Formal Methods for Safe Design of Autonomous Systems Dedicated to Risk Management

Sophie Coudert, Tullio Tanzi

Formal Methods for Safe Design of Autonomous Systems Dedicated to Risk Management

Sophie Coudert and Tullio Joseph Tanzi

Telecom ParisTech, Sophia Antipolis, France
{sophie.coudert,tullio.tanzi}@telecom-paristech.fr

Abstract. A new generation of Autonomous systems (UAVs, ROVERs, etc.) is coming that will help improve the situational awareness and assessment, especially in difficult conditions like disasters. Rescuers should be relieved from time-consuming data collection tasks as much as possible and at the same time, Autonomous systems should assist data collection through a more insightful and automated guidance thanks to advanced sensing capabilities. In order to achieve this vision, two challenges must be addressed though. The first one is to achieve a sufficient autonomy. The second one relates to the reliability with respect to accidental (safety) or even malicious (security) risks. This however requires the design of new embedded architectures to be more autonomous, while mitigating the harm they may potentially cause. Increased complexity and flexibility requires resorting to modelling, simulation and formal verification techniques in order to validate such critical aspects.

Keywords: Autonomous systems, Formal methods, Safety, Security.

1 Introduction

According to Guhar-Sapir and Hoyois [14], in 2012, naturally triggered disasters (earthquakes, landslides, and severe weather, such as tropical cyclones, severe storms, floods) killed a total of 9,655 people, and 124.5 million people became victims, worldwide. Although those numbers were well below the 2002-2011 annual averages (107,000 people killed and 268 million victims), economic damages did show an increase to above-average levels (143 billion USD). When a natural disaster occurs in a populated area, it is mandatory to organize disaster management operations quickly and eff ectively in order to assist the population, to reduce the number of victims, and to mitigate the economic consequences [14], [9], [23], [22]. A non-optimal organization causes supplementary losses and delays in resuming the situation to normal (http://www.un-spider.org/).

At any time, the rescue teams need immediate and relevant information concerning the situations they have to face: disaster evolution, surviving persons, critical zones, access to refugee camps, spread assistance tools, etc. The required information is provided by a comprehensive data handling system fed with data generally produced by organizations and space agencies involved in the International Charter Space and Major Disasters.

As explained in [23], new approaches and the use of new technologies are required for a more efficient management, before, during, and after a crisis. Every specific action at each step of the crisis must be specifically taken into account. For that purpose, new dedicated tools and methodologies are required to better handle crisis situations.

We present in this paper a new approach for which a new generation autonomous systems may help improve the response of rescue teams. In particular, we discuss how autonomous systems may support rescue teams and help them to more systematically explore their surroundings. We discuss the important challenges that must be addressed in terms of navigation and detection autonomy, as well as reliability in terms of safety and security, in order to enable the seamless use of such systems by non-experts persons.

1.1 Potential Applications of Autonomous Systems

We have identified three critical fields in the relief missions: i) communications and coordination, ii) recognition field, and finally iii) search operations.

During such an event, maintaining a communication link with the various actors of the response on the one hand and with victims on the other hand is crucial. Unfortunately, when the communication infrastructure has been hit, rescue teams rely essentially on radios or satellite communications. This link remains essential even in the face of non-catastrophic circumstances, like for instance a major black-out in a network (electricity, water, etc.).

Autonomous systems might extend the communication range available as they may be deployed as mobile radio relays. They may also convey messages in a disruption tolerant network (DTN) fashion, during their normal operations, typically between the actors involved. Data sensing results have to be communicated as they are produced, and will serve for the coordination of relief operations. In this sense, the systems should also be autonomous in deciding which data to pre-process and to communicate in order to establish operational priorities.

The detection and the monitoring of the impact of natural disasters on terrain are mainly performed by space borne and air borne relying on radio and optical instruments. Due to limitations in the time window observation attached to optical instruments (i.e. no observation at night or in presence of cloud cover), radio observations are available 24/7 and relatively insensitive to atmospheric conditions: these are therefore particularly useful during the “Response phase” when information must be delivered to the disaster cell with a delay as short as possible delay [27], [18], [26].

Autonomous systems may bring significant improvements with respect to those issues. They can be easily equipped with various kinds of sensors in addition to optical ones depending on their potential mission. Their capacities make it easy to observe below a cloud cover. For example, search and rescue teams may carry UAVs and deploy them upon need on site, for instance to explore some flooded area in order to find a practicable path to victims, or a ruined building. In this respect, UAVs extend the exploration range of rescue teams while at the same time improving their safety in areas that may reveal dangerous for their own safety. The senseFly UAV has for instance demonstrated the automated mapping capabilities of small drones and how they could improve the lives of victims in the aftermath of the Haïti 2010 earthquake, by enabling the authorities to quickly draw maps of devastated areas [4].

After a brief presentation of context and an introduction to safety and security aspects (sections 1 and 2), section 3 is a brief high level introduction to formal methods. Section 4 provides some more concrete example of applying them, in the domain of autonomous systems and risk avoidance. Section 5 is a discussion about present and future development followed by conclusion.

