CONTENTS

CHAPTER 1 Introduction
  1.1 What Is Software Architecture? 1
     1.1.1 Software Design Levels, 4
  1.2 An Engineering Discipline for Software 5
     1.2.1 What Is Engineering?, 6
     1.2.2 The Current State of Software Technology, 10
  1.3 The Status of Software Architecture 15
  1.4 The Plan of This Book 17

CHAPTER 2 Architectural Styles
  2.1 Architectural Styles 19
  2.2 Pipes and Filters 21
  2.3 Data Abstraction and Object-Oriented Organization 22
  2.4 Event-Based, Implicit Invocation 23
  2.5 Layered Systems 25
  2.6 Repositories 26
  2.7 Interpreters 27
  2.8 Process Control 27
     2.8.1 Process-Control Paradigms, 28
     2.8.2 A Software Paradigm for Process Control, 30
  2.9 Other Familiar Architectures 31
  2.10 Heterogeneous Architectures 32
CHAPTER 3 Case Studies

3.1 Key Word in Context 33
  3.1.1 Solution 1: Main Program/Subroutine with Shared Data, 34
  3.1.2 Solution 2: Abstract Data Types, 35
  3.1.3 Solution 3: Implicit Invocation, 36
  3.1.4 Solution 4: Pipes and Filters, 37
  3.1.5 Comparisons, 38

3.2 Instrumentation Software 39
  3.2.1 An Object-Oriented Model, 39
  3.2.2 A Layered Model, 40
  3.2.3 A Pipe-and-Filter Model, 41
  3.2.4 A Modified Pipe-and-Filter Model, 41
  3.2.5 Further Specialization, 42
  3.2.6 Summary, 42

3.3 Mobile Robotics 43
  By Marco Schumacher
  3.3.1 Design Considerations, 43
  3.3.2 Solution 1: Control Loop, 44
  3.3.3 Solution 2: Layered Architecture, 45
  3.3.4 Solution 3: Implicit Invocation, 47
  3.3.5 Solution 4: Blackboard Architecture, 49
  3.3.6 Comparisons, 51

3.4 Cruise Control 51
  3.4.1 Object View of Cruise Control, 53
  3.4.2 Process-Control View of Cruise Control, 53
  3.4.3 Analysis and Discussion, 58
  3.4.4 Summary, 60

3.5 Three Vignettes in Mixed Style 60
  3.5.1 A Layered Design with Different Styles for the Layers, 60
  3.5.2 An Interpreter Using Different Idioms for the Components, 63
  3.5.3 A Blackboard Globally Recast as an Interpreter, 66

CHAPTER 4 Shared Information Systems 69

4.1 Shared Information Systems 69

4.2 Database Integration 70
  4.2.1 Batch Sequential, 70
  4.2.2 Simple Repository, 71
  4.2.3 Virtual Repository, 75
  4.2.4 Hierarchical Layers, 79
  4.2.5 Evolution of Shared Information Systems in Business Data Processing, 80

4.3 Integration in Software Development Environments 82
  4.3.1 Batch Sequential, 83
  4.3.2 Transition from Batch Sequential to Repository, 83
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.3.3 Repository</td>
<td>85</td>
</tr>
<tr>
<td>4.3.4 Hierarchical Layers</td>
<td>86</td>
</tr>
<tr>
<td>4.3.5 Evolution of Shared Information Systems in Software Development Environments</td>
<td>88</td>
</tr>
<tr>
<td>4.4 Integration in the Design of Buildings</td>
<td>88</td>
</tr>
<tr>
<td>4.4.1 Repository</td>
<td>89</td>
</tr>
<tr>
<td>4.4.2 Intelligent Control</td>
<td>90</td>
</tr>
<tr>
<td>4.4.3 Evolution of Shared Information Systems in Building Design</td>
<td>91</td>
</tr>
<tr>
<td>4.5 Architectural Structures for Shared Information Systems</td>
<td>93</td>
</tr>
<tr>
<td>4.5.1 Variants on Dataflow Systems</td>
<td>93</td>
</tr>
<tr>
<td>4.5.2 Variants on Repositories</td>
<td>94</td>
</tr>
<tr>
<td>4.6 Some Conclusions</td>
<td>95</td>
</tr>
</tbody>
</table>

