Formal Methods in Systems Integration: Deployment of Formal Techniques in INSPEX

Richard Banach, Joe Razavi, Suzanne Lesecq, Olivier Debicki, Nicolas Mareau, Julie Foucault, Marc Correvon and Gabriela Dudnik

Abstract Inspired by the abilities of contemporary autonomous vehicles to navigate with a high degree of effectiveness, the INSPEX Project aims to create a minaturised smart obstacle detection system, which could find use in a wide variety of leading edge smart applications. The primary use case focused on in the project is producing an advanced prototype for a device which can be attached to a visually impaired or blind (VIB) person's white cane, and which, through the integration of a variety of minaturised sensors, and of the processing of their data via sophisticated algorithms, can offer the VIB user greater precision of information about their environment. The increasing complexity of such systems creates increasing challenges to assure their correct operation, inviting the introduction of formal techniques to aid in maximising system dependability. However, the major challenge to building such systems resides at the hardware end of the development. This impedes the routine application of top-down formal methods approaches. Some ingenuity must be brought to bear, in order that normally mutually hostile formal and mainstream approaches can contribute positively towards system dependability, rather than conflicting unproductively. This aspect is illustrated using two strands of the INSPEX Project.

Richard Banach and Joseph Razavi
School of Computer Science, University of Manchester, Oxford Road, Manchester, M13 9PL, U.K., e-mail: {richard.banach, joseph.razavi}@manchester.ac.uk

Suzanne Lesecq, Olivier Debicki, Nicolas Mareau and Julie Foucault
CEA, LETI, Minatec Campus, 17 Rue des Martyrs, F-38054 Grenoble Cedex, France. e-mail: {suzanne.lesecq@cea.fr, olivier.debicki@cea.fr, nicolas.mareau, julie.foucault}@cea.fr

Marc Correvon and Gabriela Dudnik
CSEM SA, 2002 Neuchatel, Switzerland. e-mail: {marc.correvon, gabriela.dudnik}@csem.ch

1 Introduction

The contemporary hardware scene is driven, to a large extent, by the desire to make devices smaller and of lower power consumption. Not only does this save materials and energy, but given the commercial pull to make mobile phones increasingly capable, when small low power devices are incorporated into mobile phones, it vastly increases the market for them. The smartphone of today is unrecognisable (in terms of the facilities it offers) from phones even as little as a decade old. This phenomenon results from ever greater advances in system structure, and from the trend to incorporate minaturised sensing technologies that were well beyond the state of the art a short while ago. This trend continues unabated, and also massively propels advances in the Internet of Things.

The availability of such minaturised devices inspires the imagination to conceive novel applications, previously unrealised due to technological barriers. The INSPEX Project is the fruit of one such exercise in imagineering. Taking the autonomous vehicle [15] as inspiration, along with the data fusion that enables autonomous vehicles to elicit enough information about their environment from the data gathered by a multitude of sensors to navigate sufficiently safely that autonomous vehicles 'in the wild' are forseen within a few years [28, 35], INSPEX aims to minaturise a similar family of sensors to create a device that offers comparable navigational support to a wide variety of smaller, more lightweight applications.

In the remainder of this paper we do the following. In Section 2 we cover the potential application areas for INSPEX, pointing to the key VIB use case that forms the focus of the project. In Section 3 we focus more narrowly on the technical elements of the VIB use case. In Section 4 we address ourselves to the deployment of formal modelling and verification technologies within the INSPEX development activity. We focus on two areas within which formal techniques were deployed in INSPEX, namely in the power management design and in the data acquisition pathway. Section 5 contains discussion and concludes.

