Formal Modelling and Verification as Rigorous Review Technology: An Inspiration from INSPEX

Richard Banach[1], Joseph Razavi[1], Olivier Debicki[2], Suzanne Lesecq[2]

[1]School of Computer Science, University of Manchester,
Oxford Road, Manchester, M13 9PL, U.K.
{richard.banach,joseph.razavi}@manchester.ac.uk
[2]Commissariat à l'Énergie Atomique et aux Énergies Alternatives,
17 Rue des Martyrs, F-38054 Grenoble Cedex, France
{olivier.debicki,suzanne.lesecq}@cea.fr

Abstract. Reviews of various kinds are an established part of system development, but rely on the vigilance and thoroughness of the human participants for their quality. The use of formal methods as part of the toolkit deployed during review can increase those elements of dependability that formal methods do best to support. A methodology that proposes that formal techniques are used alongside conventional system construction practices during review is introduced. These can reduce the human burden of ensuring review quality, even if the coupling between the formal and conventional strands is not itself formally enforced.

The approach advocated was inspired by experience of the use of formal methods in the INSPEX Project.[1] This project targets the creation of a minaturised smart obstacle detection system, to be clipped onto a visually impaired or blind (VIB) person's white cane, that would give aural feedback to the user about obstacles in front of them. The increasing complexity of such systems itself invites the use of formal techniques during development, but the hardware challenges preclude the application of textbook top-down formal methods. The use of formal methods in INSPEX is *ad hoc*, and the methodology proposed is an abstraction from the practical experience.

1 Introduction

Reviews of various kinds have been part and parcel of system development methodologies since systems of more than a trivial size began to be conceived. To the extent that reviews and inspections engage with the details of system

[1] The INSPEX 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.

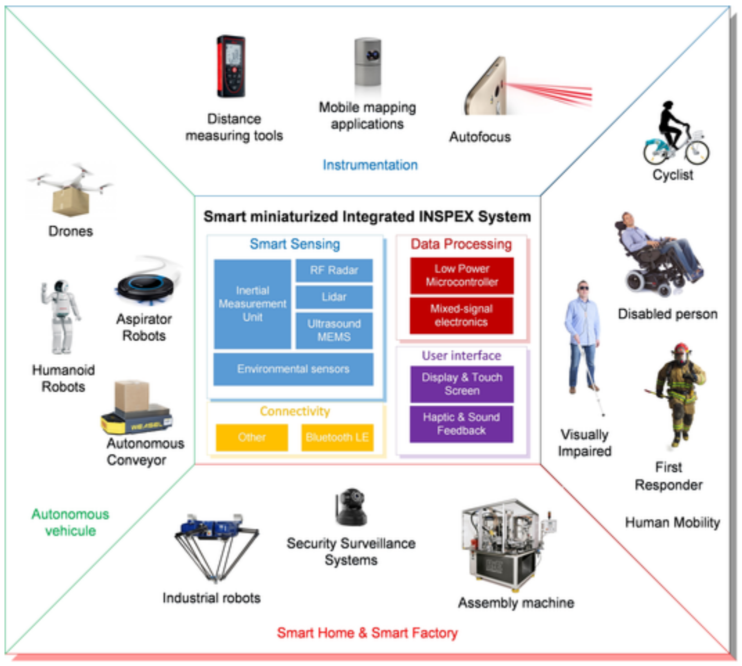

Fig. 1. A selection of potential INSPEX applications.

functionality, they rely on the vigilance and thoroughness of the human participants to ensure the quality of the review results in terms of delivering a reliable outcome. But since human vigilance is a fallible attribute, while a good review outcome encourages belief in the system's dependability, it does not guarantee it. The use of formal methods as part of the toolkit deployed during reviews can increase those elements of dependability that formal methods do best to support.

In this paper, a methodology that proposes that formal modelling and verification activities should take place alongside conventional system construction practices is introduced. When this is done, the insights from the formal strand of the work must be reconciled with the insights from the conventional strand. But since formal techniques can enforce such aspects as consistency and completeness very effectively, when this approach is pursued, the human burden of ensuring these via reviews and inspections is reduced, even if the coupling between the formal and conventional strands is not itself formally enforced.

