Developing Safe and Dependable Sensornets

Iain Bate[1], Yafeng Wu[2], John A. Stankovic[2]
[1]Department of Computer Science, University of York, York, YO10 5GH, UK
[2]Department of Computer Science, University of Virginia, Charlottesville, USA
iain.bate@cs.york.ac.uk, {yw5s, stankovic}@cs.virginia.edu

Abstract

Sensornets are being widely proposed as a solution technology in a wide number of applications, e.g. health care. As part of this work some key challenges for the safety and sensornet communities are established in part by developing parts of a safety case for a fire detection system in a skyscraper. We then demonstrate how some of these issues can be resolved by modifying earlier work on Run Time Assurance of applications to satisfy some key safety and dependability requirements in the context of a sensornet used as part of a fire fighting system.

1 Introduction

A great deal of work has been performed developing sensornets for a range of applications. They are a classic example of a Cyber Physical System (CPS) where a complex computing system provides essential services to its human operators in order to make the operators more efficient and effective. The essential nature of the services extend beyond merely the functional or real-time properties but also encompass safety and dependability. Two distinct problems are emerging from the sensornet research field. Firstly for many applications of sensornets the operators could become reliant on the information provided to an extent a failure, systematic or random, could lead to a hazard and ultimately injury or loss of life. A common misconception is that a system such as a sensornet can directly cause a hazard, however in fact a processing device needs to be operating as part of a wider system specifically including physical effectors to cause harm. Even then for many applications, e.g. fire fighting, there is a highly-trained operator who makes decisions based on the information presented. Therefore the sensornet is classed as safety related as it can only contribute to the overall safety of a system, e.g. a building, and the case that it is safe or dependable must be be made as part of a larger system. Secondly there is growing evidence that the deployment of sensornets have reliability issues. The issues include systems failing to initialise properly, failures being caused by changing environments, and longer term availability where failures (e.g. battery power becoming exhausted) not being detected and addressed.

2 Proposed Approach

Figure 1 illustrates how this issue might be handled. The architecture represents both the logical and physical structure, however it is worth noting that the safety kernel and user interface could exist on any of a range of devices.

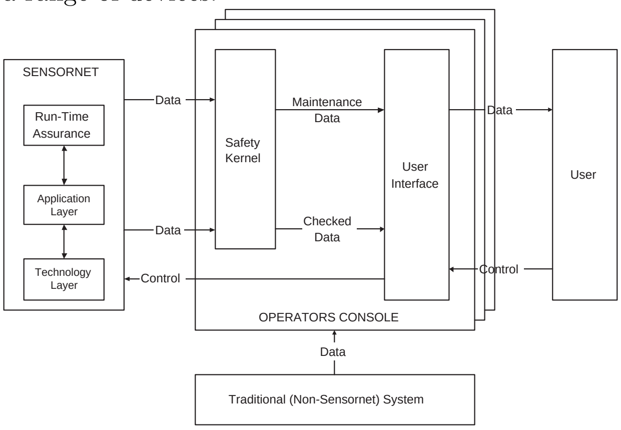


Figure 1. Proposed Architecture

The sensornet is split into an application layer, a technology layer and the Run Time Assurance (RTA) which allows us to reason about the behaviour, including that of failure, by the use of an abstraction layer. The application layer is responsible for sensor fusion in order to fuse all the data into an appropriate form for communicating to the user. In contrast the technology layer will provide the data which the fusion algorithms operate upon. Another key feature is much of the flow is unidirectional for particular reasons. For instance the safety kernel, which should be kept simple, and isolated such that it cannot be affected by the sensornet or RTA itself. If it identifies an issue it only stops the propagation of the potentially hazardous data and flags the issue to the relevant user(s) rather than instigate a full tolerance policy. This decision is taken by the user. All data flows from the sensornet via the safety kernel to the user.

