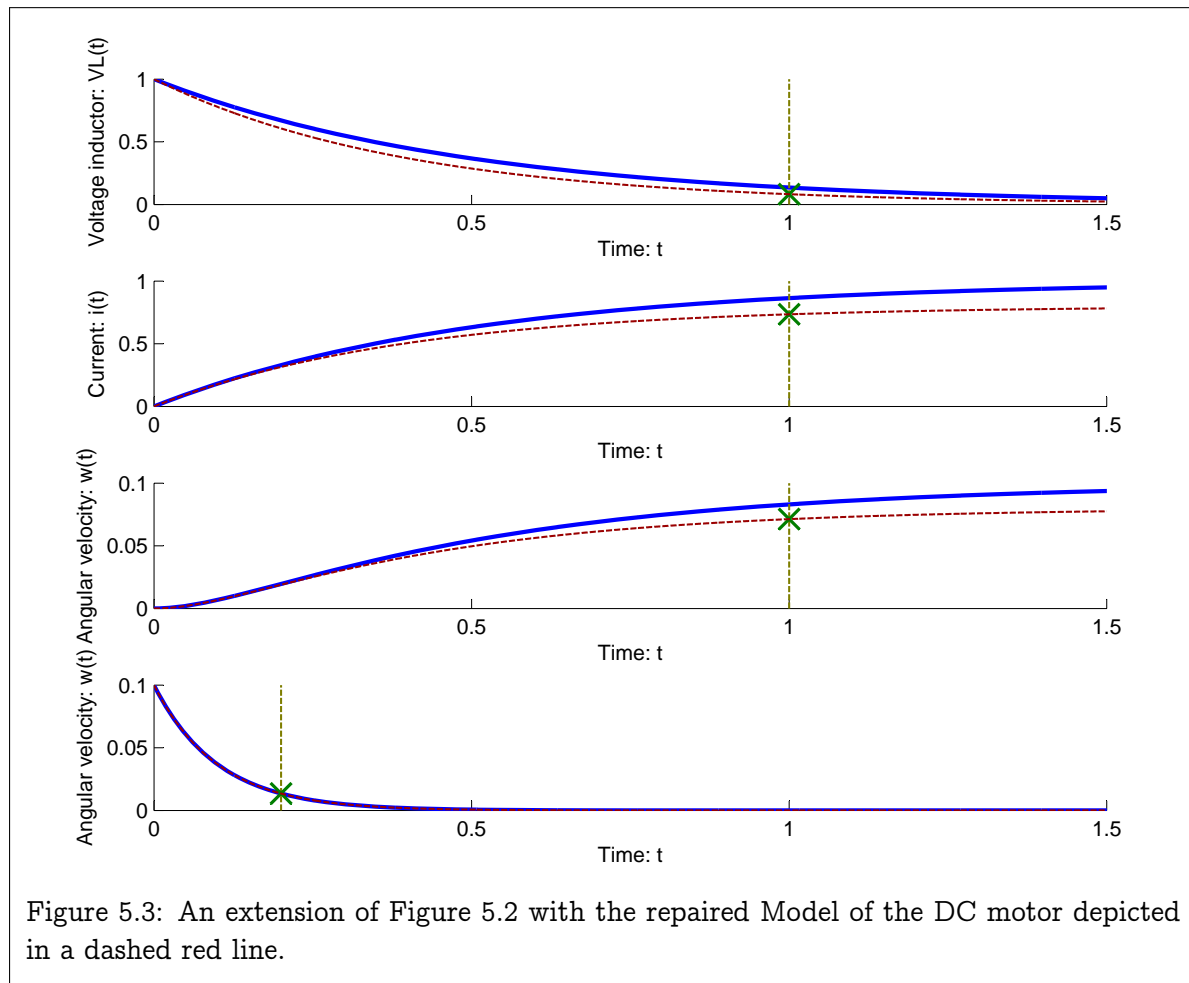
All in all, with the correct detection and isolation the ontologically false Analytical Redundancy Relation in the DC motor's diagnostic system, empirical evidence has been provided supporting meta-diagnosis claim of being a general theory for reasoning about Models in implemented diagnostic systems, these being logic-based or not.

5.2 Meta-diagnosing other meta-systems

In the previous section, I tried to convince the reader that the general character of meta-diagnosis is not restrained to the formalism on which diagnostic systems are based. This time, the second objective of this chapter will be addressed, namely, to provide evidence that meta-systems are not limited to implemented diagnostic systems, and meta-diagnosis can be used to detect and isolate abnormalities in Models of prediction, planning or learning systems. I will do so by focusing on an implemented abnormality prediction system, and meta-diagnose it.

As for the organisation, in Subsection 5.2.1 a logical framework of Model-based abnormality prediction will be exposed; a framework complemented with a series of abnormality prediction systems' and abnormality predictions' properties and property relations exposed in Subsection 5.2.2. These are the necessary conditions for studying the meta-diagnosis of such meta-systems. From this basis, the need for meta-diagnosing Models of implemented abnormality prediction systems will be illustrated through a DC motor example. Finally, in

¹Although electrical wires in the coils of a DC motor armature are usually made of copper, in this example iron was used to present a uniform framework for the whole section.



Subsection 5.2.4, the DC motor abnormality prediction believed system will be successfully meta-diagnosed, hence providing empirical evidence that meta-diagnosis can be used to detect and isolate abnormalities in Models of meta-systems other than implemented diagnostic systems.

5.2.1 Characterising abnormality predictions

In order to introduce the meta-diagnosis of implemented abnormality prediction systems, the characterisation of these entities, as well as of abnormality predictions, is needed. This characterisation will be profoundly based on the work of Subsection 2.3.2.

First, recall a statement from the introductory chapter: “If we see explanation as a reasoning process where, by and large, some observations over a period of time and some scientific and mathematical theories are used to determine a description of the real-world at some earlier time, then we can see: prediction as a task where some observations over a period of time and some scientific and mathematical theories are used to determine a future description of the real-world (...)”. Thus, both scientific and mathematical theories, observations and the description

of the real-world can be seen as a common factor between explanation and prediction; and believed systems (cf. Definition 9) and believed system observations (cf. Definition 15) come as natural pieces of an abnormality prediction system. The first two Models of Model-based abnormality prediction are, hence, a gift from Model-based diagnosis. Recall the notation of section 2.3:

- SD , is the believed system description,
- $COMPS$, are the believed system components,
- P , are the believed system parameters,
- $BP \subseteq P$, are the boundary believed system parameters,
- CXT , the believed system contexts (where each element m of CXT maps a partition $\{IP, OP\}$ of BP),
- OBS_P , are the believed system parameter observations (observations about boundary system-parameters), and
- OBS_{CXT} , the believed system context observations.

This will be the exact same notation used in Model-based abnormality prediction. Note, however, that since the notion of context and its added value has already been exposed in along this dissertation, for the sake of simplicity a single context will be assumed hereafter.

With both the notions of believed system and believed system observations re-introduced, it is time to focus on the differences between diagnosis and abnormality prediction, i.e. past and future. In explanation we say that some *explanans* explain some explananda-observations if the latter can be inferred from the former and the context-observations, both at some earlier time, along with some scientific and mathematical theories. As for prediction, we say that some *predictions* are predicted by some predicta-observations if the former can be inferred from the latter and the context-observations, both at some earlier time, along with some scientific and mathematical theories. This being so, as the reader may note, explanation and prediction are two faces of the exact same coin; and if A is an explanan of B , then B is a prediction of A ². From this vision, consistent with the work of Poole in [86] let us move into what constitutes an abnormality prediction problem:

Definition 75 (Abnormality prediction problem). *APP, an abnormality prediction problem, is a tuple $(SD, COMPS, OBS, HYP)$.*

Note that while diagnostic problems depend on nothing but observations, in abnormality prediction problems one may also want to select those predictions of believed system components normal and abnormal behaviour where some observables take a given value. This is the object of the theory HYP:

Definition 76 (Prediction hypotheses). *The prediction hypotheses, HYP, is a finite set of first-order sentences. Semantically, prediction hypotheses represent [future] observations (cf. Principle 4).*

As for the solutions of abnormality prediction problems they rely, as it was the case with diagnostic problems, on health states (cf. Definition 18). It is time to commit to a logic behind

²For a discussion on what causes are, the reader is encouraged to refer to [92].

abnormality prediction. As it was the case with Model-based diagnosis, I will be using first-order logic with equality; for it is already defined in literature and has an expressive power necessary to tackle the examples hereafter. Once again and as discussed in Section 2.3, note that this choice shall not be seen as a hard directive toward the choice of a logic and the reader is encouraged to refer to the works of Shoham [97], Shanahan [95] or Thielscher [108]. This being said, an abnormality prediction is defined as follows:

Definition 77 (Abnormality prediction). *Let $APP = (SD, COMPS, OBS, HYP)$ be an abnormality prediction problem to be solved, and let $\{HYP_{pred}, HYP_{cons}\}$ be a partition of HYP . An abnormality prediction for APP is the smallest non-empty set AP of health states (called, in this context, abnormality prediction hypotheses) σ such that:*

$$SD \cup OBS_{CXT} \cup OBS_P \cup HYP_{cons}$$

is satisfiable, and

$$SD \cup OBS_{CXT} \cup OBS_P \cup HYP_{cons} \models HYP_{pred} \cup \{\bigvee_{\sigma \in AP} \sigma\}.$$

A series of natural consequences arise from this definition:

1. “abduction-based” shall be substituted by that of “deduction-based”; for no abduction takes place in abnormality prediction.
2. “consistency-based” abnormality predictions are those corresponding to the situation where $HYP_{pred} = \emptyset$; while, otherwise, we are presented with “deduction-based” abnormality predictions.
3. the notions of partial (cf. Definition 20), kernel (cf. Definition 21), model-theoretic and proof-theoretic diagnosis (cf. Definition 23) find a perfect equivalent in the abnormality-prediction framework.