The approach advocated was inspired by experience of the use of formal methods in the INSPEX Project. This project targets the creation of a minaturised smart obstacle detection system, in turn, inspired by the sensor constellations and their supporting software that underpin the intelligence of contemporary automated vehicle driving systems. While such a system can have many potential applications —see Fig. 1, which illustrates a selection of potential use cases for an INSPEX-like system— the specific goal of INSPEX is the creation of a

TRL4 prototype device to clip on to a visually impaired or blind (VIB) person's white cane. This would give aural feedback to the user about obstacles in front of them, in order that the risks of collisions, and of accidents, can be diminished. The increasing complexity of such multi-sensor systems creates challenges for ensuring their correct operation, inviting the introduction of formal techniques to help maximise system dependability. Still, the preponderant challenge to building such systems resides at the hardware end of the development, and this impedes the routine application of top-down formal methods, resulting in in an *ad hoc* approach to the use of formal techniques in INSPEX. The methodology proposed in this paper is an abstraction from this practical experience.

The rest of this paper is as follows. The next two sections focus on INSPEX, illuminating the background to our proposal. Thus, Section 2 overviews the INSPEX VIB system, and Section 3 discusses the INSPEX design approach, and the areas in which formal techniques were deployed during the development. We concentrate on the issue that provided the inspiration for our proposed methodology, and on how formal modelling and verification evolved *de facto* into a rigorous code inspection technique. The abstract methodological framework we propose is described in Section 4, which may be read without reference to the earlier sections if desired. Section 5 concludes.

2 The INSPEX VIB System

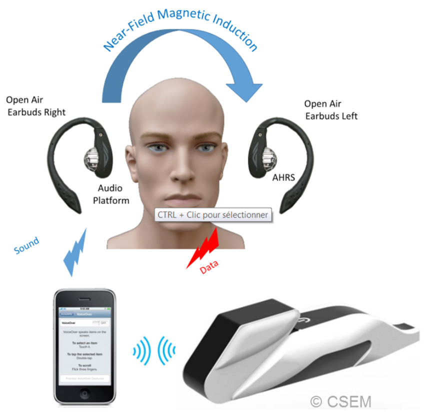

Fig. 2. The complete INSPEX system for the VIB use case.

Given the trend in sensors towards smaller, lighter and more power efficient devices, the INSPEX concept envisages a plethora of possible applications for a smart device that is truly capable of fine grained 3D spatial awareness. A range of these is indicated in Fig. 1.

As stated earlier, the INSPEX Project itself focuses on the VIB use case. Fig. 2 shows the complete INSPEX system for this use case from the user's perspective. The complete system consists of three modules. There is the mobile detection device which contains the main sensors (of which more shortly). There is a pair of open air earbuds which transmit a binaural audio representation of the environment sensed by the detection unit to the user. And there is a smartphone. This receives the information from the detection unit, information about the user's head orientation from the earbuds, and other helpful signals from any relevant smart beacons that there might be in the surroundings, and computes an audio signal depicting this information to be presented to the user, which is then sent to the earbuds via Bluetooth.

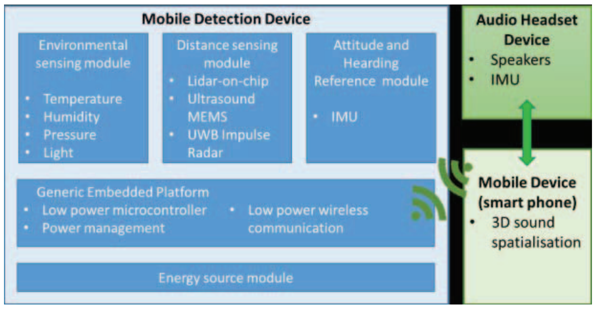

Fig. 3. The architecture of the INSPEX VIB system.

The detection unit is the module that is fixed to a VIB person's white cane. Its architecture is shown in the left part of Fig. 3. It contains the sensors that generate the data needed for the rest of the system. 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, an inertial measurement unit (IMU) and a generic embedded platform (GEP). The latter gathers all the data generated and performs all the computations needed to support all the other devices mentioned.

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.

The smartphone that performs the processing needed to convert the geometrical information provided by the detection unit into an aural signal, needs to take into account the movement of the user's head, this being independent from the movement of the white cane. So the binaural earbuds contain an IMU sensor to detect movement, and this information is transmitted to the smartphone. The earbud system is designed by French SME GoSense. Similar remarks regarding movement apply to the main detection unit which also contains an IMU sensor. The smartphone takes all of this into account in computing an aural image which is stationary in 3D space, thus enabling meaningful perception of obstacle location by the user. Fig. 3 shows all the contributing elements and the Bluetooth connections between modules.

