ISO/IEC JTC 1/SC 22/OWGV N 0185

Proposed revision of Sub-clause 5.2

Date	16 April 2009
Contributed by	Robert Karlin
Original file name	5-2 rewrite.doc
Notes	Drafted as a proposed disposition of comment UK-37 to PDTR 24772

The goal in the creation of a programming language is to provide a tool that may be used by the writers of software to manipulate data and produce a desired result. Some programming languages are general purpose languages, while others are targeted to specific tasks or needs. Even general purpose languages may be created with a specific user in mind, for example C was created by and for engineers, while COBOL was created for business analysts and programmers.

All humans are different. Constructs that may be easily understood by mathematicians may be confusing to a business analyst. Also, similar constructs in different languages (the use of '=' in COBOL for comparison, and in C for assignment), and similar constructs within the same language (the use of parentheses for both function parameter lists, and for array subscripts) can add to this confusion. In addition, many languages provide multiple syntaxes to accomplish the same task, and coders will chose those syntaxes that make the most sense to them, again adding additional confusion in program creation and maintenance.

Humans are also fallible, and can only comprehend a limited number of interactions within a process before that process must be subdivided into smaller segments. In addition, stress, whether internal pressures and deadlines or external influences totally unrelated to the task at hand, can interfere with the process of software creation. These factors combined with language constructs that may be confusing can lead to failures due to human fallibility. These failures can include:

- Cognitive failures, which are failures of design and implementation. These may be
  - faulty reasoning and
  - incomplete solutions due to lack of time and effort in comprehending the task
- Knowledge failures, which are failures of training and environment. These can include
  - incomplete and/or incorrect knowledge of appropriate language semantics,
  - incomplete and/or incorrect knowledge of how the chosen syntax will be executed by a particular implementation, and
  - incomplete and/or incorrect knowledge of the internal and external interactions of the various software components involved
- Judgment failures, which are failures of choice and will, and may include
  - selection of simpler but vulnerable constructs in place of more robust but more time

consuming solutions and
- selection of terse less understandable constructs in place of verbose maintainable solutions

This technical report identifies issues that can increase the likelihood of errors due to cognitive limitations, and recommends ways that can be used to mitigate or eliminate these errors. Some of the mechanisms recommended in this technical report include reducing the cognitive effort necessary in reading existing source code, reducing the amount of knowledge need by readers of existing source code, and reducing the probability that incorrect developer knowledge will result in unpredictable code execution.