2 INSPEX Application Use Cases

Fig. 1 gives an indication of the range of applications that the INSPEX imagineering effort generated. The figure is divided into four broad aplication areas. Working left to right, we start with some examples of small autonomous vehicles. Autonomous navigation for these demands the small size, weight and power requirements that INSPEX seeks to provide. Small airborne drones have demands that are very similar, and as their number increases, their navigation and collision avoidance needs increase correspondingly. Considerations of size, weight and power also impinge on humanoid robots and specialised devices such a floor cleaning robots. INSPEX navigation capabilities will also increase autonomy and flexibility of use for factory based transport robots, which have to be prepared to avoid unexpected obstacles, unless their environment is sufficiently tightly constrained.

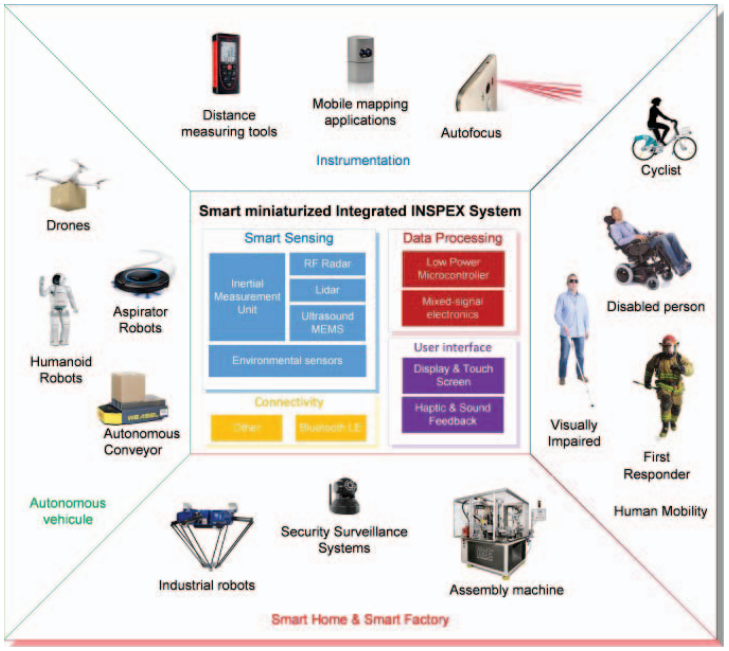

Fig. 1 Potential INSPEX use cases.

At the bottom of Fig. 1 we see some examples concerned with large enclosed environments, such as highly automated factories featuring assembly lines consisting of hundreds of robots. To increase the flexibility of reconfiguration of these, increased autonomy in the participating robots is one necessary ingredient. INSPEX, appropriately deployed, can significantly assist in meeting this requirement. The issue becomes the more forceful when the robots involved are mobile, since along with the need to be more smart, they particularly need to avoid harm to any humans who may be working nearby. Security surveillance systems, traditionally relying on infra-red sensors, can also benefit from the extra precision of INSPEX.

At the top of Fig. 1 we see some examples concerned with distance estimation. Modern distance measuring tools typically make use of a single laser beam whose reflection is processed to derive the numerical result. For surfaces other than smooth hard ones, the measurement arrived at may be imprecise, for various reasons. INSPEX can perform better in such situations by combining readings from a number of sensors. A very familiar application area for such ideas is autofocus in cameras. These days, camera systems (typically in leading edge phones) employ increasingly sophisticated algorithms to distinguish foreground from background, to make up for varying lighting conditions, and generally to compensate for the users lack of expertise in photography. INSPEX can add to the capabilities available to such systems.

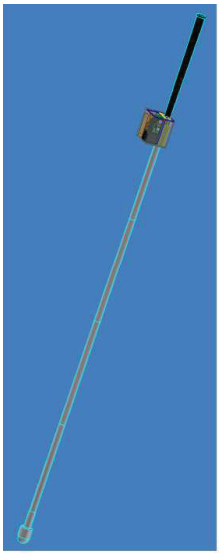

Fig. 2 The INSPEX white cane addon.

