**Computer Program Verification: Improvements for Human Reasoning**

By

Wayne David Heym, Ph.D.

The Ohio State University, 1995

Bruce W. Weide, Adviser

To ably create or modify computer programs that behave according to specification, programmers find it necessary to reason about their programs’ behavior. We have formalized, in a direct, natural way, the informal pattern of reasoning generally used with programs written in modular, imperative languages. This formal system provides a solid basis against which to check the soundness and (relative) completeness of an informal reasoning method.

Formal proof that a program meets a specification can be done in this new system or in existing systems (e.g., calculating weakest preconditions using Hoare-style axioms or using symbolic execution). Each system prescribes a way to translate the program-specification pair to a mathematical assertion whose truth implies that the program satisfies the specification. Alternative systems are distinguished, however, by how well they fit programmers’ informal reasoning methods. Programmers think of the effect that the execution of a given statement will have on variables’ values, and they consider what conditions must hold for those values in each branch of the program. The new method is organized accordingly, unlike previous methods, which are organized for the convenience of mathematicians.
COMPUTER PROGRAM VERIFICATION:
IMPROVEMENTS FOR HUMAN REASONING

DISSERTATION

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

By

Wayne David Heym, B.Phil., M.S., M.S.

* * * * *

The Ohio State University

1995

Dissertation Committee:                              Approved by
Bruce W. Weide                                      Adviser
William F. Ogden
Stuart H. Zweben

Department of Computer and Information Science
For my loving wife, Kimberly Wells Heym
ACKNOWLEDGMENTS

My advisor, Bruce W. Weide, has guided and encouraged me through all the stages of graduate study, ever since my first visit to the campus. I am grateful for his wisdom in suggesting the topic for this dissertation. The number of helpful suggestions and ideas Bruce has offered in our regular meetings together is beyond counting, and his steadfast encouragement has helped me to persevere. I am thankful for William F. Ogden and his insightful, skeptical approach to my work. More than anyone else, he has guided the form of the proof rules, the explanation of the semantics, and the presentation of the proofs of soundness and relative completeness. Bill’s hearty and healthy skepticism led the way to identifying the crucial ideas in the soundness proofs. Stuart H. Zweben encouraged me to work with him on testing and was instrumental in helping me learn most of what I know about the subject. He persevered in getting me to properly defend, with reference to the literature, the claim that my proposed new method, the indexed method of correctness proof, is more natural than the existing method. Stu’s careful reading and corrections of early drafts were invaluable, and he completed his readings with amazing, and much appreciated, speed. These three professors regularly meet with a group of students interested in the effective production of quality software from the standpoint of reusability—the Reusable Software Research Group (RSRG) at The Ohio State University. The participants in RSRG have helped me learn many of the important issues in software engineering. This thesis has benefited greatly from the RSRG’s intelligently dedicated effort to learn how to do software right.

We gratefully acknowledge the support of the National Science Foundation through grants CCR-9111892 and CCR-9311702; the Advanced Research Projects Agency under ARPA contract number F30602-93-C-0243, monitored by the USAF Material Command, Rome Laboratories, ARPA order number A714; and the Army Research Office through grant DAAH04-95-1-0457.
VITA

July 3, 1956 ........................................... Born - Cleveland, Ohio

1978 .................................................. B.Phil. Interdisciplinary Studies, Miami University

1980 .................................................. M.S. Operations Research, Cornell University

1980-1983 .......................................... Programmer/Analyst I, Eastman Kodak Company

1984-1988 .......................................... Research Support Specialist, Cornell University

1988-1989 .......................................... University Fellow, The Ohio State University

1989 .................................................. M.S. Computer and Information Science, The Ohio State University

1989-1991 .......................................... Graduate Teaching Associate, The Ohio State University

1991-1995 .......................................... Graduate Research Associate, The Ohio State University

1994-1995 .......................................... Instructor of Computer Science, Otterbein College

Publications

Research Publications


**Fields of Study**

Major Field: Computer and Information Science

Studies in:

- Software Engineering
  - Prof. Bruce W. Weide
- Programming Languages
  - Prof. Wolfgang W. Kuechlin
- Theory of Computation
  - Prof. Timothy J. Long
# Table of Contents

DEDICATION ................................................................. iii

ACKNOWLEDGMENTS ....................................................... v

VITA ........................................................................ vii

