Librería

Bonilla y Asociados

desde 1950

Título:

Autor:

Editorial:

Tema:

Sinopsis

Precio: $400.00

Año: 2002

Edición: 1ª

ISBN: 9780201733860

Software Craftsmanship is a call to arms for programmers: an impassioned manifesto that restores the developer to a central role in large-scale projects, and shows developers how to master the skills they need to succeed in that role. Software Craftsmanship transcends "software engineering," demonstrating that quality software can't simply be "manufactured": it must be built by craftspeople with pride in their work, and a personal commitment to excellence.KEY TOPICS:In Software Craftsmanship, Pete McBreen focuses on the craft of software development, explaining why current "software engineering" techniques often fail, and offering programmers a new path to excellence. Just as the modern carpenter benefits from better tools, materials, and understanding, the modern programmer can benefit from better computers, reusable components, and more robust languages -- but only if he or she is prepared to treat the software profession as a true "craft." McBreen explains what software "craftsmanship" means, how its affects users, and how it changes the developer's relationship with customers. He introduces the concepts of software apprentices and journeymen, shows what can (and can't) be learned from the software engineering movement, and presents specific steps you can take now to move towards craftsmanship in your work -- and your organization.MARKET:For all programmers, software engineers, application developers who want to gain greater personal satisfaction from their work, and for project managers who want to lead more successful projects while building great teams and retaining great developers.

Preface.

I. QUESTIONING SOFTWARE ENGINEERING.

1. Understanding Software Engineering.

The Paradox of Software Engineering.

The Modern Definition of Software Engineering.

Librería

Bonilla y Asociados

desde 1950

Is Software Engineering a Good Choice for Your Project?

2. The Problems with Software Engineering.

Can Software Development Be Made Systematic and Quantified?

The Hazards of the Good Enough Software Approach.

What Is the Alternative to Software Engineering?

3. Understanding Software Development.

Software as Capital.

Does the Division of Labor Work for Software Development?

One Size Does Not Fit All.

4. Finding a Better Metaphor Than Software Engineering.

Finding a Better Metaphor Than Software Engineering.

The Craft of Software Development.

Parallels with Traditional Craftsmanship.

The Resurgence of the Craft of Software Development.

II. SOFTWARE CRAFTSMANSHIP.

5. Putting People Back into Software Development.

Craftsmanship Is About Getting Better at Software Development.

Craftsmanship Encourages Developers to Write Great Software.

Teléfonos: 55 44 73 40 y 55 44 72 91

www.libreriabonilla.com.mx

A Call to Arms.

6. Craftsmanship Is the Opposite of Licensing.

Craftsmanship Is Personal.

Licensing Is an Illusion.

Craftsmanship Focuses on the Individual.

III. IMPLICATIONS OF SOFTWARE CRAFTSMANSHIP.

7. How Craftsmanship Affects the Users of Systems.

Software Craftsmanship Works Because Software Is Easy to Copy.

Craftsmen Have a Different Relationship with Their Users.

Great Software Deserves to Be Signed.

Craftsmen Need Demanding Users.

Software Craftsmanship Leads to Collaborative Development.

8. Customers Have a Different Relationship with Craftsmen.

Setting Realistic Delivery Dates.

Exposing the Fallacy of Good Enough Software.

Allowing Software Craftsmen to Take Credit for Their Work.

Start Exploiting the Difference in Productivity Between Developers.

But How Do We Know How Good a Developer Really Is?

Customers Make a Cost/Quality Trade-off When Choosing Craftsmen.

Customers Have Long Term Relationships with Software Craftsmen.

Customer Interests Are Aligned with the Interests of Software Craftsmen.

9. Managing Craftsmen.

Software Craftsmen Are Not Hired Hands.

Good Developers Are More Valuable Than Their Managers.

Software Craftsmen Have a Different Relationship with Their Managers,

Managing Great Developers Is a Pleasure and a Privilege.

Software Craftsmen Like Creating Applications.

Managing Software Craftsmen Is Different.

