STRATEGIES FOR REAL-TIME SYSTEM SPECIFICATION
This page intentionally left blank
STRATEGIES FOR REAL-TIME SYSTEM SPECIFICATION

By

Derek J. Hatley   Imtiaz A. Pirbhai

Dorset House Publishing
353 West 12th Street
New York, New York 10014
To Sue
DJH

To my father,
Baba Anwar Shah Taji,
and in memory of my mother
IAP
Acknowledgments

First and foremost, we thank Wendy Eakin, our editor and publisher, who gently but firmly led these two neophyte authors through the rigors of writing a book. You too should thank her, dear Reader: For most of our careers we have worked on or with engineering specifications and standards, and if you are at all familiar with those kinds of documents, you will understand that she is all that stands between you and total boredom.

Many people have contributed in many ways to the work presented here, and while we can mention only a few, we thank you all. Bruce Chubb, Angelo Dimitriou, Rex Morin, and Rod Wierenga gave management support for development and introduction of the requirements method. Dave Bulman sat in on the method’s original conception, and, with his great knowledge and experience, helped us sift out the good ideas from the bad. The original Methods Team consisted of Kathy Hornbach, Bill Kelly, Martie Lorenz, John McCreary, and George Wood. Don Morrow remains the preeminent practitioner of the method itself, while John TenCate is master of the art of transforming the resulting model into a real system. Linda Goggins succeeded in the enormous task of coordinating and compiling the first (manual) application of the method. Kent McPherson performed some miracles with \LaTeX, the document preparation system for this book, to make it work the way we wanted.

Tim Petersen, Irv Reese, Dick Schoenmann, and Peter Sutcliffe provided the management support needed for the development of the architecture method, as well as support for the further expansion and acceptance of both the requirements and architecture methods in the user community.

Dan Bacon, Rodney Bell, James Bouhana, Bob Doering, Peter Hruschka, Lou Mazzucchelli, Don Morrow, Rick Swanborg, Tony Wasserman, and Bob Weisickle reviewed various drafts of the book and improved it immeasurably with their insights.

Finally, we give a very special acknowledgment to Paul Gartz. Paul has been an enthusiastic champion of these methods, and an inspiration to the rest of us working on them. His aggressive canvassing on behalf of the methods has been responsible for their acceptance by management and the industry.
## Contents

List of Figures ................................................. xv

Foreword ................................................... xxiii

Preface .................................................... xxv

### PART I: The Overall Strategy

1 Overview ..................................................... 3
   1.1 The Birth of the Requirements Model ................. 4
   1.2 The Birth of the Architecture Model ................. 7
   1.3 Compatibility of the Models ......................... 7
   1.4 Applicability of the Models ......................... 8
   1.5 The System Life Cycle ................................ 8

2 The Role of the Methods .................................. 11
   2.1 Structured Methods: What They Are ................. 11
   2.2 System Requirements Model ......................... 13
   2.3 System Architecture Model ......................... 19
   2.4 System Specification Model ......................... 25
   2.5 The Development Life Cycle ......................... 27
   2.6 Structured Methods: What They Are Not ............ 31
   2.7 Summary ............................................. 32

### PART II: The Requirements Model

3 Overview ..................................................... 35
   3.1 The Structure of the Model .......................... 37
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>The Process Model</td>
<td>41</td>
</tr>
<tr>
<td>4.1</td>
<td>Data Context Diagrams</td>
<td>41</td>
</tr>
<tr>
<td>4.2</td>
<td>Data Flow Diagrams</td>
<td>44</td>
</tr>
<tr>
<td>4.3</td>
<td>Leveling and Balancing</td>
<td>47</td>
</tr>
<tr>
<td>4.4</td>
<td>The Numbering System</td>
<td>49</td>
</tr>
<tr>
<td>4.5</td>
<td>Data Flows</td>
<td>49</td>
</tr>
<tr>
<td>4.6</td>
<td>Data Stores</td>
<td>52</td>
</tr>
<tr>
<td>4.7</td>
<td>Process Specifications</td>
<td>53</td>
</tr>
<tr>
<td>4.8</td>
<td>Interpreting the Process Model</td>
<td>56</td>
</tr>
<tr>
<td>4.9</td>
<td>Summary</td>
<td>59</td>
</tr>
<tr>
<td>5</td>
<td>The Control Model</td>
<td>61</td>
</tr>
<tr>
<td>5.1</td>
<td>Control Context Diagrams</td>
<td>61</td>
</tr>
<tr>
<td>5.2</td>
<td>Control Flow Diagrams</td>
<td>64</td>
</tr>
<tr>
<td>5.3</td>
<td>Control Flows</td>
<td>67</td>
</tr>
<tr>
<td>5.4</td>
<td>Data Conditions</td>
<td>69</td>
</tr>
<tr>
<td>5.5</td>
<td>Control Stores</td>
<td>70</td>
</tr>
<tr>
<td>5.6</td>
<td>Control Specifications</td>
<td>70</td>
</tr>
<tr>
<td>5.7</td>
<td>Process Controls</td>
<td>73</td>
</tr>
<tr>
<td>5.8</td>
<td>Summary</td>
<td>76</td>
</tr>
<tr>
<td>6</td>
<td>Finite State Machines</td>
<td>77</td>
</tr>
<tr>
<td>6.1</td>
<td>Combinational Machines</td>
<td>79</td>
</tr>
<tr>
<td>6.2</td>
<td>Sequential Machines</td>
<td>83</td>
</tr>
<tr>
<td>6.3</td>
<td>Incorporating Finite State Machines into CSPECs</td>
<td>89</td>
</tr>
<tr>
<td>6.4</td>
<td>Summary</td>
<td>97</td>
</tr>
<tr>
<td>7</td>
<td>Timing Requirements</td>
<td>98</td>
</tr>
<tr>
<td>7.1</td>
<td>Repetition Rate</td>
<td>99</td>
</tr>
<tr>
<td>7.2</td>
<td>Input-to-Output Response Time</td>
<td>99</td>
</tr>
<tr>
<td>7.3</td>
<td>Summary</td>
<td>102</td>
</tr>
<tr>
<td>8</td>
<td>Requirements Dictionary</td>
<td>103</td>
</tr>
<tr>
<td>8.1</td>
<td>Primitive Attributes</td>
<td>104</td>
</tr>
<tr>
<td>8.2</td>
<td>Group Structure</td>
<td>105</td>
</tr>
<tr>
<td>8.3</td>
<td>Dictionary Data Bases</td>
<td>107</td>
</tr>
<tr>
<td>8.4</td>
<td>Summary</td>
<td>110</td>
</tr>
<tr>
<td>9</td>
<td>Requirements Model Interpretation and Summary</td>
<td>111</td>
</tr>
<tr>
<td>9.1</td>
<td>The Requirements Model Interpreted</td>
<td>111</td>
</tr>
<tr>
<td>9.2</td>
<td>Requirements Model Summary</td>
<td>113</td>
</tr>
</tbody>
</table>
## CONTENTS

