Configuration Verification

Avi Harel https://avi.har-el.com/

Ergolight consulting ergolight@gmail.com

Abstract

The article suggests a model, based on two case studies, of the generation of configuration errors. The design goal proposed is to detect and warn about such errors before operation. Configuration errors are due to situational exceptions, in which the configuration does not comply with the operational scenario. The conclusion is that the verification should be based on testing that the configuration complies with the scenario.

Objective

Typically, configuration errors are attributed to careless operation. The article presents a model of configuration errors, and a framework for eliminating these errors by integration engineering.

Situational consistency

During the operation, various states in various components need to change to fit the functional needs. A situation that complies with the functional needs may be regarded as consistent with respect to the supported function.

Operational reliability relies on situational consistency. The operation might fail when the situation is not consistent with a function.

Case studies

The model of configuration errors was obtained by analysis of partly published investigation reports of two incidents of launching missile unintentionally: Cheongung (2019) and BrahMos (2022).

The Cheongung case study

Cheongung is a medium-range, anti-aircraft guided missile, intended to strike a hostile aircraft at an altitude of around 40 kilometers. In March 2019, one of these missiles was launched unintentionally during a routine maintenance check, after the mechanics failed to replace an operational cable with a test cable. The missile was unintentionally launched when the operational cable received a mock launch signal for a testing purpose. More presented by Harel (2024, A).

This case study demonstrates the need to verify the compliance of the information flowing in the system with the operative scenario, each time the scenario changes.

The BrahMos case study

BrahMos is a supersonic land attack missile with nuclear capabilities. On 9 March 2022, during routine maintenance, a land-based version of BrahMos was fired accidentally into Pakistan. From its initial course, the object suddenly maneuvered towards Pakistani territory and violated Pakistan's air space, ultimately falling near Mian Channu. More presented by Harel (2024, B).

Anatomy of operational failure

Traditional root-cause analysis (RCA)

Traditional RCA is reactive, namely, specific to an incident. The goal of reactive failure RCA is to generate a model describing the root cause of the failure. In reactive RCA we look for a single main important cause of the specific accident, a unique source for the unfortunate event. The goal is to ensure that the failure does not repeat. The conclusions are often circumstantial, justified by quoting Murphy's Law.

The failure in the case studies was that the configuration of the missile launcher did not match the operational scenario, when changing to maintenance:

- In the Cheongung case, the incident was due to failure to maintain situation compliance following a change of the operational scenario from operational to maintenance.
- In the BrahMos case, the incident was due to failure to disconnect the connectors from the junction box.

In these case studies, the system did not test if the configuration complies with the scenario, and did not provide warnings about the failure thereof. The engineering challenge about configuration errors is to enforce the verification testing at the time of scenario change. The case studies suggest the need for scenario-based configuration testing.

In proactive failure RCA we define models based on the analysis, intended for runtime verification testing, by design. The impact of the models is demonstrated and evaluated by hypothetical application to the case studies.

Configuration exceptions

In the case studies, the failure was due to failure to comply with the scenario:

- In the Cheongung incident, the configuration included an operation cable, which did not fit the maintenance scenario
- In the BrahMos incident, the configuration included connections to the junction box, which did not fit the transition to the inspection scenario.

The problem in the case studies was that the compliance was not defined explicitly, and the verification relied on the human operators, instead on automation.

Invisible risks

When the exceptions are fuzzy, it is possible to divert to the exceptions, and such diversion is unnoticed. In most of the systems, most of the failures are unknown, because neither the designers nor the customers can notice them, and therefore they are not aware of them. Only when the costs are high do we bother to look for ways to prevent repeating the diversion. Only a small part of the diversions is noticed, namely, when their costs are noticeable.

In case of an incident, when the designers notice a failure, they often attribute it to an operator's error. They are obliged to pay attention to a failure only in case of an accident, namely, when the costs are extremely high.

We should assume that the number of risky situations is huge, because we can see only those that are costly, and because we do not bother to detect and investigate low-cost events.

Operational errors

Analysis of many accidents has shown that the term human error is just a name for operational failure that the human operator was not able to prevent (cf. Dekker, 2007). Examples:

- Cheongung, the failure to detect that the operation cable was not replace was attributed to the operation team, instead of to a design mistake
- BrahMos, the failure to disconnect the junction box was attributed to the operation team, instead of to a design mistake.