The RTA is responsible for checking the capability of the sensornet to continue to provide the end-to-end application functionality needed, however its operation is then checked by the safety kernel to fulfil the requirement of "Quis custodiet ipsos custodes?" By establishing what parts of the system are healthy in an appropriate fashion, the unhealthy or failed parts can be determined. Using RTA [2] fits well with our aim to abstract our failure analysis away from specific design features and operational behaviours. To provide a fail safe operation a traditional fire detection system is included. The existence of the Traditional System as well as the sensornet raises an important issue as to why have both and what the benefits of having a sensornet might be. Having both is a usual tactic in safety-related systems at least for an interim period while the technology and experience base matures, however if a dependability case could be made without it then it could be omitted. The sensornet provides a more flexible and maintainable means of meeting the application requirements than the Traditional System. Taking the fire fighting system as an example, a traditional system can only detect a fire in the first place but does not allow the user to detect the existence and voracity of the fire after that point or provide additional functions like the calculation of egress routes unlike the sensornet. Other checks that could be performed by the safety kernel include cross referencing the results from the sensornet against the traditional fire detection system. For example if the Traditional Fire Detection system indicates a fire and the sensornet doesn't, the inconsistency could be flagged for further investigation.

3 Assuring a Firefighting System

The firefighting system [2] is intended for use on larger buildings, e.g. skyscrapers, with each room having a number of different sensors, e.g. temperature and humidity. Based on these sensors a decision is made as to whether there is a fire in the building. To date if a fire is detected then a signal is sent to a central monitoring station. The information from all the rooms is then used to initiate and control appropriate firefighting measures. As part of the firefighting system it is envisaged the firefighters themselves will receive information on which rooms have a fire and in the case of evacuation what the best route out of the building would be. Other users of the information would be the buildings supervisor who would have responsibility for ensuring an operational fire fighting system, the people who have to maintain the sensornet, and the fire chief(s) with responsibility for coordinating the firefighting.

3.1 Safety Strategy

For reasons of space a full dependability argument and safety analyses are not included. The basis for the argument that was produced is the sensornet does not introduce new hazards, or make existing hazards worse or more likely. Knowledge of hazards were derived using well-recognised hazard analysis techniques like HAZard and OPerability (HAZOP) and software variants of it such as Software Hazard Analysis and Resolution in Design (SHARD) [1]. Techniques such as HAZOP typically apply guidewords to key interfaces of systems and examine the impact. The hazard resulted in the following five DSRs.

1. *DSR1* - The fire detection system should detect a fire in a room within X minutes, where X is dependent on the building's safety case and the need to evacuate the building within a given amount of time.
2. *DSR2* - The fire detection system should be able to determine whether there is a fire based on Y sensors out of Z being operational and sensors not providing meaningful outputs (i.e. there is a significant value error) should be detected, e.g. by RTA. The value of Z would be derived based on a failure analysis, e.g. FTA, and knowledge of each sensor's failure rate along with the Mean Time To Repair (MTTR) and Mean Time Between Failures (MTBF) requirements for the application. The value of Y would again use failure analysis but this time ones based on the accuracy of data from sensors and associated algorithms.
3. *DSR3* - The communication system should be tolerant to appropriate attacks and bit error patterns.
4. *DSR4* - A failure to provide an updated egress route within V seconds is detected.
5. *DSR5* - A failure of the RTA should be detected within W seconds. This is a classic case of Quis custodiet ipsos custodes? In the case of our architecture it could be the safety kernel receives information from the RTA and hence checks its liveness by a simple heart beat mechanism.

3.2 Design of the Sensornet System

Given an appropriate set of DSRs, the next part of this work considers how to instantiate parts of the architecture in Figure 1. Here two important parts are considered. Firstly what are some of the principal trade-offs within the design of the application and how they might affect the other key properties, primarily safety, dependability and battery life. Secondly how this application might be protected by RTA and the safety kernel.

3.2.1 Application Model and Design

The starting point for the application design is the SNEDL model presented in [2]. The reason for choosing this model is it presents a baseline where the existing system was designed for the same application without the benefit of considering safety and dependability as a primary driver. There are also significant benefits of using SNEDL to represent the model as it is recognised as being Turing complete. Figure 2 shows the original model that was presented in [2].