**PART III: Building the Requirements Model**

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>Overview</td>
<td>119</td>
</tr>
<tr>
<td>10.1</td>
<td>Model Users and Builders</td>
<td>119</td>
</tr>
<tr>
<td>10.2</td>
<td>The Sources of Requirements</td>
<td>120</td>
</tr>
<tr>
<td>10.3</td>
<td>The Model Building Process</td>
<td>121</td>
</tr>
<tr>
<td>11</td>
<td>Getting Started</td>
<td>124</td>
</tr>
<tr>
<td>11.1</td>
<td>User Requirements Statements</td>
<td>124</td>
</tr>
<tr>
<td>11.2</td>
<td>Separating Data and Control</td>
<td>125</td>
</tr>
<tr>
<td>11.3</td>
<td>Establishing the System Context</td>
<td>129</td>
</tr>
<tr>
<td>11.4</td>
<td>Partitioning the Top Levels</td>
<td>132</td>
</tr>
<tr>
<td>11.5</td>
<td>Summary</td>
<td>135</td>
</tr>
<tr>
<td>12</td>
<td>Developing the Model’s Structure</td>
<td>137</td>
</tr>
<tr>
<td>12.1</td>
<td>Abstraction and Decomposition</td>
<td>137</td>
</tr>
<tr>
<td>12.2</td>
<td>The Seven-Plus-or-Minus-Two Principle</td>
<td>138</td>
</tr>
<tr>
<td>12.3</td>
<td>Grouping and Decomposing Processes</td>
<td>139</td>
</tr>
<tr>
<td>12.4</td>
<td>Grouping and Decomposing Flows</td>
<td>140</td>
</tr>
<tr>
<td>12.5</td>
<td>Naming Processes and Flows</td>
<td>147</td>
</tr>
<tr>
<td>12.6</td>
<td>Use of Stores</td>
<td>149</td>
</tr>
<tr>
<td>12.7</td>
<td>Functionally Identical Processes</td>
<td>150</td>
</tr>
<tr>
<td>12.8</td>
<td>De-emphasizing the Control Model</td>
<td>151</td>
</tr>
<tr>
<td>12.9</td>
<td>Control Intensive Systems</td>
<td>153</td>
</tr>
<tr>
<td>12.10</td>
<td>The Dilemma of Detail: Requirements Versus Design</td>
<td>155</td>
</tr>
<tr>
<td>12.11</td>
<td>The Final Product</td>
<td>156</td>
</tr>
<tr>
<td>12.12</td>
<td>Summary</td>
<td>156</td>
</tr>
<tr>
<td>13</td>
<td>Preparing Process Specifications</td>
<td>158</td>
</tr>
<tr>
<td>13.1</td>
<td>The Role of Process Specifications</td>
<td>158</td>
</tr>
<tr>
<td>13.2</td>
<td>The Different Types of PSPECs</td>
<td>159</td>
</tr>
<tr>
<td>13.3</td>
<td>Some Important Signal Conventions</td>
<td>162</td>
</tr>
<tr>
<td>13.4</td>
<td>Structured English</td>
<td>165</td>
</tr>
<tr>
<td>13.5</td>
<td>Annotating with Comments</td>
<td>167</td>
</tr>
<tr>
<td>13.6</td>
<td>Summary</td>
<td>167</td>
</tr>
<tr>
<td>14</td>
<td>Preparing Control Specifications</td>
<td>169</td>
</tr>
<tr>
<td>14.1</td>
<td>Avoiding Control Specifications</td>
<td>169</td>
</tr>
<tr>
<td>14.2</td>
<td>Combinational Control</td>
<td>170</td>
</tr>
<tr>
<td>14.3</td>
<td>Sequential Control</td>
<td>176</td>
</tr>
<tr>
<td>14.4</td>
<td>Multi-Sheet CSPECs</td>
<td>182</td>
</tr>
</tbody>
</table>
## CONTENTS

<table>
<thead>
<tr>
<th>Page</th>
<th>Chapter</th>
</tr>
</thead>
<tbody>
<tr>
<td>14.5</td>
<td>Fitting CSPECs In</td>
</tr>
<tr>
<td>14.6</td>
<td>Summary</td>
</tr>
<tr>
<td><strong>15</strong> Defining Timing</td>
<td>190</td>
</tr>
<tr>
<td>15.1</td>
<td>Timing Overview</td>
</tr>
<tr>
<td>15.2</td>
<td>Response Time Specification</td>
</tr>
<tr>
<td>15.3</td>
<td>Summary</td>
</tr>
<tr>
<td><strong>16</strong> Managing the Dictionary</td>
<td>194</td>
</tr>
<tr>
<td>16.1</td>
<td>Flow Types</td>
</tr>
<tr>
<td>16.2</td>
<td>Dictionary Symbols</td>
</tr>
<tr>
<td>16.3</td>
<td>Summary</td>
</tr>
<tr>
<td><strong>PART IV: The Architecture Model</strong></td>
<td>201</td>
</tr>
<tr>
<td><strong>17</strong> Overview</td>
<td>203</td>
</tr>
<tr>
<td>17.1</td>
<td>Requirements-to-Architecture Template</td>
</tr>
<tr>
<td>17.2</td>
<td>Architecture Model Symbols</td>
</tr>
<tr>
<td><strong>18</strong> Architecture Diagrams</td>
<td>211</td>
</tr>
<tr>
<td>18.1</td>
<td>Architecture Context Diagrams</td>
</tr>
<tr>
<td>18.2</td>
<td>Flows and Interconnects</td>
</tr>
<tr>
<td>18.3</td>
<td>Architecture Flow Diagrams</td>
</tr>
<tr>
<td>18.4</td>
<td>Architecture Interconnect Diagrams</td>
</tr>
<tr>
<td>18.5</td>
<td>Summary</td>
</tr>
<tr>
<td><strong>19</strong> Architecture Dictionary and Module Specifications</td>
<td>225</td>
</tr>
<tr>
<td>19.1</td>
<td>Architecture Module Specification</td>
</tr>
<tr>
<td>19.2</td>
<td>Architecture Interconnect Specification</td>
</tr>
<tr>
<td>19.3</td>
<td>Timing Requirements</td>
</tr>
<tr>
<td>19.4</td>
<td>Architecture Dictionary</td>
</tr>
<tr>
<td>19.5</td>
<td>Summary</td>
</tr>
<tr>
<td><strong>20</strong> Completing the Architecture Model</td>
<td>235</td>
</tr>
<tr>
<td>20.1</td>
<td>Allocation to Hardware and Software</td>
</tr>
<tr>
<td>20.2</td>
<td>The Hardware and Software Architectures</td>
</tr>
<tr>
<td>20.3</td>
<td>The Complete Architecture Model</td>
</tr>
</tbody>
</table>
27 Home Heating System 326
  27.1 Problem Statement 326
  27.2 Requirements Model 330
  27.3 Architecture Model 340

28 Vending Machine 342
  28.1 Customer Dialogue 342
  28.2 Requirements Model 344
  28.3 Architecture Model 355

APPENDICES 363

APPENDIX A: Standard Symbols and Definitions 363
  A.1 Introduction 363
  A.2 Standard Symbols 363
    A.2.1 Data flow 363
    A.2.2 Control flow 364
    A.2.3 CSPEC bar 364
    A.2.4 Process 365
    A.2.5 Store 365
    A.2.6 Terminator (source or sink, also external) 365
    A.2.7 Architecture module 366
    A.2.8 Information flow vector 366
    A.2.9 Information flow channel 367
  A.3 Requirements Model 367
    A.3.1 Context diagrams 368
    A.3.2 Flow diagrams 370
    A.3.3 Process and control specifications 372
    A.3.4 Timing specifications 376
    A.3.5 Requirements dictionary 377
    A.3.6 Requirements model balancing rules 379
  A.4 Architecture Model 382
    A.4.1 Architecture context diagram 382
    A.4.2 Architecture flow and interconnect diagrams 384
    A.4.3 Architecture module and interconnect specifications 386
    A.4.4 Architecture dictionary 388
    A.4.5 Balancing rules 390
CONTENTS

APPENDIX B: Making the Models into Documents 392
   B.1 Organizing the Models .................................... 392
   B.2 Military Standards ....................................... 396

APPENDIX C: Information Modeling: The Third Perspective 398

References 403

Index 405
List of Figures

A Map of the Book ........................................... xxvi

1.1 The universal hierarchy of systems ..................... 4
1.2 The total system life cycle .............................. 9

2.1 Vending machine DFD .................................... 14
2.2 Vending machine PSPEC ................................. 14
2.3 Composite data entries from vending machine dictionary .................. 15
2.4 Vending machine CFD ................................... 16
2.5 Vending machine CSPEC ................................. 17
2.6 Primitive control entries from vending machine dictionary ............... 18
2.7 The structure of the requirements model ................ 18
2.8 Vending machine AFD ................................... 21
2.9 Vending machine AMS ................................... 21
2.10 Vending machine architecture dictionary entries .......... 22
2.11 Vending machine AID ................................... 22
2.12 Vending machine AIS ................................... 23
2.13 Augmented vending machine architecture dictionary ................. 23
2.14 The structure of the architecture model .................. 24
2.15 The formation of the system specification ................. 25
2.16 Requirements-to-architecture iterative loop ................ 26
2.17 System, software, and hardware specification structure .......... 28
2.18 Overlapping systems development phases .................. 28
2.19 Development life cycle spiral ........................... 30