On the right of Fig. 1 we see the use cases for human centred applications. We see the VIB use case which forms the focus of the INSPEX project, and which will be discussed in detail later. There are also other prominent use cases. The first responder example includes cases like firefighters, who need to be able to enter hazardous environments such as smoke filled rooms, in which normal visibility is impossible. An aid like an INSPEX device can be of immeasurable help, in giving its users some orientation about the space in which they find themselves, without resorting to tentative feeling about, which is what firefighters are often reduced to. Other applications include the severely disabled who may have impediments to absorbing the visual information from their surroundings. And the able bodied too can benefit from INSPEX, when visibility is severely reduced. Although the cases of heavy fogs which reduced visibility to almost zero are thankfully history, today's mega-cities now feature smogs due to different sources of atmospheric pollution which can be just as bad.

3 The INSPEX VIB White Cane Use Case

Although a large number of use cases are envisaged for a system such as INSPEX, the primary use case addressed within the INSPEX Project is the smart white cane to assist visually impaired and blind persons. Fig. 2 shows a schematic of one possible configuration for the attachment of a smart addon to a standard type of white cane. The white cane application needs other devices to support the white cane addon, in order that a system usable by the VIB community ensues. Fig. 3 shows the overall system architecture.

As well as the Mobile Detection Device addon to the white cane, there is an Audio Headset containing extra-auricular binaural speakers and an inertial measurement unit (IMU) — the latter so that an audio image correctly oriented with respect to 3D space may be projected to the user, despite the user's head movements. Another vital component of the

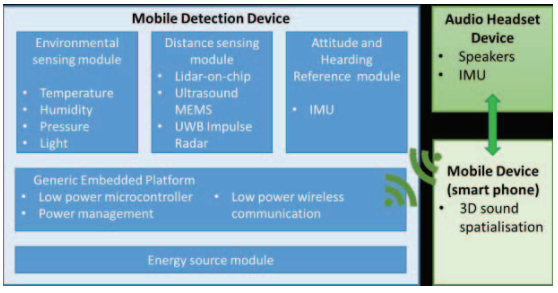

Fig. 3 The architecture of the INSPEX system.

system is a smartphone. This correlates the information obtained by the mobile detection device with what is required by the headset. It also is able, in *smart city* environments, to receive information from *wireless beacons* which appropriately equipped users can access. This enables the whole system to be even more informative for its users.

The white cane add-on contains the sensors that generate the data needed to create the information that is needed by the user. The chief among these comprise a short range LiDAR, a long range LiDAR, a wideband RADAR, and a MEMS ultrasound sensor. Besides these there are are the support services that they need, namely an Energy Source Unit, environmental sensors for ambient light, temperature and humidity, another IMU and a Generic Embedded Platform (GEP).

The main sensors are subject to significant development and minaturisation by a number of partners in the INSPEX project. The short range LiDAR is developed by the Swiss Center for Electronics and Microtechnology (CSEM) and the French Alternative Energies and Atomic Energy Commission (CEA). The long range LiDAR is developed by the Tyndall National Institute Cork and SensL Technologies, while the wideband RADAR is also developed by CEA. The MEMS ultrasound sensor is from STMicroelectronics (STM). Cork Institute of Technology (CIT) design the containing enclosure and support services, while the audio headset is designed by French SME GoSense.

The GEP has a noteworthy challenge to confront. Data from the sensors comes in at various times, and with varying reliability. Distance measurements from the sensors are just that, merely distance data without any notion of direction, or orientation with respect to the user. The latter is elucidated by reference to data from the IMU in the mobile detection device. Data from both the IMU and directional sensors is timestamped, since freshness of data is crucial in providing information to the user that is not only accurate but timely. This enables distance sensor data to be aggregated by time and IMU data.