This model was based on a more traditional sensornet design featuring only the sensornet and the operators' user interface, i.e. there was no traditional fire fighting system as a backup or safety kernel. The basic concept for this design is that each room would have a number of nodes, each node has one sensor with there being two principal types of sensor - temperature and humidity. Each node would periodically sample its sensor and compare its value against a threshold that there might be a fire. If there is a positive result, then the node with the humidity sensor would broadcast its result. Each temperature node would decide whether there is a fire or not if its own sensed value exceeded the threshold and it received a positive result from a humidity node. If the temperature node decides there is a fire it forwards it towards the operator who will then take appropriate action.

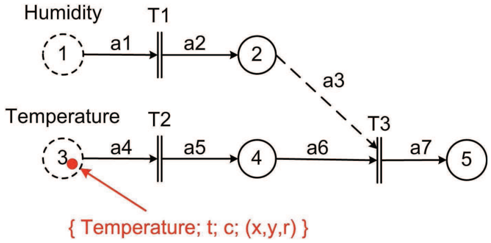


Figure 2. Original System Design

Based on the originally derived DSRs in section 3.1, the original design meets the first DSR (i.e. *DSR1* - fire is detected within X minutes) and partially the second one (i.e. *DSR2* - an inability to detect a fire is reported but no warnings are given that nodes have failed which might affect this ability). Partially meeting *DSR2* raises an important trade-off. On the one hand identifying every omission could lead to too many warnings, which undermines trust in the system and nodes, as it could just indicate a difficult communications environment. However regular failures indicate that a design change might be needed, e.g. increasing the radio signal strength used, that the node needs replacing, and most importantly when there are only two nodes left and there is then an additional failure then there is no longer an ability to detect a fire. This suggests that nodes not providing a value, or at least those that have not reported a value for some time, should have their issues logged and maintenance planned accordingly. The final three DSRs are not covered by the current RTA.

Fault Tree Analysis (FTA) was performed based on an assumption there are no systematic failures and each node has its own power source, e.g. battery. The analysis showed that a failure to detect a fire can occur for a variety of reasons including if there is not one sensor of each type remaining. This can be caused if either the RTA hasn't detected previous omissions or if the maintenance has not yet been carried out. This exposes an issue with the current design as nodes fail silently until a point is reached where there is insufficient nodes left to detect a fire, i.e. one temperature and one humidity.

3.2.2 Revised Design

For a safer more dependable system, it would be proposed that the system model from Figure 1 is used and the RTA tests improved so that as soon as any node is suspected of failure it is reported and maintenance planned and that the RTA test does not fail silent. The latter of these is achieved by the RTA reporting both positive and negative trials. To save resources the negative trials, i.e. those that do not highlight a problem, could be reported less regularly with the periodicity of these being chosen based on a *time at risk* assessment. Based on the previous assessment of the original design, Figure 3 is a proposed design for a more dependable and safer system. For this revised model the actual detection logic of a fire is unchanged, i.e. it relies on one temperature one humidity sensor having its sensed value exceed a particular threshold. There are however some key differences in that a number of extra places and tokens have been added. These are explained further in the pseudo-code in Figure 4.