As so, the third Model can be defined:

Definition 78 (Abnormality prediction algorithm). *An abnormality prediction algorithm, noted A , is a algorithm with an underlying set of inference rules aiming at computing abnormality predictions (cf. Definition 77).*

Before ending this subsection, note the concept of abnormality prediction system:

Definition 79 (Abnormality prediction system). *An abnormality prediction system is a tuple $(SD, COMPS, OBS, HYP, A)$.*

5.2.2 Defining and relating properties in abnormality prediction systems and abnormality prediction results

Up to this point, a framework of abnormality prediction has been defined. Hence, in the same way a series of diagnostic systems’ and diagnoses’ properties and property relations were exposed in Chapter 2 so they could be meta-diagnosed; such will also be done in the abnormality-prediction-wise.

With respect to abnormality prediction systems’ properties, since believed system descriptions, believed system observations, diagnostic problems, diagnostic algorithms and diagnoses

have a perfect equivalent in the abnormality prediction side; then so do their properties. In a nutshell, every property exposed in Subsection 2.4.1 is both applicable to diagnostic systems and to abnormality prediction systems (naturally moderated by the difference in both concepts).

Concerning property relations, things are not so easy; for they depend on the concepts of explanation and prediction themselves. Therefore, I will not aim at providing an extensive account on abnormality prediction systems' property relations and will, instead, focus on nothing but those theorems that will be used in the remainder of the present section.

Now, abnormality prediction systems are typically built with objective in mind: computing valid abnormality predictions in every situation. Therefore, the necessary conditions to fulfil such objective have to be stated; conditions which are yet another consequence of the insights provided by Theorem 1. The following three theorems address such issue.

Theorem 8. *If SD is an ontologically true believed system description, then for every abnormality prediction problem $(SD, COMPS, OBS, HYP)$ with ontologically true believed system observations and prediction hypotheses:*

1. *every consistency-based model-theoretic abnormality prediction AP_{M-T} is valid.*
2. *some deduction-based model-theoretic abnormality prediction AP_{M-T} are not valid.*

Proof.

1. First, if SD , OBS and HYP are ontologically true, then $SD \cup OBS \cup HYP$ is satisfiable for, by Definition 1, SD , OBS and HYP share at least a model \mathcal{M} such that $\exists_{\mathcal{P}, \mathcal{R} \in \Omega} \exists_{\Gamma(\mathcal{M}, \mathcal{P})} (\mathcal{P} \subseteq \mathcal{R}) \wedge (\mathcal{R} \models \Psi)$. Suppose that the consistency-based model-theoretic abnormality prediction AP_{M-T} for the abnormality prediction problem $(SD, COMPS, OBS, HYP)$ is not valid, i.e. $\sigma_T \notin AP_{M-T}$. In this case, from Definition 77 and the previous result, $SD \cup OBS \cup HYP \cup \{\sigma_T\}$ is unsatisfiable. From Definition 25, $\{\sigma_T\}$ must have a model \mathcal{M} such that $\exists_{\mathcal{P}, \mathcal{R} \in \Omega} \exists_{\Gamma(\mathcal{M}, \mathcal{P})} (\mathcal{P} \subseteq \mathcal{R}) \wedge (\mathcal{R} \models \Psi)$. Combining all the arguments we get that $\neg(\exists_{s \in \text{Mod}(SD \cup OBS \cup HYP)} \exists_{\mathcal{P}, \mathcal{R} \in \Omega} \exists_{\Gamma(\mathcal{M}, \mathcal{P})} (\mathcal{P} \subseteq \mathcal{R}) \wedge (\mathcal{R} \models \Psi))$; and either the believed system description, the believed system observations or the prediction hypotheses are not ontologically true.
2. Suppose an ontologically true but incomplete believed system description. Suppose also a sentence φ in HYP_{pred} such that: $SD \cup OBS \cup HYP_{\text{cons}} \not\models \{\varphi\}$ and $SD \cup OBS \cup HYP_{\text{cons}} \models \{\neg\varphi\}$. In this case, no abnormality prediction satisfies the required condition and, as so, the AP_{M-T} is not valid.

■

Theorem 9. *If \mathcal{A} is a sound and complete abnormality prediction algorithm, then $AP_{M-T} = AP_{P-T}$ for a given abnormality prediction problem $(SD, COMPS, OBS, HYP)$.*

Proof. Trivial from the definition of model-theoretic and proof-theoretic abnormality prediction and Definitions 35 and 77. ■

Corollary 10. *If A is a sound and complete abnormality prediction algorithm and SD is an ontologically true believed system description, then for every abnormality prediction problem $(SD, COMPS, OBS, HYP)$ with ontologically true observations and prediction hypotheses:*

1. *every consistency-based proof-theoretic abnormality prediction, $AP_{P,T}$, is valid.*
2. *some abduction-based proof-theoretic abnormality prediction, $AP_{P,T}$, are not valid.*

Proof.

Trivial from Theorems 8 and 9. ■

With a series of abnormality prediction systems' properties and property relations, we are now ready to meta-diagnose abnormalities in Models of abnormality prediction systems.

5.2.3 When Models hurt prediction: a DC motor trial

As stated in the beginning of the present section, in this subsection I will illustrate how abnormalities in Models of implemented abnormality prediction systems can affect the computed predictions. This, in turn, will provide a basis example for the remainder of this section. As so, I will start by elaborating a scientific and mathematical theory representing a DC motor prediction-wise. Then, I will expose how an implemented abnormality prediction system based on this theory fails to compute a valid prediction.

Modelling a DC motor

Consider once again Mr. Tortuga and his car. Satisfied by the results obtained with the new ARR-based permanent magnet DC motor diagnostic system, Tortuga decided to continue his experiences by implementing an abnormality prediction system for this same motor, whose representation is depicted in Figure 5.1. First, suppose that Tortuga noticed that the iron-made electrical wiring on the armature (represented by its equivalent perfect line, resistor R and inductor L in Figure 5.1) had a tendency to slowly become oxidised. This being so, Tortuga realised that when a critical penetration level of iron oxide was achieved, the wires tended either to break due to the motor vibrations or to generate enormous amounts of heat leading to internal fires in the motor. As so, Tortuga thought it would be a very good idea to predict when a given penetration level of iron oxide was attained in order to fully replace the DC motors' armature wirings before all the important damage takes place. Second, suppose also that after a series of bad experiences with motors that tore themselves apart, Tortuga understood that 1) the viscous friction in the rotor (represented by its equivalent coefficient of viscous friction f in Figure 5.1) has a tendency to increase with the number of cycles, 2) that this friction generated vibrations in the motor, and 3) that these same vibrations, in turn, generated even more friction, and so on until the motor tears itself apart. In a nutshell, he realised that the coupling between vibrations and friction generated an exponential increase in friction that had to be quickly predicted before huge damages in the motor take place.

Interested as he certainly is, Tortuga started by asking for Mr Zapato's help into obtaining a series of informations concerning iron corrosion and iron oxide. This way, he first gathered some data on the penetration of iron oxide on his island conditions and understood that the function $p(t) = 1.5 \cdot 10^{-4} \cdot \ln(1 + 100 \cdot (\frac{t}{365})^2)$ perfectly interpolated all the gathered observations for a new unoxidised piece of iron; for as more and more iron gives rise to iron oxide, the inner iron core becomes more protected and corrosion slows down. Moreover, he measured the length and determined the radius of the iron wire to be, respectively, 216.5m and $2.5 \cdot 10^{-3}$ m. Finally, Mr Zapato told him that an iron core radius of less than $1.2 \cdot 10^{-3}$ m would entail breaking through vibrations and potential fires. Using all these informations, he built a scientific and mathematical theory on how the iron core radius evolves with time as oxidation takes place. This is depicted in Figure 5.4.

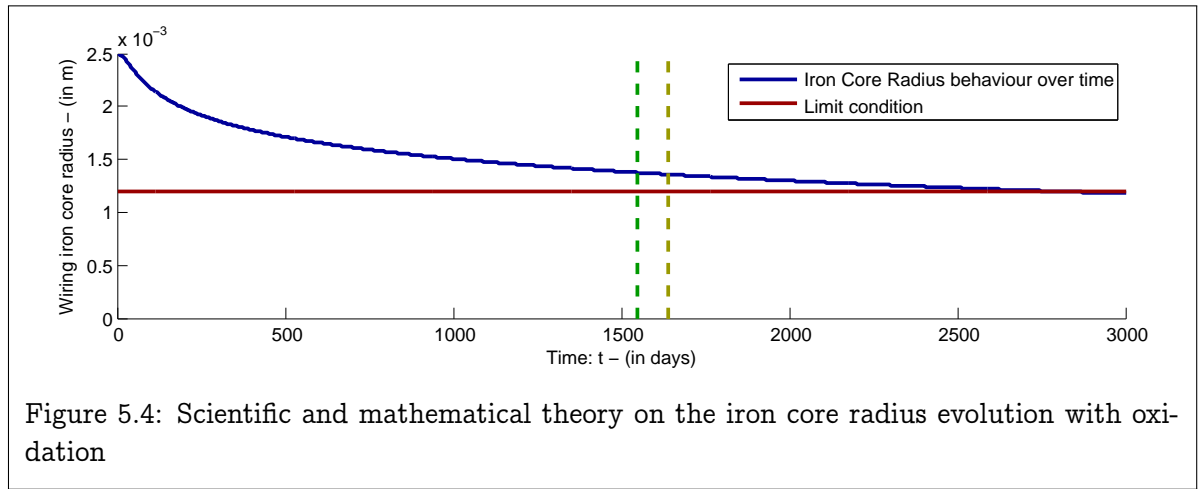


Figure 5.4: Scientific and mathematical theory on the iron core radius evolution with oxidation

Once the Model of the iron core radius was ready, Tortuga then focused on the viscous friction in the DC motor. With Mr Zapato's help, he set up a series of run-to-failure tests and concluded that: 1) the function $f(t) = f_0 \cdot e^{(\frac{t}{2000})^{50}}$ (with f_0 being the initial value of the coefficient of viscous friction) interpolated all the observations gathered on those tests (where the coefficient of friction was measured through its impact on the angular velocity of the DC motor); and 2) the DC motor tears itself apart when the coefficient of viscous friction exceeds the value of 1N.m.s. This is depicted in Figure 5.5.