LIST OF TABLES .......................................................... xiii

LIST OF FIGURES .......................................................... xv

CHAPTER | PAGE
---|---
I  Introduction ......................................................... 1
   1.1 How People Want to Reason About Program Behavior ....... 3
   1.2 Formal Bases for Reasoning ..................................... 5
   1.3 The Problem ....................................................... 8
   1.4 The Thesis ......................................................... 9
   1.5 Traditional Formal Reasoning Is Not Natural .............. 9
      1.5.1 An Example of Traditional Reasoning .................. 11
      1.5.2 An Example of More Natural Reasoning ............... 13
      1.5.3 Changing the Postcondition ............................. 24
      1.5.4 Conclusions ................................................. 26
   1.6 Importance of Proving Soundness and Completeness ....... 27
   1.7 Outline of Dissertation ....................................... 30

II  Syntax and Semantics ................................................ 31
   2.1 Syntax ............................................................ 32
      2.1.1 Aspects of the Syntax That Are Context-Free ....... 32
      2.1.2 Aspects of the Syntax That Are Not Context-Free .... 38
### 2.2 Semantics ................................. 41
  2.2.1 Semantic Space ............................. 43
  2.2.2 Semantic Definition ....................... 48
  2.2.3 Validity .................................. 57

### III Proof Rules .............................. 61
  3.1 Assertive Program Language Subsets ............. 63
  3.2 How the Rules are Defined ..................... 68
  3.3 The Context Attribute .......................... 70
  3.4 The Bridge Rule ................................ 72
  3.5 The Rule for assume ............................. 78
  3.6 The Rule for Procedure Call ...................... 80
  3.7 Rules for Selection ............................. 83
  3.8 The loop while Rule .............................. 89
  3.9 The Rule for confirm ............................. 92
  3.10 The Rule for Empty Guarded Blocks .............. 95
  3.11 The Rule for alter all ........................... 95
  3.12 The Rule for Consecutive assume Statements .... 98
  3.13 The assume-confirm Rule ......................... 103
  3.14 The Rule for Consecutive confirm Statements ... 103
  3.15 The Rule of Inference Bridging Predicate Logic and the Indexed Method .................. 106
  3.16 Example Application of the loop while Rule .... 107
  3.17 Summary .................................... 110

### IV Soundness and Relative Completeness .......... 111
  4.1 Soundness .................................... 111
  4.2 Relative Completeness ............................ 112
  4.3 General Auxiliary Lemmas ........................ 119
    4.3.1 Proof Rules Are Well-Formed ................. 119
    4.3.2 Factoring Statement Sequences ................ 123
    4.3.3 Shallow Lemmas .............................. 123
    4.3.4 Deeper Lemmas ............................... 125
    4.3.5 Negative-Branch-Condition Independence .... 138
    4.3.6 Internal-Index Independence ................. 153
  4.4 Soundness Lemmas ............................... 154
  4.5 Relative Completeness Lemmas .................... 178
4.5.1 Related Valid Program Differing in Assertions Only . . . . . 178
4.5.2 System of Rewrite Rules Is Terminating . . . . . . . . 178
4.5.3 Preservation of Invalidity in the Program Direction . . . . . 181
4.5.4 The loop while Rule . . . . . . . . . . . . . . . . . . . . . . . 191
4.5.5 The Procedure Call Rule . . . . . . . . . . . . . . . . . . . . 193
4.6 Relaxing a Simplifying Assumption . . . . . . . . . . . . 194
4.7 Summary . . . . . . . . . . . . . . . . . . . . . . . . . . . 196

V Conclusion . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 197

5.1 Informal and Formal Indexed Methods . . . . . . . . . . 197
5.2 Relationships Among Methods . . . . . . . . . . . . . . 198
5.3 Opportunities for Future Work . . . . . . . . . . . . . . 200
  5.3.1 Miscellaneous Issues . . . . . . . . . . . . . . . . . . 200
  5.3.2 The “loop exit when” Rule . . . . . . . . . . . . . . 201
  5.3.3 Investigating the Worth of the Indexed Method . . . 204
5.4 Contributions . . . . . . . . . . . . . . . . . . . . . . . . . 205

