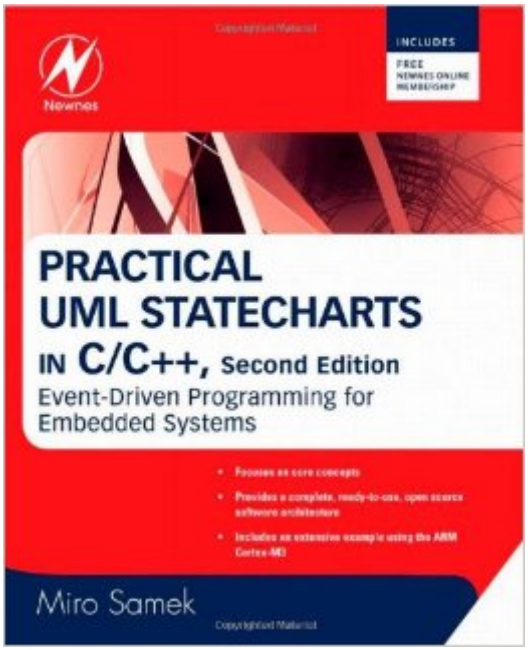
Practical UML Statecharts In C/C++: Event-Driven Programming For Embedded Systems



Synopsis

Practical UML Statecharts in C/C++ Second Edition bridges the gap between high-level abstract concepts of the Unified Modeling Language (UML) and the actual programming aspects of modern hierarchical state machines (UML statecharts). The book describes a lightweight, open source, active object (actor) framework, called QP that enables direct manual coding UML statecharts and concurrent event-driven applications in C or C++. This book is presented in two parts. In Part I, you get a practical description of the relevant state machine concepts starting from traditional finite state automata to modern UML state machines followed by state machine coding techniques and state-machine design patterns, all illustrated with executable examples. In Part II, you find a detailed design study of a generic real-time framework indispensable for combining concurrent, event-driven state machines into robust applications. Part II begins with a clear explanation of the key event-driven programming concepts such as inversion of control (”Hollywood Principle”), blocking versus non-blocking code, run-to-completion (RTC) execution semantics, the importance of event queues, dealing with time, and the role of state machines to maintain the context from one event to the next. This background is designed to help software developers in making the transition from the traditional sequential to the modern event-driven programming, which can be one of the trickiest paradigm shifts. The lightweight QP active object framework goes several steps beyond the traditional real-time operating system (RTOS). In the simplest configuration, QP runs on bare-metal microcontroller completely replacing the RTOS. QP can also work with almost any OS/RTOS to take advantage of the existing device drivers, communication stacks, and other middleware. The accompanying website to this book (state-machine.com/psicc2) contains complete open source code for QP and the free QM graphical modeling tool for QP, ports to popular processors, including ARM Cortex-M, ARM7/9, MSP430, AVR/AVR32, PIC24, RX, etc., as well as QP ports to operating systems, such as Linux, Windows, and Android.

Book Information

Paperback: 728 pages
Publisher: CRC Press; 2 edition (October 1, 2008)
Language: English
ISBN-10: 0750687061
ISBN-13: 978-0750687065
Product Dimensions: 7.5 x 1.5 x 9.2 inches
Shipping Weight: 2.8 pounds (View shipping rates and policies)

Average Customer Review: 4.2 out of 5 starsÂ Â See all reviewsÂ (26 customer reviews)
Best Sellers Rank: #165,881 in Books (See Top 100 in Books) #12 inÂ Books > Computers & Technology > Programming > Software Design, Testing & Engineering > UML #14 inÂ Books > Computers & Technology > Hardware & DIY > Microprocessors & System Design > Embedded Systems #67 inÂ Books > Engineering & Transportation > Engineering > Electrical & Electronics > Digital Design

Customer Reviews

If you are an accomplished or aspiring embedded systems programmer and resort to the "superloop" as your foremost implementation method, you absolutely MUST read this book.While not new concepts to the industry, the concepts presented in Miro's book are certainly not commonplace in many of the embedded systems code I've seen over many years in the industry.While UML makes for a clean, now defacto standard for presenting concepts, the real beauty in Miro's book is the use of a readily-available frameworks for immediately adopting the many lessons-learned and provided in the book. These complete, well written and excellently coded frameworks are available under the GPL and an alternate, low-cost license for those who wish to keep their code private.For strong C programmers, the implementation of the frameworks is a delight as one reads through the code and is treated to an incredibly simple, lightweight and extremely powerful system that will forever put an end to your future plans to "superloop" again--albeit for any system where C (or C++) is adequately supported by the hardware. For anyone admitting that "we've always done superloop and it has always worked in the past," get ready for a real-time, highly responsive system that actually IS event driven and thrives in as little as a few bytes of RAM and perhaps a K of ROM.For those who must hold on to the "superloop is king" mindset, consider the consequences of adding to your superloop an entirely new set of features and how that may affect the timing through your loop. I started with the "dining philosophers" example code discussed in the book and ported it to my own board (using a Renesas H8S-2166 microcontroller) and augmented the code to be responsive to external events (mechanical switches).

The man that wrote this book is a genius. This book explains in great detail, how firmware can created by drawing UML state charts, clicking a button to build the project and then clicking another button to upload the firmware to a target. The book is very well written. It is clear, concise and extremely informative.The book describes how to use QM, a graphical tool for designing firmware,

by creating UML state charts. It also describes how to use QP, a framework for managing concurrent, hierarchical state machines, (Active Objects) that can run under Linux, Windows and a multitude of different micro-controllers.The book explains what hierarchical state machines and Active Objects are. It describes how and why they would be used. It also describes QK, which is a preemptive real time executive that is very fast and has an extremely tiny foot print.Another topic discussed in the book is QSpy, a powerful tool for logging software trace information, used for debugging the firmware running on the target, in real time.There are detailed examples in the book that explain how to use events, through the use of a messaging system.Later in the book porting of QP or QK to other embedded platform, is explained in detail.One of the best features of QP and QM, is that they can be downloaded for free, from SourceForge.net for non-commercial purposes. Very cool!!!If it sounds like I am very impressed with the book, I am. I used the book to learn how to use QM and QP for the verification and validation testing of the firmware, of a medical device used to help people, suffering from epilepsy. The testing was necessary for FDA cerification,QM was used to design a test harness, which ran on a PC.

Programming Languages, Requirements for Programming Systems, Decision-Making Aids The Object Primer: Agile Model-Driven Development with UML 2.0 What Customers Want: Using Outcome-Driven Innovation to Create Breakthrough Products and Services: Using Outcome-Driven Innovation to Create Breakthrough Products and Services Serial Port Complete: COM Ports, USB Virtual COM Ports, and Ports for Embedded Systems (Complete Guides series) Embedded Linux: Das Praxisbuch (X.systems.press) (German Edition) JOP Reference Handbook: Building Embedded Systems with a Java Processor