From these two scientific and mathematical theories, Tortuga then proceeded into studying, from a different perspective, the electrical and mechanical equations describing the behaviour of the DC motor. As so, he started by deciding he would only gather observations for prediction when the DC motor is in steady state; for this would remove every possible errors committed when modelling the transient behaviour of the motor. Hence, from Equations 5.1 and 5.2 in steady state, he discovered that:

$$\omega_R(\infty) = \frac{\frac{V_S \cdot K_T}{R} - \tau_L}{\frac{K_T \cdot K_v}{R} + f} \quad i(\infty) = \frac{\frac{\tau_L \cdot K_v}{f} - V_S}{\frac{K_T \cdot K_v}{f} + R}$$

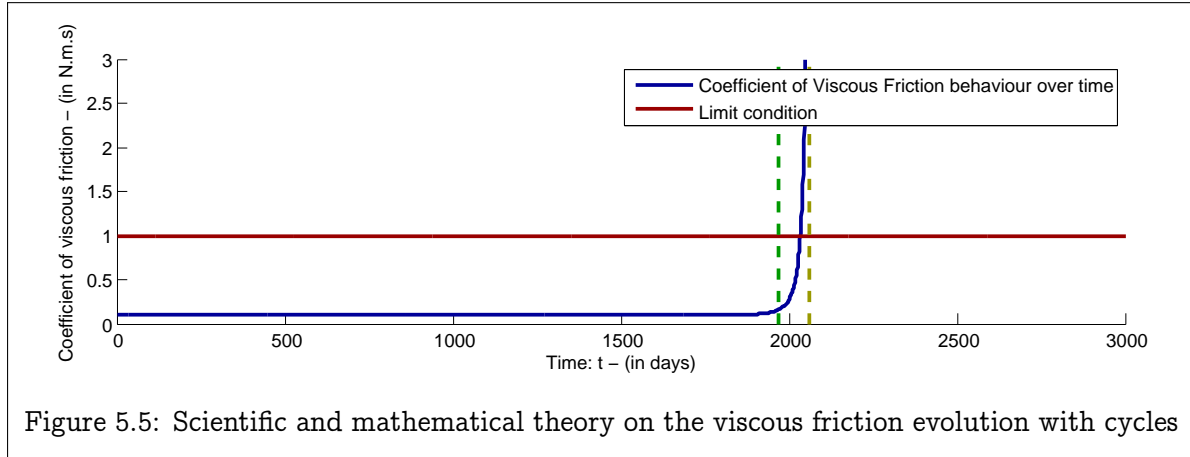


Figure 5.5: Scientific and mathematical theory on the viscous friction evolution with cycles

once again with $K_v = K_T$ (cf. Subsection 5.1.1).

Tortuga also wondered about the relation between the resistance of the electrical iron wire and the amount of iron oxide formed. Since the iron core and the iron oxide act as two resistors in parallel, the total resistance is given by:

$$R = \frac{1}{\frac{1}{R_{core}} + \frac{1}{R_{oxide}}}$$

Furthermore, each individual resistance R_x is determined by $\rho_x \cdot \frac{l}{S} \cdot (1 + \alpha_x \cdot (T - T_0))$, where the factor $(1 + \alpha_x \cdot (T - T_0))$ has been detailed in Subsection 5.1.4, ρ_x is the resistivity of the material x , and l and S are, respectively, the length and cross-sectional area of the wire. In this case: 1) $l = 216.5\text{m}$, 2) $S_{core} = \pi \cdot (r_0 - p(t))^2$ (with $r_0 = 10^{-3}\text{m}$), 3) $S_{oxide} = \pi \cdot [r_0^2 - (r_0 - p(t))^2]$, 4) $\alpha_{core} = \alpha_{oxide} = 6.2 \cdot 10^{-3}\text{K}^{-1}$, 5) $\rho_{core} = 1 \cdot 10^{-7}\Omega \cdot \text{m}$ and 6) $\rho_{oxide} = 6 \cdot 10^2\Omega \cdot \text{m}$.

Finally, Tortuga decided not to add any more sensors to the ones in his implemented diagnostic system (cf. Section 5.1). Therefore, the angular velocity of the shaft ω_R and the current i can be measured. Moreover, he decided to always measure ω_R and i when the DC motor is considered to be in steady state.

With all these data, he could finally define a believed system (cf. Definition 9) where $\text{COMPS} = \{\text{Electric}, \text{Mechanic}\}$, representing the electrical wire and the mechanical components of the DC motor, and SD is the theory containing the following sentences:

$$\begin{aligned}
S_1: \quad & i = \frac{\frac{\tau_L \cdot K_v}{f} - V_s}{\frac{K_T \cdot K_v}{f} + R} \\
S_2: \quad & \omega_R = \frac{\frac{V_s \cdot K_T}{R} - \tau_L}{\frac{K_T \cdot K_v}{R} + f} \\
S_3: \quad & R = \frac{1}{\frac{1}{R_{core}} + \frac{1}{R_{oxide}}} \\
S_4: \quad & R_{core} = \rho_{core} \cdot \frac{l}{\pi \cdot (r_0 - p)^2} \cdot (1 + \alpha_{core} \cdot (T - T_0)) \\
S_5: \quad & R_{oxide} = \rho_{oxide} \cdot \frac{l}{\pi \cdot [r_0^2 - (r_0 - p)^2]} \cdot (1 + \alpha_{oxide} \cdot (T - T_0)) \\
S_6: \quad & p = 1.5 \cdot 10^{-4} \cdot \ln(1 + 100 \cdot (\frac{t_0^1 + t}{365})^2) \\
S_7: \quad & f = f_0 \cdot e^{(\frac{t_0^2 + t}{2000})^{50}} \\
S_8: \quad & p_{pred} = 1.5 \cdot 10^{-4} \cdot \ln(1 + 100 \cdot (\frac{t_0^1 + t_{pred}}{365})^2) \\
S_9: \quad & f_{pred} = f_0 \cdot e^{(\frac{t_0^2 + t_{pred}}{2000})^{50}} \\
S_{10}: \quad & \text{Ab(Electric)} \Leftrightarrow ((r_0 - p_{pred}) < r_{lim}) \\
S_{11}: \quad & \text{Ab(Mechanic)} \Leftrightarrow (f_{pred} > f_{lim})
\end{aligned}$$

together with the axioms for arithmetic and so on; and with: K_v , K_T , τ_L , f_0 , l , r_0 , T_0 , ρ_{core} , ρ_{oxide} , α_{core} , α_{oxide} , r_{lim} and f_{lim} defined as before. Moreover, t , t_0^1 and t_0^2 are variables. This is the Model that will be used for prediction. Note that, for the sake of simplicity, Tortuga used a propositional logic, for he is only interested in abnormality predictions at a given instant of time. This generates two pairs of redundant sentences: S_6 and S_8 , and S_7 and S_9 ; that represent the functions p and f at two different instants: the instant where prediction takes place, and the predicted instant. Such redundancy could have been avoided by using the more expressive first-order language.

Predicting abnormalities on the DC motor

As seen before, both believed systems (cf. Definition 9) and believed system observations (cf. Definition 15) can be seen as a common factor between explanation and prediction. Since the former has been treated above, it is time to focus on the latter. Suppose that, each time Tortuga rides his car, the DC motor is started with a drive voltage of 1V. One fine day, after waiting for the armature current and shaft angular velocity to reach their steady state values, Tortuga determined i and ω_R to be, respectively, 0.2382A and 0.01507rad/s. Moreover, suppose that Tortuga decided to predict the health state of his DC motor 90 days later. Finally, to ensure the validity of his predictions (cf. Corollary 10), he decided to use a consistency-based approach. Hence, the set of believed system observations is: $\text{OBS} = \{V_s = 1, i = 0.2382, \omega_R = 0.01507, t = 0\}$; and the set of prediction hypotheses is: $\text{HYP} = \{t_{pred} = 90\}$.

Based on the believed system, believed system observations and prediction hypotheses above, the kernel model-theoretic abnormality previsions (cf. Subsection 5.2.1) for the abnormality prevision problem (SD, COMPS, OBS, HYP) can be determined to be (with $\text{HYP}_{pred} = \emptyset$): $\text{KAP} = \{\neg \text{Ab(Electric)}, \text{Ab(Mechanic)}\}$; with $t_0^1 = 1549$, $t_0^2 = 1969$, $p_{pred} = 0.001364$ and

$f_{pred} = 7.21506$. The green and yellow lines in Figures 5.4 and 5.5 represent, respectively, the prediction and predicted instants' values of the iron core radius and the viscous friction coefficient.

Having these results in his hands, Tortuga decided to continue riding his car for 90 days in order to test them. After all, he could spare one single DC motor for this noble cause. Strangely, after these 90 days had passed, the DC motor did not tear itself apart. Tortuga is now facing a meta-diagnostic problem that needs to be solved: why did the implemented abnormality prediction system foresee a mechanic abnormality while his own perception is that 90 days later every component is normal?

5.2.4 Meta-diagnosis

In the previous sections and chapters we focused on diagnostic systems. In that context, to meta-diagnose a given meta-system one had to identify it, along with its meta-goals, identify the components and their meta-goals, describe meta-components and meta-components' normal/abnormal behaviour on believed meta-systems and identify meta-observables based on the believed meta-system description. If meta-diagnosis claim of being a general theory is verified, such modelling process should be exactly the same for abnormality prediction systems, with the concept of meta-system extended to take these objects into account.