BIBLIOGRAPHY . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 207


**List of Tables**

<table>
<thead>
<tr>
<th>TABLE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Nonterminal Symbols Whose Definitions Are Assumed</td>
<td>33</td>
</tr>
<tr>
<td>2 Multipliers for Function Meas</td>
<td>179</td>
</tr>
</tbody>
</table>
# List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Human Reasoning and Formal Bases</td>
<td>8</td>
</tr>
<tr>
<td>2</td>
<td>A Specification of Procedure Set_Maximum</td>
<td>10</td>
</tr>
<tr>
<td>3</td>
<td>An Implementation for Procedure Set_Maximum</td>
<td>11</td>
</tr>
<tr>
<td>4</td>
<td>Traditional Back Substitution Method: First Step</td>
<td>12</td>
</tr>
<tr>
<td>5</td>
<td>Traditional Back Substitution Method: Second Step</td>
<td>12</td>
</tr>
<tr>
<td>6</td>
<td>Traditional Back Substitution Method: Third Step</td>
<td>13</td>
</tr>
<tr>
<td>7</td>
<td>Traditional Back Substitution Method: Fourth Step</td>
<td>14</td>
</tr>
<tr>
<td>8</td>
<td>Traditional Back Substitution Method: Final Assertion</td>
<td>14</td>
</tr>
<tr>
<td>9</td>
<td>Indexed Method: Mark Between-Statement Spaces</td>
<td>15</td>
</tr>
<tr>
<td>10</td>
<td>Indexed Method: Mark Each Branch Condition</td>
<td>16</td>
</tr>
<tr>
<td>11</td>
<td>Indexed Method: Write Obligation at Last Position</td>
<td>17</td>
</tr>
<tr>
<td>12</td>
<td>Indexed Method: Replacing an Assignment Statement with a Fact</td>
<td>18</td>
</tr>
<tr>
<td>13</td>
<td>Indexed Method: Replacing the Second if-then Statement with Two Facts</td>
<td>19</td>
</tr>
</tbody>
</table>
14 Indexed Method: Replacing Statements Inside Branch Conditions with Facts ........................................... 20
15 Indexed Method: Replacing the First if-then Statement with Two Facts 21
16 Indexed Method: Sequence of Facts and Obligations ................. 22
17 Indexed Method: Final Assertion ................................. 23
18 Another Correct Implementation for Procedure Set_Maximum .... 25
19 An Improved Specification: Procedure Useful_Set_Max ............. 25
20 Evidence of Unsoundness: First Step ............................. 28
21 A Specification of Procedure Stay_two .......................... 29
22 Evidence of Unsoundness: Second Step .......................... 29
23 Evidence of Unsoundness: Third Step ............................ 29
24 Context-free Grammar of Assertive Programs ........................ 39
25 Specification of Procedure Set_State_by_Addition .................. 40
26 Specification of Procedure Add .................................... 41
27 Definition of Domains in the Semantic Space ....................... 45
28 Six Projection Functions on Environments .......................... 48
29 Notation for Environment Named “env” ............................ 49
30 Nested Loops and Old Variable Names ............................ 56
31 Transforming Programs to Mathematical Assertions in Phases ....... 62
32 Context-free Grammar of Subsets of Assertive Programs ........ 66

xvi
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Grammar Productions Restricted for <code>&lt;p_body&gt;</code></td>
<td>67</td>
</tr>
<tr>
<td>Grammar Productions Restricted for <code>&lt;in_code&gt;</code>, <code>&lt;cd_prefix&gt;</code>, <code>&lt;cd_suffix&gt;</code>, and <code>&lt;cd_kern&gt;</code> Inside Selection or Iteration, But Not Part of Procedure Body</td>
<td>67</td>
</tr>
<tr>
<td>Grammar Productions Restricted for <code>&lt;in_code&gt;</code>, <code>&lt;cd_prefix&gt;</code>, <code>&lt;cd_suffix&gt;</code>, and <code>&lt;cd_kern&gt;</code> Outside Selection and Iteration, and Not Part of Procedure Body</td>
<td>67</td>
</tr>
<tr>
<td>Grammar Productions Restricted for <code>&lt;top_lev_code&gt;</code></td>
<td>68</td>
</tr>
<tr>
<td>Example Subgoal</td>
<td>70</td>
</tr>
<tr>
<td>Example Context</td>
<td>71</td>
</tr>
<tr>
<td>Application of Krone’s Procedure Declaration Rule</td>
<td>71</td>
</tr>
<tr>
<td>Equations Defining the Bridge Rule</td>
<td>73</td>
</tr>
<tr>
<td>Grammatic Derivation of Body of Change_X</td>
<td>74</td>
</tr>
<tr>
<td>Grammatic Derivation of Stows_Added(Body of Change_X)</td>
<td>75</td>
</tr>
<tr>
<td>Application of the Bridge Rule: Inst(\mathcal{M})</td>
<td>76</td>
</tr>
<tr>
<td>Equations Defining the Rule for <code>assume</code></td>
<td>78</td>
</tr>
<tr>
<td>First Application of Rule for <code>assume</code></td>
<td>79</td>
</tr>
<tr>
<td>Equations and Additional Syntactic Restriction Defining the Rule for Procedure Call</td>
<td>81</td>
</tr>
<tr>
<td>First Application of Procedure Call Rule</td>
<td>82</td>
</tr>
<tr>
<td>Equations Defining the Rule for Selection in the Absence of an <code>else</code> Clause</td>
<td>84</td>
</tr>
</tbody>
</table>

xvii
Definition of $\mathcal{P}$ for the Rule for Selection in the Presence of an else Clause ........................................ 85

