Measuring Dependability as a Mean Failure Cost

Ali Mili,

College of Computing Science,

New Jersey Institute of Technology,

Newark NJ 07102-1982 mili@cis.njit.edu

April 2, 2007

1 Challenging the Mean Time to Failure

The Mean Time to Failure (MTTF) is a commonly accepted measure for system reliability. Some variations of it (MTTD, MTTE) have also been adopted to measure other dependability attributes [1]. In this short note we wish to submit a tentative challenge to this measure, propose an alternative, and discuss how the alternative can be deployed.

When we say that a system S has an MTTF M, we mean that the mean time to the failure of the system with respect to some implicit specification R is M. In doing so, we are usually making two implicit assumptions:

- *Independence with respect to subspecifications*. A complex specification is typically the aggregate of many individual requirements; the stakes attached to meeting each requirement vary from one requirement to another. Yet the MTTF makes no distinction between requirement; failing any requirement counts as a failure.

- *Independence with respect to stakeholders*. Typically the operation of a system involves many stakeholders, who have different stakes in the system meeting any given requirement.

Taking these dependencies into account, we must now consider a substitute for the MTTF, which varies according to which specific requirement is violated, and according to which stakeholder we consider. We propose the concept of *Mean Failure Cost per Unit of Time* (MFC), which reflects how much a particular stakeholder stands to lose, on average, as a result of possible failure, per unit of time. Given a system whose MTTF is M, we can de-

cide whether to rely on this system or not by comparing M to the time T we expect to use the system: If the ratio $\frac{T}{M}$ is small enough, whence the likelihood of failure during the period of usage is small enough, then we can decide to use it. By contrast, given a system whose MFC is M, we can decide whether to rely on this system or not by comparing M to the mean benefit (B) that we gain from using this system per unit of time.

2 Quantification Infrastructure

To estimate the MTTF of a system, we only need to model its probability of failure with respect to its specification, R. By contrast, to estimate the MFC we need the following information:

- The probability of the system to meet various subspecifications of R.
- A matrix that shows, for each relevant stakeholder, the associated failure cost with respect to each subspecification.

What makes the estimation of MFC difficult is that the decomposition of a specification R into subspecifications is not orthogonal, in the sense that subspecifications may overlap arbitrarily and that the decomposition is not unique. Also, decompositions may vary from stakeholder to stakeholder. To fix our ideas, we consider a sample/ simplistic example of a flight control system for a passenger airplane, for which we list some sample subspecifications, some sample stakeholders, and some sample failure costs.

Sample Subspecifications:

- Ensure a smooth ride.
- Ensure adherence to flight vector within governmental guidelines.
- Ensure timeliness/ adherence to flight schedule.
- Ensure fuel efficiency.
- Ensure adherence to noise pollution standards.
- Ensure responsiveness to Auto Pilot settings.
- Ensure that the reverse thrust is never applied in mid air.
- Ensure that the landing gears are always activated prior to landing.
- Ensure that the aircraft speed does not fall below the stalling speed.
- ... .

This list is neither complete (of course), nor orthogonal (many requirements overlap a great deal). As for stakeholders, we consider:

- The airplane's pilot.
- The passengers.

- The Aviation authorities (e.g. FAA).
- The CEO of the airline.
- The CEO of the aircraft manufacturer.
- The CEO of the insurance company ensuring the airline.
- Environmental activists/ organizations.
- Residents in the neighborhood of the airports (departure, destination of the flight).
- The beneficiary of a passenger's life insurance.

Sample Failure Costs:

- CEO of the Airline, Fuel efficiency: Bottom Line.
- CEO of the Airline, Timeliness: Company's reputation for timeliness.
- CEO of the Airline, Smoothness of the flight: Loyalty of the passengers.
- CEO of the aircraft manufacturer, Fuel efficiency: corporate image, selling point.
- Environmental activists, fuel efficiency: carbon imprint.
- Reverse thrust, passenger: Life.
- Reverse thrust, insurance company: Claims.
- Reverse thrust, airline company: Reputation for safety.

For a given stakeholder, the the Mean failure Cost can be estimated by considering the probability of failure with respect to all relevant subspecifications (i.e. those subspecification whose associated failure cost is non-zero), along with the associated failure costs for the stakeholder. By extension, a party who has no stake in the operation of a system views the MFC of the system as zero (one could argue that for this party the MTTF of the system is as good as infinite).

One of the consequences of this model is that the distinction between reliability and safety is blurred: what is usually referred to as *safety* is merely reliability with respect to a subspecification whose associated failure cost (with respect to some implicit stakeholder) is very high. The proposed model makes no distinction between low failure cost and high failure cost, and integrates a continuum of subspecification with varying failure costs. Hence MFC encompasses not only reliability, but also safety.

What we must consider now is how to estimate the mean failure cost, indeed what is its formula. To begin to understand this question, we consider the refinement structure of specifications, because it is at the core of how specifications are structured as aggregates of subspecifications.