In Tortuga's case, he is looking for the reason behind the invalid abnormality predictions computed. Moreover, for the sake of the argument, imagine that Tortuga assumes the implemented abnormality prediction algorithm computing abnormality predictions is sound and complete, that the believed system observations are ontologically true and that the meta-system is correctly perceived as a well-implemented abnormality prediction system. Hence, we are confronted with a problem similar to the one of Subsection 3.2.1, i.e. with "the problem of ontologically false believed system descriptions". This being so:

1. The meta-system is made of:
 - a sensed part of reality correctly perceived as a well-implemented abnormality prediction system $S = (SD, COMPS, OBS, \mathcal{A})$; where SD, COMPS and OBS are those of the previous subsection.
 - the meta-goal of computing valid proof-theoretic consistency-based abnormality predictions.
2. Using Corollary 10 and the problem hypotheses, the meta-components can be chosen to represent every implemented SD-sentence, whose associated meta-goal is their ontological truth. Moreover, implemented sentences S_6 and S_8 , describing the same part of reality can be described by a single believed meta-system meta-component. This is also the case of implemented sentences S_7 and S_9 .
3. Every SD-sentence s represented by a believed meta-system meta-component S is modelled, at a meta-system level as: $\neg M\text{-}Ab(S) \Rightarrow s$.
4. From Corollary 10 and the problem hypotheses we get that if every believed meta-system meta-component is normal then the AP_{P-T} is valid. To assess AP_{P-T} validity we need to meta-observe σ_T and take into account that $AP_{M-T} = AP_{P-T}$ in our conditions.

In a nutshell, believed meta-system meta-components can be chosen as $M-COMPS = \{S_1, S_2, S_3, S_4, S_5, S_{6\&8}, S_{7\&9}, S_{10}, S_{11}\}$; and the following believed meta system description, $M-SD$, can be built:

$$\begin{aligned}
\neg M-Ab(S_1) &\Rightarrow \left(i = \frac{\frac{\tau_L \cdot K_v}{f} - V_S}{\frac{K_T \cdot K_v}{f} + R} \right) \\
\neg M-Ab(S_2) &\Rightarrow \left(\omega_R = \frac{\frac{V_S \cdot K_T}{R} - \tau_L}{\frac{K_T \cdot K_v}{R} + f} \right) \\
\neg M-Ab(S_3) &\Rightarrow \left(R = \frac{1}{\frac{1}{R_{core}} + \frac{1}{R_{oxide}}} \right) \\
\neg M-Ab(S_4) &\Rightarrow \left(R_{core} = \rho_{core} \cdot \frac{l}{\pi \cdot (r_0 - p)^2} \cdot (1 + \alpha_{core} \cdot (T - T_0)) \right) \\
\neg M-Ab(S_5) &\Rightarrow \left(R_{oxide} = \rho_{oxide} \cdot \frac{l}{\pi \cdot [r_0^2 - (r_0 - p)^2]} \cdot (1 + \alpha_{oxide} \cdot (T - T_0)) \right) \\
\neg M-Ab(S_{6\&8}) &\Rightarrow \left(p = 1.5 \cdot 10^{-4} \cdot \ln(1 + 100 \cdot (\frac{t_0^1 + t}{365})^2) \right) \wedge \\
&\quad \wedge \left(p_{pred} = 1.5 \cdot 10^{-4} \cdot \ln(1 + 100 \cdot (\frac{t_0^1 + t_{pred}}{365})^2) \right) \\
\neg M-Ab(S_{7\&9}) &\Rightarrow \left(f = f_0 \cdot e^{(\frac{t_0^2 + t}{2000})^{50}} \right) \wedge \left(f_{pred} = f_0 \cdot e^{(\frac{t_0^2 + t_{pred}}{2000})^{50}} \right) \\
\neg M-Ab(S_{10}) &\Rightarrow [Ab(Electric) \Leftrightarrow ((r_0 - p_{pred}) < r_{lim})] \\
\neg M-Ab(S_{11}) &\Rightarrow [Ab(Mechanic) \Leftrightarrow (f_{pred} > f_{lim})]
\end{aligned}$$

extended with the appropriate axioms for arithmetic and so on. As for meta-observations, they contain all the believed system observations at system-level. Moreover, this theory is extended by taking into account the unobserved abnormality in the two components analysed by Tortuga represented by the believed system components: Electric and Mechanic. As so, $M-OBS = \{V_s = 1, i = 0.2382, \omega_R = 0.01507, t = 0, t_{pred} = 90, \neg Ab(Electric), \neg Ab(Mechanic)\}$

With a believed meta system and some meta-observations the consistency-based kernel meta-diagnoses for the meta-diagnostic problem $M-DP = (M-SD, M-COMPS, M-OBS)$ can now be computed. It contains the following kernel meta-diagnosis hypotheses:

$$\begin{array}{cc}
\{M-Ab(S_{11})\} & \{M-Ab(S_{7\&9})\} \\
\{M-Ab(S_1)\} & \{M-Ab(S_2)\}
\end{array}$$

Using such results, Tortuga inspected the sentences S_1 , S_2 , S_{11} and S_7 and S_9 . With the help of Mr. Zapato he understood what the problem was. In fact, while the passage of time affects corrosion independently of DC motor's usage, this is not the case with viscous friction. As so, Tortuga found that sentences S_7 and S_9 needed to be corrected to reflect how much time the DC motor had been used.

5.3 Originality and Related work

I started this chapter by affirming that, the main objectives of this dissertation being achieved, it was time to focus on meta-systems and on the unproven claim regarding the meta-diagnosis thesis' generality. Accordingly, I defined the main objectives of the present chapter to be concerned with providing evidence for the usage of meta-diagnosis to detect and isolate abnormalities in Models of 1) non-logical-based implemented diagnostic systems, and 2) meta-systems corresponding to entities other than implemented diagnostic systems. My contribution in this chapter appears, thus, naturally related to these two axes.

Concerning the meta-diagnosis of non-logical-based implemented diagnostic systems, in Section 5.1, it offers to the naked eye an original contribution towards the usage of meta-diagnosis for reasoning in the Model-based diagnosis framework of analytical redundancy relations. In fact, if on one hand it is true that some works exist in the FDI community for reasoning in the presence of possibly abnormal - or "uncertain", or "failing to correctly represent noise", and so on - Models [23][45][2], even if such works invariably make some hypotheses about the unknown, about how approximate the Model is to reality or about how the reality is structured, among many others hypotheses described, for instance, by [84]; on the other hand it is also true that, to my knowledge, the FDI community is chronically missing of a method in its arsenal to detect and isolate abnormalities in Models of implemented diagnostic systems.

Nevertheless, there is much more to this section than meets the eye. In fact, this contribution brings with it the much stronger idea that if a diagnostic problem encoded in a non-logical formalism can be mapped into the framework of Section 2.3, then our theory of meta-diagnosis can be used to reason about abnormalities in the Models of its underlying diagnostic system. In a nutshell, the *Bridge* used in that section can be seen as a metaphor for a whole set of mappings.

As for the meta-diagnosis of meta-systems representing entities other than implemented diagnostic systems, in Section 5.2, it is once again a trojan horse. Straightforwardly, a greater idea lies behind the superficial contribution concerning the usage of meta-diagnosis for reasoning about abnormalities in Models of implemented abnormality prediction systems; the idea that if one can describe some meta-goals of a meta-system in the framework of Section 3.1, then one can meta-diagnose such meta-system. Finally, it appears to me that the characterisation of abnormality predictions and the definition and relation of properties in its underlying systems is a collateral yet original contribution related to the work of Poole in [86].

Chapter 6

A final perspective

Last words are for fools who haven't said enough.

- Karl Marx

This dissertation was dedicated to reasoning about Models, i.e. scientific and mathematical theories, observations and rules of inference. Motivated by implemented [Airbus] diagnostic systems' problems, its main message was that detecting and isolating abnormalities in [Airbus] Models is possible, both in theory and in practice. In the remainder of the present book, I will start by navigating the reader through all the arguments given in order to reach this and other conclusions and finish with a glimpse into the future.

6.1 Summary of the contribution

The introductory chapter announced that Models are condemned to suffer from a series of abnormalities; abnormalities that affect, in particular implemented [Airbus] diagnostic systems and their computed results. This being so, building a theory and developing a tool for detecting and isolating such abnormalities became the prime goal of this dissertation. Moreover, as different heterogeneous implemented Airbus diagnostic systems, suffering from distinct abnormalities, may compute different diagnoses, developing methods and specifying tools for: 1) checking the consistency between subsystem-level diagnoses and 2) validating and comparing the performance of these implemented diagnostic systems; became two other objectives of this dissertation.

Since the first step into solving a problem is understanding and formalising it, I started by redefining the classic framework of Model-based diagnosis from formal model-theoretic perspective, based on the difference between the cognitive and the real-worlds. Such framework

enabled me to express exactly what abnormalities are and how they relate to the properties of the computed diagnoses. Chapter 2 served such purpose.

The material of the second chapter was then exploited in the third one. From the idea that an implemented diagnostic system can be, itself, seen as a real-world artifact to be diagnosed, I developed a theory of meta-diagnosis enabling the detection and isolation of abnormalities in Models of implemented diagnostic systems and explanation in general. Then, I implemented such theory in a tool, called MEDITO, and tested it against the classical Polybox toy problem.

Applying the developed theory of meta-diagnosis at Airbus was the main motivation behind the fourth chapter. In order to so, and since the implemented diagnostic systems from different Airbus aircraft programs (A380, A400M, A350 and research) have different syntaxes and semantics, I started by establishing a common framework of diagnosis at Airbus, naturally mapped to the developments of Chapter 2. A gateway was then opened to the direct usage of MEDITO for detecting and isolating abnormalities in Models of implemented Airbus diagnostic systems. Moreover, I solved the two other industrial objectives behind this book. As for the first one, relying once again in Airbus' common framework of diagnosis, I defined a theoretical condition for subsystem-level diagnoses' consistency; and developed a tool, called CONCKER, for verifying such condition at Airbus. Finally, I tackled the problem of validating and comparing the performance of different implemented diagnostic systems at Airbus. Unsurprisingly using the common framework of diagnosis, I exploited the fact that the properties of diagnoses are intimately related to the abnormalities in Models of implemented diagnostic systems - courtesy of Chapter 2 developments - and built a method and tool, called DPAT, for computing such properties.

With all the initial objectives tackled, the fifth chapter of this book was devoted to convincing the reader that meta-systems can be generalised, both in the formalism used and on their scope. To do so, I started by successfully meta-diagnosing an implemented Analytical-Redundancy-Relation-based diagnostic system. Finally, an implemented abnormality prediction system, whose theoretical bases were given in the same chapter, was meta-diagnosed.

