The Unbearable Lightness of Distributed Programming: A Survivor's Guide

Bran Selic

It is widely believed that, with the introduction of such straightforward extensions like remote procedure call, distributed systems programming is no different than more classical forms of programming. However, in contrast to most traditional software, the logical aspects of distributed programs are inextricably connected to the physical aspects of the environment in which they execute. The net result can be overwhelming complexity that, in essence, reflects the inevitable complexity of the physical world. Our only chance of success against such overwhelming odds is to counter-balance this inherent complexity by simplicity of system design. In this talk, we first examine the essential characteristics of distributed systems that make them so challenging and then describe certain basic architectural patterns that can help us achieve the goal of simplicity. We also show how these patterns can be captured using the industry-standard Unified Modeling Language.

Bran Selic is Principal Engineer at Rational Software in Kanata, Canada. His experience covers real-time software, fault-tolerant distributed systems, and object-oriented modeling techniques. He has written a popular textbook on object-oriented programming for real-time systems and is a member of the core team that has defined the Unified Modeling Language (UML) standard. Bran is also an adjunct professor at Carleton University, serves on the editorial board of several computer journals, and is a member of the IFAC Technical Committee on Real-Time Software Engineering.