

<<激情式软件开发>>

图书基本信息

书名：<<激情式软件开发>>

13位ISBN编号：9787564042288

10位ISBN编号：7564042281

出版时间：2011-3

出版时间：北京理工大学出版社

作者：麦凯

页数：344

版权说明：本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问：<http://www.tushu007.com>

## <<激情式软件开发>>

### 内容概要

The writing of this 《激情式软件开发(英文版)》, at this time, can be attributed to Dr. Alan George who gave me an opportunity to do a reasonably large project that fits the methodology described in this book. This project in turn led to an opportunity to try to teach students and others associated with the project, the principles and concepts behind the project itself; hence the book. Alan also provided input on some of the content, such as the ideal characteristics of software.

## <<激情式软件开发>>

### 作者简介

Dr. Kenneth McKay is a Professor of Operations Management and Information Systems. Department of Management Sciences. Faculty of Engineering, University of Waterloo. Canada. He has been involved with computer systems for over four decades, and has developed dozens of software systems ranging from relational databases and interactive math software to factory scheduling tools. During this time. Dr. McKay has focused on systems that are centred on the user. and which require innovative solutions. In "Software Development on Adrenalin. " methods and concepts are shared for how to understand the user requirements, and how to undertake software development when there are few. if any, existing examples to follow.

## <<激情式软件开发>>

### 书籍目录

#### Part S/W Development: a Personal View

##### Overview

#### Chapter 1 Introduction

##### 1.1 ZenTai

##### 1.2 Mushing

#### Chapter 2 High Velocity Mushing

##### 2.1 Low Volume vs. High Volume Development

##### 2.2 From a Toothpick to a Decorated Living Christmas Tree

##### 2.3 Agile & Extreme -- an Overview

##### 2.4 Preconditions

##### 2.5 Bite Sized Pieces

#### Chapter 3 Experience and Expertise

#### Part Understanding the Problem & Thinking Through the Conceptual Solution

##### Overview

#### Chapter 4 Understanding THE Problem

##### 4.1 Understanding

##### 4.2 Good Questions

##### 4.3 Questioning and Understanding

##### 4.4 Listening Is Reading

#### Chapter 5 Modeling

##### 5.1 Abstract Modeling -- the Art of Seeing

##### 5.2 Finite State Automata

##### 5.3 Process Mapping

##### 5.4 Ishikawa's Fishbone Diagrams

#### Chapter 6 Field Analysis

##### 6.1 Ethnographic Methods

##### 6.2 Analysis Under Hostile Fire

##### 6.3 The Quick "Drive-by" Analysis

#### Chapter 7 User Engagement

##### 7.1 Stating the Obvious

##### 7.2 Styles of Engagement

##### 7.3 Other Sources of Insight -- Support and QA

##### 7.4 Stakeholder Checklist

##### 7.5 Feedback and Suggestions -- Caveat Emptor

#### Chapter 8 ZenTai -- the Value Equation

##### 8.1 面值-- Value

##### 8.2 The Value Framework

##### 8.3 Life Cycles

##### 8.4 Society or Organizational Structure

##### 8.5 Interactions

##### 8.6 Information

##### 8.7 Impact, Value

##### 8.8 Utility or Futility?

<<激情式软件开发>>

8.9 Value Analysis -- an Example  
8.10 Process Models and Value Mapping  
8.11 Exploiting Existing Technology  
8.12 Conclusion  
Chapter 9 ZenTai -- the Comfort Zone  
9.1 安心-- Comfort  
9.2 Sources of Discomfort  
9.3 Increasing the Comfort Level  
9.4 A Comfort Analysis  
Chapter 10 ZenTai -- the Experience Factor  
10.1 经验-- Experience  
10.2 Prior Experience  
10.3 Experiencing  
Chapter 11 ZenTai -- Evolution  
11.1 进化-- Evolution  
11.2 Environmental Evolution  
11.3 Functional Evolution  
Chapter 12 Pulling It All Together  
Chapter 13 Universal Requirement Factors  
13.1 The Human Element  
13.2 The Synthetic Element  
Chapter 14 ZenTai Summary  
Part Architecture & Design  
Overview  
Chapter 15 Universal Designs  
Chapter 16 The Big Picture  
16.1 What Is Meant by the Phrase: Big Picture?  
16.2 Good Architecture  
16.3 Layered Analysis  
16.4 Interface Definitions, and Protocols  
Chapter 17 Designing for Change  
17.1 Technology  
17.2 The Problem  
17.3 Users  
Chapter 18 Stability & Robustness  
18.1 Levels I through V-- Infrastructure Stu~  
18.2 Levels VI through VII--Your Stuffl  
Chapter 19 Tempus/Temporis  
Chapter 20 Task Oriented Design  
Chapter 21 Design Sufficiency  
Part Level Rapids & Mushing  
Overview  
Chapter 22 Management  
22.1 The Management Challenge  
22.2 Good Management  
22.3 Strategic, Tactical, and Operational  
22.4 Management Skill & Training