2 Reliability: Safety and Security

Legal and ethical constraints arise out of potential risks incurred by the use of an autonomous systems with respect to both victims and rescuers. For instance, a UAV crash may cause damages and harm people; as another example, privacy requires the release of any footage of a disaster to be controlled. Satisfying those constraints requires taking into account the risks linked with fault-tolerance (like for instance in case of the violation of real-time deadlines of safety-critical tasks) and security (sensed data piracy, hijacking, etc.) and to prevent them with adequate countermeasures. Mitigating or preventing those risks involves the introduction of multiple security and safety mechanisms into components as well as in the overall architectural design, which we detail per domain below.

A common way to ensure that a sensor (navigation or payload) has not failed or that its data have not been corrupted by interference, noise or attacks is to perform a plausibility check. Those checks can be done on sensor data in isolation. Many embedded functions also rely on the data produced by several sensors, and checks will consist in ensuring their consistency. Typically, the stabilisation system relies on data issued from the gyroscope, altimeter, etc. Such a verification must be used in our system architecture to improve both the safety of critical function and to limit the impact of attacks.

The deployment of Autonomous systems for such applications will also bring up societal challenges. Indeed, the apparition of a UAV may be terrifying to an unprepared victim (Predator Syndrome), which might reduce the effectiveness of the detection operations. In contrast, victims may not notice UAVs flying at a high altitude and therefore fail to signal their position as they would for an aircraft. New standards will probably have to be defined in this respect.

Furthermore, in critical applications regarding safety, a solution must be validated before it is applied on the field. It is forbidden to test solutions directly on the real system for safety reasons. Any attempt of modification or intrusion of a safety system must be approached with the utmost care. The proposed solutions must be validated, and, if possible, formally validated. It is the case of prevention systems for accidents, installed on highways. They also are in charge of gathering information in the case of an accident actually occurring. In this context, we aim to provide an approach to validate the behaviour of a critical system before its effective realization (rapid prototyping). Since the tests are not covering all the possibilities, we then move to formally verifyingthe critical properties. This formal analysis is made possible by the mathematical grounds used.

Regarding the critical applications, meaning the applications for which a system failure could cause important damages to the people and/or the installations, it is possible to gain enough confidence about the systems functioning by imposing suitable development and validating methods. Norms indicating the methods to follow can be found. For instance, the techniques used in a safety functional analysis highlight external specifications defects that may have gone unnoticed during the various stages of the study. The past few years have witnessed progresses made in the field of operating reliability for the technical parts of risky systems. Nevertheless, some issues still remain:

- What level of reliability should we allocate to each component?
- How do we assess this reliability?
- What are the main default risks?
- How do we identify them?
- How do we measure them?
- …

To conclude this first part, it is necessary to constitute an approach based on formal methodology able to insure the safety of the autonomous system behaviour. Mathematical models and formal methods are tools with interesting features to ensure these objectives.

3 Formal Methods Overview

As pointed by [21] in a chapter about formal methods, mathematical analysis modelling is not new in engineering. The specificity of formal methods is their close relation to computer science, mechanisation and automating. They originate in modern formal logics which characterises reasoning rules as pure shapes and then allow to mechanically check valid reasoning as strict combination of valid rules. The soundness of such reasoning relies on the mathematical proof of each rule — the inference it allows is always true (roughly speaking, in set theory) — and on the mathematical proof of the combination principles: combining valid rules leads to valid deductions or inferences. This kind of approach has then been extended to other way of reasoning on mathematical models such as model checking for example which allows to establish properties on state machines by going through all possible states.

Thus the first historical interest of formal tools was to avoid human mistakes in reasoning. The profit was a very strong guaranty but the cost was very high. Indeed, formal reasoning requires each elementary step be explicit and is thus very complexe. For example, proving that $a+b+c+d+e=e+a+c+b$ requires multiple application of rules $x+y=y+x$ and $(x+y)+z=x+(y+z)$. Model checking is limited by the number of states of FSM (Finite State Machines), etc. On the other hand, formalizing allows some total or partial automating. As computers were slow, it was of limited interest but as computing capabilities increased a lot, things have changed. And the last three decades have seen the development of many enhancements, complementary technics, associated tools, methodologies, etc. (see [5]). Thus, although intrinsic complexity and induced cost remains an obstacle to the substantial use of formal methods in industry, the scope of these methods has extended to more than only guaranteeing critical small systems or small critical parts of systems.

Experience around formal methods leads to the emergence of some keys trends in recurrent difficulties to apply them and solutions to overcome these difficulties. These solutions have led to develop knowledge useful beyond the simple aim to rigorously prove properties. Here we summarise this in three points without claiming to be exhaustive.

Abstraction, refinement and early modelling. The idea is simple: omitting information which does not matter with respect to an expected property makes it easier to establish this property. For example, you can ignore identities of people in a queue if you are only interested in the size of the queue. Abstraction provides simpler models and thus significally simplifies applying formal technics. It reduces the number of states of state machines for model checking. It reduces the number of formulas and makes them simpler for logically specified properties. In brief, it makes checking feasible where it would not be without abstraction. The major issue for a reliable guaranty is then to ensure that omitted information actually doesn't matter. Much scientific results in formal methods address this concern: they characterize reliable relations between abstract and concrete models to ensure good abstractions.