Software Craftsmen Push for What They Need.

10. Becoming a Software Craftsman.

Software Craftsmanship Is a Rejection of Narrow Specialization.

Craftsmanship Requires Dedication.

How Does a Person Become a Software Craftsman?

The Craft Tradition Has Endured for Centuries.

11. Mastering the Craft.

What Does a Master Software Craftsman Look Like?

Use Your Old-timers.

Librería

Bonilla y Asociados

desde 1950

Mastery Implies the Use of Stable Technologies.

Developing Mastery Takes Time.

Mastery Implies Taking Responsibility for Passing on the Craft.

12. Apprentice Developers.

We Must Reverse the Decline in the Quality of Developer Training.

Becoming an Apprentice Is a Significant Step.

Apprenticeship Instills Lifelong Learning.

The Role of Apprentices.

An Apprenticeship Is a Significant Investment of Time and Energy.

13. Journeymen Developers.

Where Journeymen Fit in the Craft Tradition.

Journeymen Developers.

Journeymen Are Focused on Delivering Applications.

Journeymen Play a Key Role in Software Craftsmanship.

IV. REPOSITIONING SOFTWARE ENGINEERING.

14. Software Engineering Projects.

Software Engineering Is Designed for Large Systems Projects.

Software Engineering Projects Are Diverse and Varied.

15. Hazards of the Software Engineering Metaphor.

Teléfonos: 55 44 73 40 y 55 44 72 91

www.libreriabonilla.com.mx

Librería

Bonilla y Asociados

desde 1950

You Cannot Do Software Engineering on a Low Budget.

Software Engineering Encourages Scientific Management.

Software Factories: The Production Line for Software.

Reuse over Time Is Hazardous.

The Myth of the Standardized Software Development Process.

Software Engineering Forces Us to Forget the Individual.

We Need More Variety in Our Development Processes, Not Less.

16. Learning from Software Engineering.

Size and Complexity Matter.

Applications Need to Be Well Structured.

Change Can Be Expensive Unless You Allow for It.

Communication Inside the Team and with Users Is Crucial.

Producing Accurate Estimates Is Very Expensive.

V. WHAT TO DO ON MONDAY MORNING.

17. Experience_ The Best Indicator of Project Success.

Choose Software Craftsmen Based on Their Reputations.

Evaluate Craftsmen Based on Their Reputations and Portfolio.

Auditioning a Software Craftsman.

Teléfonos: 55 44 73 40 y 55 44 72 91

www.libreriabonilla.com.mx

Let Your Software Craftsman Pick the Rest of the Development Team.

Collaborative Development.

Avoid Bleeding-Edge Technology If At All Possible.

Paying for Experience.

Be Prepared to Be Amazed.

Design for Testing and Maintenance.

Think Applications, Not Projects.

Maintenance Teams Should Refuse to Accept Bad Applications.

18. Design for Maintenance.

Software Craftsmen Prefer Nonproprietary, Open Source Tools.

Great Software Is Global.

Software Craftsmen Need to Fight Back Against Planned Obsolescence.

Great Software Needs to Be Given a Great User Interface.

Maintainable Software Is Easy to Diagnose.

The Hazards of Outsourcing.

You Can Still Use Outside Craftsmen to Create Your Application.

Maintenance Is the Most Important Part of the Life of Any Application.

Not All Software Has to Be Maintainable.

Design for Testing and Maintenance Is Not Rocket Science.

Librería

Bonilla y Asociados

desde 1950

19. Perpetual Learning.

Creating a Learning Environment.

Mastering the Craft of Software Development.

Choose Training Courses Very Carefully.

Encourage Your People to Be Visible in the Software Development Community.

Becoming a Reflective Practitioner.

Epilogue.

Acknowledgements.

Index. 0201733862T08072001

Software Engineering -- Advanced [PTG: AW PROFESSIONAL] (Computer Science)
Object-Oriented Programming (Computer Science)

Teléfonos: 55 44 73 40 y 55 44 72 91

www.libreriabonilla.com.mx