Once the data has been correctly aggregated, it is passed to the module in the GEP that computes the *occupation grid*. This is a partition of the 3D space in front of the user into cells, each of which is assigned a probability of its being occupied by some obstacle. The occupation grid idea is classical from the autonomous vehicle domain, but in its standard implementation, involves intensive floating point computation [28, 35]. This is too onerous for the kind of lightweight applications envisaged by the concept of INSPEX. Fortunately INSPEX is able to benefit from a highly efficient implementation of the occupation grid, due to a careful analysis of the computations that are needed to derive a good occupation grid result [13]. The integration of all the hardware and software activities described, constitutes a non-trivial complex systems undertaking.

The wide range of sensors and their concomitant capabilities in the INSPEX white cane application is necessitated by the detailed needs of VIB persons navigating around the outdoors environment (in particular). Although a standard white cane can give good feedback to its user regarding the quality and characteristics of the ground in front of them, especially when the ground texture in the urban environment is deliberately engineered to exhibit a range of standard textures signifying

specific structures [31], it gives no information about hazards to be found higher up. It is a fact of life for VIB persons, that, like it or not, unanticipated collisions with obstacles at chest or head height are an unavoidable occurrence [25]. Many VIB persons are prone to wearing some sort of headgear, more or less involuntarily, to try to mitigate the worst effects of such unanticipated high level collisions. The possibility of alleviating this situation, even in the absence of other use cases, makes for ample justification for the development of INSPEX.

4 Formal Modelling and Verification in INSPEX

By now, formal techniques of system development have had a substantial history. After the early years, and the widespread perception that such approaches were 'hard' and did not scale, there was a concerted effort to dispel this view in classic works such as [19, 11, 12]. It was increasingly recognised, especially in niche areas, that formal techniques, wisely deployed, can add a measure of dependability not achievable by other means.[1] It became recognised that tools, particularly ones that worked in a reasonably scalable way,[2] were key to this [34, 33]. This spurred the idea of 'Grand Challenges' in verification, one purpose of which was to both test and further inspire the scalability of tools [23, 38, 39]. Later surveys include [3, 8], and this trend is also evident in [5].

The classic way of applying formal approaches is top-down. One starts with an oversimplified, but completely precise, definition of the desired system. This is then enriched, via a process of formal refinement, to take into account more system detail in order to address more of the system's requirements. Eventually one gets close enough to the code level that writing code is almost a transcription, or the code can be generated automatically.

There are many variations, small and large, on this basic idea. An early account is in [30]. The Z approach is represented by [32, 21]; the VDM apporach is in [22, 17]; TLA+ is in [24]; Alloy in [1]. There are many others. The B-Method, of which more later, is represented by [2, 4, 29].

Accompanying these developments grew the subdiscipline of behaviour oriented, or process oriented descriptions of system behaviour. Early references are [20, 26, 7]. Not long afterwards, it was observed that many process oriented properties of interest for systems conformed to a so-called model checking pattern, and this led to an explosion of research and tool building, since model checking could then be completely automated, leading to tools that could work in a push-button manner, and that could be embedded in development environments, in which they worked 'behind the scenes', i.e. without explicit user control or invocation. Among the tools

[1] In some niche areas, the recognition came as a direct result of painful and expensive failure, the Pentium Bug and Arianne Disaster being iconic examples.

[2] It became apparent at this time that scalable formal tools were not an impossible dream, even if the degree of scalability was not as great as typically found in conventional approaches.

in this style that have proved to be of interest for the INSPEX project are FDR [16], NuSMV [27], Uppaal [36].

Whereas all the preceding approaches relied on there being a model of the system that was presented in a relatively abstract language, the growing power and scalability of tools generated an interest in techniques that worked directly on implementation level code. By now there are many well established tools that input an implementation in a given language such as C or C++, and that take this implementation and then analyse it directly for correctness properties [37]. Very often these properties are predefined runtime correctness properties concerning commonly introduced programmer errors, such as (the absence of) division by zero or (the absence of) null pointer dereference. Some however, e.g. [6, 9] allow more application specific properties to be checked.