As shown by **Error! Reference source not found.**, refinement is the methodological counterpart of abstraction: mathematical relations are similar and the diff erence is the order in which models are produced. Abstraction extracts property-relevant information from a more concrete model while refining adds details to an abstract model to move toward complete model.

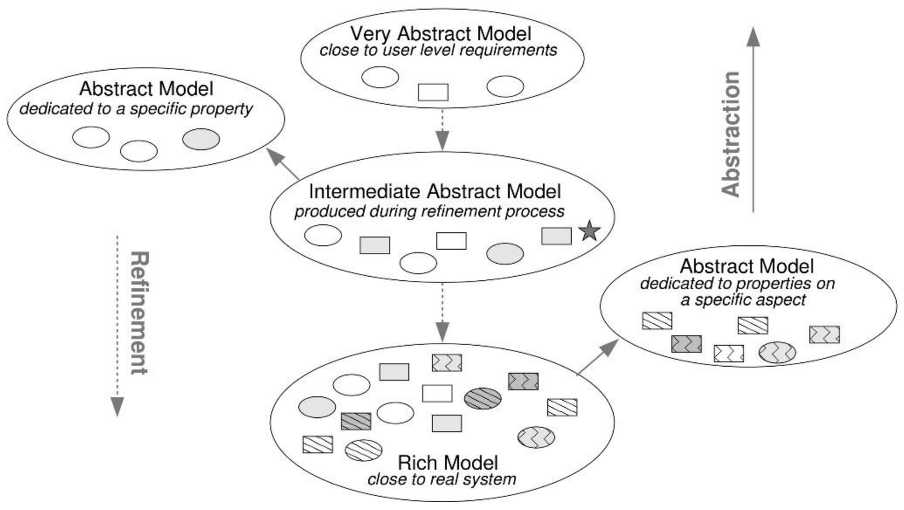
Very Abstract Model
close to user level requirements
Abstract Model
dedicated to a specific property
Abstraction
Intermediate Abstract Model
produced during refinement process
Refinement
Abstract Model
dedicated to properties on a specific aspect
Rich Model
close to real system

Fig. 1. Abstraction – Refinement

Properties can be checked at each level and mathematical guaranty about relations between models ensures high level properties are inherited by lower levels. Another similar aspect is composition: diff erent parts or agents of a system can be specified separately and these specifications are then combined using specific rules. Preservation (or not) of properties by composition is also mathematically assured but additional verifications are sometime necessary.

The perfect warranty is proof (interactive or automatic, depending of method) but the amount of induced work often leads to compromises. Despite this restriction, experience shows that using techniques of formal methods and the use of abstraction, very early in the design process seriously improves the reliability of the result. Another gain is early detection of high level design errors. It avoids the enormous cost induced by the late detection and long and costly ex-post checks. Formal guaranty applies only in the case of mathematical models. Human expertise is very important both to design the detailed model corresponding to the real system and to specify the required high level properties.

Diversifying the use of formal models. Not everything can be proved, but once formal model have been produced, they are also used to apply a wide variety of techniques taking benefit of rigor and abstraction. Here we just give an idea of some of these approaches. Formal techniques allow rapid prototyping. Formal model can be animated and, for example, used for validation: a human can observe the global behaviour of the system before it is implemented. Using computing and abstraction, rapid simulation can be implemented by optimized algorithms and although all paths and states of the system are not covered, the automatically tested part of the system is then much more substantial compared to previous simulation approaches. Formal specification can also be used to automatically produce test sets with a good understanding of the scope of these sets. As high level specification describe "what" the system does and not "how", they can be used to provide infered high level results to compare with the expected ones (i.e. they provide an "oracle"). It is well known that confronting two solutions to obtain a safer result is a good approach when these solutions are very diff erent; here the specification plays the role of one solution. Some works exists also about automatic generation of human readable documentation. As development of formal methods is quite recent, complementary tools using their models continue to emerge nowadays and this presentation is not exhaustive. And last but not least, it is widely recognized that formal modelling alone suffices to highly enhance the design process. Indeed as formal specifying is exigent, it forces to make all requirement precise and unambiguous which leads to debugging them at the beginning of the design process, which is very rentable.

Making specification more user-friendly and providing methodologies. Models must properly represent reality and thus be validated by humans. Unfortunately expertise both in application domain and formal notation is rare which compromises the use of formal technics. Tools are developed to overcome this problem. For example, graphical notations are used to describe systems in an intuitive way and the result is then automatically translated into formal description. Previously mentioned animation of models or automatic generation of natural language helps humans to confront formal specifications with their expectations without formal expertise. Such tools are often integrated in environments with associated methodologies combining formal and unformal verifications. Much has been done to make use of formal methods more attractive and convincing and to allow experts of application domain to play their role.

It would be unrealistic to believe that use of formal methods can be done without formal method experts. The way in which specifications are written has a significant impact on the efficiency of the tools or the size of the proofs. Scientific work continues to propose guidelines, methodologies and associated tools for finding good abstraction and structuring. Concrete expertise is also required on tools themselves. For example in some context a semi-automatic theorem prover may be more powerful using a property written “b=a” than using the same property written “a=b”. The lack of such expertise is an obstacle but it is an inherent problem as formal methods are recent. The situation is changing and some businesses begin to have their own team of specifiers. For example Parisian metropolitan company (RATP) uses the B method since the nineties and has its own team of specialists.