3.1 Composite chart of the requirements model ................. 38

4.1 The process structure (highlighted) ..................... 42
4.2 Data context diagram ................................... 43
4.3 Data flow diagram ...................................... 45
4.4 Transaction centers .................................... 46
4.5 Four levels of data flow diagrams superimposed .......... 47
<table>
<thead>
<tr>
<th>List of Figures</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.6 Leveling, numbering, and parent/child relationships</td>
<td>48</td>
</tr>
<tr>
<td>4.7 Data flows</td>
<td>51</td>
</tr>
<tr>
<td>4.8 Splitting flows between levels</td>
<td>52</td>
</tr>
<tr>
<td>4.9 Two examples of process specifications</td>
<td>54</td>
</tr>
<tr>
<td>4.10 The structures of structured English</td>
<td>57</td>
</tr>
<tr>
<td>4.11 The primitive network</td>
<td>58</td>
</tr>
<tr>
<td>5.1 The control structure (highlighted)</td>
<td>62</td>
</tr>
<tr>
<td>5.2 Control context diagram</td>
<td>63</td>
</tr>
<tr>
<td>5.3 Control flow diagram</td>
<td>64</td>
</tr>
<tr>
<td>5.4 Variations on CSPEC bars</td>
<td>66</td>
</tr>
<tr>
<td>5.5 Comparison between discrete and continuous signals</td>
<td>68</td>
</tr>
<tr>
<td>5.6 A data condition</td>
<td>69</td>
</tr>
<tr>
<td>5.7 The model as a feedback control loop</td>
<td>72</td>
</tr>
<tr>
<td>5.8 Process controls</td>
<td>74</td>
</tr>
<tr>
<td>5.9 Controlling a transaction center</td>
<td>75</td>
</tr>
<tr>
<td>6.1 Decision table for a heating system</td>
<td>81</td>
</tr>
<tr>
<td>6.2 Condensed decision tables for a heating system</td>
<td>82</td>
</tr>
<tr>
<td>6.3 Rotary combination lock: a sequential machine</td>
<td>84</td>
</tr>
<tr>
<td>6.4 State transition diagram</td>
<td>85</td>
</tr>
<tr>
<td>6.5 State transition table</td>
<td>87</td>
</tr>
<tr>
<td>6.6 State transition matrix</td>
<td>88</td>
</tr>
<tr>
<td>6.7 State transition matrix (alternate form)</td>
<td>89</td>
</tr>
<tr>
<td>6.8 Combinational CSPEC, with its DFD and CFD</td>
<td>91</td>
</tr>
<tr>
<td>6.9 Sequential CSPEC, with its DFD and CFD</td>
<td>92</td>
</tr>
<tr>
<td>6.10 Block diagram of composite CSPEC</td>
<td>93</td>
</tr>
<tr>
<td>6.11 Composite CSPEC</td>
<td>94</td>
</tr>
<tr>
<td>6.12 Typical multi-sheet control specification</td>
<td>96</td>
</tr>
<tr>
<td>7.1 Response time specifications</td>
<td>100</td>
</tr>
<tr>
<td>7.2 Timing diagram to supplement the response time specification</td>
<td>101</td>
</tr>
<tr>
<td>8.1 Typical primitive continuous signal definitions</td>
<td>104</td>
</tr>
<tr>
<td>8.2 Typical primitive discrete signal definitions</td>
<td>104</td>
</tr>
<tr>
<td>8.3 Requirements dictionary symbols</td>
<td>106</td>
</tr>
<tr>
<td>8.4 Dictionary listing</td>
<td>108</td>
</tr>
<tr>
<td>8.5 Indented explosion of a large group flow</td>
<td>109</td>
</tr>
<tr>
<td>9.1 Primitive network with control structure</td>
<td>112</td>
</tr>
<tr>
<td>Figure</td>
<td>Title</td>
</tr>
<tr>
<td>--------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>11.1</td>
<td>Signal and process categories</td>
</tr>
<tr>
<td>11.2</td>
<td>Typical PSPEC: a primitive requirements statement</td>
</tr>
<tr>
<td>11.3</td>
<td>Assuming the worst: all conceivable functions included</td>
</tr>
<tr>
<td>11.4</td>
<td>Functions reduced after checking system scope</td>
</tr>
<tr>
<td>11.5</td>
<td>Diagram with a first-cut control flow</td>
</tr>
<tr>
<td>11.6</td>
<td>Composite top-level data and control flow diagram</td>
</tr>
<tr>
<td>11.7</td>
<td>DFD 0: Control &amp; Monitor Auto</td>
</tr>
<tr>
<td>11.8</td>
<td>CFD 0: Control &amp; Monitor Auto</td>
</tr>
<tr>
<td>12.1</td>
<td>Correction of uneven flow distribution</td>
</tr>
<tr>
<td>12.2</td>
<td>Correction of clustered processes</td>
</tr>
<tr>
<td>12.3</td>
<td>Splitting and merging data flows</td>
</tr>
<tr>
<td>12.4</td>
<td>Decomposing data flows</td>
</tr>
<tr>
<td>12.5</td>
<td>Splitting off unused flows</td>
</tr>
<tr>
<td>12.6</td>
<td>Flows with common members</td>
</tr>
<tr>
<td>12.7</td>
<td>Multiple flows</td>
</tr>
<tr>
<td>12.8</td>
<td>Two-way flows</td>
</tr>
<tr>
<td>12.9</td>
<td>The half-arrow convention</td>
</tr>
<tr>
<td>12.10</td>
<td>A flow tree</td>
</tr>
<tr>
<td>12.11</td>
<td>Functionally identical processes</td>
</tr>
<tr>
<td>12.12</td>
<td>Composite chart of the model for control intensive system</td>
</tr>
<tr>
<td>13.1</td>
<td>A PSPEC with equations</td>
</tr>
<tr>
<td>13.2</td>
<td>A tabular PSPEC</td>
</tr>
<tr>
<td>13.3</td>
<td>A flight profile used in a PSPEC</td>
</tr>
<tr>
<td>13.4</td>
<td>Geometrical relationships used in a PSPEC</td>
</tr>
<tr>
<td>13.5</td>
<td>Continuous, discrete-valued, and time-discrete signals</td>
</tr>
<tr>
<td>14.1</td>
<td>A full decision table</td>
</tr>
<tr>
<td>14.2</td>
<td>A reduced decision table</td>
</tr>
<tr>
<td>14.3</td>
<td>An ambiguous set of input combinations</td>
</tr>
<tr>
<td>14.4</td>
<td>Alternative representation of transaction center control</td>
</tr>
<tr>
<td>14.5</td>
<td>Data flow diagram showing throttle control</td>
</tr>
<tr>
<td>14.6</td>
<td>Control specification showing throttle control</td>
</tr>
<tr>
<td>14.7</td>
<td>A multi-sheet state transition matrix</td>
</tr>
<tr>
<td>14.8</td>
<td>Organization of a composite CSPEC</td>
</tr>
<tr>
<td>14.9</td>
<td>A floating CSPEC</td>
</tr>
<tr>
<td>15.1</td>
<td>Dictionary, response time, PSPEC, and CSPEC timing specifications</td>
</tr>
<tr>
<td>Figure</td>
<td>Title</td>
</tr>
<tr>
<td>--------</td>
<td>-------</td>
</tr>
<tr>
<td>16.1</td>
<td>Flow types and their attributes</td>
</tr>
<tr>
<td>17.1</td>
<td>Architecture model components</td>
</tr>
<tr>
<td>17.2</td>
<td>Requirements template</td>
</tr>
<tr>
<td>17.3</td>
<td>Architecture template</td>
</tr>
<tr>
<td>17.4</td>
<td>Architecture template for the Automobile Management System</td>
</tr>
<tr>
<td>17.5</td>
<td>Architecture model layering</td>
</tr>
<tr>
<td>17.6</td>
<td>Architecture module symbol</td>
</tr>
<tr>
<td>17.7</td>
<td>Information flow vector symbol</td>
</tr>
<tr>
<td>17.8</td>
<td>Information flow channel symbols</td>
</tr>
<tr>
<td>18.1</td>
<td>ACD for a flight management system</td>
</tr>
<tr>
<td>18.2</td>
<td>ACD for the Automobile Management System</td>
</tr>
<tr>
<td>18.3</td>
<td>AFD for a flight management system</td>
</tr>
<tr>
<td>18.4</td>
<td>AFD for a data acquisition subsystem</td>
</tr>
<tr>
<td>18.5</td>
<td>Architecture template for a flight management system</td>
</tr>
<tr>
<td>18.6</td>
<td>Architecture template for a data acquisition subsystem</td>
</tr>
<tr>
<td>18.7</td>
<td>AFD 0: Automobile Management System</td>
</tr>
<tr>
<td>18.8</td>
<td>AFD layering</td>
</tr>
<tr>
<td>18.9</td>
<td>AFD 2: Shaft Interface Module</td>
</tr>
<tr>
<td>18.10</td>
<td>AFD 5: Throttle Interface Module</td>
</tr>
<tr>
<td>18.11</td>
<td>AID for a flight management system</td>
</tr>
<tr>
<td>18.12</td>
<td>Redundancy of architecture modules</td>
</tr>
<tr>
<td>18.13</td>
<td>Redundancy of architecture channels</td>
</tr>
<tr>
<td>18.14</td>
<td>AID 0: Automobile Management System</td>
</tr>
<tr>
<td>18.15</td>
<td>AID 2: Shaft Interface Module</td>
</tr>
<tr>
<td>19.1</td>
<td>Architecture model support components</td>
</tr>
<tr>
<td>19.2</td>
<td>Architecture module specification</td>
</tr>
<tr>
<td>19.3</td>
<td>A DFD, CFD, and their CSPEC to be allocated</td>
</tr>
<tr>
<td>19.4</td>
<td>Splitting CSPECs for allocation</td>
</tr>
<tr>
<td>19.5</td>
<td>Traceability matrix for the Automobile Management System</td>
</tr>
<tr>
<td>19.6</td>
<td>Interconnect channel and corresponding AIS</td>
</tr>
<tr>
<td>19.7</td>
<td>A typical AIS</td>
</tr>
<tr>
<td>19.8</td>
<td>Partial architecture dictionary for the Automobile Management System</td>
</tr>
<tr>
<td>20.1</td>
<td>Partitioning between hardware and software</td>
</tr>
<tr>
<td>20.2</td>
<td>System, hardware, and software requirements hierarchy</td>
</tr>
<tr>
<td>20.3</td>
<td>System, hardware, and software architecture hierarchy</td>
</tr>
</tbody>
</table>
LIST OF FIGURES

