Formal Specification Techniques for Engineering Modular C Programs /

Tan, Yang Meng

Springer US,
1996

Electronic books

Monografía

Software is difficult to develop, maintain, and reuse. Two factors that contribute to this difficulty are the lack of modular design and good program documentation. The first makes software changes more difficult to implement. The second makes programs more difficult to understand and to maintain. Formal Specification Techniques for Engineering Modular C Programs describes a novel approach to promoting program modularity. The book presents a formal specification language that promotes software modularity through the use of abstract data types, even though the underlying programming language may not have such support. This language is structured to allow useful information to be extracted from a specification, which is then used to perform consistency checks between the specification and its implementation. Formal Specification Techniques for Engineering Modular C Programs also describes a specification-driven, software re-engineering process model for improving existing programs. The aim of this process is to make existing programs easier to maintain and reuse while keeping their essential functionalities unchanged. Audience: Suitable as a secondary text for graduate level courses in software engineering, and as a reference for researchers and practitioners in industry

Título: Formal Specification Techniques for Engineering Modular C Programs by Yang Meng Tan

Editorial: Boston, MA Springer US 1996

Descripción física: 1 online resource (xvi, 213 pages)

Mención de serie: The Springer International Series in Software Engineering 1384-6469 1

Contenido: 1 Introduction -- 1.1 The Problems and the Approach -- 1.2 Larch/C Interface Language -- 1.3 Related Work -- 1.4 Key Lessons Learned -- 1.5 Research Contributions -- 2 Overview of LCL -- 2.1 Larch -- 2.2 LCL Basics -- 2.3 LCL Function Specification -- 2.4 LCL Abstract Type Specification -- 2.5 Historical Note -- 3 Supporting Programming Styles -- 3.1 Specified Interfaces and Data Abstraction -- 3.2 Specified Interfaces in C -- 3.3 Abstract Types in C -- 3.4 Tool Support: LCLint -- 3.5 Summary -- 4 Specification Techniques -- 4.1 Requirements of the Program -- 4.2 The Design of the PM Program -- 4.3 Overview of Specification Techniques -- 4.4 The date Interface -- 4.5 The trans Traits -- 4.6 The trans Interface -- 4.7 The trans-set Interface and Trait -- 4.8 The position Traits -- 4.9 The position Interface -- 4.10 Summary -- 5 Redundancy in Specifications -- 5.1 Testing

Specifications -- 5.2 Semantics of Claims -- 5.3 Claims Help Test Specifications -- 5.4 Claims Help Specification Regression Testing -- 5.5 Claims Highlight Specification Properties -- 5.6 Claims Promote Module Coherence -- 5.7 Claims Support Program Reasoning -- 5.8 Claims Support Test Case Generation -- 5.9 Experiences in Checking LCL Claims -- 5.10 Claims or Axioms? -- 5.11 Summary -- 6 Reengineering Using LCL -- 6.1 Software Reengineering Process Model -- 6.2 A Reengineering Exercise -- 6.3 Effects of Reengineering -- 6.4 Specification Tool Support -- 6.5 Summary -- 7 The Semantics of LCL -- 7.1 Basic LCL Concepts -- 7.2 LCL Storage Model -- 7.3 LCL Type System -- 7.4 LCL Function Specification -- 7.5 LCL Module -- 7.6 Type Safety of Abstract Types -- 7.7 Summary -- 8 Further Work and Summary -- 8.1 Further Work -- 8.2 Summary -- A LCL Reference Grammar -- B Relating LCL Types and LSL Sorts -- B.1 Modeling LCL Exposed Types with LSL Sorts -- B.2 Assigning LSL Sorts to LCL Variables -- B.3 Assigning LSL Sorts to C Literals -- C LCL Built-In Operators -- D Specification Case Study -- D.1 The char Trait -- D.2 The cstring Trait -- D.3 The string Trait -- D.4 The mystdio Trait -- D.5 The genlib Trait -- D.6 The dateBasics Trait -- D.7 The dateFormat Trait -- D.8 The date Trait -- D.9 The security Trait -- D.10 The lot Trait -- D.11 The list Trait -- D.12 The lot list Trait -- D.13 The kind Trait -- D. 14 The transBasics Trait -- D.15 The transFormat Trait -- D.16 The transParse Trait -- D.17 The trans Trait -- D.18 The trans-set Trait -- D.19 The income Trait -- D.20 The positionBasics Trait -- D.21 The positionMatches Trait -- D.22 The positionExchange Trait -- D.23 The positionReduce Trait -- D.24 The positionSell Trait -- D.25 The positionTbill Trait -- D.26 The position Trait -- D.27 The genlib Interface -- D.28 The date Interface -- D.29 The security Interface -- D.30 The lot-list Interface -- D.31 The trans Interface -- D.32 The trans_set Interface -- D.33 The position Interface -- E Getting Larch Tools and Information -- References

Copyright/Depósito Legal: 934970392 968650831

ISBN: 9781461541257 electronic bk.) 1461541255 electronic bk.) 9781461368502 1461368502

Materia: Computer science Software engineering Computer science. Software engineering.

Enlace a formato físico adicional: Print version 9781461368502

Punto acceso adicional serie-Título: Springer International Series in Software Engineering 1
