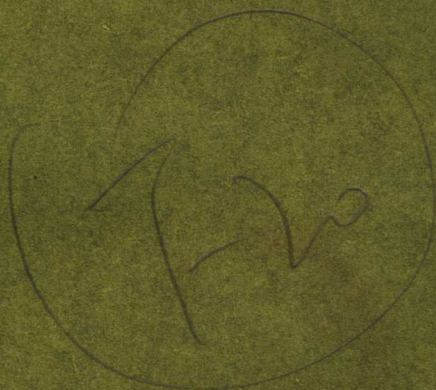


**Fourth International Workshop On  
Computer-Aided Software Engineering**



7  
**Fourth International Workshop on  
Computer-Aided Software  
Engineering**

**ADVANCE WORKING PAPERS**

Irvine, California  
December 5-8, 1990

***Editors:***

Ronald J. Norman  
Roger Van Ghent

***Co-Sponsors:***

International Workshop on CASE, Inc.  
IEEE Computer Society



**IEEE Computer Society Press**  
Los Alamitos, California

Washington • Brussels • Tokyo

---

The papers in this book comprise the proceedings of the meeting mentioned on the cover and title page. They reflect the authors' opinions and are published as presented and without change, in the interests of timely dissemination. Their inclusion in this publication does not necessarily constitute endorsement by the editors, the IEEE Computer Society Press, or The Institute of Electrical and Electronics Engineers, Inc.

Published by



IEEE Computer Society Press  
10662 Los Vaqueros Circle  
P.O. Box 3014  
Los Alamitos, CA 90720-1264

Copyright 1990. International Workshop on CASE Inc.

Copying without fee is permitted provided that copies are not made or distributed for direct commercial advantage and credit to the source is given. To copy papers contained herein which have copyright notices on them, contact the copyright owner as indicated by that copyright notice.

The Workshop logo is a trademark of the Int'l. Workshop on CASE Inc. All rights reserved.

Cover and Workshop logo designs by Ann Casady, Gill Fishman Associates.

IEEE Computer Society Press Order Number 2129  
Library of Congress Number 90-85552  
IEEE Catalog Number 90CH2956-1  
ISBN 0-8186-2129-X (paper)  
ISBN 0-8186-6129-1 (microfiche)  
ISBN 0-8186-9129-8 (case)  
SAN 264-620X

Additional copies can be ordered from:

IEEE Computer Society Press  
Customer Service Center  
10662 Los Vaqueros Circle  
P.O. Box 3014  
Los Alamitos, CA 90720-1264

IEEE Computer Society  
13, Avenue de l'Aquilon  
B-1200 Brussels  
BELGIUM

IEEE Computer Society  
Ooshima Building  
2-19-1 Minami-Aoyama,  
Minato-Ku  
Tokyo 107, JAPAN

IEEE Service Center  
445 Hoes Lane  
P.O. Box 1331  
Piscataway, NJ 08855-1331



THE INSTITUTE OF ELECTRICAL AND  
ELECTRONICS ENGINEERS, INC.

# TABLE OF CONTENTS

## EXISTING SYSTEMS

### CASE '89 Results

|                                                          |   |
|----------------------------------------------------------|---|
| 1989 Maintenance and Reverse Engineering Track . . . . . | 2 |
|----------------------------------------------------------|---|

### Maintenance & Management

|                                                            |    |
|------------------------------------------------------------|----|
| Design Maintenance Systems . . . . .                       | 10 |
| <i>I.D. Baxter</i>                                         |    |
| Towards Integration of CASE Back-end Tools . . . . .       | 14 |
| <i>S. Jarzabek</i>                                         |    |
| CASE Requirements for Enhancing Existing Systems . . . . . | 16 |
| <i>R.G. Mays</i>                                           |    |