Of course, the idea for the INSPEX VIB system did not come out of thin air. A number of white cane add-ons are already available on the market, for example [23, 26, 20]. The INSPEX system is more complex though, and utilises more sensors, in order to give users more complete and more precise information, and this is the source of the added complexity of INSPEX, compared with these earlier systems.

3 The INSPEX Design Approach and the Role of Formal Methods in INSPEX

INSPEX is, first and foremost, a hardware systems integration project. Without working hardware, the project achieves nothing. So the overwhelming emphasis in the project is on overcoming the physical challenges in bringing the equipment to life — everything from the detailed properties of the sensors and their physical signals, to the minaturisation of the constituent devices to the extent possible, to the movement of these signals and their mutual isolation, to the significant software challenges of a design of such complexity, to the properties of the main INSPEX device container, with its need for robustness and durability under a variety of weather conditions while at the same time permitting each sensor to transmit its signal and receive the corresponding reflection.

If formal approaches are to be used to help control the complexity of such an undertaking (a decision taken early in INSPEX), some methodological novelty is going to be unavoidable compared with the familiar way that such techniques are applied in practice [13, 7, 1, 4]. One consequence of this was that it was not clear at the outset what the best strategies for applying formal techniques in INSPEX would be. In the end, the most useful approach turned out to be to use Event-B with its Rodin toolkit [2, 22]. This conveniently supported formal modelling and verification. Some use was also made of Blast [6] and of PRISM [18]. Below, we outline the areas of the project upon which it was decided to focus the use of formal techniques, and we elaborate one area, the sensor reading pathway, which spurred the conception of a methodologically distinct use of formal technologies as a formalised contributor to development review.

3.1 Modelling INSPEX Power Management

Given that the main INSPEX device is intended for the maximum possible minaturisation and portability, striving for the minimum possible expenditure of power is a clear necessity. Accordingly, one strand of the INSPEX Project entailed the development of a power management strategy, in order to eke out the capacity of the power supply system to the greatest degree possible.

If we are candid, the sheer challenge of bringing the various not-off-the-shelf hardware components that form the core of the INSPEX prototype to an adequate level of development, is sufficiently taxing that switching everything on and having it working demonstrably is itself considered a resounding success. Nevertheless, as the system progresses towards a commercial product, sophisticated tuning of power use will become important, so design of a power management strategy was included as a strand of the INSPEX development.

In a complex system such as INSPEX, each sensor and subsystem has its own power consumption characteristics. But an exclusive focus on the individual subsystems risks paying too little attention to issues of global coordination. For this reason, a higher level perspective towards power management was adopted, which made it an ideal candidate for predicting power consumption behaviour using modelling via formal techniques.

The approach used for the modelling work centred on Event-B and its Rodin toolkit [2, 22], the latter an being outcome of the RODIN [21], DEPLOY [9] and ADVANCE [3] projects. PRISM [18] was used to handle the quantitative aspects. Being targetted at future needs, the power management work was much closer to a textbook formal development, and thus of less interest for this paper.

3.2 Modelling and Verification of the Sensor Readings Pathway

In contrast to the power management work, the handling of the data from the sensors is not only mission critical (without it the hardware does not work at all), but complex (because of the optimisations used), and it also relies on preexisting code (because the relevant software is an outgrowth of code used in earlier projects, as well as containing portions for future deployment). The complexity of this subsystem warranted the employment of formal techniques to improve assurance. However, a pure top-down approach to formal modelling and verification was not practical.

The information from the sensors is gathered by the acquisition software. This accepts interrupts from the short and long range LiDARs, the RADAR, the MEMS (ultrasound) and the IMU (inertial measurement unit). These need to be timestamped so that the freshness of the data can later be taken into account. At regular intervals, the available fresh data is packaged and transmitted to the fusion software. The fusion software then uses an approach to fusion based on Bayesian estimation [15] to compute an *occupation grid*, which is an estimate of which sections of the 3D space in front of the user are occupied by obstacles. While conventional techniques for data fusion [14] are computationally too intensive for a minaturised application like INSPEX, in [10] there is a much more lightweight approach to the occupation grid problem that makes it suitable for adoption in INSPEX. The granularity of the estimate that is obtained is constrained by the quality of the information received, by the bandwidth of the Bluetooth connection to the smartphone, by many pragmatic hardware and architectural considerations, and by the computational power available. Moreover, the detailed operation of the data fusion algorithm is a tightly protected commercial secret of CEA,[2] so only the most basic information about its input interface is available.

