Modeling dependability for a diverse set of stakeholders

Duy Huynh, Dept. of Computer Science, University of Maryland
Marvin V. Zelkowitz, Dept. CS, University of Maryland and Fraunhofer Center Maryland
Victor R. Basili, Dept. CS, University of Maryland and Fraunhofer Center Maryland
Ioana Rus, Fraunhofer Center Maryland
{duy,mvz,basili}@cs.umd.edu, irus@fc-md.umd.edu

1. Introduction

Users today want software that is not only reliable and efficient, but is dependable. NASA is also concerned about space missions that need to operate for several years without human intervention and funded the High Dependability Computing Program (HDCP), of which this research is part. But how does one build and evaluate such software? This paper addresses a modeling technique for computing dependability.

IFIP Working Group WG 10.4 defines dependability as "the trustworthiness of a computing system which allows reliance to be justifiably placed on the service it delivers, enables these various concerns to be subsumed within a single conceptual framework." Reliability certainly plays a major role in dependability. However, other attributes affect a user's perception of being trustworthy: performance, or time to execute a command, is a factor; ease of use or of maintaining the software also affects a user's perception of the software; security issues also are a concern.

Different stakeholders, may have a different view of dependability, even for the same system. An example of a set of stakeholders might be users, developers, legislators, and decision-makers. A user may be more concerned about usability and availability, while the developer of the software may be more interested in maintainability and performance.

Therefore, not only is dependability a multi-attribute property, but it also differs among classes of stakeholders. How can we measure this dependability and satisfy each stakeholder community? We simplify the problem to the following: Given the following assumptions:

1. Dependability is a vector of attributes.
2. There exists a set of stakeholders, each having dependability requirements (a vector with a minimal attribute value assigned to each dependability attribute).
3. Given one or more systems (i.e., potential solutions) with known dependability (i.e., a known value for each of the attributes).

Do any of these systems meet the dependability needs for all of the stakeholders?

Since our model assumes multiple attributes, we need to compare attributes that have different characteristics. We normalize all data by converting attribute values into 0..1 ranges using utility functions. A 0 utility means "no value" and a 1 utility satisfies all needs. Intermediate values provide partial satisfaction of that attribute. By converting each attribute value into its corresponding utility, we can provide uniform analysis across the entire vector space of dependability needs.

We have chosen the 2-dimensional graphical model of the radar (Kiviat) graph as our representation of dependability. Each axis in the graph represents the utility of a different dependability attribute. In Figure 1, R_1 and R_2 each represent the utility (dependability) requirements over 6 attributes for two stakeholders and D represents the utility of a system for these attributes. Does D satisfy either stakeholder?

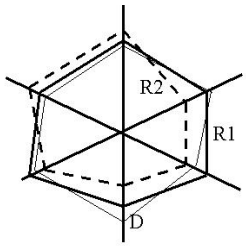

Figure 1. Dependability Representation

Let D(x) represent the value of D for attribute x and let $R_i(x)$ represent the corresponding dependability requirement for attribute x and stakeholder i (represented as utility functions). If $\forall i, \forall x, D(x) > R_i(x)$, then D satisfies all the requirements and is an appropriate solution. What if some attribute value is not sufficient; that is, $\exists x, \exists i, D(x) \leq R_i(x)$?

We need a mechanism to choose among several "good," but imperfect systems. Our solution is part algorithm to identify dependability needs and part process that identifies a negotiation strategy that each stakeholder can employ to reach a consensus on a solution.

2. Computation of dependability

We define the center of mass (COM) as the dependability of the system. As an analogy with physical properties, the COM represents the joint influence of all of the attributes. If A_i is the point (x_i, y_i) lying on axis i, the distance between A_i and the origin is the value of attribute i. The center of mass is calculated as follows:

$$X_{COM} = (x_0 + x_1 + \ldots + x_n) / n$$
$$Y_{COM} = (y_0 + y_1 + \ldots + y_n) / n$$

(Not all attributes have equal weights. A more accurate COM is given by a function g(x) over the attribute set x, such that $\Sigma\, g(x) = 1$ with COM computed as:

$$X_{COM} = (g(0)*x_0 + g(1)*x_1 + \ldots + g(n)*x_n)$$
$$Y_{COM} = (g(0)*y_0 + g(1)*y_1 + \ldots + g(n)*y_n)$$

In this note, for simplicity, we assume each attribute has equal weight.)