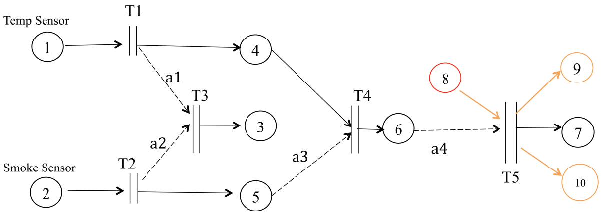
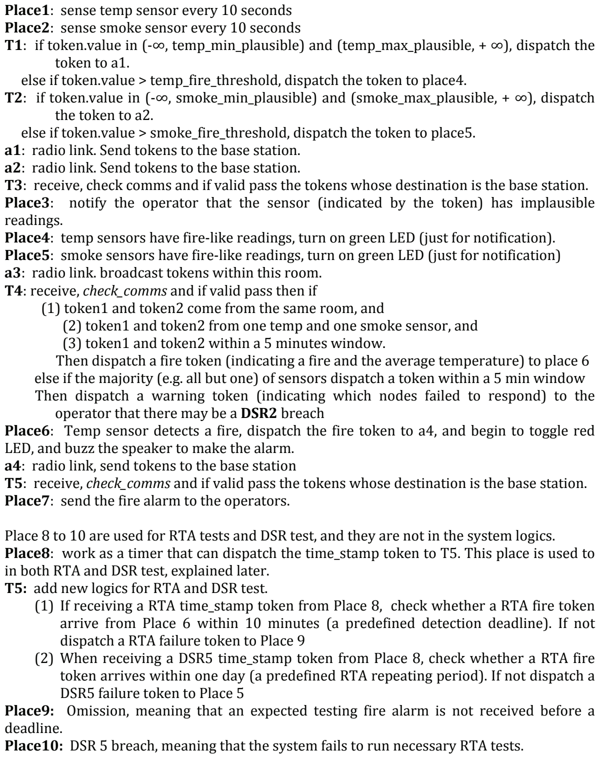


Figure 3. Revised SNEDL Design

In the pseudo code a number of extra tests are included based on the findings of the safety assessment in the previous section. For instance at token T4, a warning message is dispatched if any of the sensors do not provide an alert within 5 minutes when the majority of the other sensors have. This indicated a possible failure to meet DSR1 and hence should be at least logged and alerted to the operator if appropriate. The time out of 5 minutes is currently used an example and as the system is refined with further application knowledge may be adapted as appropriate. It is noted that every warning may not be alerted to the operator in order that the operator doesn't become overloaded with irrelevant information. For example since the communications within a sensornet is inherently unreliable, then a decision might be taken if the number of healthy sensors is greater than X then don't alert the operator until there have been three successive warnings from the same node as this would imply a more permanent problem. If there was less than or equal to X healthy sensors then an immediate alert might be raised as the sensornet would be closer to reaching a state that it couldn't comply with DSR1.

The following is an explanation of how the DSRs previously established are met.