Definition of $\mathcal{M}$ for the Rule for Selection in the Presence of an else Clause .................................................. 86

Second Application of Rule for assume ........................................... 87

Application of Rule for Selection in the Presence of an else Clause . 88

Definition of $\mathcal{P}$ for the loop while Rule ................................. 90

Definition of $\mathcal{M}$ for the loop while Rule ................................. 91

Equations Defining the Rule for confirm ........................................ 93

Application of Rule for confirm ................................................... 94

Third Application of Rule for assume ............................................. 96

Second Application of Procedure Call Rule ................................. 97

Equations Defining the Rule for Empty Guarded Blocks ............... 98

First Application of Rule for Empty Guarded Blocks ................... 99

Application of Three Different Rules ......................................... 100

Equations and Additional Syntactic Restriction Defining the Rule for alter all ......................................................... 101

Seven Applications of the Rule for alter all .............................. 101

Equations Defining the Rule for Consecutive assume Statements ... 102

Four Applications of the Rule for Consecutive assume Statements ... 102
<table>
<thead>
<tr>
<th>Page</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>66</td>
<td>Equations Defining the <strong>assume-confirm</strong> Rule .......... 103</td>
</tr>
<tr>
<td>67</td>
<td>An Application of the <strong>assume-confirm</strong> Rule .......... 104</td>
</tr>
<tr>
<td>68</td>
<td>Equations Defining the Rule for Consecutive <strong>confirm</strong> Statements .. 104</td>
</tr>
<tr>
<td>69</td>
<td>An Application of the Rule for Consecutive <strong>confirm</strong> Statements .. 105</td>
</tr>
<tr>
<td>70</td>
<td>Five Rule Applications ........................................ 106</td>
</tr>
<tr>
<td>71</td>
<td>Mathematical Assertion in Context $C'$ .......................... 107</td>
</tr>
<tr>
<td>72</td>
<td>Example: Iteration ................................................. 108</td>
</tr>
<tr>
<td>73</td>
<td>Application of the <strong>loop while</strong> Rule ......................... 109</td>
</tr>
<tr>
<td>74</td>
<td>A Valid Assertive Program ........................................ 113</td>
</tr>
<tr>
<td>75</td>
<td>An Invalid Assertive Program ..................................... 115</td>
</tr>
<tr>
<td>76</td>
<td>An Assertive Program That Is Valid According to Functional Semantics ........................................ 117</td>
</tr>
<tr>
<td>77</td>
<td>Environments Defined for $\mathcal{P}$ of the Rule for Selection in the Presence of an <strong>else</strong> Clause ....................... 156</td>
</tr>
<tr>
<td>78</td>
<td>Environments Defined for $\mathcal{M}$ of the Rule for Selection in the Presence of an <strong>else</strong> Clause ....................... 157</td>
</tr>
<tr>
<td>79</td>
<td>Environments Defined for $\mathcal{P}$ of the Rule for Selection in the Absence of an <strong>else</strong> Clause ....................... 161</td>
</tr>
<tr>
<td>80</td>
<td>Environments Defined for $\mathcal{M}$ of the Rule for Selection in the Absence of an <strong>else</strong> Clause ....................... 162</td>
</tr>
<tr>
<td>81</td>
<td>Environments Defined for $\mathcal{P}$ of the <strong>loop while</strong> Rule .......................... 164</td>
</tr>
<tr>
<td>82</td>
<td>Environments Defined for $\mathcal{M}$ of the <strong>loop while</strong> Rule .......................... 165</td>
</tr>
<tr>
<td>Page</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td>83</td>
<td>Environments Defined for $\mathcal{P}$ and $\mathcal{M}$ of the Rule for Procedure Call</td>
</tr>
<tr>
<td>84</td>
<td>The Rule for a Procedure Call That Is Not the First Statement Within a $\textit{whenever}$ Statement</td>
</tr>
<tr>
<td>85</td>
<td>Action Directions Differ Among the Three Methods</td>
</tr>
<tr>
<td>86</td>
<td>Redefinition of $\langle \text{iter} \rangle$</td>
</tr>
<tr>
<td>87</td>
<td>Definition of $\mathcal{P}$ for a Proof Rule That Would Handle the “$\textit{loop exit when}$” Statement Having Exactly Two $\textit{exit when}$ Constructs</td>
</tr>
<tr>
<td>88</td>
<td>Definition of $\mathcal{M}$ for a Proof Rule That Would Handle the “$\textit{loop exit when}$” Statement Having Exactly Two $\textit{exit when}$ Constructs</td>
</tr>
</tbody>
</table>