6.2 A glimpse into the future

Straightforwardly, this dissertation is far from being a complete account. Hence, I hereafter provide the reader with some of the main questions that either remain unanswered or whose answer has not been developed in this book.

6.2.1 The notion of knowledge

Scientific theories and observations were presented, in this work, as sets of sentences stating some facts about the real-world; sentences that are usually assumed ontologically true at a meta-level. If instead of adopting such approach, one sees these sentences as statements of what we know, what we believe, what we think is possible, and so on, then the assumption of ontological truth can be displaced from a meta-level to the level of those theories themselves.

In terms of a logic formalism, this can be done with the same classical logics as before, even if one usually prefers modal epistemic logics such as the S5 or KD45 for its elegance and for its already exploited connection with the philosophical accounts ¹ [61, 67, 46]. The reader is encouraged, for instance, to see the parallel between the assumption of ontological truth and the truth axiom in these logics. Formalising our Models using these concepts is a future axis of development.

6.2.2 The notion and logic of explanation

The notion and logic of explanation have been discussed several times along this dissertation. Straightforwardly, I have committed to one of the many possible notions of explanation conjectured throughout the times. Moreover, for simplicity reasons, I have made all the developments using classical logics, although I stated that the results obtained were applicable to other logics. Hence, a future axis of development would be the effective usage of other notions and better-suited logics of explanation, such as, for instance, those of Shoham [97] and Thielscher [108].

6.2.3 The logics handled by MEDITO

When introducing MEDITO, I stated that, implementing CHOCO as a Constraint Satisfaction Problem solver, it was only able to handle first-order theories. While this is enough to handle, for example, set theory (eg. Zermelo-Fraenkel set theory with the axiom of choice) and many aspects of natural language, it is still unable to take into account, for instance, some subtleties of these language [18]. This is why it would be interesting for MEDITO to be able to manage other logics such as, for instance, second-order logic.

6.2.4 The formalism of Meta-diagnosis

Model-based meta-diagnosis has been defined using a logical formalism. Despite such choice, the fifth chapter provided empirical evidence that non-logical-based implemented diagnostic systems, such as Analytical-Redundancy-Relations-based ones, could be meta-diagnosed. Doing so, however, required two transformations to and from an ARR-based formalism to a logical one; transformations which have underlying costs. As so, it would be important to directly implement meta-diagnostic systems in other formalisms.

In Mr. Tortuga's DC-motor example of Chapter 5, for instance, since Definition 19 and Definition 46 are syntactically equivalent and [32] tells us that under some assumptions "FDI and DX views agree on diagnoses", then one could certainly try to investigate the possibility of directly computing meta-diagnoses through an FDI analytical redundancy approach.

In this perspective, one could take, for instance, the first M-SD-sentence in the DC motor problem:

¹Note that basic modal logics can be seen as classical logics using the standard translation.

$$\neg \mathbf{M}\text{-}\mathbf{Ab}(\mathbf{S}_1) \Rightarrow [\mathbf{Closed}(\mathbf{circuit}) \wedge \neg \mathbf{Ab}(\mathbf{L}) \wedge \neg \mathbf{Ab}(\mathbf{R}) \wedge \\ \wedge \neg \mathbf{Ab}(\mathbf{K}) \Rightarrow (V_{L_{obs}} - [V_s - R \cdot i(t) - K \cdot \omega_R(t)] = 0)]$$

and use the analogy between both worlds to formulate a meta-ARR with a meta-residual:

$$m - r_1 = (r_1 - r_{1_{obs}})$$

with $r_1 = 0$ if $(V_{L_{obs}} - [V_s - R \cdot i(t) - K \cdot \omega_R(t)]) = 0$ and 1 otherwise; and $r_{1_{obs}} = 0$ if no component in the structure of ARR1 is meta-observed as faulty and 1 otherwise. Moreover, rejecting the exoneration and no-compensation hypotheses a $m - r_1 < 0$ could be interpreted as a possible misdetection and not as a fault in the model. Finally, the M-ARR would have the structure {ARR1}.

6.2.5 The selection of test-cases

Meta-diagnosis (implemented in MEDITO) and diagnosis performance assessment (implemented in DPAT) rely on a series of sets of observations, called test-cases, for achieving its goals. At this point, two important difficulties arise. First, one usually wants test-cases to cover all the situations the implemented diagnostic system may be confronted to throughout his life. Second, one is typically fond of those sets of tests-cases that are as small as possible; for big sets of test-cases require more computations with potentially redundant results. Now, selecting a minimal complete set of test-cases is far from being trivial and remains an open point in this dissertation.

6.2.6 The repair of Meta-systems

I left what is maybe the most important incomplete axis of this dissertation to the end. In order to introduce it, I encourage the reader to note that when an implemented diagnostic systems determines a component of a given system to be abnormal, then one simply needs to replace it by a normal one. In a nutshell, repairing a system is usually done through the replacement of abnormal components by normal ones. Unfortunately, things are not so simple in meta-diagnosis if the isolated abnormalities do not correspond to implementation errors. In fact, suppose an implemented SD-sentence is found to be ontologically false. In this case, in order to repair its associated implemented diagnostic system one must either: 1) completely remove this sentence, at the cost of a loss of precision in the computed diagnoses; or 2) re-define that sentence to guarantee its ontological truth (eg. through learning). The treatment of latter case is not covered by this book, and it is maybe the most interesting perspective for the future.

Appendix A

Model-theory

In this dissertation, I use the notions of: logic, tuple, signature, structure, signature interpretation, correct structure interpretation, function, relation, constant, variable, symbol, substructure, extension, model, isomorphism, closed term, closed sentence, ground term, ground sentence, theory, satisfiability, entailment, syntax, semantics, inference, logical implication, consequence, theorem and complete theory. These concepts are clarified hereafter. The material presented is completely brought from the works of Hodges [62], Marker [74] and Hedman [56]; and should be used as a reference for model-theory in this dissertation.

First of all, a **structure** \mathcal{M} is an mathematical object specified by:

- A (possible empty) set called the **domain** or **universe** of \mathcal{M} written M . Each element of M is called an element of the structure \mathcal{M} .
- A (possible empty) set of **constant elements** of \mathcal{M} , each named by one or more **constants**. If c is a constant, $c^{\mathcal{M}}$ indicates the constant element named by c .
- For each positive integer n , a (possible empty) set of n -ary **relations** on M (subsets of M^n). Each relation is named by one or more n -ary **relation symbols**. If R is a relation symbol, $R^{\mathcal{M}}$ indicates the relation named by R .
- For each positive integer n , a (possible empty) set of n -ary **operations** on M . Each operation is named by one or more n -ary **function symbols**. If F is a function symbol, $F^{\mathcal{M}}$ indicates the **function** named by F .

Moreover, every well-ordered sequence of elements of a structure will be written \overline{m} , \overline{n} , and so on. A **tuple** in \mathcal{M} is a finite sequence of elements of \mathcal{M} ; and if a tuple has length n it is referred to as an n -tuple. Finally, if f is a function and \overline{m} is an n -tuple, $f\overline{m}$ means (fm_1, \dots, fm_n) .

The **signature** \mathcal{L} of the structure \mathcal{M} is specified by the set of constants, relation symbols and function symbols of \mathcal{M} (we assume \mathcal{L} can be read off uniquely from the structure). Note, however, that constants, functions and relations of a structure can be named by different symbols, i.e. a signature can be **interpreted** in several different ways. If \mathcal{M} has signature \mathcal{L} , we say \mathcal{M} is an \mathcal{L} -structure.

Let \mathcal{M} and \mathcal{P} be two \mathcal{L} -structures with domains M and P respectively. An **homomorphism** $f: \mathcal{M} \rightarrow \mathcal{P}$ is a function f from M to P with the following properties:

- For each constant c of \mathcal{L} , $f(c^{\mathcal{M}}) = c^{\mathcal{P}}$.
- For each positive integer n , each n -ary relation symbol R of \mathcal{L} and n -tuple \bar{m} from \mathcal{M} , if $\bar{m} \in R^{\mathcal{M}}$ then $f\bar{m} \in R^{\mathcal{P}}$.
- For each positive integer n , each n -ary function symbol F of \mathcal{L} and n -tuple \bar{m} from \mathcal{M} , $f(F^{\mathcal{M}}(\bar{m})) = F^{\mathcal{P}}(f\bar{m})$.

Now, an **embedding** of \mathcal{M} into \mathcal{P} is a homomorphism $f: \mathcal{M} \rightarrow \mathcal{P}$ which is injective and satisfies the following stronger version of the second condition:

- For each positive integer n , each n -ary relation symbol R of \mathcal{L} and each n -tuple \bar{m} from \mathcal{M} , $\bar{m} \in R^{\mathcal{M}} \Leftrightarrow f\bar{m} \in R^{\mathcal{P}}$.

An **isomorphism** is a surjective embedding. An isomorphism between the structures \mathcal{M} and \mathcal{P} will be noted $\mathcal{M} \cong \mathcal{P}$.

Another interesting concepts are the ones of **extension** and **substructure**. If \mathcal{M} and \mathcal{P} are \mathcal{L} -structures with $\mathcal{M} \subseteq \mathcal{P}$, and the inclusion map $i: \mathcal{M} \rightarrow \mathcal{P}$ is an embedding, then we say \mathcal{P} is an extension of \mathcal{M} or that \mathcal{M} is a substructure of \mathcal{P} . This is noted $\mathcal{M} \subseteq \mathcal{P}$.

Now, every language is built up from the atomic formulas of a signature, let us say \mathcal{L} ; where atomic formulas are some strings of symbols including the symbols of \mathcal{L} . Moreover, every language has a stock of **variables**. These are symbols that serve as temporary labels for the elements a structure. As for the **terms** of the signature \mathcal{L} , these are strings of symbols defined as follows:

- Every variable is a term of \mathcal{L} .
- Every constant of \mathcal{L} is a term of \mathcal{L} .
- If $n > 0$, F is an n -ary function symbol of \mathcal{L} and t_1, \dots, t_n are terms of \mathcal{L} then the expression $F(t_1, \dots, t_n)$ is a term of \mathcal{L} .
- Nothing else is a term of \mathcal{L} .