3 Refinement Structure

We approach this problem from the standpoint of relational specifications. The results we present here have been explored in the context of relational specifications, though we suspect that they hold in other refinement calculi as well. Requirements specifications are represented by relations, which map system inputs to correct systems outputs. We define a partial ordering on relational specifications, under the name of *refinement ordering*. This ordering, which we represent by the symbol $\sqsupseteq$, is reflexive, transitive and antisymmetric. It is relevant to us because a system S is correct with respect to a (relational) specification R if and only if S refines R. Because the refinement ordering is transitive, we infer that if R refines R', then any system that is correct with R is correct with R'.

If we turn to lattice properties of the refinement ordering, we find that it has lattice-like properties, in the following sense. Any two specifications R and R' which admit a common upper bound have a least upper bound, which we denote by $R \cup R'$. We refer to the least upper bound of R and R' as the *join* of R and R', and we interpret it as the specification that captures all the requirements of R and all the requirements of R'. While the least upper bound exists conditionally, the greatest lower bound of two specifications is defined for any two specifications. We represent it by $R \sqcap R'$, and we find that it reflects requirements information that is common between R and R'.

Perhaps as a consequence of the conditional nature of the join and the unconditional nature of the meet, the lattice of refinement has a universal lower bound, which is represented by the empty relation, but it has no universal upper bound. Instead, maximal elements of the refinement lattice are total deterministic relations. Figure 1 represents the outline of this lattice-like structure. This refinement structure allows to reinterpret some concepts we have been discussing informally, such as:

- That R_i is a subspecification of R: we can write this as:

$$R \sqsupseteq R_i.$$

- That R is the aggregate of subspecifications R_1, R_2, R_3, ... R_n:

$$R = R_1 \sqcup R_2 \sqcup R_3 \sqcup .. \sqcup R_n.$$

- That R_1 and R_2 are independent requirements, in the sense that refinement one of them does not imply refining any part of the other:

$$R_1 \sqcap R_2 = \phi.$$

4 Estimating/ Approximating the Mean Failure Cost

We consider a system S and a specification R, and we are interested in estimating the mean failure cost of sys-

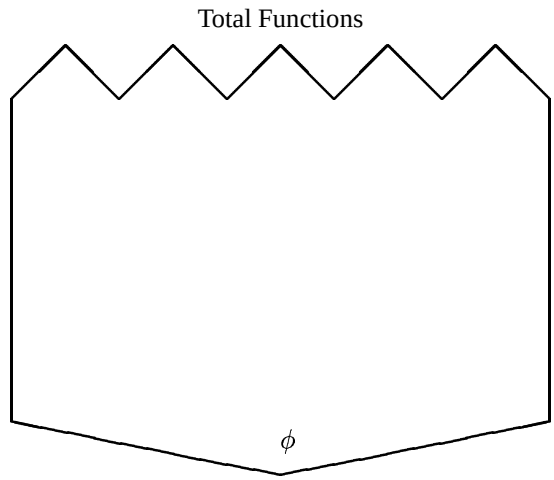

Figure 1: Refinement Lattice Structure

tem S with respect to specification R for some implicit stakeholder K, which we denote by $\Gamma(R)$. We consider a simple case, where specification R is the aggregate of subspecifications R_1, R_2, R_3, ..., R_n, that are mutually independent (i.e. refining any one of them does not imply refining any part of any other subspecification). Then, we argue that the mean failure cost of S with respect to R is:

$$\Gamma(R) = \sum_{i=1}^{n} Pr(\overline{S \sqsupseteq R_i}) \times \Gamma(R_i),$$

where Pr represents the probability of an event. In other words, the mean failure cost of the aggregate specification is the weighted sum of the failure costs of the various components of the specification, weighted by the probability of failing with respect to each component.

The very unique situation above is the only one that is simple. In general, if we apply the formula above to an arbitrary collection of subspecifications, two things happen that make the results wrong: Because subspecifications overlap, many failure costs are counted more than once; also, because subspecifications overlap, failing to meet one is not independent from failing to meet another. One way to resolve this is to redecompose the specification into independent subspecifications, but this is rather very impractical, since the failure costs that we know of pertain to overlapping subspecifications.

Another approach, that which we are pursuing, is to explore identities that involve failure costs between of subspecifications. Examples of such identities include:

- $$\Gamma(R_1 \sqcup R_2) = \Gamma(R_1) + \Gamma(R_2) - \Gamma(R_1 \sqcap R_2).$$

- $$min(\Gamma(R_1), \Gamma(R_2)) \leq \Gamma(R_1 \sqcap R_2).$$

- $$R \sqsupseteq R' \Rightarrow \Gamma(R) \geq \Gamma(R').$$

Generalizations of the formula above are under investigation.

References

[1] Fabrice Stevens, Tod Courtney, Sankalp Singh, Adnan Agbaria, John F Meyer, William H Sanders, and Partha Pal. Model based validation of an intrusion tolerant information system. In *Proceedings, SRDS*, pages 184–194, 2004.