To eliminate human errors, we need to understand how the operation fails. The challenge is to get enough evidence to understand how errors are generated.

Error proofing

To be on the safe side, we should protect the system from all risky situations, because we cannot tell when one of them might be disastrous. Practically, this implies that error proofing ought to be a key topic of systems engineering.

Enforcing situational consistency

To enforce situational consistency, the system should adapt to scenario changes. We may distinguish between two types of situational variables: controllable variables, such as state variables, and uncontrollable variables, typically continuous variables, obtained by measurements.

In the Cheogung case study the problem was with the physical cable connection, which we cannot enforce by software. However, we can enforce verification, by alerting the operators to violation of the situational rules. In the BrahMos case study there was a problem with the physical connection to the junction box, but also a problem with the 'live mode' that did not comply with the 'inspection' scenario. The problem with the physical connection may be solved by verification and alerting. The mode error may be prevented by scenario-driven mode setting.

Situational rules

To enable the situation adaptation, the scenario should be defined explicitly, and implemented as a primary system variable. Otherwise, different system components might assume different scenarios, and the corresponding situations might be different. The scenarios should be defined such that a scenario change involves:

- Enforcing consistent change of the adaptable variables
- Verification of the consistency of unadaptable variables.

In the Cheogung case study, the scenario change was from functional to testing. The scenario implies several mode variables, one of them, the configured cable, is relevant to this event. The rules for scenario-mode compliance should be:

- During functional operation, the system components should be connected by an operational cable.
- During testing, the system components should be connected by a test cable.

The operation in the incident involved violation of the second rule, namely, the system was connected by an operational cable even after changing to a testing scenario. Apparently, the design of the Cheongung missile did not include such tests.

In the BrahMos case study, the enforcement is applicable to the mode change, such that in the 'inspection' scenario should enforce the 'inert state'. On the other hand, in this case, we cannot enforce physical cable connection by software, but we can enforce verification, and alerting on violation of the situational rules. The situational rules applicable to this case study are:

Scenario-driven setting rules

- The system should notify and disable launching the combat missile when in 'live state' when the scenario is 'inspection'

Configuration rules

- Whenever the combat connectors are connected to the connection box and the scenario is 'inspection', the system should notify the exception and disable the Mobile Autonomous Launcher commander from launching the combat missile.

Enforcing the situational rules

To enable the situation verification, the system requirement should specify the operational scenarios, the system design should manage the scenarios, and the software program should verify that the implementation complies with the design. The focus of situation verification is on the mode compliance with the scenario according to the rules and on disabling the operation when they do not comply with the

scenario. The design challenge is to specify the rules defining the scenario-mode compliance and the reaction to violating these rules.

Activity rules

Activity may be defined in terms of changes in the system situation. Accordingly, activity rules are about situational changes. The activity rule applicable to the Cheogung case study is:

- On change to the testing scenario, the system should prompt the operators to disconnect the operational cable and to replace it with a testing cable.

The activity rule applicable to the BrahMos case study is:

On change to the 'inspection' scenario, the system should:

- Apply the scenario-driven rule: enforce the 'inert state', and
- Prompt the operators to disconnect the combat connectors from the connection box.

Error proofing

To be on the safe side, we should protect the system from all risky situations, because we cannot tell when one of them might be disastrous. Practically, this implies that error proofing ought to be a key topic of systems engineering.

Enforcing situation awareness

To enforce the operator's awareness of the rule violation we need to apply two mechanisms:

- Situational verification: ongoing notification, as long as the configuration does not comply with the rules
- Activity verification: an alarm on changing from normal to exceptional configuration.

Conclusion

The case studies demonstrate a need for early detection of configuration errors and mode errors. The method demonstrated here is based on rules for associating the configurations and modes with the scenarios.

The vision proposed here is that system engineering standards may include a chapter on when and how to apply the method for scenario-driven operation.

References

Dekker, S 2007. Just Culture: Balancing Justice with Accountability.

Harel, A 2024. (A) Configuration verification: the Cheongung case study,
DOI: 10.13140/RG.2.2.27364.18564

Harel, A 2024. (B) Scenario-driven operation: the BrahMos case study,
DOI: 10.13140/RG.2.2.25686.46405