

OBJEKTORIENTIERTE ENTWICKLUNG VON EMBEDDED REAL-TIME SYSTEMS mit der UNIFIED MODELING LANGUAGE (UML)

Ein dreitägiger Workshop von und mit Dr. Peter Hruschka

Eingebettete Systeme und Echtzeitsysteme stellen zusätzliche Anforderungen an die Entwickler: Wir müssen nicht nur die richtige Funktionalität liefern (in immer kürzeren Zeiträumen), sondern das auch noch unter manchmal sehr harten Zeitanforderungen, Zuverlässigkeits- und Robustheitsanforderungen und vielen anderen Randbedingungen. Die Systeme kooperieren oft mit einer Vielzahl anderer Systeme und müssen nach diesen Vorgaben von außen gestaltet werden. Verteilung auf mehrere Standorte, mehrere Prozessoren oder parallele Prozesse auf einzelnen Prozessoren erschweren zusätzlich die Entwicklung.

Seit einigen Jahren steht uns die standardisierte Notation der UML zur Verfügung, um Systeme in einer weltweit verständlichen Notation zu modellieren. Die meisten Lehrbücher und Seminare zeigen Beispiele aus dem kommerziellen Umfeld. In diesem Workshop lernen Sie, wie man die Standardnotation der UML zur Entwicklung von Embedded Realtime-Systems verwenden kann.

Die besondere Bedeutung der dynamischen Modelle (Verhaltensmodelle), der Einfluß vieler nicht-funktionaler Anforderungen und Randbedingungen (allen voran Zeitanforderungen und die Einbettung in andere Systeme bzw. Geräte) beeinflussen die Vorgehensweise bei Echtzeitsystemen. Sie lernen ein pragmatisches Vorgehensmodell kennen, das neben dem Finden von wichtigen Systemprozessen viel Wert auf die Verteilung von logischen Aktivitäten auf Standorte, Prozessoren und Tasks legt.

An allen 3 Tagen werden Sie das Gelernte durch praktische Übungen direkt umsetzen. Sie sehen die UML-Modelle wachsen, wie es auch in Ihren Projekten der Fall sein sollte, und erleben die Lernkurve am eigenen Leib.

Zielgruppe:

Dieses Seminar spricht vor allem Analytiker, Designer und Projektleiter an, die für die Abwicklung von objektorientierten Projekten im Bereich von Embedded Realtime-Systemen Verantwortung tragen. Wertvolle Anregungen bekommen auch alle Qualitätssicherer, sowie Methoden- und Standardisierungsverantwortliche.

Der Referent:

Dr. Peter Hruschka ist Prinzipal der Atlantic Systems Guild, einer weltweit führenden Gruppe von Methodenberatern, Buchautoren, Trainern und Beratern. Er ist auch Partner des System Bauhaus, einer internationalen tätigen, in Deutschland ansässigen Gruppe von OO-Beratern. Dr. Hruschka hat mehr als 20 Jahre Erfahrung in der Vermittlung moderner Software-Entwicklungsmethoden. Er berät derzeit einige sehr große Projekte sowohl in der Echtzeitindustrie wie auch im kommerziellen Umfeld. Als Mitglied des Fachbeirats der führenden deutschen OO-Zeitschrift OBJEKTspektrum schreibt er regelmäßig Kolumnen zum Thema „OO-Analyse, -Design und -Management“.

Inhaltsübersicht:

Der Entwicklungsprozeß für Embedded Real-Time Systems im Überblick

- Was ist so besonders an Real-Time Systemen?

- Die Bedeutung von funktionalen und nicht-funktionalen Anforderungen und Randbedingungen

- Die UML: eine standardisierte Notation für Systemmodelle

Die Analyse von Embedded Real-Time Systems

- Die Abgrenzung des Systems zur Umwelt

- Finden und Modellieren der wichtigsten Systemprozesse (mit Use Cases)

- Stilvorgaben für Use Case Spezifikationen

- Präzisierung des Ablaufs durch Aktivitätsdiagramme

- Ausnützung der natürlichen Parallelität

- Die wichtigsten Klassen des Systems finden

- Beispielhafte Abläufe und Testvorgaben mittels Interaktionsdiagrammen vorgeben

- Zwei wichtige Arten von Klassen: Entity-Klassen und Steuerungsklassen

- Präzise Modellierung des Verhaltens durch StateCharts

- Der effektive Einsatz von StateCharts (mit Schachtelung und Parallelität)

- Strukturvorschläge für das Requirementsdokument zu verschiedenen Zeitpunkten

Der Entwurf von Echtzeitsystemen

- Verteilungsentscheidungen treffen

- Deploymentdiagramme zur Dokumentation der Verteilung

- Parallele Prozesse entscheiden (Tasking, Synchronisation, Kommunikation)

- Komponenten bilden und als Komponentendiagramme darstellen

- Schnittstellen spezifizieren: Das Konzept der Interfaces

- Beispielarchitekturmuster für Embedded Systems

- Patterns und Collaborations zur Lösung wiederkehrender Probleme

- Neue Arten von Klassen: Der technologische Ring

- Strukturvorschläge für das Architekturdokument zu unterschiedlichen Zeitpunkten

Zusammenhang zwischen Requirements und Architektur

- Der iterative, inkrementelle Entwicklungsprozeß

- Architekturgetriebene Entwicklung

- Traceability (Nachvollziehbarkeit) von Requirements

- Abgeleitete Requirements

- Die Wartung und Weiterentwicklung von Systemen

- Umgang mit großen Systemen und Teilsystemen

- Die Verwendung von Standardkomponenten und (Teil-)Fertiglösungen