Beyond Mere Logic
A Vision of Modeling Languages for the 21st Century

Bran Selic
University of Toronto, Canada

Abstract: Traditional computer languages are all ultimately based on mathematical logic, which, after all, is the foundation of practically all of modern mathematics. This is a natural outcome of the initial algorithmically-oriented applications of electronic computing machines and is even revealed in how we've chosen to name these devices (i.e., "computers"). One obvious aspect of this is reflected in the fact that values in programs are typically represented by "logical" data types, such as integers, reals, or strings, which are quite intentionally shorn of any physical connotations. Consequently, in cases where such data is intended to represent relevant physical quantities, such as length or communication bandwidth, the association with the corresponding physical dimensions is typically informal, through convention. This has led to some catastrophic and expensive failures, such as the case of the unfortunate Mars Lander spacecraft, which was attributed to an undetected mismatch between metric and imperial systems measures. The informal nature of the association between values expressed in programs and their corresponding physical dimensions can also greatly complicate proper verification of such software. Whereas a great deal of effort has been expended in evolving various type theories for computer languages in order to avoid mismatches between "pure" data types, very little has been done to help us with problems with "physical" data types.

In the past, this was perceived as a concern primarily for the relatively specialized field of real-time computing. However, as more and more software involves interactions with the physical world, this deficiency is becoming more obvious, more pervasive, and more critical. Thus, with the growth of the Internet, many modern software systems are physically distributed and, consequently, highly sensitive to physical phenomena, such as communication delays, equipment failures, out-of-sequence events, and the like. In other words, more and more software is becoming "real-time". In this talk, we focus on the issues involved in the somewhat contradictory relationship between the orderly logical world of traditional software and the complex and sometimes unpredictable physical world with which it interacts. Specifically, we look at how computer languages should be constructed to deal more effectively with this complex combination.

BRIEF BIOGRAPHY

Bran Selic is currently President of Malina Software Corp., a Canadian company that provides consulting services. He is also a part-time research scientist at Simula Research Laboratory where he works on system architecture modeling for embedded, real-time systems. In 2007, Bran retired from IBM Canada, where he was an IBM Distinguished Engineer responsible for setting the strategic direction for software development tools for the technical systems space. In addition, he is an adjunct professor of computer science at the University of Toronto and at Carleton University (Ottawa, Canada), as well as a visiting lecturer and researcher at the University of Sydney (Australia) and at INSA (Lyon, France). With close to 40 years of practical experience in designing and implementing large-scale industrial software systems, Bran has pioneered the application of model-driven development methods in real-time and embedded applications. In this domain, he has led numerous research projects, authored and edited several textbooks and numerous technical papers and chaired major conferences dedicated to these problems. In addition, he was one of the principal contributors to several technical standards related to OMG’s Model-Driven Architecture initiative, including the widely adopted UML standards.