Thus the sensor readings acquisition software is complex, it is concurrent (because all the sensors involved act concurrently and with varying levels of reliability), and it must cope with a wide range of timescales (the LiDARs and RADAR act almost instantaneously, while by comparison, the ultrasound takes orders of magnitude longer to respond). On the one hand, this makes it a prime candidate for scrutiny via formal techniques with the aim of reinforcing its dependability. On the other, it raises interesting methodological questions regarding quite how one might go about doing that.

[2] It remains a secret, not withstanding the IP protection provisions of the INSPEX Project's Grant Agreement, Consortium Agreement, and even bilateral software access and nondisclosure agreements between consortium partners.

The way this challenge was approached involved an eclectic mix of top-down and bottom-up elements. When the task was started, there was already in existence a large body of code, developed conventionally. It was clear that an understanding of this had to be gained before much progress could be made. So at the outset, there was a lot of informal discussion regarding the purpose of the code and its main constituents. This yielded enough information to enable some initial formal modelling to take place, but was not yet precise enough that detailed refinement would be productive in creating models that accurately reflected the reality of the code (as opposed to merely embodying the imagination of the model's creators regarding desirable lower level properties of the code).

To go further, the code itself had to be examined in detail. Facing up to a large body of C code that embodies the elements mentioned above is a considerable challenge. The implementation level code is replete with a large amount of low level detail that is not directly pertinent to the concerns of high level correctness. And quite apart from that, the scenario being described begs the question: *what exactly does high level correctness amount to*?

In a textbook style formal methods aware development milieu, there is some process that captures the requirements, sharpens the focus to a specification, after which a formal development proceeds in stages, eventually resulting in implementation code. Some of the earier phases of this are captured in documentation of one kind or another.

In a hardware centred project such as INSPEX, the focus being so much on the hardware has the following consequence. The hardware constrains (at a very low level of abstraction) what the software can do to such an extent that little documentation is produced. In essence, the low level code defines itself. And so, from a higher level viewpoint, correctness criteria have to be elicited from the code via a combination of: understanding the code, abstracting from relevant parts of the code, reflection about the code, and reconciling the conclusions evinced from this process with what 'makes sense' in the context of what is known about the application and its context. Not unnaturally, all of this takes quite some time.

The fruits of this activity can be summarised as follows. There are some relatively self-evident high level properties, stating for example that sensors have to be OFF before they can be switched ON, and *vice versa*. These are easy to model and verify without much deep investigation. Beyond that, comes the recognition that the software maintains a buffer in which sensor data is recorded. Saying that there is a buffer (a familiar notion) does not elaborate exactly how that buffer is used. In the INSPEX context, the following facts pertinent to the buffer have to be taken into account.

Messages from the various sensors (LiDARs, RADAR, MEMS US, IMU, etc.) arrive at the Generic Embedded Processor (GEP) and are inserted into the buffer. To be useful in terms of the 3D objective of the application, they have to be associated with a time and orientation. Timing can be handled by the real time clock of the GEP, while orientation is inferred from the IMU. But the IMU is a separate sensor, sending its data at time points different from those at which the other sensors send. To mitigate this, a consecutive pair of IMU data messages are used as brackets, within which the data messages which arrive from other sensors and which are timestamped with values between the the timestamps of the two IMU brackets are aggregated into a ‘frame’, and this frame is the basic unit which is passed to the fusion software. Obviously, what has just been described goes well beyond what is one’s first thought on hearing the word ‘buffer’, and presents a non-trivial undertaking for the formal modelling and verification activity, especially considering that the devices involved are subject to failures of varous kinds and the system has to be robust against these.

It should be clear already that what was involved in adding the assurance obtainable via verification to the sensor readings acquisition software entailed a deep analysis of the existing code, and contemplation of how its high level purpose could be formalised. Thus, the formal modelling and verification amounted *de facto* to a sophisticated kind of code inspection, and this is what has inspired us to make the more abstract methological proposal in the remainder of this paper.