20.4 A vertical slice through the architecture model ............ 239
21.1 Hierarchical nature of natural systems .................. 246
21.2 Hierarchical nature of systems development ............. 247
22.1 Outside-in approach to architecture development ......... 249
22.2 Input and output buffers for the requirements model ...... 250
22.3 Input and output processes for the Automobile Management System ........................................ 251
22.4 Enhancements to the Automobile Management System DFD for user interface processing .................. 252
22.5 DFD/CFD for the driver interface process ................ 253
22.6 User interface buffer to the requirements model .......... 254
22.7 Enhancements for maintenance and self-test buffer ....... 255
22.8 Maintenance processing added to the requirements model .. 256
22.9 Enhanced DFD for Automobile Management System ........ 257
22.10 Enhanced CFD for Automobile Management System ....... 258
22.11 Architectural enhancements to the requirements model ... 259
23.1 ACD for the Automobile Management System ............ 264
23.2 Generic DFD/CFD/CSPEC .................................. 265
23.3 AFD 0: Automobile Management System ................ 266
23.4 AID 0: Automobile Management System ................ 267
23.5 Architecture module specification for the Automobile Management System ................................... 268
23.6 Architecture interconnect specification for the Automobile Management System .............................. 269
23.7 Architecture dictionary for the Automobile Management System .................................................. 269
23.8 Architecture development process ........................ 272
24.1 Hierarchical nature of the Automobile Management System .... 273
24.2 DFD of requirements allocated to the automobile management computer module ............................. 274
24.3 CFD of requirements allocated to the automobile management computer module ............................. 275
24.4 AFD for automobile management computer hardware configuration .............................................. 277
24.5 Requirements allocated to hardware ...................... 277
24.6 Enhanced hardware requirements ......................... 278
24.7 AFD for shaft interface module .......................... 278
LIST OF FIGURES

24.8 AID for shaft bus receiver .......................... 278
24.9 Enhanced DFD for the automobile management computer software ........................................ 280
24.10 Enhanced CFD for the automobile management computer software ........................................ 281
24.11 A structure chart ........................................ 282
24.12 Software development structure ........................ 283
24.13 Hardware and software architecture development process ...................................................... 284

25.1 Iterative process for developing system requirements and architecture ........................................ 287
25.2 System modeling process ................................ 288
25.3 Hardware and software partitioning process .......... 289
25.4 Hardware and software modeling process ............ 289
25.5 Application of requirements and architecture modeling to large systems .............................. 290
25.6 Integrated system, hardware, and software model ................................................................. 291

A.1 Data flow: A solid arc with a name .................... 364
A.2 Control flow: A dashed arc with a name .............. 364
A.3 CSPEC bar: A short unlabeled bar ...................... 364
A.4 Process: A circle with a name and a number ......... 365
A.5 Store: A pair of parallel lines containing a name .... 365
A.6 Terminator: A rectangle containing a name ........... 366
A.7 Architecture module: A rounded rectangle containing a name and number ................................. 366
A.8 Information flow vector: A solid or dashed line with a name ..................................................... 367
A.9 Architecture information flow channels ................ 367
A.10 Components of the requirements model ............... 368
A.11 Sample data context diagram .......................... 369
A.12 Sample control context diagram ....................... 370
A.13 Sample data flow diagram ................................ 371
A.14 Data flow diagram naming and numbering rules ...... 372
A.15 Sample control flow diagram .......................... 373
A.16 DFD and CFD naming and numbering .................. 373
A.17 Samples of process specifications ...................... 374
A.18 Process specification naming and numbering rules ... 375
A.19 Generic CSPEC for combinational system ............ 376
A.20 Sample CSPEC for combinational system ............. 377
A.21 Generic CSPEC for sequential system ................. 378
LIST OF FIGURES

A.22 Sample CSPEC for sequential system ...................... 379
A.23 Illustration of flow decomposition in the requirements
dictionary .................................................. 381
A.24 Components of the architecture model .................... 382
A.25 Architecture template ................................... 383
A.26 Sample architecture context diagram ...................... 383
A.27 Sample architecture flow diagram ......................... 385
A.28 AFD naming and numbering rules ......................... 386
A.29 Sample architecture interconnect diagram ............... 387
A.30 Sample architecture module specification ................. 388
A.31 Sample architecture interconnect specification .......... 389
A.32 Example of enhancements to dictionary .................. 389

C.1 Requirements projections of a system ...................... 399
C.2 Architecture template with three requirements projections 400
Foreword

“Most systems people use the term real-time rather loosely,” the young manager said. We were seated over dinner with three members of her staff and some other managers who took part in the day’s seminar. “They say they’ve got a real-time constraint when they’re worried about impatient insurance brokers or bankers sitting in front of their terminals. A real-time system, in their minds, is just one that needs to be ‘quick as a bunny.’ If they fail to meet that constraint, their users might be inconvenienced or even annoyed. When we use the term, it means something rather different.”

Her co-workers began to smile, knowing what was coming. “We build systems that reside in a small telemetry computer, equipped with all kinds of sensors to measure electromagnetic fields and changes in temperature, sound, and physical disturbance. We analyze these signals and transmit the results back to a remote computer over a wide-band channel. Our computer is at one end of a one-meter long bar and at the other end is a nuclear device. We drop them together down a big hole in the ground and when the device detonates, our computer collects data on the leading edge of the blast. The first two-and-a-quarter milliseconds after detonation are the most interesting. Of course, long before millisecond three, things have gone down hill badly for our little computer. We think of that as a real-time constraint.”