<<激情式软件开发>>

- Chapter 23 Risk Management
  - 23.1 Risk Analysis
  - 23.2 Development
  - 23.3 Operational Considerations
  - 23.4 Risk Identification
- Chapter 24 Project Management
  - 24.1 Early Phases of Project Management
  - 24.2 Detailed Functionality and Planning
  - 24.3 Budgets and Plans
  - 24.4 Degrees of Certainty
  - 24.5 Slack and Project Elasticity
  - 24.6 Critical Paths
  - 24.7 Resource Flexibility
  - 24.8 Multiple Plans
  - 24.9 Dancing with the Devil
- Chapter 25 Planning vs. the Plan
- Chapter 26 Aversion Dynamics
- Chapter 27 Reliance on Technology
- Chapter 28 User Interface Principles
- Chapter 29 The Toothpick
- Chapter 30 Factoring
- Chapter 31 Coding
- Chapter 32 Testing
- Chapter 33 Tool Smithing
  - 33.1 Passive Tools
  - 33.2 Active Tools
- Chapter 34 Documentation
- Chapter 35 Client and Developer Build Cycles
- Chapter 36 At the Helm
- Chapter 37 Operational Control & Tracking
  - 37.1 Tasks -- Who Does What
  - 37.2 Detailed Plan Contents -- Start of Week
  - 37.3 Weekly Updates and Reflection
  - 37.4 Monthly Level Details
- Chapter 38 Team Design
- Chapter 39 Mission Critical Systems
- Chapter 40 Final Thoughts
- References

## 章节摘录

版权页：插图：The Agile philosophy is not restricted to the code phase and rapid delivery of functionality. You can also do the understanding and designing bits in an agile way. The next set of chapters probe the ZenTai concepts and it is just as important to engage the user during the ZenTai design phase as it is in the code crafting phase. There should always be some form of thinking before doing, else you are really wasting people's time and money. But, many people seem to bail early on the thinking side and use the Agile concept as an excuse. At least that is my own personal observation. Thinking is hard and it hurts. It is not tim. It is a lot more fun to start wailing on some poor code and showing ill-thought through functionality under the premise Of this is how to figure out the functionality, claiming that this is how the Agile manifesto says how to do it (not). You need to think enough to get to a reasonable starting point and user engagement during the design helps you do that. There is nothing in the Agile texts and guidelines that says you have to be an idiot, that you should not think, and that you should use it as an excuse for sloppy development. When you take the time to engage the user and to understand the problem, it helps you understand the user's value chain, the most important comfort issues, areas where evolution will be important, and the potential and impact of experience. It helps you understand how the computer technology can help the user. When you are building relatively small systems for a limited audience or specialized functions, it is usually easy to identify and engage the key or representative user. For example, when I build planning and scheduling systems for a factory, there are usually several users involved and that is it. You know their names, where they sit and you can dialog with them. Management might want certain fields on reports and certain features to be included, and there might be a number of these folk putting in their two cents worth. However, when it comes down to banging on keys, there are few users to really engage. This is the same in small businesses, or focused applications such as customer database development, charity donor systems, etc.

## <<激情式软件开发>>

### 编辑推荐

《激情式软件开发》由北京理工大学出版社出版。



<<激情式软件开发>>

版权说明

本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问:<http://www.tushu007.com>