1. *DSR1* - This is tested by RTA synthesising sensor signals that indicate there is a fire. If there are tokens going to Place 9 after the RTA test, the system has the DSR1 problem.
2. *DSR2* - This is tested by injecting implausible value in either Place 1 or 2, and check whether Place 3 receives tokens.
3. *DSR3* - During RTA tests, choose randomly from expected bit errors and attacks, and replicate its behaviour on all radio links, and then check whether we still get right results. if not then a DSR3 warning is sent to the operator. This test is performed at all communications links that have a radio link based on a random probability.
4. *DSR4* - The egress functionality is not currently represented in the SNEDL model or pseudo code for reasons of space.
5. *DSR5* - Based on a random probability, the RTA does not provide any results for an hour (i.e. no tests are performed) and this should result in a DSR5 warning token being raised at Place 5.



```
Place1:  sense temp sensor every 10 seconds
Place2:  sense smoke sensor every 10 seconds
T1:  if token.value in (-∞, temp_min_plausible) and (temp_max_plausible, + ∞), dispatch the
          token to a1.
   else if token.value > temp_fire_threshold, dispatch the token to place4.
T2:  if token.value in (-∞, smoke_min_plausible) and (smoke_max_plausible, + ∞), dispatch
          the token to a2.
   else if token.value > smoke_fire_threshold, dispatch the token to place5.
a1:  radio link. Send tokens to the base station.
a2:  radio link. Send tokens to the base station.
T3:  receive, check comms and if valid pass the tokens whose destination is the base station.
Place3:  notify the operator that the sensor (indicated by the token) has implausible
readings.
Place4:  temp sensors have fire-like readings, turn on green LED (just for notification).
Place5:  smoke sensors have fire-like readings, turn on green LED (just for notification)
a3:  radio link. broadcast tokens within this room.
T4: receive, check_comms and if valid pass then if
        (1) token1 and token2 come from the same room, and
          (2) token1 and token2 from one temp and one smoke sensor, and
          (3) token1 and token2 within a 5 minutes window.
         Then dispatch a fire token (indicating a fire and the average temperature) to place 6
     else if the majority (e.g. all but one) of sensors dispatch a token within a 5 min window
     Then dispatch a warning token (indicating which nodes failed to respond) to the
         operator that there may be a DSR2 breach
Place6:  Temp sensor detects a fire, dispatch the fire token to a4, and begin to toggle red
LED, and buzz the speaker to make the alarm.
a4:  radio link, send tokens to the base station
T5:  receive, check_comms and if valid pass the tokens whose destination is the base station.
Place7:  send the fire alarm to the operators.

Place 8 to 10 are used for RTA tests and DSR test, and they are not in the system logics.
Place8:  work as a timer that can dispatch the time_stamp token to T5. This place is used to
in both RTA and DSR test, explained later.
T5:  add new logics for RTA and DSR test.
     (1) If receiving a RTA time_stamp token from Place 8,  check whether a RTA fire token
         arrive from Place 6 within 10 minutes (a predefined detection deadline). If not
         dispatch a RTA failure token to Place 9
     (2) When receiving a DSR5 time_stamp token from Place 8, check whether a RTA fire
         token arrives within one day (a predefined RTA repeating period). If not dispatch a
         DSR5 failure token to Place 5
Place9:  Omission, meaning that an expected testing fire alarm is not received before a
deadline.
Place10:  DSR 5 breach, meaning that the system fails to run necessary RTA tests.
```

Figure 4. Pseudo-code the Revised Design

Given a model where RTA is used to help enhance safety and dependability, the next issue to be raised is what test cases are needed. An important issue raised in [2] was the issue of test case reduction since it is not feasible for resource reasons to test everything especially on-line. The tests to be performed are as follows:

1. *No failure case* - intended to check the normal behaviour of the sensornet. Three sub-cases are considered sufficient as part of reducing the number of test case based on well-established boundary testing principles. The cases are: zero or one sensors (the exact number chosen at random) indicate there is a fire in which case no fire should be reported; two sensors indicate there is a fire which is the boundary value condition; and more than two sensors (the exact number chosen at random) indicate there is a fire .
2. *Failure to send a value* - Using the same three sub-cases above between one and the maximum number of sensors will have an omission failure synthesised. In the first two case at least, and possibly the final case if there are sufficient failures, there will be a failure to detect a fire, which in the first sub-case is correct, and the omissions should be identified and reported as warnings.
3. *Communications failures* - This test is the same as *Failure to send a value* except that instead of an omission failure a randomly chosen communications failure across a radio link will be chosen. This should result in a warning and the message not being propagated. The fact the message is not propagated will lead to the same overall result as omission above.
4. *No RTA tests* - Based on a random probability, all RTA tests will be suspended for over an hour to ensure that the safety kernel identifies the failure to meet DSR5.

4 Conclusions

In summary using the case study has demonstrated how some of the significant issues facing designers of safe and dependable system featuring sensornets can be tackled. An architecture has been proposed that clearly separates ensuring the application semantics are met. The architecture features an improved version of RTA.

References

[1] J. McDermid, M. Nicholson, D. Pumfrey, and P. Fenelon. Experience with the application of HAZOP to computer-based systems. In *Proceedings of the 10th Annual Conference on Computer Assurance*, pages 37–48, 1995.

[2] Y. Wu, K. Kapitanova, J. Li, J. Stankovic, S. Son, and K. Whitehouse. Run time assurance of application-level requirements in wireless sensor networks. In *Proceedings of the ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN 2010)*, pages 197–208, 2010.