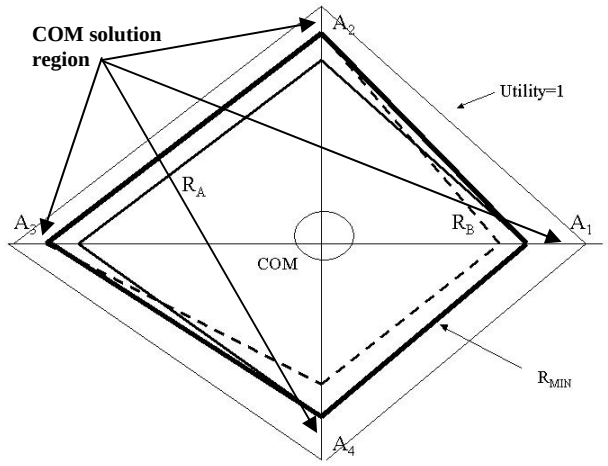

Figure 2. Computation of Center of Mass Region

Assume $R_i(x)$ is dependability requirement for attribute x on dependability vector R_i. D_i are potential solutions:

1. Compute the minimal dependability solution R_{min} as follows: $\forall i$, $R_{min}(i) = \forall j \max(R_j(i))$. That is, $R_{min}(i)$ is the largest acceptable value for each attribute.
2. If $\forall i\ D(i) \geq R_{min}(i)$ then that solution is acceptable to all stakeholders.
3. If no solution is acceptable, choose the best solution. Compute the COM of all points x such that $\forall i$, $x(i) \geq R_{min}(i)$. That is, allow x to range between $R_{min}(i)$ and a utility of 1 for all attributes i. This forms a region (COM in Figure 2).
4. Compute the average dependability for each D_i. $Avg_i = \Sigma\, D_i(j)/n$. We want this average dependability to be at least as great as the average of the required attribute values. That is, $Avg_i \geq \Sigma\, R_{min}(j)/n$.
5. For each D_i that obeys the inequality of step 4, if its center of mass is in COM, then that solution is acceptable. (It has sufficiently high average dependability and all attributes are close to $R_{min}(i)$.
6. If no solution satisfies 5, do either step 6(a) or 6(b):
   a. Choose solution i with max avg_i dependability.
   b. Choose the solution where the dependability values are more consistent with the desired solution, i.e., choose i that minimizes max $|D_i(x)-R_{min}(x)|$.

In step 6(a) you are choosing the highest average dependability. However, this permits a solution, which allows a low value for an attribute. On the other hand, 6(b) requires all attribute values to be as close as possible to the desired value. In both cases, by step 4 we ensure that the average dependability is at least as great as the minimal desired solution.

Computing dependability attribute values requires utility functions for each attribute (Figure 3). From the results of previous studies, it is feasible to elicit utility values from stakeholders as a first approximation of what is needed. Therefore, we will survey stakeholders asking questions similar to the following:

- What is the utility of a measured value of this behavioral property (e.g., for MTBF)?
  - What is the utility of a MTBF of 10 hours, 2 hours, ½ hour, …? (i.e., , How useful will this be to you?)
  - If a system exists, what is the utility of the current MTBF? (i.e., What is the utility of that current system with respect to this measure? Even if the stakeholder doesn't know the MTBF value, which is likely, this allows us to normalize the values for the other answers.)
- What are your needs with respect to this property (e.g., What do you expect the MTBF to be)?

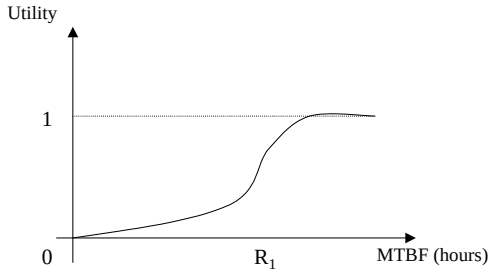

Figure 3: Utility of MTBF

After getting the stakeholder's expected value on each behavior property, we can interpolate the values to build a utility function in the [0, 1] range.

3. Negotiation

If multiple (or no) solutions remain, this process can be used within a negotiation strategy. Stakeholders can modify their required dependability needs, $R_i(x)$ or decide to use either 6(a) or 6(b) of the algorithm for choosing a solution. This then provides a basis for making such decisions.

This work is still preliminary and we are now collecting data to determine its effectiveness in measuring dependability. We are looking at multiple implementations of web software (e.g., web browsers, instant messaging systems) to evaluate this model for various stakeholders.

Acknowledgement

This work is partially supported by the High Dependability Computing Program (NASA agreement NCC-2-1298 to Carnegie Mellon University) subcontract to the University of Maryland and Fraunhofer Center.