Letter to the Editor

Dear CrossTalk Editor,

In the article "Better Communication Through Better Requirements," CrossTalk April 2004, Michael Hillelsohn states, "An effective requirement communicates clearly to all parties …" Let's examine the following assumption underlying this article and most current requirements work: One can communicate clearly with all parties (stakeholders) about system needs by relying primarily on natural language (e.g., English, French, etc.) expressions.

I believe this assumption is false.

Natural language has value in introductions and overviews, but is unsuitable for precise or complete specification – even if scrubbed. Its words (components) and sentences (structures) are inherently ambiguous, and its expressions are bulky. To deal with this problem, branches of mathematics (algebra, statistics, calculus) have been developed to express precise relationships about various facets of our world. Mathematics, however, has its own problems with readability.

This suggests that the needs of cost-effective requirements specification might be met with a compromise that blends the familiarity of (well-defined) domain terminology with the structured expressions of mathematics.

As an example, consider the need for functionality to create a valid order. To understand this requirement, you must understand the terms *order*, *valid order*, and *create a valid order*. Order is both a domain entity and a system entity. As a system entity it has attributes, value ranges, and relationships with other system entities (e.g., customers). Valid order is a [hairy] Boolean expression involving attributes and relationships of several entities. Create is an action that should be specified by defining the conditions that are TRUE after a successful create (i.e. post-conditions) that are not TRUE before.

Failure to supply these detailed definitions at requirements time, means that they will be supplied later and most likely will not be effectively validated. Therefore, the system must fail in test or production to reveal misunderstandings or omissions. Precision always happens (e.g., code), if a working system is produced. The issue is not if precision, but when it first appears, who provides it, and when it can be validated.

David Gelperin
LiveSpecs Enterprises
david@livespecs.com
Web Site: <www.livespecs.com>