While direct checking of implementations would appear to be a panacea for verification, it nevertheless risks overemphasising low level system properties at the expense of the higher level view. When we recognise that deciding what the system *should be* is always a human level responsibility, and that formal approaches can only police the consistency between different descriptions of the same thing, abandoning the obligation to independently consider the abstract high level view of the system risks abandoning a valuable source of corroboration of the requirements that the system is intended to address. It is this kind of 'stereoscopic vision' on what the system ought to do and to be that constitutes the most valuable contribution that a top-down formal approach makes to system development, quite aside from the formal consistency checking.

In normal software developments, one starts the process with a good idea of the capabilities of software in general, so in principle, it is feasible to use a relatively pure top-down approach. Likewise in most hardware developments that take place at the chip level, one starts the process with a good idea of the capabilities of the technology platform that will be used, and working top-down is perfectly feasible (and in fact is unavoidable given the scale of today's chips). In both of these cases deploying top-down formal techniques (if the choice is made to do so) is feasible.

In INSPEX however, the development of the devices at the physical level is a key element of ongoing project activity, and the low level properties of all the devices used in the INSPEX deliverable are contingent and emergent to a significant extent. This makes the naive use of top-down approaches problematic, since there is no guarantee that the low level model that emerges from a top-down development process will be drafted in terms of low level properties that are actually reflected in the devices available, since the constraints on the system's behaviour that are directly attributable to physics are simply incontestable. As a result of this, the approach to incorporating formal techniques in INSPEX was a hybrid one. Top-down and bottom-up approaches were pursued concurrently, with the aim of meeting in the middle.

The next sections cover how this hybrid approach was applied in two of the INSPEX Project's activities, namely the design of the power management strategy for the mobile detection device module, and in the verification of the data pathway from the sensors to the data fusion application.

4.1 Power Management Formal Modelling and Verification

In INSPEX, power management poses a number of challenges. As stated earlier, the concentration of effort in INSPEX is on engineering a suitable outcome at the hardware systems level. Each sensor and subsystem creates its own problems. However they all share a common goal, one common to all mobile systems, of making the smallest demand on the power system that is possible. However, a focus on individual submodules risks paying insufficient attention to issues of coordination. A higher level view offers a number of benefits.

The first benefit is an issue of correct functioning. A naive combination of low level modules, each of them correct in itself, is not guaranteed to generate in a straightforward manner (from a systems level perspective), a globally correct behaviour. For example a submodule might conceivably be left running when it ought not to be running as an unexpected consequence of some complex sequence of events. The second benefit is the issue of global optimality. Focusing on the low level prevents the global optimisation of performance (in this case power saving) by balancing criteria from competing interests originating in diverse submodules.

A formal approach rooted in a higher level view can assist in both of these aspects of the development. Formal techniques are suited *sans pareil* to targeting correctness aspects of a development. Moreover, they are capable of capturing the global consequences of a collection of submodels when they are combined into a single entity, since they do not suffer from the variability of focus that humans can exhibit when they concentrate on one or another aspect of an activity.

Power management design in INSPEX proceeded top-down. From a human perspective this might mean considering broad properties of the power regime first, descending to low level detail at the end — this would fly in the face what has been stated above since what is most incontestable about the design is the low level properties of individual sensors etc. We reconcile these views by observing that formally, 'top level' properties are those that will not be contradicted in subsequent steps of development. This implies that they will be the most primitive rather than the most far reaching among the properties that the system satisfies.

