The Influence of Software Reuse on Programming Language Design

DISSERTATION

Presented in Partial Fulfillment of the Requirements for
the Degree Doctor of Philosophy in the Graduate
School of the Ohio State University

By

Douglas Eugene Harms, B.S., M.S.

* * * * *

The Ohio State University
1990

Dissertation Committee:  Approved by

Dr. Bruce W. Weide

Dr. Neelam Soundararajan  Adviser

Dr. Timothy J. Long

Department of
Computer and Information Science
To my parents, Clarence and Mary Ann Harms,

my wife, Mary Beth,

and daughters, Rebecca Elizabeth, Gretchen Marie,

and Alisha Anne
ACKNOWLEDGEMENTS

I express sincere appreciation to my adviser, Bruce Weide, for his patience and guidance during this research, and for his confidence. I also thank the other members of my committee, Timothy Long and Neelam Soundararajan, for finding time in their busy schedules to make constructive suggestions for both the research and dissertation organization.

I am grateful to the members of the Reusable Software Research Group at Ohio State, especially Bill Ogden, Stu Zweben, Mike Stovsky, Murali, Joan Krone, and Joe Hollingsworth. This research has been positively influenced and directed by the ideas and suggestions expressed during our many discussions.

I am indebted to the Muskingum College community for supporting me in both financial and moral senses during the years it has taken me to achieve this goal. I especially thank Dan Van Tassel, Sam Speck, Jim Smith, Ralph Hollingsworth, Ray Rataiczak, and Art DeJong.

I am also thankful for the encouragement and support provided by the members of the United Methodist Church in New Concord and St. Andrew Presbyterian Church in Columbus.

Finally, and most importantly, I am grateful to my wife, Mary Beth, and children, Rebecca, Gretchen, and Alisha, who made this adventure possible with their understanding and constant encouragement.
VITA

June 10, 1957 .................................................. Born — Hillsboro, Kansas

1979 ............................................................. Bachelor of Science
Muskingum College
New Concord, Ohio

1979-1981 ...................................................... System Programmer
NCR Corporation
Cambridge, Ohio

1981-1983 ...................................................... Lecturer
Muskingum College
New Concord, Ohio

1983 ............................................................. Master of Science
The Ohio State University
Columbus, Ohio

1983-present ................................................... Assistant Professor
Muskingum College
New Concord, Ohio

PUBLICATIONS

Types, Copying, and Swapping: Their Influences on the Design of Reusable Software Components, with B.W. Weide, Department of Computer and Information Science, The Ohio State University, Columbus, OH, March 1989, OSU-CISRC-3/89-TR13.

Efficient Initialization and Finalization of Data Structures: Why and How, with B.W. Weide, Department of Computer and Information Science, The Ohio State University, Columbus, OH, March 1989, OSU-CISRC-3/89-TR11.


Swapping — A Desirable Alternative to Copying, with B.W. Weide, Department of Computer and Information Science, The Ohio State University, Columbus, OH, January 1988, OSU-CISRC-1/88-TR2.

**FIELDS OF STUDY**

Major Field: Computer and Information Science — Software Engineering
# TABLE OF CONTENTS