I had been lecturing that day on the use of data flow modeling techniques for system specification. There had been the odd question about the use of such techniques for real-time systems, and my (typically glib) answer was that the techniques were applicable, though perhaps not totally sufficient. After all, my own earliest work with data flow modeling had been at Bell Labs and at La CEGOS Informatique, in both cases working on real-time systems. Or were those real-time systems? Maybe they were really just systems that needed to be ‘quick as a bunny’? The young manager’s graphic example had left me in some doubt.

It has always been clear that something more than the basic tools of structured analysis are needed to specify timing and synchronization requirements.
In the simplest case, that “something” could be as trivial as a set of textual annotations, perhaps directly on the data flow diagrams. But for even slightly more complicated cases, there is the possibility of linked timing and ordering constraints that may involve a dozen or more system components. There has been a need for a systematic way to deal with such constraints.

Over the last few years, I began to hear favorable reports of some real-time modeling techniques called the Hatley/Pirbhai Extensions. This multi-perspective approach combined data flow decomposition with model components constructed in control- and information-space. The result appealed to me as a specification technique that was not only applicable, but for most cases sufficient for real-time systems. I contacted the developers and started to try out their extensions.

The act of writing a Foreword is a kind of endorsement. It says, if nothing else, “Read this book, it couldn’t hurt.” In this case, I can be considerably more positive than that. I can tell you that I learned valuable new techniques from Hatley and Pirbhai, and that I apply them regularly on real-world real-time applications.

October 1987
Camden, Maine

Tom DeMarco
The Atlantic Systems Guild
Preface

This book describes two methods for specifying, respectively, the requirements for and the design structure of software-based systems. Although the methods grew up around real-time embedded systems, systems of all types and sizes have benefited from them, largely because of their flexibility and adaptability. The methods can be viewed as an integrated toolkit from which all the tools are compatible, but only those that are useful for the particular job need be used.

The subject of this book is neither computer science nor software. Although the vast majority of the systems to which the methods are applied will in fact be implemented using software in digital computers, the methods do not address how to write that software or how to design that hardware. What the methods do address is how to specify the problems that the hardware and software must solve.

Organization and audience of the book

The book is intended for a wide range of readers. We assume that you are involved, or at least interested, in software-based systems, but that your needs may range from just a general understanding to an in-depth working knowledge. We have tried to organize the book to make it easy for you to select those parts that are of specific interest to you and to avoid those that are not.

The figure on the following page shows the layout of the book. Part I provides an overview from which everyone will benefit. Parts II and IV describe what the methods are, and if you only need to understand the specifications resulting from their use, then these two parts are enough for you to read. If you need to prepare specifications, then you will also need to read Parts III and V, which describe how to use the methods. Part VI contains examples of the application of the methods, and we conclude the book with appendices and a brief bibliography.

Decide what your area of interest is—requirements analysis, design, or both—and what depth of understanding you need—general familiarity, use
of the completed specifications, or building the specifications—and it will be clear from the figure which parts of the book you should read.

Derek J. Hatley
Wyoming, Michigan

Imtiaz A. Pirbhai
Seattle, Washington
STRATEGIES FOR REAL-TIME SYSTEM SPECIFICATION
This page intentionally left blank
Chapter 3
Overview

Our strategy is to establish what the desired end product is, before describing how to produce it. Consequently, in Part II, we describe in detail the completed requirements model, as it would be received by the user, and in Part III, we describe how to construct the model. The principal tools of the requirements model are flow diagrams: graphical models of signal flows and processes acting on those flows. Flow diagrams consist of data flow diagrams and control flow diagrams. The former are essentially the same as the DFDs of DeMarco [4]; the latter are similar to DFDs but with some important and unique differences.

The other components of the requirements model are process specifications and the requirements dictionary (which are similar to DeMarco’s mini-specifications and data dictionary), and control specifications (which are unique to this method).

You might wonder why we are describing here those parts of the method that have been described so often, and so well, before. The answer is, first, that we want the book to stand alone as a complete description of the whole method. Second, the earlier descriptions allowed a commendable degree of flexibility in the ways the method could be used. This has resulted in many different conventions being adopted in its use. It is therefore important to explain the particular conventions we have adopted in extending the method. If you have already been using “basic” structured analysis, you might very well find that some of our conventions are quite different from yours.

Key features of the method are

- It is a purely abstract model of the requirements, not a physical model of the design. It thus represents what has to be designed, not how to design it.
• It is diagrammatic, exploiting our ability to absorb visual information more effectively than textual or verbal information ("a picture is worth a thousand words").

• It hides unnecessary detail at any given level, and provides for progressive decomposition from that level down—the information hiding, abstraction, and decomposition principles put forward by Parnas [13]—thus allowing presentation of the requirements from a top-level overview down to the most intricate detail.

• It meets the needs of large, complex systems, including real-time systems, for representation of a process structure and a control structure. It does this by integrating DeMarco's structured analysis with finite state machine theory. Neither of the methods alone can meet these needs.

• It has built-in consistency checks within and between all its components. This feature of structured analysis has been extended to the whole structure, taking the guesswork out of the rigor and consistency of the model.

• It is self-indexing, another feature of structured analysis that has been successfully extended. It is easy to find your way around the resulting model without referring to page or paragraph numbers (although formal documentation requirements usually force us to keep these numbers).

The first of these features—the abstract nature of the model—deserves special emphasis, because it seems to be the one that newcomers to the method find hardest to grasp. Perhaps the best way to approach it is to understand the motivation for wanting such a feature, and this is best understood by reviewing the history of structured methods development.

Although structured methods generally represent their subject matter in a top-down fashion, the methods themselves have evolved in a bottom-up fashion. As software-based systems grew larger and more complex, the first obvious problem to be overcome was unmanageable and incomprehensible code. The need to solve this problem gave rise to structured programming in the form of constraints on module size, entry and exit points, branching, and so on. It also gave rise to high order languages, and, more recently, to program design languages.

Structured programming improved the quality of the code within individual modules, but still, as systems continued to grow in size and complexity, the interactions between the modules became as complex as the earlier code had been. Structured design methods, in which modules are constrained to communicate with each other in an organized way, were developed to deal
with this complexity. These methods improved the quality of the software structure external to the modules, such that if both structured design and structured programming methods were conscientiously applied, the resulting system would work exactly as the designer intended.

Unfortunately, it still often happens that what the designer intended is not what the user wanted! The reason for this is that as the technology allows us to make systems that have ever larger capacity and complexity, the requirements themselves for such systems become large, complex, and difficult to describe and understand. Thus, the need was established for a method to represent the functional requirements of a system in an organized and understandable way, and in a way that is entirely independent of the eventual design and implementation of the system. Given such a model, various designs can be developed and compared without changing the requirements representation.

DeMarco's structured analysis method was designed to meet this need for data processing systems, and the extended structured analysis method described here was designed to meet the need for more general systems.

The model does, unfortunately, have a machine-like appearance; moreover, we are used to models that are intended to represent the actual design structure or implementation. As should now be clear, this is not the intent here. Again, it is a purely abstract model, with highly idealized properties, whereby each of its processes is independently data triggered and infinitely fast—unlikely characteristics for a real system. These properties are deliberately chosen for the very reason that we can ignore the constraints of the physical world, and concentrate exclusively on the required functionality.

As we build a requirements model, we follow this separation principle by including those characteristics that are of concern to the users, and deliberately excluding those that are not. The latter are left to the designer, and the moment we allow them to migrate into the requirements model, then we are including items that are not requirements; in so doing, we are placing unnecessary and artificial constraints on the designer.

We will remind you of this property at appropriate places throughout Parts II and III to emphasize its importance in each aspect of the model.

### 3.1 The Structure of the Model

Figure 3.1 is a composite chart, which enlarges on the diagram of Figure 2.7, and illustrates how the major components of the requirements model relate to each other. We will describe the major roles and relationships of these components before going into their individual details in the following chapters.
Figure 3.1. Composite chart of the requirements model.
At the top of Figure 3.1 are the data context diagram and the control context diagram. These show, respectively, the data and the control signals flowing between the system and the external entities with which the system must communicate. They also show the overall function of the system in the form of a single process. These diagrams represent the highest-level, or most abstract, view of the requirements, and show the central role the system is required to perform within its external environment.