**CHAPTER 5 Architectural Design Guidance**

5.1 Guidance for User-Interface Architectures  
*by Thomas G. Lane*

5.1.1 Design Spaces and Rules                                     97
5.1.2 A Design Space for User-Interface Architectures            100
5.1.3 Design Rules for User-Interface Architecture              110
5.1.4 Applying the Design Space: An Example                     111
5.1.5 A Validation Experiment                                    113
5.1.6 How the Design Space Was Prepared                          114
5.1.7 Summary                                                    115

5.2 The Quantified Design Space                                   116
*by Toru Asada, Roy F. Swonger, Nadine Bounds, and Paul Duerig*

5.2.1 Overview                                                   116
5.2.2 Background                                                 116
5.2.3 Quantified Design Space                                    120
5.2.4 Conclusion                                                 127

**CHAPTER 6 Formal Models and Specifications**

6.1 The Value of Architectural Formalism                          129
6.2 Formalizing the Architecture of a Specific System            130
6.3 Formalizing an Architectural Style                            133
6.3.1 Filters                                                    134
6.3.2 Pipes                                                      135
6.3.3 Pipe-and-Filter System                                     136
6.4 Formalizing an Architectural Design Space                    139
6.5 Toward a Theory of Software Architecture                     142
6.6 What Next?                                                   142
6.7 Z Notation Used in This Chapter                               143
CHAPTER 7 Linguistic Issues

7.1 Requirements for Architecture-Description Languages 147
7.1.1 The Linguistic Character of Architectural Description, 148
7.1.2 Desiderata for Architecture-Description Languages, 151
7.1.3 Problems with Existing Languages, 155

7.2 First-Class Connectors 160
7.2.1 Current Practice, 160
7.2.2 Problems with Current Practice, 161
7.2.3 A Fresh View of Software System Composition, 165
7.2.4 An Architectural Language with First-Class Connectors, 166
7.2.5 The Promise of Explicit Architectural Notations, 171

7.3 Adding Implicit Invocation to Traditional Programming Languages 172
7.3.1 Introduction, 172
7.3.2 Adding Implicit Invocation to Ada, 174
7.3.3 Evaluation, 181

CHAPTER 8 Tools for Architectural Design 183

8.1 UniCon: A Universal Connector Language 183
8.1.1 Components and Connectors, 185
8.1.2 Abstraction and Encapsulation, 186
8.1.3 Types and Type Checking, 187
8.1.4 Accommodating Analysis Tools, 188

8.2 Exploiting Style in Architectural Design Environments 190
8.2.1 What Is Architectural Style?, 190
8.2.2 Automated Support for Architectural Design, 192
8.2.3 Observations about Environments for Architectural Design, 202

8.3 Beyond Definition/Use: Architectural Interconnection 204
8.3.1 Implementation versus Interaction, 205
8.3.2 Example, 206
8.3.3 The WRIGHT Model of Architectural Description, 208
8.3.4 Reasoning about Architectural Descriptions, 210
8.3.5 A Brief Explanation of Our Use of CSP, 211

CHAPTER 9 Education of Software Architects 213

9.1 Philosophy and Course Overview 213
9.1.1 Objectives, 213
9.1.2 Approach, 214

9.2 Course Description 215

9.3 Assignments 218
9.3.1 Purpose, 218
9.3.2 Readings, 219
9.3.3 Architectural Development Tasks, 220
9.3.4 Formal Modeling, 222
9.3.5 Analysis and Interpretation of a System, 222

9.4 Evaluation 223
   9.4.1 Lessons from the Initial Offering, 223
   9.4.2 Conclusions About Teaching Software Architecture, 225

Bibliography 227

Index 239