Proposal to SC22 Future Directions Study Group regarding Support of High Integrity Applications

Prepared by Jim Moore, 12 January 2005, James.W.Moore@ieee.org

At its 2004 plenary meeting, ISO/IEC JTC1/SC22 formed a study group, chaired by Jonathan Hodgson to consider future work for SC22. A sub-group of that study group was organized under Jim Moore to consider new work in the area of support for High Integrity Applications. The sub-study group was directed to report its results by 15 January 2005 in preparation for an anticipated March meeting of the study group.

Mr. Moore implemented the direction by forming an open-ended study group collaborating via an email distribution list manager. There are currently 32 subscribers to the list. The group collaborated for approximately four months. Based on his perception of the results, Mr. Moore drafted this report and circulated it to the distribution list for comment. Comments were received from two persons and appropriate changes were made to the draft, resulting in the current report.

Summary of Findings

The High Integrity Sub-Study Group discussed a variety of topics related to programming language support for high integrity applications. At this time, the group is not yet prepared to draft a New Work Item Proposal for consideration by SC22. However, the sub-study group recommends a narrowing and deepening of the study to deal with two specific topics:

- Usage guidelines for programming languages to increase the integrity of applications
- Metalanguage constructions (and other language-independent means) to support reasoning and analysis of the integrity of applications

Recommendation to SC22 and the SC22 Future Directions Study Group

To implement its findings, the sub-study group recommends that SC22 create a distinct study group "Programming Language Support for High Integrity Applications." The study group would be responsible for developing New Work Item Proposals for the two topics mentioned above. It would also be responsible for continued operation of the existing mailer to serve as an incubator for additional topics.

Detailed Results

The two initially selected topics are briefly described below:

TOPIC 1: Usage guidelines for programming languages to increase the integrity of applications:

Remarks on Scope:

- Any specific programming language contains constructs that are semantically vague or difficult to use. Furthermore, some languages permit a wide range of "implementation dependency" in the semantics provided by compilers and underlying operating systems. Programs containing these constructs often do something different than the coders intended.
- In addition, we see a number of "common-mode failures" that seem to emerge across a wide variety of programming languages.
- Software quality can be improved by providing guidance to programmers on how to avoid weak spots--both language-specific and language-independent ones.
- One can consider usage guidelines to serve a function similar to language subsetting but usage guidance provides a less prescriptive and more flexible approach which can serve (possibly) as a precursor to language subsetting.

- A fair number of usage guidelines already exist for specific languages. It would be a contribution, though, to consider a broad range of languages from a commensurate viewpoint. This would provide the collateral benefit of assisting users in selecting languages appropriate to their needs.
- It is possible that common-mode failures can be treated in a manner that is language-independent. For example, instead of providing exemplar coding sequences in Ada, Basic, C++, ... Z, one might use a technology similar to "design patterns" to suggest language-independent solutions.

Form of Deliverable:
What is proposed for now is continuing study leading to a New Work Item Proposal. The form of document likely to emerge from a standardization effort would be a Technical Report, possibly of Type 3 (a document inherently unsuitable to be a standard) or of Type 2 (a document potentially suitable for standardization, but on which work still continues).

In addition, a standardization effort might result in recommendations to other working groups (and other standards organizations) that language specifications should be tightened, that implementation dependencies should be narrowed, or even that language features should be redefined.

Organizational Structure:
Thinking ahead, a suitable structure for developing such a product would be a (Special) Working Group that is staffed not only by National Body representatives, but by representatives from other SC22 working groups as well as liaison representatives for language outside the purview of SC22.

Who Should be Approached:
Aside from the obvious -- SC22 National Bodies and SC22 Working Groups -- other organizations that standardize programming languages. Some possibilities include INCITS and the Sun Java Community.

TOPIC 2: Metalanguage constructions (and other language-independent means) to support reasoning and analysis of the integrity of applications

Remarks on Scope:
- Recent history has shown that static analysis and related forms of analysis can do much to improve the quality of software. Some developers routinely use tools that search for stereotypical types of errors. Ada programmers often state that once their program compiles, it usually works -- because of the extensive static analysis required of compilers by Ada's type system. The SPARK subset of Ada utilizes static analysis to produce programs to are certain to execute without exceptions and which can be proved to have various functional characteristics.
- There is some research to indicate that static techniques can be applied to other programming languages through the use of meta-language annotations which support the reasoning performed by analysis tools.
- The primary challenge in this work is probably the extent of programming languages to be included. These techniques might be easy for Ada, highly difficult for C, and of intermediate difficulty for other languages. By precluding the more difficult languages, one could probably include more powerful mechanisms. Possibly, one could imagine a hierarchy of techniques providing increasing strength but narrowing breadth of applicability.
- The work should consider the interaction of analysis techniques with other forms of verification commonly applied in software development.

Form of Deliverable:
What is proposed for now is continuing study leading to a New Work Item Proposal. The form of document likely to emerge from a standardization effort would be a Technical Report,

probably of Type 3. One could imagine Type 2 TRs or even standards, however, that standardize the syntax and semantics of selected notations.

Organizational Structure:
Thinking ahead, a suitable structure for developing such a product would be a normal Working Group that is staffed by National Body representatives. Liaison relationships with the working groups for specific programming languages should suffice for coordination.

Who Should be Approached:
This is not yet clear. It should be the subject of further study.