Next (below the context diagrams in Figure 3.1) are two more flow diagrams—the level 1 DFD and CFD. They too represent the overall function of the system in graphical form, but in somewhat more detail than the context diagrams.

Below the level 1 diagrams are level 2 diagrams. For purposes of illustration, the figure shows just one DFD and one CFD at level 2, but, in fact, each of the several subprocesses on the level 1 diagrams may have its own level 2 DFD and CFD, so there will be several diagrams at this level. Just as the level 1 diagrams represent a more detailed statement of the system's purpose, so the level 2 diagrams represent more detailed statements of the purposes of the subprocesses on the level 1 diagram.

Similarly, level 3 diagrams decompose the processes and signal flows on the level 2 diagrams into more detailed statements, with a further multiplication of the number of diagrams. This decomposition continues level by level, each step revealing finer and finer details of the originally stated system's purpose.

The decomposition ends, for any given subprocess on the DFD side, with a concise statement of the purpose of a process using simple text, equations, tables, or diagrams. This statement constitutes a process specification, as illustrated at the bottom of the column of DFDs. A process that is described by a process specification is called a functional primitive, or just a primitive.

On the CFD side, the decomposition parallels that of the DFDs, but some of the control signals flow to and from control specifications, shown in the center column of Figure 3.1. The CSPECs describe the finite state machine functions of the system, which we will discuss in Chapter 6. Each CSPEC is associated with a specific CFD—the one adjacent to it in the diagram. It will usually control processes, but only those in the specific DFD adjacent to it in the figure. This close association of a DFD, CFD, and CSPEC is an important part of the structure and self-consistency properties of the requirements model.

The system input-to-output timing requirements are shown in the timing specification, shown between the context and level 1 diagrams (and detailed in Chapter 7). These specifications simply list events that are detected at the system inputs (the system stimuli), the corresponding events that are
required to occur at the system outputs (the system responses), and the timing constraints within which the system must generate these responses.

Finally, Figure 3.1 shows the requirements dictionary. Although it is shown outside the main structure, the RD serves a vital role throughout the model since it contains precise definitions of all the data and control flows in all their levels of decomposition.

The remaining chapters of Part II describe each component of the method in detail, but it is important not to lose track of how they fit into the overall structure of Figure 3.1 as you study them.

For the purposes of this description, we refer to the DFDs and PSPECs as the process model, the CFDs and CSPECs as the control model, and the two of them with the requirements dictionary and timing specifications as the requirements model. This is in keeping with the evolution of the methods: The process model corresponds to the classical structured analysis model; the control model is the part we added, derived from finite state machine theory; and the requirements model is the complete, integrated combination of the two.
Index

Actions, 15, 83–88, 93, 97, 178–79
logic, 93, 94, 184
naming, 177, 189
Activation table: see Process activation table
Architecture communication channels:
AIS and, 229–32
redundancy of, in AID, 221–22
Architecture context diagram (ACD), 203–204,
211–13, 263–64, 382–84
components in, 211
example, 212, 213, 383
for Automobile Management System, 264, 322
Architecture dictionary (AD), 20, 25, 203–204,
225–26, 233–34, 388–89
of Automobile Management System, 233,
325
of vending machine example, 22, 23, 358
Architecture flow diagram (AFD), 20, 25,
203–204, 213–20, 264–68
decomposition of, 215–17
examples, 215, 216
naming and numbering rules, 384, 386
of Automobile Management System, 218,
220, 266, 277–78, 322
of vending machine example, 21, 357
structure chart and, 282
Architecture interconnect diagram (AID),
21–22, 25, 203–204, 213, 218–24,
264–68, 385–87
decomposition of, 215–16
layering rules, 222
naming and numbering rules, 222, 385,
386
of Automobile Management System,
277–78, 323
of vending machine example, 22, 357
redundancy in, 219, 221–24
Architecture interconnect specification (AIS),
22, 23, 203–204, 229–32, 387–89
of Automobile Management System, 231,
324
of vending machine example, 23, 359
Architecture model, 7, 12, 19–27, 99, 382–91
Automobile Management System, 319–25
balancing, 238, 268–71, 287–88, 368, 390–91
components, 203–204
decomposition of, 215
hierarchy of, 207
illustration, 24
implementation of, in hardware and software,
235–40
of Home Heating System example,
340–41
summary, 24–25, 203–10, 238–40
symbols of, 207–10
vending machine example, 355–59
Architecture module, 20, 208
automobile management computer module
example, 235–36, 274–77, 280–81
decomposition of, 215
defined, 366
implementation of, 235–36
in ACD, 211
in AID, 219, 221–24
naming rule, 208, 366
number in AFD, 214
partitioning of requirements, 273–85
redundancy of, 219, 221
symbols, 208, 366
timing constraints and, 232–33
Architecture module specification (AMS), 20,
25, 215, 225–29
of Automobile Management System, 268,
324
of vending machine example, 21
traceability matrix and, 226, 229–30
Architecture template, 204–207, 217, 248–62,
279–82, 383, 400

405
benefits of, 260–62
for Automobile Management System, 206
software requirements and, 279–82
Automobile management computer module, 235–36, 274–77, 280–81
Automobile Management System, 295–325
ACD example, 213, 322
AD example, 234, 325
AFD example, 218, 220, 322
AID example, 223, 323
AIS for, 324
AMS example, 226–27, 324
architecture model of, 239, 319–50
architecture template for, 206
control diagrams for, 301
CSPEC for, 303, 308, 312, 314
data vs. control issues, 128–29
enhanced CFD for, 258, 321
enhanced DFD for, 257, 320
flow names in, 148
hierarchical nature of, 273
input and output processing for, 249–51
PSPEC of, 160, 305–306, 309, 312, 315
RD for, 316–18
response time specification, 100
safety requirements in, 245
sequential machine in, 179–82
STD in, 85, 87
STM in, 88, 89
system context, 129–32
timing specification, 304
top-level partitioning, 133–36
traceability matrix for, 230
transient signal in, 163
user interface processing, 251–54
Automated tools, 6, 206, 404
used for balancing, 49, 140, 166
used for dictionary, 107, 140, 197, 199, 395
Avionics system example, 4–7
Backus-Naur form, 105–107, 196–99
Balancing:
architecture model, 268–71, 390–91
requirements model, 47–49, 113, 123, 379–81
(see also specific entries)
Balzer, R., 139, 404
Bar symbol, 65–66, 71, 89, 93, 185, 364, 371
Boehm, B., 29, 403
Boolean equation form, 79–80, 97, 171, 198
Booth, T., 86, 173, 403
CASE tools, 6, 404
Combinational control specification, 90–93, 169–78
Combinational machine, 78–82, 114, 152, 157, 169–76
CSPEC for, 375–77
Comment:
in PSPEC, 55, 167
in requirements dictionary, 106–107
Conditional construct, 166
Consistency checking, 5, 36
of architecture model, 268–71
of CSPECs, 71
of DFDs, 49, 50
of requirements model, 36, 48, 50, 52, 113, 115
Constantine, L., 6, 282, 404
Context diagram:
context process in, 137
control, 38–39, 41, 61–63, 76, 301, 369
data, 38, 39, 41–44, 59, 63, 368
level 1 and, 130–31
nonprimitive flows and, 105–107
primitive flows in, 67–68, 104–105
timing requirements and, 99
(see also Control context diagram, Data context diagram)
Context process, 43–44, 63, 137
Continuous (analog) machine, 77–78, 97
digital computer and, 78
PSPECs and, 77
Continuous signals, 67–68, 77–78, 103, 125, 165
equation of primitive in RD, 104, 196–97
Control context diagram (CCD), 38–39, 41, 61–63, 76, 301, 369–70
of Automobile Management System, 301
of vending machine example, 344
Control flow, 13, 41, 61, 64–68, 103
defined in requirements dictionary, 67, 104, 115
determined in requirements model, 125–29
graphic convention, 61, 63, 67, 364
naming, 364
types and attributes, 195–97
Control flow diagram (CFD), 13, 15, 19, 35, 64–68, 76, 113, 371–72
allocation in architecture model, 227–29
balancing, 64, 113
composite example of, 33
control store on, 70, 76, 150, 365
control flow on, 15–16, 67–68
control signals and, 15–16, 67
Control flow diagram (CFD), 13, 15, 19, 35, 64–68, 76, 113, 371–72
allocation in architecture model, 227–29
balancing, 64
control store on, 70, 150
decomposition of, 39, 65
interface with CSPEC (bar symbol), 65–66, 71, 89, 90, 185, 364, 371–72
leveled, 39, 64, 113
link to DFD, 69, 71
naming and numbering, 64, 113–14, 372–73
of automobile management computer module, 275, 281
of Automobile Management System, 258, 302ff.
of vending machine example, 16, 356
sample, 373
Control model, 13, 15–19, 40, 61–76
control-intensive system and, 153–54
implementation details and, 151–52
purpose of, 152
vending machine example, 13–18
Control outputs, 73
(see also Process controls)
Control signal, 15–16, 67
RD and, 16
symbol for, on CFD, 15
(see also Control flow)
Control specification (CSPEC), 15, 16, 19, 39, 64–67, 70–73
balancing, 95, 368, 379–81
bar symbol on, 65–66, 71, 89, 93, 185, 364, 371–72
combinational vs. sequential control, 90–93, 169–78, 376–77
composite, 93–94, 184
decision table and, 171–74
finite state machines and, 39, 73, 89–97, 188
floating, 187
interface with CFD, 65–66, 71, 89, 90, 185, 364
multi-sheet conventions for, 95–97, 114, 182–85
naming and numbering, 71, 114
of Automobile Management System, 303, 308, 312, 314
of combinational machine, 79–82, 170–76, 375–77
of vending machine example, 17, 20
PAT in, 174–75
placement of, in model, 185–86
primitive, 153–54
process controls and, 73–76, 90
purpose of, 73, 114, 169
rule for allocation in architecture model, 227–29
state transition diagram and, 65, 90, 92, 93
timing requirements in, 188
transaction center and, 75, 175–76
users’ guide for, 95, 114, 153, 185
Control store, 70, 76, 149–50
(see also Data store)
naming, 70, 150, 365
Control structure, 5
balancing, 76
determining in requirements model, 111–13, 125–29
illustrated, 62, 112
Cruise control system example: see Automobile Management System