### Reverse Engineering & Design Recovery

|                                                                                             |    |
|---------------------------------------------------------------------------------------------|----|
| We Need Code Processing or Should CASE Care About Maintenance? . . . . .                    | 20 |
| <i>A. von Mayrhauser</i>                                                                    |    |
| ESPRIT Project REDO . . . . .                                                               | 22 |
| <i>P.F. Katsoulakos</i>                                                                     |    |
| Lessons Learned During the Development of Re-engineering Tools . . . . .                    | 24 |
| <i>W. Kozaczynski</i>                                                                       |    |
| Re-engineering of Legacy Systems Using an Object-Oriented Technology . . . . .              | 26 |
| <i>F. Lindström</i>                                                                         |    |
| User Involvement in Software Re-engineering . . . . .                                       | 28 |
| <i>J.Q. Ning</i>                                                                            |    |
| Maintaining and Re-engineering Old Software Systems . . . . .                               | 30 |
| <i>W. Scacchi</i>                                                                           |    |
| Automated Reverse Engineering and Integrating with Forward Engineering CASE Tools . . . . . | 32 |
| <i>T.L. Scott</i>                                                                           |    |
| The State of the Practice of CASE Tools Supporting Ada Reverse Engineering . . . . .        | 33 |
| <i>M.C. Smith, D.E. Mularz, and T.J. Smith</i>                                              |    |
| Re-engineering: A Requirement for an Integrated Migration Framework . . . . .               | 35 |
| <i>W.M. Ulrich</i>                                                                          |    |
| Integrating Design Recovery into CASE . . . . .                                             | 37 |
| <i>T.W. Usowicz</i>                                                                         |    |
| Tools for Reverse Engineering Existing Systems . . . . .                                    | 39 |
| <i>L. Zucconi</i>                                                                           |    |

### Salvaging Software for Reuse

|                                                            |    |
|------------------------------------------------------------|----|
| A CASE Tool Supporting Software Reuse: The PRESS . . . . . | 42 |
| <i>C. Boldyreff</i>                                        |    |

|                                                                                             |    |
|---------------------------------------------------------------------------------------------|----|
| Supporting the Software Reuse and Maintenance Practitioner: Requirements for CASE . . . . . | 45 |
| <i>V. Karakostas</i>                                                                        |    |

## MANAGING THE TRANSITIONS

### CASE '89 Results

|                                              |    |
|----------------------------------------------|----|
| 1989 Management and Practice Track . . . . . | 48 |
| Future Directions of CASE                    |    |
| CASE in Manufacturing                        |    |
| Computer Supported Cooperative Work          |    |
| Productivity Quality and Metrics             |    |
| Technology Transfer                          |    |
| Standards for CASE                           |    |
| System Metrics                               |    |

### Use and Abuse of Case

|                                                                                                               |    |
|---------------------------------------------------------------------------------------------------------------|----|
| A Role for CASE in Manufacturing . . . . .                                                                    | 58 |
| <i>J. Behm</i>                                                                                                |    |
| A Case for CASE Administration . . . . .                                                                      | 60 |
| <i>M.R. Danziger and R.F. Birney</i>                                                                          |    |
| Elements of a Realistic CASE Tool Adoption Budget . . . . .                                                   | 62 |
| <i>C.C. Huff</i>                                                                                              |    |
| The Use of CASE in Planning for Information Systems Development by UK Management . . . . .                    | 63 |
| <i>A. Leeming</i>                                                                                             |    |
| Integrating CASE Into the CIS Curriculum: Enhancing Understanding and Quality<br>in All CIS Courses . . . . . | 64 |
| <i>E.G. Rodgers</i>                                                                                           |    |
| Enabling Technologies for CASE . . . . .                                                                      | 66 |
| <i>I. Simmonds</i>                                                                                            |    |
| Potential Benefits and Problems of Customizable CASE Tools . . . . .                                          | 68 |
| <i>P. Smith, S.C. Stobart, and J.B. Thompson</i>                                                              |    |
| An Analysis of the Use of Commercial CASE Tools . . . . .                                                     | 70 |
| <i>S.C. Stobart, J.B. Thompson, and P. Smith</i>                                                              |    |