A term is said **closed** or **ground** if no variables occur in it. Moreover, if a term t is introduced as $t(\bar{x})$, this means that \bar{x} is a sequence x_0, x_1, \dots , possibly infinite, of distinct variables; and every variable which occurs in t is among the variables in \bar{x} .

As for the **atomic formulas** of \mathcal{L} , these are the strings of symbols given by the following conditions:

- If s and t are terms of \mathcal{L} , then the string $s = t$ is an atomic formula of \mathcal{L} .
- If $n > 0$, R is an n -ary relation symbol of \mathcal{L} and t_1, \dots, t_n are terms of \mathcal{L} then the expression $R(t_1, \dots, t_n)$ is an atomic formula of \mathcal{L} .

In this case, the symbol $=$ is not assumed to be a relation symbol in the signature. Moreover, when an atomic formula has no variables, it is referred to as being an **atomic sentence**. Furthermore, a **literal** of \mathcal{L} is either an atomic or a negated atomic formula of \mathcal{L} ; where the latter is a string $\neg\phi$ with ϕ being an atomic formula of \mathcal{M} . Finally, the symbol \neg shall be read as “not” and we say that $\mathcal{M} \models \neg\phi[\bar{m}]$ holds iff $\mathcal{M} \models \phi[\bar{m}]$ does not hold.

Now, if the variables \bar{x} of an atomic formula $\phi(\bar{x})$ are interpreted as names of elements \bar{m} in a structure \mathcal{M} , then ϕ makes a statement about \mathcal{M} . This statement can be **true** or **false**.

If it is true, then we say ϕ is true of \bar{m} in \mathcal{M} , or that \bar{m} satisfies ϕ in \mathcal{M} . This is noted as $\mathcal{M} \models \phi[\bar{m}]$. Formally speaking, if $\phi(\bar{x})$ is an atomic formula of \mathcal{L} with $\bar{x} = (x_0, x_1, \dots)$, \mathcal{M} is an \mathcal{L} -structure and \bar{m} is a sequence (m_0, m_1, \dots) of elements of \mathcal{M} (assumed at least as long as \bar{x} , then:

- if ϕ is the formula $s = t$ where $s(\bar{x})$ and $t(\bar{x})$ are terms, then $\mathcal{M} \models \phi[\bar{m}]$ iff $s^{\mathcal{M}}[\bar{m}] = t^{\mathcal{M}}[\bar{m}]$, and
- if ϕ is the formula $R(s_1, \dots, s_n)$ where $s_1(\bar{x}), \dots, s_n(\bar{x})$ are terms, then $\mathcal{M} \models \phi[\bar{m}]$ iff the ordered n -tuple $(s_1^{\mathcal{M}}[\bar{m}], \dots, s_n^{\mathcal{M}}[\bar{m}])$ is in $R^{\mathcal{M}}$.

Note that when ϕ is an atomic sentence, then the sequence \bar{m} can be omitted and one can write $\mathcal{M} \models \phi$ instead of $\mathcal{M} \models \phi[\bar{m}]$. We say that \mathcal{M} is a **model** of ϕ , or that ϕ is **true in \mathcal{M}** if $\mathcal{M} \models \phi$.

The language \mathcal{L} based on the signature \mathcal{L} consists of **formulas**, that is, strings of symbols built using the terms, atomic formulas and the literals of the signature \mathcal{L} , together with rules of grammar, some logical symbols (\forall, \exists, \wedge or \vee , for instance) variables and punctuation signs. The notation $\mathcal{M} \models \phi[\bar{m}]$ is extended to fit formulas of the language \mathcal{L} by induction on the construction of the formula, depending on the language. Furthermore, a **logic** is a family of languages which differ from each other only in signature.

As for the **sentences** of the language \mathcal{L} , they are formulas with no free variables, that is, formulas in which every variable is inside a quantifier (\forall or \exists). Moreover, a **theory** is simply a set of sentences. To make a long story short, if \mathcal{M} is a \mathcal{L} -structure, then each sentence ϕ is once again either true or false in \mathcal{M} . If ϕ is true in \mathcal{M} , \mathcal{M} is said to be a **model** of ϕ and noted as $\mathcal{M} \models \phi$. Moreover, given a theory T , \mathcal{M} is said to be a model of T if $\mathcal{M} \models \phi$ for all sentences $\phi \in T$. This is written as $\mathcal{M} \models T$. A theory with at least a model is said to be **satisfiable**.

Now, let T be a theory in the language \mathcal{L} and \mathbf{K} a class of \mathcal{L} -structures. We say that T **axiomatises \mathbf{K}** , if \mathbf{K} is the class of all \mathcal{L} -structures which are models of T . This is noted $\mathbf{K} = \text{Mod}(T)$. Likewise, if both T and U are theories in the language \mathcal{L} , U **axiomatises T** if $\text{Mod}(U) = \text{Mod}(T)$.

Conversely, let \mathcal{L} be a language and \mathbf{K} a class of \mathcal{L} -structures. The \mathcal{L} **theory of \mathbf{K}** , $\text{Th}_{\mathcal{L}}(\mathbf{K})$ is the set of all sentences ϕ of \mathcal{L} such that $\mathcal{M} \models \phi$ for every structure \mathcal{M} in \mathbf{K} . Moreover, when a theory T is written to describe a particular structure \mathcal{M} , we say that \mathcal{M} is the **intended model** of T . Finally, if a theory T has models that are not isomorphic to the intended one, these are called **nonstandard models**.

This duality between the **syntax** of a language (its terms, formulas, sentences, and so on) and its related **semantics** (the structures) is extremely important; for theories define classes of structures that satisfy them, and structures define sets of sentences that are satisfied by them.

Now, the symbol \models can also be used to relate theories and sentences, and theories and other theories. Let T and U be two theories and ϕ be a sentence in the language \mathcal{L} . We say that $T \models \phi$ iff $\mathcal{M} \models \phi$ whenever $\mathcal{M} \models T$. Naturally, $T \models U$ iff $\mathcal{M} \models U$ whenever $\mathcal{M} \models T$. In these cases, T

is said to **entail** or **imply** ϕ and \mathcal{U} ; and ϕ and \mathcal{U} are said to be **logical consequences** of \mathcal{T} . A theory \mathcal{T} is said to be **complete** iff for every sentence ϕ in the language \mathcal{L} , either $\mathcal{T} \models \phi$ or $\mathcal{T} \models \neg\phi$.

As for the proof-theoretic symbol \vdash it can also be used to relate theories and sentences. Let \mathcal{T} be a theory and ϕ be a sentence in a language \mathcal{L} . $\mathcal{T} \vdash \phi$ if ϕ is deducible from \mathcal{T} in some proof calculus. In this case, we say that ϕ is **inferred** or **proved** from \mathcal{T} , and ϕ is a **theorem** of \mathcal{T} .

The last concept I will be exposing is the notion of **interpretation of a structure**. Let \mathcal{L} and \mathcal{K} be signatures, \mathcal{M} a \mathcal{K} -structure and \mathcal{P} a \mathcal{L} -structure, and n a positive integer. An (n -dimensional) **correct interpretation** Γ of \mathcal{P} in \mathcal{M} , noted $\Gamma(\mathcal{P}, \mathcal{M})$ is defined to consist of three items,

- a formula $\mu_\Gamma(x_0, \dots, x_{n-1})$ of signature \mathcal{K} ,
- for each unnested atomic formula $\phi(y_0, \dots, y_{j-1})$ of \mathcal{L} , a formula $\phi_\Gamma(\bar{x}_0, \dots, \bar{x}_{j-1})$ of signature \mathcal{K} in which the \bar{x}_i are disjoint n -tuples of distinct variables,
- a surjective map $f_\Gamma : \mu_\Gamma(\mathcal{M}^n) \rightarrow \mathcal{P}$,

such that for all unnested atomic formulas ϕ of \mathcal{L} and all $\bar{m}_i \in \mu_\Gamma(\mathcal{M}^n)$,

- $\mathcal{P} \models \phi(f_\Gamma \bar{m}_0, \dots, f_\Gamma \bar{m}_{j-1}) \Leftrightarrow \mathcal{M} \models \phi_\Gamma(\bar{m}_0, \dots, \bar{m}_{j-1})$.

The formula μ_Γ is the **domain formula** of Γ ; and the formulas μ_Γ and ϕ_Γ are the **defining formulas** of Γ . As for the map f_Γ , it is the **coordinate map** of Γ . To **interpret a structure** \mathcal{P} in a structure \mathcal{M} is to **try to build** a correct interpretation of \mathcal{P} in \mathcal{M} . To interpret a structure does not mean that there is a correct interpretation.

Bibliography

- [1] Robert Merrihew Adams. Primitive Thisness and Primitive Identity. *Journal of Philosophy*, 76, 1979.
- [2] Olivier Adrot, Didier Maquin, and José Ragot. Fault detection with model parameter structured uncertainties. In *Proceedings of the 5th European Control Conference (ECC-99)*, 1999.
- [3] National Aeronautics and Space Administration. *Systems Engineering Handbook*. National Aeronautics and Space Administration (NASA), 2007.
- [4] Airbus. *ABD0100.1.4: Maintainability Requirements for Suppliers*, Issue F edition, 2007.
- [5] Airbus. *ABD0100.1.9: Electronic*, Issue H edition, 2008.
- [6] Aristotle. *Physics (Oxford World's Classics)*. Edited by David Bostock. Translated by Robin Waterfield. Oxford University Press, 2008.
- [7] Aristotle. *Metaphysics*. Translated by W. D. Ross. NuVision Publications, 2009.
- [8] Pierre Baldi, Soren Brunak, Yves Chauvin, Claus A. F. Andersen, and Henrik Nielsen. Assessing the accuracy of prediction algorithms for classification: an overview. *Bioinformatics review*, 2000.
- [9] Nuno Belard. Diagnostic Distance: Towards the evaluation of diagnoses and diagnostic algorithms. Technical report, Airbus, 2009.
- [10] Nuno Belard. Minutes of Meeting. On-board Diagnostic Workshops: Defining a system-level diagnostic framework. Technical report, Airbus, 2011.
- [11] Nuno Belard. Minutes of Meeting. On-board Diagnostic Workshops: Defining an aircraft-level diagnostic framework. Technical report, Airbus, 2011.
- [12] Nuno Belard. Minutes of Meeting. On-board Diagnostic Workshops: Diagnosis performance assessment. Technical report, Airbus, 2011.
- [13] Nuno Belard, Yannick Pencolé, and Michel Combacau. Defining and exploring properties in diagnostic systems. In *Proceedings of the 21th International Workshop on Principles of Diagnosis (DX-10)*, 2010.