As formal methods have essentially been developed by mathematicians moving to computer science, they naturally have been initially applied to software engineering. As a paradox, the programmer’s world was not familiar with mathematical modelling and it has probably put the brakes on their integration in real practice. Moreover time to market consideration have become more important than safety in this domain and despite all previously presented enhancements, historical cost of formal methods have been dissuasive. Of course, much of these enhancements and specialisations have targeted software engineering. But in general they may be useful for a lot of other disciplines. Logics can not only model software but potentially anything in the world. This appears in recent developments. They have been applied to computer hardware verification and then more generally to hardware–software system design such as embedded systems. Security aspects, i.e protection against attacks are also addressed. More recently, they are extended to cyber-physical systems ([13][10]) and the question of modelling not only discrete time but also continuous aspects arise. And as illustrated in section 4.3 new applications appear, for example to validate human-machine interaction([8]).

Today formal methods have attained a certain level of maturity and are relevant to explore new domains and particularly those were safety is crucial. For all these reasons we believe that formal methods may yield interesting benefits to “risk management”. And here we focus on autonomous systems in this context.

4 Autonomous Systems Illustrations

Applications of formal methods already exist in relation to risk management and autonomous systems. In this section we present some of them. The examples of the two first subsections are extracted from our previous works concerning various equipments dedicated to the safety road (cf. 4.1) or autonomous system such as UAV (cf. 4.2). The third one briefly summarise other works in the literature.

4.1 Transportation Example: Motorway

In this illustration, taken from [24], we adopt a synchronous approach to model and analyse the behaviour of devices related to safety on the highways. This approach provides guarantees on system behaviour and allows you to test its behaviour as soon as the design phases (see **Error! Reference source not found.**). The selected graphic synchronous model, which is based on the concepts of States and transitions, allows more communication with the users of the system, without recourse to the underlying synchronous language explicitly. Associated tools facilitate the implementation and operation of test scenarios. These tests can also be carried out interactively. Even more interesting is the possibility of formally establishing properties of safety. The knowledge acquired during the design phase, i.e. before the actual performance of the system, can be reflected immediately toward the design team.

The goal is the definition of an information system for the technical data necessary to maintain the level of viability and security of a highway network. The studied solution proposes to use private communication network buried along the motorway to transmit information. The network used is a basic emergency call network (RAU). If a user comes to use emergency phone at the same time as relief, its communication becomes a priority at the expense of any other operation. In case of malfunction of the classic RAU, the system must be able to offer an alternative of taking over and relief by routing the message to the next connection point operational.

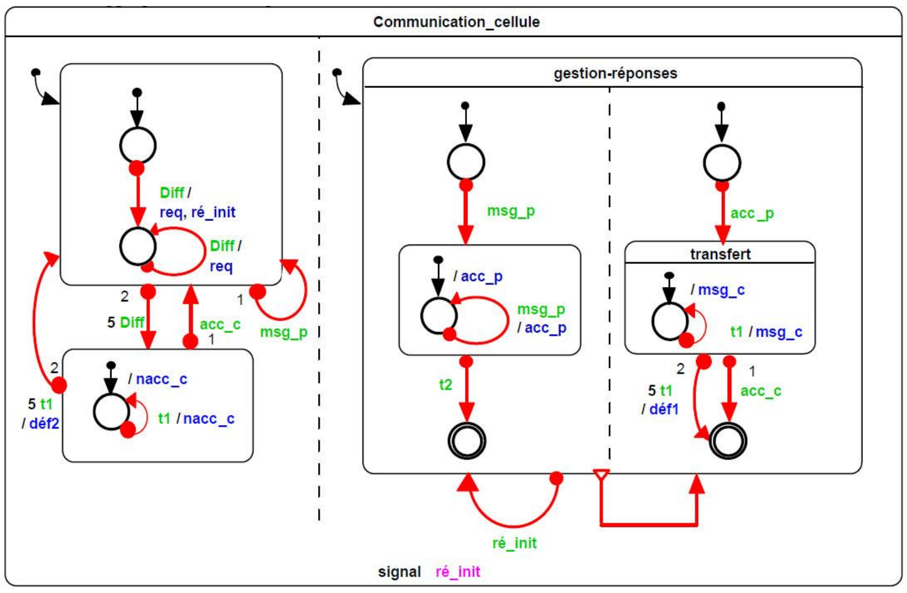

Fig. 2. Example of behaviour modelling

Given the formal nature of the model used, it is possible to prove properties related to the model. The technique that we have adopted is that of symbolic model checking. The idea is to explore exhaustively the State space of the model. This assumes that our model has a finite number of states. However, this space while finite can be so large that the audit programs are unable to analyse it. The use of a symbolic evaluation avoids an explicit construction of the state-space and makes it possible to tackle potentially very large areas.