4 Formal Modelling and Verification as Rigorous Review Technology

In this section we abstract from, and extrapolate, our experience with INSPEX described in the preceding sections, and we present an approach to the use of formal modelling and verification as an adjunct to conventional approaches to system development with an emphasis on the use of formal techniques as a review technology. The latter aspect intersects with existing ideas on reviews, both formal and less formal, that have been around for a number of decades.

The first point to note is that we are aiming at development methodologies that do not wish to, or do not have the resources to, or are unwilling to, adapt their main development path to accomodate the exigencies of a fully formalised development strategy. Nevertheless they recognise that formal approaches can yield benefits in terms of improved dependability, and wish to gain what benefits they can from a cooperative relationship with formal techniques.[3] In this context, we must assume that the formal work that is to take place is supported by resources separate from those that support the existing practices.

The fact that we are concerned with development practices that are specifically *not* led by formal development considerations, implies that we cannot be too prescriptive about how the formal and conventional practices might work together. Therefore, a considerable degree of flexibility is needed in how formal techniques might fit alongside existing conventional techniques. However, we can infer the following.

[3] In the case of INSPEX, it was the lack of resources that prevented a greater integration with formal techniques. It would have necessitated a considerable investment of time and manpower, far beyond the resources of the project, to evolve the existing practices of firmware and hardware development to bring them closer to formal approaches.

Since the conventional techniques are specifically *not* formal, the connection between the formal and conventional sides must be human mediated. For this to be possible, the formal side must replicate some of what the conventional side does; otherwise, there would be no sensible point of comparison, no bridge, between the two sides.

The latter point is the focus of costs and benefits. The costs are obvious: the additional resources that must be found to support the formal work. The benefits are to be found in the independent scrutiny that a formal reappraisal of the relevant elements of the conventional work brings. Independent scrutiny of any kind is widely recognised as being beneficial, even if its benefits are often hard to quantify, and that, far beyond purely technical considerations.

The approach outlined has its strengths and weaknesses too. An obvious strength is the much greater control over consistency and completeness that a formal, mechanically checked definition of a system possesses, when compared with a purely conventionally developed counterpart. An equally obvious weakness though, lies in the fact that the formally defined model is the translation of a human interpretation of some conventionally developed counterpart, and therefore, its reliability is wholly dependent on the reliability of the human interpretation. Part and parcel of our proposal then, is that the human interpretation of the conventional system model is likely to be more reliable than human performed consistency and completeness checking of the same conventional system model, based on the presumption that consistency and completeness checking are detailed, bureaucratic activities, better done by machines, whereas reinterpretation of a system model from a different technical perspective suits the abilities of the human imagination better. This observation has the potential to turn what is a perceived weakness, partly at least, into a strength also.

4.1 Reviews

The role of formal techniques within conventional development, as suggested heretofore, is to reappraise the conventional development, casting a diverse perspective on it, with the aim of improving its overall integrity. This brings it close to the traditional role played by reviews of one kind or aonther. We turn to this issue now.

No activity of any scale can come to a successful conclusion without an appropriate degree of oversight as it proceeds. The construction of a complex artifact such as a digital system is no exception, and for the oversight to be effective, it has to engage sufficiently with the actual technical details of the project.[4] Thus reviewing of technical progress has always been around, one way or another.

[4] The hazards of not engaging with the technical details sufficiently are well illustrated in [25], which describes how the original management of the Crossrail Project in the UK failed to stay in close touch with the technical progress (and problems), resulting in sudden announcements of delays of two or more years, and budget overruns of billions of GBP.

The idea of formalising the structure of reviews, especially reviews of the code in large software projects, was pioneered by Fagan [11, 12], whose proposals aimed to maximise the effectiveness of the reviewing process by formalising the process in a way that made the best use of the review's human participants, in particular, paying attention to the limitations of human attention span, etc. Fagan's ideas gained widespread traction in the mainstream [24, 8, 17, 16], and different kinds of review were developed to suit different stages of development and different kinds of project: e.g. requirements reviews, design reviews, as well as code reviews. Different levels of rigour for the reviewing process also emerged, ranging from a structured discussion with a wide range of stakeholders in an informal walkthrough, to much more formalised processes involving a strictly defined team of participants each of which engages in a precisely defined role within the review, which itself takes place within tightly constrained time limits and procedural norms. The article [19] contains an interesting discussion about the world of reviews of different kinds.