- [14] Nuno Belard, Yannick Pencolé, and Michel Combacau. A Theory of Meta-diagnosis: Reasoning about Diagnostic Systems. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence (IJCAI-11)*, 2011.
- [15] Nuno Belard, Yannick Pencolé, and Michel Combacau. MEDITO: a logic-based meta-diagnosis tool. In *Proceedings of the 23rd International Conference on Tools with Artificial Intelligence (ICTAI-11)*, 2011.
- [16] L. Birnbaum. Rigor Mortis: A Response to Nilsson’s “Logic and Artificial Intelligence”. *Artificial Intelligence*, 47, 1991.
- [17] Max Black. The Identity of Indiscernibles. *Mind*, 61, 1952.
- [18] George Boolos. To Be is to Be a Value of a Variable (or to Be Some Values of Some Variables)*. *Journal of Philosophy*, 81, 1984.
- [19] Claus Borgnakke and Richard Sonntag. *Fundamentals of Thermodynamics*. Wiley, 2008.
- [20] Abderrazak Boulaabi. Towards the evaluation of Airbus diagnostic algorithms’ performances. Master’s thesis, Institut Supérieur de l’Aéronautique et de l’Espace: Ecole Nationale Supérieure d’Ingénieurs de Constructions Aéronautiques, 2010. Supervisors: Nuno Belard (Airbus), Joel Bordeneuve-Guibé (ISAE-ENSICA) and Fabrice Francès (ISAE-ENSICA).
- [21] Sylvain Bromberger. Why Questions. In Robert . Colodny, editor, *Mind and Cosmos: Essays in Contemporary Science and Philosophy*. University of Pittsburgh Press, 1966.
- [22] Vittorio Brusoni, Luca Console, Paolo Terenziani, and Daniele Theseider Dupré. A Spectrum of Definitions for Temporal Model-Based Diagnosis. *Artificial Intelligence*, 102, 1998.
- [23] Jie Chen and Ron Patton. *Robust Model-Based Fault Diagnosis for Dynamic Systems*. Kluwer Academic Publishers, 1999.
- [24] Vincent Cheriére. Monitoring, Diagnosis and Architecture Guidelines for the Design of Health Monitoring Agents. Technical Report RP0831919, Airbus, 2009.
- [25] Luca Chittaro, Giovanni Guida, Carlo Tasso, and Elio Toppano. Functional and Teleological Knowledge in the Multimodeling Approach for Reasoning About Physical Systems: A Case Study in Diagnosis. *IEEE Transactions on Systems, Man and Cybernetics*, 23, 1993.
- [26] Luca Chittaro and Roberto Ranon. Hierarchical model-based diagnosis based on structural abstraction. *Artificial Intelligence*, 155, 2004.
- [27] Luca Console, Daniele Theseider Dupré, and Pietro Torasso. A theory of diagnosis for incomplete causal models. In *Proceedings of the 11th International Joint Conference on Artificial Intelligence (IJCAI-89)*, 1989.

- [28] Luca Console, Claudia Picardi, and Marina Ribaud. Diagnosis and diagnosability analysis using PEPA. In *Proceedings of 14th European Conference on Artificial Intelligence (ECAI-00)*, 2000.
- [29] Luca Console and Pietro Torasso. Integrating Models of the Correct Behavior into Abductive Diagnosis. In *Proceedings of the 9th European Conference on Artificial Intelligence (ECAI-90)*, 1990.
- [30] Luca Console and Pietro Torasso. A Spectrum of Logical Definitions of Model-based Diagnosis. *Computational Intelligence*, 1991.
- [31] Marie-Odile Cordier, Philippe Dague, Michel Dumas, François Lévy, Jacky Montmain, Marcel Staroswiecki, and Louise Travé-Massuyès. A Comparative Analysis of AI and Control Theory Approaches to Model-based Diagnosis. In *Proceedings of the 14th European Conference on Artificial Intelligence (ECAI-00)*, 2000.
- [32] Marie-Odile Cordier, Philippe Dague, Michel Dumas, François Lévy, Jacky Montmain, Marcel Staroswiecki, and Louise Travé-Massuyès. AI and Automatic Control Theory approaches of model-based diagnosis : links and underlying hypotheses. In *Proceedings of the 4th IFAC International Symposium on Fault Detection, Supervision and Safety for Technical Processes (Safeprocess'2000)*, 2000.
- [33] Charles B. Cross. Max Black on the Identity of Indiscernibles. *Philosophical Quarterly*, 45, 1995.
- [34] Newton da Costa and Steven French. *Science and Partial Truth: A Unitary Approach to Models and Scientific Reasoning*. Oxford University Press, 2003.
- [35] Randall Davis. Diagnostic reasoning based on structure and behavior. *Artificial Intelligence*, 24, 1984.
- [36] Randall Davis and Walter Hamscher. Model-Based Reasoning: Troubleshooting. Technical report, M.I.T., 1988.
- [37] Johan de Kleer. *Local Methods for Localizing Faults in Electronic Circuits*. Artificial intelligence memo. Defense Technical Information Center, 1976.
- [38] Johan de Kleer. Dynamic Domain Abstraction Through Meta-diagnosis. In *SARA*, 2007.
- [39] Johan de Kleer, Alan K. Mackworth, and Raymond Reiter. Characterizing Diagnoses and Systems. *Artificial Intelligence*, 56, 1992.
- [40] Johan de Kleer and Brian C. Williams. Diagnosing Multiple Faults. *Artificial Intelligence*, 32, 1987.
- [41] Johan de Kleer and Brian C. Williams. Diagnosis with Behavioral Modes. In *Proceedings of the 11th International Joint Conference on Artificial Intelligence (IJCAI-89)*, 1989.

- [42] John Earman. *A Primer on Determinism (The Western Ontario Series in Philosophy of Science)*. Edited by Robert S. Butts. D. Reidel Publishing Company, 1986.
- [43] Albert Einstein and Alice Calaprice. *The New Quotable Einstein*. Edited by Alice Calaprice. Princeton University Press, 2005.
- [44] Albert Einstein, Boris Podolsky, and Nathan Rosen. Can quantum mechanical description of physical reality be considered complete? *Physical Review*, 1935.
- [45] Erik Frisk and Lars Nielsen. Robust residual generation for diagnosis including a reference model for residual behavior. *Automatica*, 42, 2006.
- [46] Ronald Fagin, Joseph Y. Halpern, Yoram Moses, and Moshe Y. Vardi. *Reasoning about Knowledge*. The MIT Press, 1995.
- [47] Kenneth D. Forbus and Johan de Kleer. *Building Problem Solvers*. M.I.T. University Press, 1993.
- [48] Steven French and Décio Krause. *Identity in Physics: A Historical, Philosophical, and Formal Analysis*. Oxford University Press, 2006.
- [49] Peter Thomas Geach. *Reference and Generality: An Examination of Some Medieval and Modern Theories*. Cornell University Press, 1980.
- [50] Michael Genesereth. The use of design descriptions in automated diagnosis. *Artificial Intelligence*, 24, 1984.
- [51] Donald Gillies. The Duhem Thesis and the Quine Thesis. In Martin Curd and J. A. Cover, editor, *Philosophy of Science: The Central Issues*. W. W. Norton and Company, 1998.
- [52] Richard Wesley Hamming. Error Detecting and Error Correcting Codes. *The Bell System Technical Journal*, 1950.
- [53] Walter Charles Hamscher. *Model-based troubleshooting of digital systems*. MIT, 1988.
- [54] Katherine Hawley. *How things persist*. Oxford University Press, 2002.
- [55] Katherine Hawley. Identity and Indiscernibility. *Mind*, 118, 2009.
- [56] Shawn Hedman. *A First Course in Logic: An Introduction to Model Theory, Proof Theory, Computability, and Complexity*. Oxford University Press, 2004.
- [57] Carl Hempel. Deductive-Nomological vs. Statistical Explanation. In Herbert Feigl and Grover Maxwell, editor, *Scientific Explanation, Space, and Time, Minnesota Studies in Philosophy of Science, Volume III*. University of Minnesota Press, 1962.
- [58] Carl Hempel. *Aspects of Scientific Explanation and Other Essays in the Philosophy of Science*. The Free Press, 1965.