The models are translated into a system of Boolean equations that represent the control of the program. As there are very eff ective techniques based on the BDD (Binary Decision Diagrams) to evaluate the systems of Boolean equations symbolically, we can apply techniques of symbolic verification to our models.

In a simplified way, a security property expresses the fact that, no matter how the system changes, we won't get unwanted situations. Being able to prove this kind of property is naturally important in critical systems. In our applications, we distinguish between diff erent classes of safety properties: i) Combinatorial properties. They express that a predicate is always true for all available states (for example, a mutual exclusion between actions). ii) Properties of sequential type. These properties are usually expressed by formulas of temporal logics. It comes, for example to show that a dangerous succession of actions will never happen.

In conclusion, we thus have very valuable assistance, because the created scenarios correspond to changes that the tester may not even think about. It is clear that this technique may lead the user to discover anomalies deeply hidden in its specifications or design.

4.2 Autonomous System: Safety Drone Born

We now illustrate some results obtained from the modelling and validation methodology of a mini-drone aimed at autonomously indoor navigating. This drone was implemented on top of an existing drone platform ([19]). It uses a 720p monocular camera to capture 2D images. "Corkscrew flight" allows to reconstruct the UAV 3D environment from the captured 2D images. This requires the synchronization between the motion control (MC) subsystem and the payload management (PM) in charge of image acquisition. The UAV deduces from its environment model flight orders that are sent to the flight control agent.

The approach consists in a three-step methodology: (i) model the functions of the system, (ii) capture the candidate hardware architectures, which is defined in terms of processors, buses and memories, and finally (iii) allocate functions and their communications to the resources of the hardware architecture and study the impact of this allocation with respect to the properties assessed.

In the first iterations of the design, the main purpose of validation is not so much to search for possible deadlock situations usually studied on more accurate models than to study the load of processors and platform buses, and the impact of this load on the flight capabilities of the drone.

For instance, one can clearly see that data from the diff erent sensors (video, attitude, altitude) are sent separately to the ComputingNavigationOrders function that fuses them to interpret the scene before sending commands to the FlightControl.

Modelling tools computes the CPU load resulting from a simulation of the diff erent tasks at hand, as well as a simulation of the inter-CPU information flows triggered by function interactions. This can be used to determine that it would be quite safe to run the emergency tasks on the drone CPU for instance. Other simulations about emergency situations have also been successfully performed to verify that this partitioning of functional tasks can support emergency response functions within acceptable real-time constraints.

4.3 Other Works

A) B-method, metro and railway story. The numerous uses of the Bmethod [2, 11, 12, 17] in the domain of rail transportation [1] is an atypical example of successful early introduction of such a kind of method in industrial practice. Indeed, It relies on interactive proving of set theoretic properties, which requires lot of human expertise and manpower. Thus, thirty years ago, it was difficult to convince industry to move to such approaches. Moreover complete implementation of the method supposes to entirely prove the respect of abstract specifications and required properties by final software. This is particularly demanding and binding unlike model checking, for example, which off ers automatic tools and can be applied punctually on some specific aspects of systems. The counterpart is one of the strongest possible guaranties. Success of B method is essentially due to the eff ort made by its designer for proposing a language closed to programming languages, and also to its first well-known real-size application: METEOR, the first driver-free Parisian metro. As French RATP decided to use formal method to off er strong guaranty, they developed their own team of specifier and this first use of B-method has been followed by other ones. Then B-method provider developed a deep expertise and successfully promotes the use of the method for other railway businesses in the world. A lot of these railway applications concern autonomous and automatic systems (driver-free trains, automatic doors, etc.).

Error! Reference source not found.. provides a very simplified view of the B-method, without any technical consideration. At abstract high level main properties and operations are described without any implementation details. For example, moving can be described without speed considerations. Only necessary order between actions is provided and order depending on implementation choices is ignored. Refining consist in adding details on state and operation steps: introducing speed, door control, etc. Last level is fully deterministic (all order choices have been done) and contains all details of relevant state for software implementation. Language changes during process: high level language handles parallelism while implementation language provides programming language constructs. At each level proof obligations are generated, and proving them (with an interactive prover) ensures that all invariants are satisfied and that low level operation behaviour respects high level operation description. Thus high level invariants have not to be re-proved at low level. This relies on proved mathematical (set theoretic) foundations of the method. To have a total confidence in final software, three other conditions must hold: high level specification is well written, code generation does not introduce errors (there are certified generators) and the computer on which it runs is reliable, of course (which is not in the scope of the method).