Data abstraction, 137, 209
(see also Information abstraction)
Data condition, 19, 69–70, 76, 114
example, 69
Data context diagram (DCD), 38, 39, 41–44, 368–69
external primitive flows in, 191
of Automobile Management System, 301
of vending machine example, 344
Data flow, 41, 43, 44, 49–52, 59
arcs, 50–51
balancing, 47–49, 59, 143–44
consistency checking of, 49, 50, 52
data stores and, 52, 150
decomposition of, 50, 143
defined, 363
determined from user requirements statement, 125–29
graphic conventions, 43, 50–51, 143–47, 364
in requirements dictionary, 13, 103, 115, 144–45
naming rules, 46, 50, 143–44, 147–49, 363
primitive, 49
splitting and merging, 50, 143
types and attributes, 194–97
Data flow diagram (DFD), 13, 19, 35, 44–53, 114, 370–71
allocation in architecture model, 227–29
composite example of, 133
data signal and, 67
decomposition of, 39, 45, 47
leveling and balancing, 39, 45, 47–49, 59, 113
link to CFD, 69, 71
naming and numbering, 49, 370–74
of automobile management computer module, 274, 280
of Automobile Management System, 302ff.
of vending machine example, 14
parent/child relationship, 45, 47–49
sample, 45, 371

Data modeling: see Information modeling

Data process, determining, 128
Data store, 52–53, 59, 149–50
defined, 44, 365
flows and, 150
naming rules, 52–53, 150, 365
symbol, 44

Data structure:
determined in requirements model, 125–29

Date, C., 194, 403

Decision table (DT), 73, 79–82, 97, 114, 166, 375
Home Heating System example, 81–82
in CSPEC, 171–74
luggage lock example, 79
PAT, 90–91, 174–75
STM and, 93–94
transaction center and, 175–76

Decomposition, 36, 137–38
of flows, 140–47
of processes, 45, 139–40
7 ± 2 principle, 138–39

DeMarco, T., 5, 35, 36, 37, 56, 403
structured analysis method of, 5, 95

Design, 243–44
constraints, 203
internal timing requirement and, 98, 102, 114, 190
phase, 155

Discrete signal, 67–68, 77–78, 103–104, 125, 165
equation of primitive in RD, 104, 196–97
Don’t care condition, 80, 172, 173, 189, 194

Entity, in ACD, 211
Equation, in PSPEC, 159–60
Event, 15, 83–88, 93, 97, 178–79
asynchronous, 150
logic, 93, 94, 184
naming, 189
response time, 99–102

Feedback control loop, 71–73, 114
Finite state machine, 77–97, 125, 153
combinational, 78–82, 152, 157, 169–76, 375–77
collection model and, 40, 70–73, 76
collection structures and, 5, 113
CSPEC and, 15, 39, 73, 89–97
model, 77
sequential, 78, 83–89, 176–82, 375, 378–79
theory, 5, 36, 89

Flavin, M., 403

Flight management system example, 78
response time specification, 100
Flow arcs, 43–44, 50–51
symbol on CFD, 65
symbol on DFD, 16

Flow diagram: see Control flow diagram, Data flow diagram

Flows:
allocation in AMS, 226–29
balancing, 143–44
(see also Control flow, Data flow)

Functional primitive: see Primitive process

Functional requirements, 3–10, 13, 19
(see also Control model, Process model,
Timing requirements)

Fundamental entity, 208

Hardware design, 276–78
requirements allocated to hardware, 277
Hardware/software implementation of architecture model, 235–36, 276–78
example, 236
Hardware/software models, 288–91
Hierarchical representation, 4, 9–10, 28–29, 36, 39
of Automobile Management System, 273

Hofstadter, D., 216, 403

Home Heating System example, 326–41
Hopcroft, J., 86, 403

Implementation, system, 10, 12, 56, 114
architecture, 229
of process model, 56
timing constraints and, 232–33

Information abstraction, 5, 36, 113, 137–38

Information flow channel, 213
in AFD and AIS, 25
in AID, 219
naming rule, 209, 367
symbol, 209, 367
INDEX 409

Information flow vector, 208–209
  defined, 366
  in ACD, 211
  naming rule, 209, 367
  symbol, 209, 367
Information hiding, 36, 137, 185
Information modeling, 13, 398–401, 404
Input and output processing, 26, 204–207
Input-to-output response time, 99–102, 113
Interface definition, 7, 26, 203–207, 260
Iteration:
  of architecture model, 203, 286–91
  of development process, 10, 26–30
  of requirements model, 123, 139
Leveling:
  CFDs, 39, 64, 113
  DFDs, 39, 45, 47–49, 59, 113
  PSPECs, 53, 70
Logical function, 43
Maintenance processing, in architecture model,
  26–27, 204–205, 254–56
Martin, J., 194, 403
McMenamin, S., 12, 150, 403
Mealy sequential machine model, 86, 314
Mellor, S., 86, 404
Methods Team, 6
Military standards (documentation), 396–97
Miller, G., 138–39, 403
Modeling process, 286
Module, 20, 24
  automobile management computer example,
    274
  software, 282–85
  (see also Architecture flow diagram)
Module specification: see Architecture module
  specification
Moore sequential machine model, 86, 314
Myers, G., 6, 283, 403