- [59] Carl Hempel and Paul Oppenheim. Studies in the Logic of Explanation. *Philosophy of Science*, 1948.
- [60] Ioan Hepes, Denys Bernard, Jose Dekneudt, Vincent Cheriére, and Nuno Belard. Comparison of possible approaches of centralized diagnostic. Technical report, Airbus, 2009.
- [61] Jaakko Hintikka. *Knowledge and belief: an introduction to the logic of the two notions*. Cornell University Press, 1962.
- [62] Wilfrid Hodges. *Model Theory*. Encyclopedia of Mathematics and its Applications. Cambridge University Press, 1993.
- [63] David Hume. *A Treatise of Human Nature*. Edited by L.A. Selby-Bigge and P.H. Niddich. Oxford University Press, 1978.
- [64] Aeronautical Radio, Inc. ARINC Report 624-1: Design Guidance for Onboard Maintenance System. Technical report, Aeronautical Radio, Inc, 1993.
- [65] R. Isermann and P. Ballé. Trends in the application of model-based fault detection and diagnosis of technical processes. *Control Engineering Practice*, 1997.
- [66] Immanuel Kant. *Critique of Pure Reason (The Cambridge Edition of the Works of Immanuel Kant)*. Edited by Paul Guyer and Allen W. Wood. Cambridge University Press, 1999.
- [67] Saul Kripke. Semantical Analysis of Modal Logic. *Zeitschrift für mathematische Logik und Grundlagen der Mathematik*, 1963.
- [68] Mattias Krysander and Mattias Nyberg. Statistical Properties and Design Criteria for Fault Isolation in Noisy Systems. In *19th International Workshop on Principles of Diagnosis (DX-08)*, 2008.
- [69] Tolga Kurtoglu, Sriram Narasimhan, Scott Poll, David Garcia, Lukas Kuhn, Johan de Kleer, Arjan van Gemund, and Alexander Feldman. First international diagnosis competition 2009. In *Proc. DX-09 20th Int. Workshop on Principles of Diagnosis*, 2009.
- [70] Tolga Kurtoglu, Sriram Narasimhan, Scott Poll, David Garcia, Lukas Kuhn, Johan de Kleer, Arjan van Gemund, and Alexander Feldman. Towards a framework for evaluating and comparing diagnosis algorithms. In *Proc. DX-09 20th Int. Workshop on Principles of Diagnosis*, 2009.
- [71] Gianfranco Lamperti and Marina Zanella. Diagnosis of discrete-event systems from uncertain temporal observations. *Artificial Intelligence*, 137, 2002.
- [72] Pierre Simon Marquis De Laplace. *A Philosophical Essay On Probabilities*. Translated by F. W. Truscott and F. L. Emory. Cosimo Classics, 2007.
- [73] David Lewis. Many, but Almost One. In *Ontology, Causality, and Mind: Essays on the Philosophy of D. M. Armstrong*. Cambridge University Press, 1993.

- [74] David Marker. *Model Theory: An Introduction*. Springer-Verlag, 2002.
- [75] Jolly Mathen. On the Inherent Incompleteness of Scientific Theories. Unpublished. Available in <http://philsci-archive.pitt.edu/2299/>, 2004.
- [76] John McCarthy. Epistemological Problems of Artificial Intelligence. In *Proceeding of the 5th International Joint Conference on Artificial Intelligence (IJCAI-77)*, 1977.
- [77] John McCarthy and P. J. Hayes. Some Philosophical Problems from the Standpoint of Artificial Intelligence. In D. Mickie, editor, *Machine Intelligence 4*. American Elsevier, 1969.
- [78] Igor Mozetic and Christian Holzbaur. Controlling the Complexity in Model-Based Diagnosis. In *Annals of Mathematics and Artificial Intelligence*. John Wiley and Sons, 1993.
- [79] Pandurang Nayak and Alon Y. Levy. A Semantic Theory of Abstractions. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI-94)*, 1994.
- [80] Thomas H. Naylor and J. M. Finger. Verification of Computer Simulation Models. *Management Science*, 1967.
- [81] Nils J. Nilson. Logic and artificial intelligence. *Artificial Intelligence*, 47, 1991.
- [82] Systems Engineering Handbook Development Team of the International Council on Systems Engineering. *Systems Engineering Handbook (version 3): A guide for system life cycle processes and activities*. International Council on Systems Engineering (INCOSE), 2006.
- [83] Byunghyun Oh. Towards a mature multi-program Diagnosis Performance Assessment Tool. Master's thesis, Institut Supérieur de l'Aéronautique et de l'Espace: Ecole Nationale Supérieure de l'Aéronautique et de l'Espace, 2011. Supervisors: Nuno Belard (Airbus) and Pierre Siron (ISAE-SUPAERO).
- [84] Ron Patton, Paul Frank, and Robert Clark. *Issues of Fault Diagnosis for Dynamic Systems*. Springer, 2000.
- [85] David Poole. Normality and Faults in Logic-Based Diagnosis. In *Proceedings of the 11th International Joint Conference on Artificial Intelligence (IJCAI-89)*, 1989.
- [86] David Poole. Explanation and Prediction: An Architecture for Default and Abductive Reasoning. *Computational Intelligence*, 5, 1993.
- [87] David Poole, Randy Goebel, and Romas Aleliunas. Theorist: A logical reasoning system for defaults and diagnosis. In Nick Cercone and Gordon McCalla, editor, *The Knowledge Frontier: Essays in the Representation of Knowledge*. Springer, 1987.
- [88] Karl Popper. Science as Falsification. In *Conjectures and Refutations*. Routledge and Kegan Paul, 1963.

- [89] Karl Popper. *The Logic of Scientific Discovery (Routledge Classics)*. Routledge, 1977.
- [90] Raymond Reiter. A Theory of Diagnosis from First Principles. *Artificial Intelligence*, 32, 1987.
- [91] Pauline Ribot, Yannick Pencolé, and Michel Combacau. Diagnosis and prognosis for the maintenance of complex systems. In *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics (SMC'09)*, 2009.
- [92] Wesley C. Salmon. *Causality and Explanation*. Oxford University Press, 1998.
- [93] Robert G. Sargent. Verification and validation of simulation models. In *Proceedings of the 39th conference on Winter simulation: 40 years! The best is yet to come*, 2007.
- [94] Raymond A. Serway and John W. Jewett. *Physics for Scientists and Engineers*, volume 1. Brooks Cole, 2009.
- [95] Murray Shanahan. *Solving the Frame Problem: A Mathematical Investigation of the Common Sense Law of Inertia*. The MIT Press, 1997.
- [96] Mark Shirley and Randall Davis. Generating Distinguishing Tests based on Hierarchical Models and Symptom Information. In *Proceedings of the International Conference on Computer Design: VLSI in Computers*, 1983.
- [97] Yoav Shoham. *Reasoning About Change*. M.I.T. University Press, 1987.
- [98] Theodore Sider. *Four-Dimensionalism: An Ontology of Persistence and Time*. Oxford University Press, 2003.
- [99] Francesco Stella. Diagnostic Performance Assessment Tool: enhancing performance evaluation at Airbus. Master's thesis, Institut Supérieur de l'Aéronautique et de l'Espace: Ecole Nationale Supérieure d'Ingénieurs de Constructions Aéronautiques, 2011. Supervisors: Nuno Belard (Airbus) and Tanguy Pérennou (ISAE-ENSICA).
- [100] Peter Struss. *What's in SD?: Towards a theory of modeling for diagnosis*, pages 419–449. Morgan Kaufmann Publishers Inc., 1992.
- [101] Peter Struss and Oskar Dressler. “Physical Negation” - Integrating Fault Models into the General Diagnostic Engine. In *Proceedings of the 11th International Joint Conference on Artificial Intelligence (IJCAI-89)*, 1989.
- [102] Frederick Suppe. *The Structure of Scientific Theories*. University of Illinois Press, 1974.
- [103] Frederick Suppe. *The Semantic View of Theories and Scientific Realism*. University of Illinois Press, 1989.
- [104] Patrick Suppes. What is a scientific theory? In Sidney Morgenbesser, editor, *Philosophy of Science Today*. Basic Books, 1967.

- [105] Patrick Suppes. A comparison of the meaning and uses of models in mathematics and the empirical sciences. In *Studies in the Methodology and Foundations of Science: Selected Papers from 1951 to 1969*. D. Reidel Publishing Company, 1969.
- [106] Alfred Tarski. The Concept of Truth in Formalized Languages. In *Logic, Semantics, Metamathematics*. Oxford University Press, 1936.
- [107] CHOCO Team. choco: an Open Source Java Constraint Programming Library. Research report, Ecole des Mines de Nantes, 2010.
- [108] Michael Thielscher. The Qualification Problem: A solution to the problem of anomalous models. *Artificial Intelligence*, 131, 2001.
- [109] George Vachtsevanos, Frank L. Lewis, Michael Roemer, Andrew Hess, and Biqing Wu. *Intelligent Fault Diagnosis and Prognosis for Engineering Systems*. John Wiley and Sons, 2006.
- [110] Ezio Vailati. *Leibniz and Clarke: A Study of Their Correspondence*. Oxford University Press, 1997.
- [111] Bas van Fraassen. The semantic approach to scientific theories. In N.J. Nersessian, editor, *The Process of Science*. Martinus Nijhoff, 1987.
- [112] Daniel S. Weld. Reasoning about model accuracy. *Artificial Intelligence*, 56, 1992.
- [113] John A. Wheeler. *Frontiers of Time*. North-Holland Publishing, 1979.
- [114] John A. Wheeler. The Quantum and the Universe. In Mario Pantaleo and Francesco de Finis, editor, *Relativity, Quanta and Cosmology*, volume 2. Johnson Reprint Corporation, 1979.
- [115] John A. Wheeler. Beyond the Black Hole. In Harry Woolf, editor, *Some Strangeness in the Proportion*. Addison-Wesley, 1980.
- [116] John A. Wheeler. The computer and the universe. *International Journal of Theoretical Physics*, 21, 1982.
- [117] John A. Wheeler. Information, physics, quantum: The search for links. In Wojciech Zurek, editor, *Complexity, Entropy, and the Physics of Information*. Addison-Wesley, 1990.
- [118] David Wiggins. Identity and spatio-temporal continuity. *Australasian Journal of Philosophy*, 50, 1967.
- [119] David Wiggins. On Being in the Same Place at the Same Time. *Philosophical Review*, 77, 1968.
- [120] David L. Yeung and Raymond H. Kwong. Fault Diagnosis in Discrete-Event Systems: Incomplete Models and Learning. In *Proceedings of the 2005 American Control Conference*, volume 5, 2005.

- [121] Xiangfu Zhao and D. Ouyang. Improved algorithms for deriving all minimal conflict sets in model-based diagnosis. In *Proceedings of the 3rd International Conference on Intelligent Computing (ICIC-07)*, 2007.
- [122] Xiangfu Zhao and Dantong Ouyang. A method of combining SE-tree to compute all minimal hitting sets. *Progress in Natural Science*, 16, 2006.