Since the use of formal techniques as described above is intended to bring increased dependability to an otherwise conventional development process via the oversight that a formal reformulation can bring, and the use of formally structured reviews has the same aim via the oversight that the review process imposes, it is natural to try to blend the two approaches. That is the aim of the proposal of this paper.

To blend the two ideas, for the sake of definiteness, we have in mind a relatively formally structured review process, but in fact, this is not obligatory. The essence of our proposal is that formal techniques be used to reappraise the appropriateness of the conventionally developed system — reviews are a convenient means of crystalising the conclusions of such a process.[5]

A relatively formally structured review will have a number of formal roles, with at minimum the following. There is a moderator who ensures that when the review meeting is convened, it flows smoothly, and does not get stalled, or distracted by side issues. There is a recorder who focuses on ensuring that an accurate record of the review process is maintained, but who takes no active part in the proceedings. After that there are a number of technical personnel concerned with different perspectives on the system being developed, as suits the situation. In the next sections, we look at how this plays out at the requirements, design, and implementation levels. However, in keeping with allowing the approach to be adaptable to different kinds of development scenarios, we do not assume, at the lower levels, that the corresponding higher level activities have necessarily taken place.

4.2 The Requirements Level Process

At a high level requirements review, the technical personnel will range from stakeholder representatives to high level designers. The fact that a requirements

[5] In INSPEX, formal reviews of this kind were not constituted as such. Instead the conclusions of the modelling and verification work were captured in reports that were delivered to the conventional developers.

level review is conceivable, implies that a significant amount of requirements level documentation is envisaged to exist. In this context, we can propose a process incorporating formal technical elements as follows:

- As well as the stated technical personnel, there is an individual competent in formal modelling and verification technologies, referred to below as the FM-tech.
- Part way though the requirements definition process, preliminary requirements documents are released to the FM-tech, who begins building high level models. These may be built in any suitable formalism. Model based approaches are often closest to design and implementation paradigms, but if the requirements are mostly of a behavioural type, then temporal logic formalisms may be more useful.
- If inconsistencies or omissions are detected during model building by the FM-tech, these are queried and resolved as they arise.
- At the completion of the requirements definition process, the final set of requirements documents are released to the FM-tech, who completes model building and summarises findings in a report.
- The requirements review takes place in the standard manner. During this, the FM-tech reports findings resulting from the formal model building. Issues to be resolved are documented, for followup post-review. A criterion for satisfactory resolution is consistency between the final requirements documents and the formal models (insofar as the informal nature of the former permits).

Following the above process encourages achieving as much completeness at the requirements level at the earliest possible opportunity, yet without abandoning traditional requirements activities completely. Lack of precision in requirements is often bewailed in commentary on the system development activity as a major source of system defects. In [2, 1], as well as many other places (especially works discussing the deployment of the B-Method), the necessity of completely rewriting the requirements documents before any formal development can begin is ruefully repeated. But even if formal development from the requirements is not envisaged, requirements documentation that enjoys a demonstrable level of consistency and completeness will help to ensure a smooth development process.

4.3 The Design Level Process

After the requirements definition process (or even without there having been such a process, if the requirements are intuitively well enough understood by the system designers), system design can proceed with some precision. As before, the fact that a design review is envisaged at all, implies the creation of appropriate amount of design level documentation. Focusing on the review process again, the technical personnel involved will cover a range of concerns, but probably will not include stakeholders in the same way. A process incorporating formal technical elements can then be proposed as follows:

– As before, there is an FM-tech (or perhaps more than one) involved.
– Part way though the design definition process, preliminary documentation is released to the FM-tech who begins building intermediate level models. These ought to be characterised as refinements of the corresponding high level models (provided such high level models have been created earlier). However, due to possible lack of precision at a higher level, refinement might only become possible after some alteration of the high level models. In such cases reconsideration of consistency with high level requirements should take place, and the relevant issues should be documented.
– At the completion of the design process, the final set of design documents are released to FM-tech, who completes model building and refinement, and summarises findings in a report.
– The design review takes place in the standard manner. During this, the FM-tech reports findings resulting from the formal modelling and refinement. Issues to be resolved are documented, for followup post-review. A criterion for satisfactory resolution is thoroughgoing consistency, top to bottom.