ACKNOWLEDGEMENTS ................................................................. iii
VITA .......................................................................................... iv
LIST OF FIGURES ........................................................................ x
CHAPTER I Introduction ............................................................... 1
CHAPTER II Definitions and Framework .................................. 5
  2.1 Definitions ........................................................................ 5
  2.2 Reusability Issues ............................................................. 7
    2.2.1 What Is A “Reusable Software Component?” .............. 7
    2.2.2 Advantages of Software Reuse ................................. 8
    2.2.3 Non-Technical Impediments to Software Reuse .......... 9
    2.2.4 Technical Impediments to Software Reuse ............... 10
  2.3 Characteristics of Reusable Parts ................................... 12
    2.3.1 Formal Specification ............................................... 12
    2.3.2 Separation of Specification from Implementation ....... 15
    2.3.3 Generics ................................................................. 16
    2.3.4 Multiple Implementations ....................................... 16
    2.3.5 Efficient Implementations Possible ........................... 18
  2.4 Reusable Part Design Issues ............................................ 20
    2.4.1 Encapsulation .......................................................... 20
      2.4.1.1 Abstract Data Objects ....................................... 21
      2.4.1.2 Abstract Data Types .......................................... 24
      2.4.1.3 Primary and Secondary Operations .................. 26
    2.4.2 Specification .......................................................... 26
      2.4.2.1 Mathematical Theories ..................................... 27
      2.4.2.2 Algebraic Specification ................................. 29
      2.4.2.3 Model-Based Specification ........................... 31
  2.5 Programming Language Design Issues ............................ 32
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.7.1 Realization Copy(Stack)_Real_1 of Copy(Stack)_Template</td>
<td>142</td>
</tr>
<tr>
<td>3.7.2 Realization List_Real_1 of One_Way_List_Template</td>
<td>144</td>
</tr>
<tr>
<td>3.7.3 Other Realization Sections</td>
<td>151</td>
</tr>
<tr>
<td>3.7.4 Summary</td>
<td>152</td>
</tr>
<tr>
<td>3.8 Implementation Issues</td>
<td>152</td>
</tr>
<tr>
<td>3.8.1 Primitive Realizations for Conceptualizations</td>
<td>152</td>
</tr>
<tr>
<td>3.8.2 Lazy Initialization of Variables</td>
<td>153</td>
</tr>
<tr>
<td>3.8.3 Efficient Implementation of Generic Modules</td>
<td>154</td>
</tr>
<tr>
<td>3.9 Summary</td>
<td>154</td>
</tr>
<tr>
<td>CHAPTER IV Interaction of Programming Language and Environment Design</td>
<td>157</td>
</tr>
<tr>
<td>4.1 Programming Language Influence on Environmental Design</td>
<td>157</td>
</tr>
<tr>
<td>4.2 Environmental Influence on Programming Language Design</td>
<td>159</td>
</tr>
<tr>
<td>4.3 Summary</td>
<td>163</td>
</tr>
<tr>
<td>CHAPTER V Conclusion</td>
<td>164</td>
</tr>
<tr>
<td>5.1 Summary and Conclusions</td>
<td>164</td>
</tr>
<tr>
<td>5.1.1 Software Reusability</td>
<td>164</td>
</tr>
<tr>
<td>5.1.2 RESOLVE Programming Language</td>
<td>166</td>
</tr>
<tr>
<td>5.1.3 Interaction of Programming Language and Environment Design</td>
<td>169</td>
</tr>
<tr>
<td>5.2 Future Work</td>
<td>170</td>
</tr>
<tr>
<td>5.2.1 Enhancements</td>
<td>170</td>
</tr>
<tr>
<td>5.2.2 Accessors</td>
<td>172</td>
</tr>
<tr>
<td>5.2.3 Synoptic Comments</td>
<td>172</td>
</tr>
<tr>
<td>5.3 Contributions</td>
<td>173</td>
</tr>
<tr>
<td>APPENDIX A An Editing Environment for RESOLVE</td>
<td>175</td>
</tr>
<tr>
<td>A.1 &lt;VAR NAME&gt; Placeholders and Variables</td>
<td>177</td>
</tr>
<tr>
<td>A.2 Inserting Statements and Control Invocations</td>
<td>179</td>
</tr>
<tr>
<td>A.3 Variable Declaration</td>
<td>183</td>
</tr>
<tr>
<td>A.4 Inserting Function Invocations</td>
<td>186</td>
</tr>
<tr>
<td>A.5 If and Return Statements</td>
<td>186</td>
</tr>
<tr>
<td>A.6 Editing Assertions</td>
<td>187</td>
</tr>
<tr>
<td>A.7 Creating Conceptualizations, Types, and Operations</td>
<td>188</td>
</tr>
<tr>
<td>A.8 Creating Realizations</td>
<td>191</td>
</tr>
<tr>
<td>A.9 Selection and Deletion</td>
<td>192</td>
</tr>
<tr>
<td>BIBLIOGRAPHY</td>
<td>195</td>
</tr>
</tbody>
</table>
LIST OF FIGURES

Figure 1  Informal Description of a Template Providing Type Stack............... 19
Figure 2  Specification for a Module Providing the Generic Type Stack ............ 77
Figure 3  Specification of Function Top .................................................. 82
Figure 4  Specification for a Module Providing the Generic Type List ............... 87
Figure 5  Realization Stack_Real_1 of Stack_Template Using One_Way_List_Template ......................................................... 91
Figure 6  Abstract Effect of Swap Statement “x := y” .................................. 100
Figure 7  Implementation of Swap Statement “x := y” .................................. 101
Figure 8  Definition and Invocation of Sample Procedure ................................ 102
Figure 9  Effect of Sample Procedure Invocation ........................................ 103
Figure 10 Specification of Function Replica for Type T ................................. 105
Figure 11 Relationship Between types, Markers, and Domains ................................ 110
Figure 12 Example Type Declarations .......................................................... 112
Figure 13 Type Mappings for Example Type Declarations .............................. 113
Figure 14 Type Mappings for Realization Stack_Real_1 .................................. 116
Figure 15 Specification for a Module Providing Generic Type Bounded_Stack ....... 122
Figure 16 Bounded_Stack_Template Client .................................................... 124
Figure 17 Program Type Mappings for Bounded_Stack_Template Client ............ 125
Figure 18 Specification for a Module Providing Procedure Copy_Stack ................ 126
Figure 19 Copy_Stack_Template Client .......................................................... 127
Figure 20 Specification for a Module Providing Generic Type Array .................. 128
Figure 21 Specification for a Module Providing Type Int .................................. 132
Figure 22 Specification for a Module Providing Generic Type Reference ............ 135
Figure 23 Specification for a Module Providing an “Object-Oriented” Stack ....... 140
Figure 24 Realization Copy_Stack_Real_1 of Copy_Stack_Template .................. 142
Figure 25 Realization List_Real_1 of One_Way_List_Template .......................... 144
Figure 26 Effect of Alternate Syntax .............................................................. 161