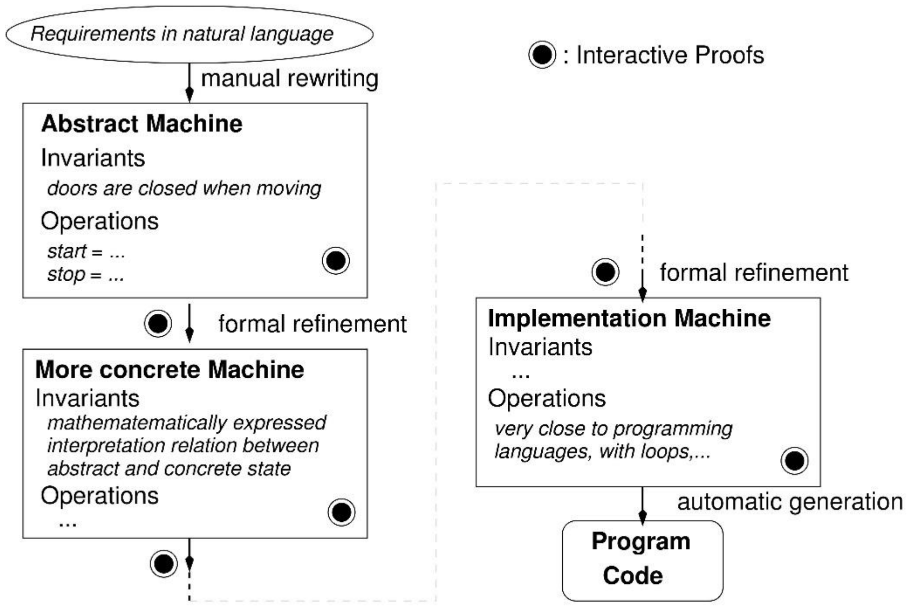

Fig. 3. Intuitive overview of the B method

B) Autonomous vehicle coordination. In this example [6], the addressed problem is coordination of autonomous vehicles and the used technique is “satisfiability decision” (more precisely “satisfying modulo theories”). In mathematical logic, satisfiability and validity are elementary concepts of semantics.

Driver-less vehicles already exist and the state of Nevada even passed legislation allowing them to operate on public roads. One of the main safety requirements is to avoid collisions. As it would be very expensive to put equipments to all crossing, the chosen solution is a distributed one: decision algorithms are located in vehicles which communicate with each other using messages and message acknowledgments.

Satisfiability is well known in logics. Considering a set of properties (logical formulas), the problem is to decide whether all these properties can be true together. Lots of algorithms have been developed and optimized about this. The goal of the authors is to apply this fully automatic proving technique to the vehicle coordination problem. The presented work evaluates feasibility on a simplified but realistic case.

The applied method consists in providing descriptions by way of usual logical formulas. Descriptions concern both the behaviour of a vehicle and the constraints to respect in order to avoid dangerous situations. Vehicles send requests to access areas of the crossing and the communication protocol together with the vehicle behaviours ensure that only one vehicle can occupy an area at any time. Using the satisfiability as technique is then asking the prover if “not safe” is possible, where “not safe” is described by formulas. Authors had to add some invariants (intermediate properties) to help the prover to automatically establish the expected result using induction. Authors also discuss the scope of the case study and find results convincing despite simplifications (essentially, one single crossing).

This approach is interesting from several points of view. Used proof principles, such as induction, are powerful ones and this does not compromise automating. As considered functions have not to be fully interpreted, the modelling can concern continuous time, infinite and dense data sets. For example, location and speed are expressed with reals and they use abstractions such are “near”, “far”, “safe-distance” for reasoning. These abstraction are mathematically related to the continuous representation (location, speed, etc.). Applying such technique as formal method to the addressed problem is quite recent and the work is academic. Although proof is automatic, it requires expertise to find the “good” invariants for helping the prover; the language is mathematical logic and no user-friendly environment is evocated. Results are promising and authors claim to be aiming for the integration of the approach in a general tool for vehicular applications in the future.

C) Confidence in autonomous systems. [15] concerns military application of autonomous vehicles, but the questions they address are very similar to our ones and the paper is a good illustration of our point, with a lot of relevant references. A serious obstacle to the use of autonomous systems is the lack of human trust in autonomy software. Thus most military UAVs are still under remote monitoring. Authors distinguish two kinds of trust: system trust and operational trust. The first one expects system to do what it is supposed to do. The second one expects system to actually help human to complete tasks successfully. The use of formal methods is presented as a solution to seriously increase both kinds of trust. Authors promote the use of formal models to verify that systems satisfy critical requirements. About this purpose, they mention existing intensive use of model checking and also robot’s software synthesis from formal models. And about operational trust they use powerful simulation based on these models and also evocate model checking to validate human-machine interaction. For this, they have developed an approach in which cognitive models predict human behaviour ([16]).

[15] focusses on two main practical problems evocated in section 3: how to produce formal models and how to exploit them in a sufficiently intuitive way to be useful for non-specialists. To solve the first problem, they present a tool which synthetises models from graphical representations of scenarios (ESC: Event Sequence Charts). As some information is sometime missing in ESCs, the tool interactively asks for it during synthesis. To solve the second problem they developed a 3D simulator for autonomy software. These developments are intended to complete the FORMAL (FOrmal Requirements Modeling and AnaLysis) toolset they use. FORMAL integrates a lot of tools for checking consistency, generating invariants, simulating and checking properties at each simulation step, and also model checkers, theorem provers, automatic test generation

and source code synthesis (many references are given in the paper). It is a good illustration of the variety of services formal methods can offer to guaranty autonomy software, and of current developments to make their use easier.