For many system types, where there have been well understood precursor systems developed by the same teams, design is the most likely starting point for development. For such systems the design stage is the earliest stage at which the kind of formal scrutiny proposed in this paper becomes possible.

4.4 The Implementation Level

Regardless of whether the activities at requirements or design level have, or have not taken place, the corresponding review processes at implementation level are always possible in principle. This is because the implementation level definition of any system is always a formal one, irrespective of whether it is one that is easily amenable to formal analysis, or whether it addresses the system requirements (whether clearly articulated or intuitively understood) either appropriately or correctly. Thus, following on from design is coding and other implementation activity, and we make a proposal in sympathy with those above, for reviewing the code that results from the implementation process.

– As before, there is an FM-tech (or perhaps more than one) involved.
– Part way though coding, some relatively complete portions of the code are released to the FM-tech who begins to assess consistency with earlier models, and begins application of source code analysis tools. Issues germane to eventual consistency are documented for resolution as work progresses.
– At the completion of coding, the final code is released to the FM-tech, who completes analysis (both human level and tool based), and summarises findings in a report.
– The code review takes place. During this, as well as the usual commentary arising from human inspection of the code, the FM-tech reports findings resulting from reconciling the refined formal models with the code, and the outputs from source code analysis tools. As always, issues to be resolved are documented, for followup post-review. A criterion for satisfactory resolution is thoroughgoing consistency, top to bottom.

4.5 Development Processes and the Involvement of Formalism

The above proposals might easily be seen as an elaboration of a process that is both a traditionally based waterfall process, and one that is rather costly. We address these two points in turn.

Regarding cost, it is true that introducing formal techniques into the development process raises costs early on. However, this has to be weighed against cost savings later down the line when faults discovered in the field have to be remedied, usually at a much higher cost. It is by now relatively well known that, done in a judicious manner, formally assisted development need not cost more, overall, than traditional development, when total system lifetime costs are properly accounted.

Furthermore, in the review scheme we proposed above, we advocated the involvement of the FM-tech from a relatively early stage. Although the maximum degree of independence of the FM-tech maximises also the diversity of perspective that the FM-tech brings to the appraisal of the system, the maximum degree of ignorance about its details maximises also the time taken —and thus the cost— of achieving a comprehensive and accurate review. We have advocated a middle way: the FM-tech should have *some* familiarity —but not too much— with the subject of the development, so that a healthy (but not unhealthy) degree of skepticism can be brought to the FM-tech's involvement.

Regarding the waterfall basis of the description, its main purpose was pedagogical, in that the clean separations of the various phases of development facilitated the explanation of our proposal. We claim that our proposal can be adapted to more agile methodologies without too much modification. In such a more agile process, successive iterations, or sprints, could be embellished with a lightweight review process gleaned from the above account. Perhaps the main apparent obstacle to doing so, though, is the ill-adaptedness of formal refinement technologies in general, to modifications of a given level of abstraction after it is once completed — such modifications can seldom be expressed as refinements. One approach to this is to simply redo the formal development after each such modification — with good tool support this is unlikely to be too burdensome in developments of modest size. An alternative approach comes through enlarging the range of processes that a formal system model can undergo, to include the increments of functionality typified in successive sprints. In [5] there is a proposal for precisely such an enhancement to formal development processes, intended for iterative development, and adapted to Event-B.

5 Conclusions

This paper takes the experience of using formal methods in the context of the INSPEX Project, and abstracts that experience to propose an approach to the use of formal techniques as an adjunct to conventional development processes. In particular, is it proposed to use formal techniques as a more rigorous version of review techniques that might form an element of conventional development processes anyway.

Thus the preceding sections overviewed the inspiration for INSPEX arising from within the autonomous automotive domain, and how the potential for minaturisation and low power consumption in the sensor families used for automotive autonomous navigation opens the door for a host of novel applications. Prime among these is one that targets the desire to assist the visually impaired and blind to navigate more safely in their environment, by providing information about the whole of the 3D space in front of the user via an aural information feed that can be comprehended by the VIB user.[6]

The adoption of formal techniques by the INSPEX Project was a consequence of the recognition that increasing complexity of such integrated multi-sensor systems as forseen in INSPEX, creates an increasing risk that errors in the design and implementation may survive into production systems undetected. This spurs the adoption of more disciplined techniques for the development of such systems, and one of the most robust approaches of this type involves the introduction of formal approaches during the development process.