The most primitive properties include the state transition diagrams of the various sensors and other components. Fig. 4 gives an example of a transition diagram for the Bluetooth submodule, rather drastically simplified from the description in [10]. To incorporate this into a wideranging formal model we used the Event-B formal-

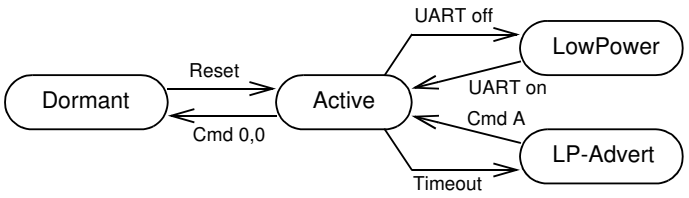

Fig. 4 A simplified Bluetooth transition diagram.

ism [4]. This enables many levels of abstraction to be formally related to each other via refinement, and is supported by the Rodin tool which features many provers and plugins [29]. A state transition diagram such as Fig. 4 can be captured in Event-B in a fragment like:[3]

```
EVENTS                                  ... ...
  Dor2Act                               Act2Dor
    WHEN state = Dormant ∧ Reset          WHEN state = Active ∧ Cmd_0,0
    THEN                                  THEN
      state := Active                       state := Dormant
    END                                   END
  LP2Act                                Act2LP
    WHEN state = LowPower ∧ UART_on       WHEN state = Active ∧ UART_off
    THEN                                  THEN
      state := Active                       state := LowPower
    END                                   END
  LPA2Act                               Act2LPA
    WHEN state = LP_Advert ∧ Cmd_A        WHEN state = Active ∧ Timeout
    THEN                                  THEN
      state := Active                       state := LP_Advert
    END                                   END
```

A formal model such as the fragment above relates to the low level real time software and firmware as follows. Each event in the model corresponds to a software or firmware command, or an interrupt routine. The guard portion, expressed in the WHEN clause of the event, corresponds to the entry condition code in the command, or scheduler code that checks the cause of the interrupt. The event's THEN clause corresponds to the software command body, or the interrupt handler routine. As stated earlier, capturing all the commands and sources of interrupt enables questions of overall consistency to be examined.

Once the low level integrity has been established, other considerations can be brought to bear. A major element is the quantitative aspect. Event descriptions as above are embellished with numerical data regarding the energetic consequences of executing the event, enabling overall conclusions about energy consumptions to be drawn. Finally, considerations of overall power management policy can be layered onto the formal model and made to correspond with the implementation code.

4.2 The Data Acquisition Pathway

Another major area in which formal techniques were deployed in INSPEX to add robustness to the software design was the data acquisition pathway. As outlined earlier, in INSPEX, there are several sensors, each working to different characteristics, but all contributing to the resolution of the spatial orientation challenge that is the *raison d'être* of INSPEX.

[3] For reasons of the confidentiality of the future commercial exploitation of the INSPEX platform, what is shown here is not actual code.

The various INSPEX sensors work at frequencies that individually can vary by many orders of magnitude. For example, the LiDARs can produce data frames with extreme rapidity, whereas the ultrasound sensor is limited by the propagation speed of pressure waves through the air, which is massively slower than the propagation characteristics of electromagnetic radiation. The ultrasound sensor, in turn, can produce data frames much faster than typical human users are able to re-orient their white canes, let alone move themselves to a significant degree, either of which requires a fresh occupation grid to be computed. This means that the data integration performed by INSPEX has to be harmonised to the pace of the human user.

The main vehicle for achieving this is the IMU. The IMU is configured to supply readings about the orientation of the INSPEX mobile detection device addon at a rate commensurate with the needs of human movement. This 'pacemaker rate' is then used to solicit measurements from the other sensors in a way that not only respects their response times but is also staggered sufficiently within an individual IMU 'window' that the energy demands of the individual measurements are not suboptimal with respect to the power management policy currently in force.

The above indicates a complex set of information receipt tasks, made the more challenging by the fact that all the sensors speak to the same receiving hardware. The goal of the information receipt tasks is to harvest a collection of data frames from the individual sensors, each timestamped by its time of measurement, and each related to a before- IMU data frame, and an after- IMU data frame, each itself timestamped. The two successive IMU data frames, and way their data might differ due to user movement, enable the interpolation of orientation of the distances delivered at various times by the other sensors.