5 Discussion About Present and Future

Presented examples illustrate how existing approaches can be used to guaranty reliability of autonomous systems and in particular autonomous systems for risk management, where safety considerations are not reduced by time-to-market ones. Current methods can be used as they are. But as their development is recent and as they are evolving, future application is promising. From this study we identify some major lines, briefly summarized in the following points. the first two ones are shared with other application domains and the last ones is a short discussion about prospective for the risk area.

Combining methods. All methods are not efficient for the same aspects or properties. For example timed automata may be used for real time constraints but the complexity of timed models leads to avoid them for time independent question. Interactive proof is very powerful but also tedious and costly, thus it should be reserved to chosen aspects. Thus a rich approach would be to combine methods, and also to combine them with non-formal ones. Such approaches already exist. Some merging are easy: for example, as b-method machines are state machine, they can allow some model checking or model animation. Hybrid approaches can strongly enhance confidence in result but require some complementary fundamental research to offer mathematical guaranty of consistence between models of the different methods.

Using formal methods early in design process. As pointed in part 3 experiment shows that using formal methods at requirement and abstract levels is very profitable. A well designed general system (before details are added) is easier to verify, handle and develop. Some approaches such as the B-method or [25] are intrinsically top-down and another main reason for this is that it makes efficient applying of formal methods easier. Another complementary reason is that applying formal methods to already developed systems implies to extract formal models from low-level descriptions and rebuild abstraction where intentional information have disappeared (replaced by operational one). It requires a difficult reverse engineering process. This delicate process is not easy and potentially error-prone.

Prospective. Paragraph 4.*3*.C) already shows that new application domains lead to new ideas about how to use formal models. Other recent evolving may be very interesting for the risk field. Among them, let us cite the extension of these methods to cyber-physical-systems: systems with sensors and strong interaction with physical environment. And another very relevant aspect is that formal methods may not only be used to model computing systems but also environment and physical systems. As the B-method presented above is used to develop software, a more recent version targets system modelling ([3]). Theory is similar but methodology is different: operations are replaced by events; high level specifies the problem and low level specifies the solution. As example, [3] presents the behaviour of traffic lights controlling the access to an island. The high level specification only says that no more than x cars can be on the island at a same time. The high level property is the real safety one and totally abstracts the way to obtain it. Then refinement gradually introduces a bridge, lights, sensors, etc. And only last level introduces the controlling software driving the lights, and its communications with sensors. This approach allows to prove that a solution correctly solves a problem expressed at a pure user level (see also [7]). Its implementation remains difficult today as efficient decomposition (avoiding that adding an event leads to redo all proofs) is not easy. However guidelines are studied for this [20] and experience shows that with good decomposition the amount of proofs is easily balanced by the benefit of the analysis: proofs can be relatively direct without a lot of distortions to cope with the formalism.

It results that beyond designing autonomous systems, it is particularly interesting to explore specialisation of formal methods to risk issues, as people in this field do not have the same needs and the same culture as those working in software development, and as domain requires strong guaranty about multiple aspects.

In order to conceive a risk-sharing system, one must answer those questions well before their exploitation. In this context, it is important to have access to tools allowing the assessment of production and support policies about reliability and efficiency and the systems operational performances, especially its availability, safety, robustness, etc. Using a simulation is part of this process. It allows to support the technological approach upstream and to help overcoming various kinds of difficulties: i) complexity of the system: dynamic, multiple resources interacting, gradual deployment; ii) lack of feedback; iii) necessity of multiple and partial approaches.

6 Conclusion

The design of autonomous systems, such as Drones for example, intended to carry out complex missions is an important challenge. The realisation of such a stand-alone system designed to evolve in its post-disaster conditions is still more demanding, mainly in the design of the architecture that must support this autonomy with a very good level of security and efficiency.

The use of the techniques of simulation on test cases, coupled with mathematical proof techniques allow to size and check the properties and safety features. The analysis focuses on functional and non-functional properties for safety and performance aspects. This new approach allows us to find the limits and to validate the consistency of complex systems without having to build real prototypes. It also lets us collect information on a new system, low-cost and at an early stage of the project. A lot of dysfunction, when using the system, appears when the operational conditions are diff erent from those defined for the design. Checking and verifying all options in a design is not always possible before its realisation.

Complex systems are difficult to test and validate in extreme conditions. Designers can provide the use of a prototype to refine their approach. However, the prototype can hardly be comprehensive and fully representative of the final behaviour. It goes the same for test conditions. By building a virtual prototype, able to model more critical behaviours, it is possible to reproduce a functioning according to operational conditions. This process integration from the early stages of the project, allows having retroactive eff ect immediately on the design.

Tools of modelling and verification are an important improvement to the mastering of complexity. This allows to improve confidence in performance, safety and security properties needed by the real operational conditions. Moreover, present evolution of formal techniques and recent applications to risk management issues are promising attractive opportunities. Benefits for risk field must be increased and diversified in the future.