The fact that as a primarily hardware led project, the development route would need to be grounded in conventional hardware design techniques to yield results on time and within cost, entailed considerable creativity in aligning the usual practice in the embedded field with the usual practice in the formal domain. This enforced novelty in the application of formal technologies in INSPEX was paramount in inspiring the idea that salient aspects of this experience could be generalised to make them applicable more widely.

We thus outlined the use of formal techniques in INSPEX. On the one hand, there was a basically top-down approach for the power management strategy modelling exercise, which proceeded in a manner relatively recognisable as a top-down methodology. On the other hand there was a much more bottom-up approach for the verification of the sensor readings pathway, the discovery of the relevant correctness criteria there, and their reconciliation with the existing implementation.

The latter led to the main novel contribution of this paper, namely the proposal that formal modelling and verification can be used to form a significant addition to the power of review approaches in conventional system development. When the unavoidably unforgiving nature of formal systems is brought into the review process, primarily as a consistency and completeness enforcement tool, the successful completion of the modelling and verification task confirms that nothing essential has been left out at the given level of abstraction. This is something that is left to the vigilance of the human reviewer in the conventional review process, and is much harder to achieve there, given that it involves recognising what has been erroneously left out, as well as recognising what might be wrong with what has been put in.

[6] The ‘first responder’ use case, shown in Fig. 1, refers especially to firefighters who often have to work in smoke-filled envirnments, and thus experience issues similar to VIB persons. It thus forms a natural follow-up to the VIB use case.

References

1. Abrial, J.R.: Formal Methods in Industry: Achievements, Problems Future. In: Proc. ACM/IEEE ICSE 2006. pp. 761–768 (2006)
2. Abrial, J.R.: Modeling in Event-B: System and Software Engineering. CUP (2010)
3. ADVANCE: (2011), European Project ADVANCE. IST-287563 http://www.advance-ict.eu/
4. Banach, R. (ed.): Special Issue on the State of the Art in Formal Methods, Journal of Universal Computer Science, vol. 13, (5) (2007)
5. Banach, R.: Retrenchment for Event-B: UseCase-wise Development and Rodin Integration. Form. Asp. Comp. 23, 113–131 (2011)
6. BLAST Tool: (2011), https://forge.ispras.ru/projects/blast/
7. Bowen, J., Hinchey, M.: Seven More Myths of Formal Methods. IEEE Software 12, 34–41 (1995)
8. Braude, E., Bernstein, M.: Software Engineering: Modern Approaches. Wiley (2011)
9. DEPLOY: (2008), European Project DEPLOY. IST-2007.1.2 No. 214158 http://www.deploy-project.eu/
10. Dia, R. and Mottin, J. and Rakotavao, T. and Puschini, D. and 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. 13841–13847 (2017)
11. Fagan, M.: Design and Code Inspections to Reduce Errors in Program Development. IBM Systems Journal 15, 182–211 (1976)
12. Fagan, M.: Advances in Software Inspections. IEEE Trans. Soft. Eng. SE-12, 744–751 (1986)
13. Hall, A.: Seven Myths of Formal Methods. IEEE Software 7, 11–19 (1990)
14. Hall, D.: Mathematical Techniques in Multisensor Data Fusion. Artech House (2004)
15. Kedem, B., De Oliveira, V., Sverchkov, M.: Statistical Data Fusion. World Scientific (2017)
16. Pfleeger, S., Atlee, J.: Software Engineering. Pearson (2010)
17. Pressman, R.: Software Engineering: A Practitioner's Approach. McGraw Hill (2005)
18. PRISM Tool: https://www.prismmodelchecker.org/
19. Radice, R.: Software Inspections (2002), http://www.methodsandtools.com/archive/archive.php?id=29
20. Rango: (2018), http://www.gosense.com/rango/
21. RODIN: (2005), European Project RODIN. IST-511599 http://rodin.cs.ncl.ac.uk/
22. RODIN Tool: (2018), http://www.event-b.org/
23. Smartcane: (2017), https://www.phoenixmedicalsystems.com/assistive-technology/smartcane/
24. Sommerville, I.: Software Engineering. Pearson (2015)
25. UK Terrestrial TV Channel 5: Crossrail: Where Did It All Go Wrong? (2019), 3 August
26. Ultracane: (2017), https://www.ultracane.com/