Timing is evidently of critical importance in the management of the incoming data. This notwithstanding, all the tasks that handle these information management duties are executed at the behest of the generic embedded device's low level scheduler. The scheduler used belongs to the real time operating system employed in the GEP, which is a version of FreeRTOS [18].

Turning to the formal modelling of what has just been described, it may well seem that the complexity of the situation might defeat efforts to add useful verification to the design. The situation is helped considerably by the existence of a formal model of the FreeRTOS scheduler [14]. This is in the kind of state oriented model based form that can be made use of in the modelling and verification of the data acquisition pathway in INSPEX. Accordingly, the properties of the FreeRTOS scheduler derived in [14] can be translated into the Event-B framework used for INSPEX and then fed in as axioms in the Event-B models that contribute to the INSPEX data acquisition pathway verification.

Within this context, the rather delicate modelling of timing issues indicated above can be based on a sensible foundation. The complexities of the behaviour of the INSPEX data acquisition pathway imply that relatively straightforward models of time, as typically included in timed tools, are not sufficiently incisive to capture the potentially problematic aspects of the system, so a bespoke approach to the modelling of time within Event-B is needed, and this consequently drives the structure of the verification activity.

5 Discussion and Conclusions

In the preceding sections, we introduced the INSPEX Project and its intended use cases, before homing in on the VIB white cane add-on use case which forms the focus of the project itself. The main objective of this paper was to describe the use of formal techniques within INSPEX, to which we addressed ourselves in Section 4. This contained a summary of the deployment of formal techniques in the data acquisition pathway and the power management design.

Given the practical constraints of the project, it was impossible to follow a pristine top-down formal development approach in combining formal and more mainstream techniques. Given that the two approaches were being pursued concurrently, one of the greatest challenges that arises is to keep both activities in step. Little purpose is served by verifying the correctness of a design that has been superseded and contradicted in some significant aspect. The formal activity therefore paid significant attention to checking whether the growing implementation continued to remain in line with what had previously been formally modelled and verified. This way of working contributed the greatest element of novelty to the combined use of formal and conventional techniques in the project, and constitutes a stimulus for finding further novel way of reaping the benefits of fusing the two approaches.

Acknowledgement: This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 730953. The work was also supported in part by the Swiss Secretariat for Education, Research and Innovation (SERI) under Grant 16.0136 730953. We thank them for their support.

References

1. Alloy Homepage. http://alloy.mit.edu/
2. Abrial, J.R.: The B-Book: Assigning Programs to Meanings. Cambridge University Press (1996)
3. Abrial, J.R.: Formal Methods in Industry: Achievements, Problems Future. In: Proc. ACM/IEEE ICSE 2006, pp. 761–768 (2006)
4. Abrial, J.R.: Modeling in Event-B: System and Software Engineering. CUP (2010)
5. Andronick, J., Jeffery, R., Klein, G., Kolanski, R., Staples, M., Zhang, H., Zhu, L.: Large-Scale Formal Verification in Practice: A Process Perspective. In: Proc. ACM/IEEE ICSE 2012, pp. 374–393 (2012)
6. Astrée Tool: http://www.astree.ens.fr/
7. Baeten, J.: Process Algebra. Cambridge Tracts in Theoretical Computer Science. Cambridge University Press (1990)
8. Banach, R. (ed.): Special Issue on the State of the Art in Formal Methods, *Journal of Universal Computer Science*, vol. 13, (5) (2007)
9. BLAST Tool: https://forge.ispras.ru/projects/blast/
10. Bluetooth Guide: http://ww1.microchip.com/downloads/en/DeviceDoc/50002466B.pdf
11. Bowen, J., Hinchey, M.: Seven More Myths of Formal Methods. IEEE Software **12**, 34–41 (1995)