References

1. Sil4 railway software, http://www.clearsy.com/en/our-specific-know-how/b-method/, 2016
2. Abrial, J.R.: The B-book: Assigning Programs to Meanings. Cambridge University Press, New York, NY, USA (1996).
3. Abrial, J.R.: Modeling in Event-B: System and Software Engineering. Cambridge University Press, New York, NY, USA, 1st edn. (2010).
4. Ackerman, E.: Drone adventures uses uavs to help make the world a better place. IEEE Spectrum, (May 2013).
5. Almeida, J.B., Frade, M.J., Pinto, J.S., de Sousa, S.M.: Chapter 2. An Overview of Formal Methods Tools and Techniques. Springer-Verlag London, 1st edn. (2011).
6. Asplund, M., Manzoor, A., Bouroche, M., Clarke, S., Cahill, V.: A Formal Approach to Autonomous Vehicle Coordination. In: Proc. of the 18th International Symposium of Formal Methods. LNCS, vol. 7436, pp. 52–67. Springer (2012).
7. Banach, R.: The Landing Gear Case Study in Hybrid Event-B, pp. 126–141. Springer International Publishing, Cham (2014).
8. Bolton, M.L., Bass, E.J., Siminiceanu, R.I.: Using formal verification to evaluate human-automation interaction: A review. IEEE Trans. Systems, Man, and Cybernetics: Systems 43(3), 488–503 (2013).
9. Chatterjee, R., Fruneau, B., Rudant, J., Roy, P., Frison, P., Lakhera, R., Dadhwal, V., Saha, R.: Subsidence of Kolkata (Calcutta) city, India during the 1990s as observed from space by diff erential synthetic aperture radar interferometry technique. Remote Sensing of Environment 102 (1-2), 176–185 (2006).
10. Drechsler, R., Khne, U.: Formal Modeling and Verification of Cyber-Physical Systems. In: 1st International Summer School on Methods and Tools for the Design of Digital Systems, Bremen, Germany, September 2015. Springer Vieweg (2015).
11. Fantechi, A.: Twenty-five years of formal methods and railways: What next? In: SEFM 2013, Madrid, Spain, September 23-24, 2013, Revised Selected Papers. pp. 167–183 (2013).
12. Fantechi, A., Fokkink, W., Morzenti, A.: Some Trends in Formal Methods Applications to Railway Signaling, pp. 61–84. John Wiley & Sons, Inc. (2012).
13. Fitzgerald, J., Gamble, C., Larsen, P.G., Pierce, K., Woodcock, J.: Cyber-physical systems design: Formal foundations, methods and integrated tool chains. In: FormaliSE 2015. Florence, Italy, May 18, 2015. pp. 40–46. IEEE (2015).
14. Guha-Sapir, D., Hoyois, P., Below, R.: Annual disaster statistical review 2012: The number and trends. (CRED) Brussels, Belgium (2012).

15. Heitmeyer, C.L., Leonard, E.I.: Obtaining trust in autonomous systems: Tools for formal model synthesis and validation. In: FormaliSE 2015. pp. 54–60. IEEE Press, Piscataway, NJ, USA (2015).
16. Heitmeyer, C.L., Pickett, M., Leonard, E.I., Archer, M.M., Ray, I., Aha, D.W., Trafton, J.G.: Building high assurance human-centric decision systems. Autom. Softw. Eng. 22(2), 159–197 (2015).
17. Lecomte, T., Burdy, L., Leuschel, M.: Formally checking large data sets in the railways. CoRR abs/1210.6815 (2012).
18. Lefeuvre, F., Tanzi, T.: International union of radio science, international council for science (icsu), joint board of geospatial information societies (jbgis). In: United Nations office for outer Space Aff airs (OOSA) (2013).
19. Ranft, B., Dugelay, J.L., Apvrille, L.: 3d perception for autonomous navigation of a low-cost mav using minimal landmarks. In: Proc. of IMAV'2013, Toulouse, France, 17-20 Sept (2013).
20. Sato, N., Ishikawa, F.: Separation of considerations in event-b refinement toward industrial use. In: Proc. of FMSEE&T 2015, co-located with FM 2015), Oslo, Norway, June 23, 2015. pp. 43–50 (2015).
21. Sommerville, I.: Chapter 27. formal methods. In: Pearson (ed.) Software Engineering 9th edition (Teaching Book) (2011).
22. Tanzi, T., Lefeuvre, F.: Radio Sciences and Disaster Management. C.R. Physique 11, 114–224 (2010).
23. Tanzi, T., Perrot, P.: T´el´ecoms pour ling´enierie du risque (in French). editions herms, ed. Collection Technique et Scientifique des T´el´ecoms (2009).
24. Tanzi, T., Andr, C.: Mod´elisation synchrone appliqu´ee la suˆret´e de fonctionnement. In: 12`eme Colloque National de Suˆret´e de Fonctionnement, 28-30 mars 2000. Montpellier - France (2000).
25. Tanzi, T., Apvrille, L., Dugelay, J.L., Roudier, Y.: Uavs for humanitarian missions: Autonomy and reliability. In: Global Humanitarian Technology Conference (GHTC), 2014 IEEE. pp. 271–278. IEEE (2014).
26. Tanzi, T.J., Lefeuvre, F.: The Contribution Of Radio Sciences to Disaster Management. In: Gi4DM 2011. Antalya, Turkey (May 2011).
27. Wilkinson, P., Cole, D.: The Role of the Radio Sciences in the Disaster Management. Radio Science Bulletin 3358, p. 4551 (2010).