Naming rules:
  abbreviations used, 148–49
  in AFD, 384, 386
  of processes and flows, 46, 50, 138, 140,
    147–49, 363–67
Numbering system:
  of flow diagrams, 49, 370–75
  of multi-sheet CSPECs, 95–97, 114
Objects (flows), naming, 46, 148
Operational requirements specification, 192
Page-Jones, M., 128, 283, 404
Palmer, J., 12, 150, 403
Parnas, D., 36, 137, 139, 283, 404
Partitioning, 30
  in Automobile Management System, 133–36
  modeling process, 286–91
  of architecture model for hw/sw implementa-
    tion, 235–36, 276–78, 289
  of requirements model, 113
Physical configuration, 24, 25
Physical modules, 203–10
Physical requirements, 3, 7, 43, 203–210
Primitive flow, 67–68, 103–105
  attributes of, 104–105, 196–97
  decomposing groups of, 140
  defined, 103, 194
  examples of, 104
  grouping, 140, 194
  symbol in dictionary, 106
Primitive network, 58, 112
Primitive process, 13, 39, 48, 53, 58, 59,
  111–15
  CFD and, 65
  network of, 58, 112
  PSPEC and, 13, 65
Process, 45–46
  allocation in AMS, 226–29
  controls, 73–76, 111
  data, 128
  defined, 365
  functionally identical, 150–51
  naming rule, 46, 147–49, 365
  numbering system, 49
  on CCD, 61
  on CFD, 64–65
  parallel, 56
  physical, 208
  primitive, 13, 39, 49, 53, 58–59, 65,
    111–15
  sequential, 55
  symbol for, 41
  timing requirements in, 191
Process activation table (PAT), 15, 90, 91,
  174–75, 375
  example, 91
  of vending machine example, 17, 20
Process activators, 73
  CSPEC and, 186–87
Process controls, 73–76, 90
  in leveled DFDs, 74
Processing, input and output:
  Automobile Management System, 249–51
INDEX

in architecture model, 249–51
Processing, user interface, 26–27, 251–54
Process model, 13, 40, 41–60
interpreting, 56–59
balancing, 53, 70, 159, 166
comment in, 55
conditional statement in, 125–26
continuous machines and, 77
data condition and, 69–70
data signal and, 67
eamples, 54, 129, 374
functional primitives, 53
leveling and balancing, 53, 70
naming and numbering rules, 374–75
of vending machine example, 14
role of, 158–59
structured English in, 53, 55–56, 165–66
transient signal in, 162–63
types of, 159–62
user requirements and, 55, 158
Process structure:
categories, 127
in requirements model, 113
Project dictionary:
abbreviations and, 148–49
list of word types, 166
Prototyping, 30
Putnam, L., 404
Putnam model, 6, 404
Real-time systems, 36
characteristics of, 3, 77, 98
development methods for, 11ff.
Reundancy requirement, 26, 27
data store and, 53
in AFD, 218
in AID, 219, 221–24
in Automobile Management System, 277
Reliability requirement, 222–23, 243
Requirements:
data vs. control, 125–29, 152
defined, 10, 12
functional, 3, 99
physical, 3, 24
processing, in PSPECs, 158–68
timing, 39–40, 98–102, 113–15, 190–93, 232–33, 244
Requirements dictionary (RD), 13, 19, 35, 40, 103–10, 377–79
attributes of, 194–97
cockpit display example, 105
data flow types in, 194–96
flow decomposition in, 381
flows in, 67, 103, 115
in a computerized data base, 107–108, 194
indentured explosion of, 107, 109
naming rules, 379
of Automobile Management System, 316–18
of vending machine example, 15, 18, 354
primitives in, 103–107
symbols in, 105–107, 198–99, 378
timing signals in, 190–93
Requirements model, 4, 7, 12, 13–19, 25–27, 35–40, 111–15
adding physical constraints, 248–62
allocation of components in Automobile Management System, 226–29
builders, 120–23
components of, 37–40, 113–15, 367–81
consistency checking of, 36, 49, 50, 52, 113, 115
decomposition and, 45, 122–23, 137–38
enhanced, 248–62
implementation details and, 152
leveling and balancing, 47–49, 113, 379–81
naming and numbering in, 47ff., 113
of Home Heating System, 330–39
of vending machine example, 344–54
overview of, 18–19, 35–40, 113–15
primitives in, 113
redundancy and, 26–27, 53
role of automata theory, 89
self-indexing, 5, 36, 115
signals in, 162–63
timing in, 190–93
tools of, 35
users, 119–21
Response time specification, 13, 19, 99–102, 115
(see also Timing specification)
Schuldt, G., 13, 404
Self-test processing, 26–27, 254–55
Sequential control specification, 90, 92, 93
Sequential machine, 78, 83–89, 95, 97, 114, 152, 157, 176–82, 375
combination lock example, 83–84
CSPEC for, 378, 379
math representation, 178
Mealy and Moore models, 86, 178–79
Seven-plus-or-minus-two principle, 138–39, 177, 214, 370, 403

Signals:
  attributes, 196–97
  continuous vs. transient, 162–63, 165
  control vs. data, 67–68, 70, 76, 111–12
  discrete vs. continuous, 67–68, 77–78, 103–104, 165
  external primitive, 99, 104
types, 127, 196

Sink, 43, 44

Software architecture, 277, 282–85

Source, 43, 44

Specification model, system, 24, 25–27

STARS Methodology Conference, 295

State, 85–89, 97, 177
  naming, 189

State transition diagram (STD), 15, 65, 73, 83–86, 90–97, 177–78, 375, 379
  of Automobile Management System, 182, 303
  of combination lock example, 84
  of vending machine example, 17, 20

State transition matrix (STM), 73, 87–89, 93, 97, 114, 177–78
  examples, 88, 89
  multi-sheet CSPECs and, 182–85

State transition table (STT), 73, 87, 93, 97, 114, 177, 184, 375
  example, 87

Store: see Control store, Data store

Structure chart, 282

Structured analysis, 5, 35–37, 43, 52, 53, 95
  control requirements and, 188
  leveling and balancing principles of, 93
  primitive network and, 113
  process model and, 40

Structured design, 6, 36, 282–84

Structured documentation, 392–97

Structured English, 53, 55–56, 59
  examples, 57
  in PSPECs, 159, 165–66
  structure of, 57

Structured methods:
  defined, 11ff.
  history of, 4–8, 36–37
  iteration and, 27

Structured programming, 36–37

Swartout, W., 139, 404

System analysts, role of, 120–23

System context, 44, 122, 129–32

System design:
  building the architecture model, 243–45
  factors, 98–99, 243–45

System life cycle, 8–10
  cost, 31
  design, 56
  implementation phase, 8, 32, 56, 261
  requirements definition phase, 56, 120

System maintenance, 26, 29
  cost, 10
  phase, 261

System model, 287–88

Systems:
  control-intensive, 153–54
  hierarchy of, 4, 10, 28–29, 246–47

System scope, 44, 122, 129–32

Systems development process:
  illustration of, 26
  iteration and, 10, 26, 30, 139
  life cycle of, 27–31

System specification:
  hierarchical nature, 3, 28–29
  leveled, 4, 28–29, 39
  methods of, 3ff.
  model, 24, 25–27
  (see also Architecture model, Requirements model)

Tabular PSPEC, 15, 161
  (see also Process activation table, State transition diagram)

Technology-dependent model, 12, 26, 244, 257–60
  (see also Architecture model)

Technology-nonspecific model, 248–62
  (see also Requirements model)

Terminator, 368
  defined, 365
  naming, 366
  on ACD, 211
  on CCD, 61, 63
  on DCD, 41, 43–44
  symbol, 41

Testing requirements, 9, 26, 29, 31
  phase, 261

Timing diagram, 101, 115

Timing requirements, 39–40, 98–102, 113–15
  190–93
  in architecture model, 229, 232–33, 244
  in CSPEC, 164, 188, 191
  in PSPEC, 164
  in requirements model, 39–40, 98–102, 113–15
  repetition rate, 99, 102, 114–15, 188, 190–93
response time, 99–102, 114–15, 190–93
balancing, 193
of Automobile Management System, 304
of vending machine example, 347
(see also Response time specification)
Traceability matrix, 226, 229–30, 388
of Automobile Management System, 230
of vending machine example, 358
Transaction center, 46
control, 75, 175–76
Transient signal, 162–63
Transition arcs, 85–86, 178
Truth table: see Decision table

Ullman, J., 86, 403
Understandability, 5, 123, 149, 156, 159, 177, 214
User interface:
in Automobile Management System, 251–54
processing, 26–27, 251–54
specification of, 260
User requirements, 37, 98, 119–23, 124–25
PSPEC and, 55, 158
Users’ guide, to CSPECs, 95, 114, 153, 185
Vending machine example, 12–23, 342–59
Ward and Mellor convention, 86, 404
Ward, P., 86, 404
Yourdon, E., 6, 283, 404