12. Clarke, E., Wing, J.: Formal Methods: State of the Art and Future Directions. ACM Comput. Surv. **28**, 626–643 (1996)
13. Dia, R., Mottin, J., Rakotavao, T., Puschini, D., Lesecq, S.: Evaluation of Occupancy Grid Resolution through a Novel Approach for Inverse Sensor Modeling. In: Proc. IFAC World Congress, FAC-PapersOnLine, vol. 50, pp. 13,841–13,847 (2017)
14. Divakaran, S., D'Souza, D., Kushwah, A., Sampath, P., Sridhar, N., Woodcock, J.: Refinement-Based Verification of the FreeRTOS Scheduler in VCC. In: Butler, Conchon, Zaidi (eds.) Proc. ICFEM-15, vol. 9407, pp. 170–186. Springer LNCS (2015)
15. Fausten, M.: Evolution or Revolution: Architecture of AD Cars. In: Proc. IEEE ESWEEK (2015)
16. FDR Tool: https://www.cs.ox.ac.uk/projects/fdr/
17. Fitzgerald, J., Gorm Larsen, P.: Modelling Systems: Practical Tools and Techniques for Software Development. Cambridge University Press (1998)
18. FreeRTOS: https://www.freertos.org/
19. Hall, A.: Seven Myths of Formal Methods. IEEE Software **7**, 11–19 (1990)
20. Hoare, C.: Communicating Sequential Processes. Prentice-Hall (1985)
21. ISO/IEC 13568: Information Technology – Z Formal Specification Notation – Syntax, Type System and Semantics: International Standard (2002). http://www.iso.org/iso/en/ittf/PubliclyAvailableStandards/c021573_ISO_IEC_13568_2002(E).zip
22. Jones, C.: Systematic Software Development Using VDM. Prentice-Hall (1990). Second edition
23. Jones, C., O'Hearne, P., Woodcock, J.: Verified Software: A Grand Challenge. IEEE Computer **39**(4), 93–95 (2006)
24. Lamport, L.: Specifying Systems, The TLA+ Language and Tools for Hardware and Software Engineers. Addison-Wesley (2002)
25. Mandruchi, R., Kurniavan, S.: Mobility-Related Accidents Experienced by People with Visual Impairment. Insight: Research and Practice in Visual Impairment and Blindness (2011)
26. Milner, R.: Communication and Concurrency. Prentice-Hall (1989)
27. NuSMV Tool: nusmv.fbk.eu/
28. Qu, Z.: Cooperative Control of Dynamical Systems: Applications to Autonomous Vehicles. Springer (2009)
29. RODIN Tool: http://www.event-b.org/ http://sourceforge.net/projects/rodin-b-sharp/
30. de Roever, W.P., Engelhardt, K.: Data Refinement: Model-Oriented Proof Methods and their Comparison. Cambridge University Press (1998)
31. Rosburg, T.: Tactile Ground Surface Indicators in Public Places. In: Grunwald (ed.) Human Haptic Perception: Basics and Applications. Springer, Birkhauser (2008)
32. Spivey, J.: The Z Notation: A Reference Manual, second edn. Prentice-Hall International (1992)
33. Stepney, S.: New Horizons in Formal Methods. The Computer Bulletin pp. 24–26 (2001)
34. Stepney, S., Cooper, D.: Formal Methods for Industrial Products. In: Proc. 1st. Conf. of B and Z Users, vol. 1878, pp. 374–393. Springer, LNCS (2000)
35. Thrun, S., Burgard, W., Fox, D.: Probabilistic Robotics. MIT Press (2005)
36. UPPAAL Tool: http://www.uppaal.org/
37. Wikipedia: List of tools for static code analysis: https://en.wikipedia.org/wiki/List_of_tools_for_static_code_analysis
38. Woodcock, J.: First Steps in the The Verified Software Grand Challenge. IEEE Computer **39**(10), 57–64 (2006)
39. Woodcock, J., Banach, R.: The Verification Grand Challenge. JUCS **13**, 661–668 (2007)