### Technology Transfer

|                                                                               |    |
|-------------------------------------------------------------------------------|----|
| Preventing Expensive CASE Tools Shelfware . . . . .                           | 74 |
| <i>L.K. Aharonian</i>                                                         |    |
| Managing the Transitions . . . . .                                            | 75 |
| <i>B. Beccue</i>                                                              |    |
| CASE Systems, Software Engineering Environments, and the Human Side . . . . . | 76 |
| <i>R. Becker</i>                                                              |    |
| Issues in CASE Technology Transfer . . . . .                                  | 78 |
| <i>D.E. Cooke</i>                                                             |    |
| Some Issues in CASE Technology Transfer . . . . .                             | 80 |
| <i>W.L. McCoy</i>                                                             |    |
| CASE - Transition Hazards and Requirements for the 90's . . . . .             | 82 |
| <i>A.C. O'Hara, Jr.</i>                                                       |    |
| Implementing CASE . . . . .                                                   | 84 |
| <i>A.R. Perrelli</i>                                                          |    |
| CASE Implementation Strategy . . . . .                                        | 86 |
| <i>A. Schiemann</i>                                                           |    |

## Methods & Tools To Support the Transitions

|                                                                                    |    |
|------------------------------------------------------------------------------------|----|
| Overcoming the Hurdles Between Phases . . . . .                                    | 90 |
| <i>J.C. Anderson and L.S. Marshall</i>                                             |    |
| On Transformations During Software Development . . . . .                           | 92 |
| <i>G. Boloix</i>                                                                   |    |
| Enhancing the Technology Transfer Process Through Explanation Facilities . . . . . | 94 |
| <i>B. Cox and J. Jenkins</i>                                                       |    |
| Mapping CASE Technology into the Information Systems Model Curriculums . . . . .   | 96 |
| <i>D. Jankowski</i>                                                                |    |

## Transition From Research to Product

|                                                                    |     |
|--------------------------------------------------------------------|-----|
| A Distributed Software Engineering Environment (DISSENV) . . . . . | 100 |
| <i>K. Känsälä and S. Kalliomäki</i>                                |     |
| CASE Environment from Individuals to Team . . . . .                | 102 |
| <i>E. Teratsuji</i>                                                |     |

## GROUP AND PROCESS MANAGEMENT

### Group Work Support Tools

|                                                                                                       |     |
|-------------------------------------------------------------------------------------------------------|-----|
| A User Interface Management System (UIMS) for an Integrated Systems Development Environment . . . . . | 106 |
| <i>A.L. Godinho and F.C. Lopes</i>                                                                    |     |
| Enterprise Analyzer: Supporting CASE with Electronic Meeting Systems . . . . .                        | 108 |
| <i>J.F. Nunamaker, Jr., V. Ramesh, G. Hayes, R.M. Daniels, Jr.,<br/>A.R. Dennis, and D.R. Vogel</i>   |     |

### Metrics, Measurement & Risk Analysis/Management

|                                                                                                                   |     |
|-------------------------------------------------------------------------------------------------------------------|-----|
| CASE Tools and Workstation Productivity . . . . .                                                                 | 112 |
| <i>B. Cureton</i>                                                                                                 |     |
| Cost Estimation Model for CASE Based Projects . . . . .                                                           | 119 |
| <i>L. Mancini</i>                                                                                                 |     |
| CASE and Software Science Metrics . . . . .                                                                       | 124 |
| <i>C.A. Argila</i>                                                                                                |     |
| Synergy Between Software Metrics and Cooperating Human Groups Through Software Development Environments . . . . . | 125 |
| <i>V. Côté and Richard St-Denis</i>                                                                               |     |
| A Methodology for CASE Cost-Effectiveness Validation . . . . .                                                    | 126 |
| <i>R.J. Freeman</i>                                                                                               |     |
| CASE, Measurement and Management of the Software Process . . . . .                                                | 128 |
| <i>D.R. Jeffery</i>                                                                                               |     |
| Metric Feedback: Will It Help Us Improve the Way We Teach Software Engineering? . . . . .                         | 130 |
| <i>F.J. Mowle</i>                                                                                                 |     |
| An Approach to Evaluate Extent of Software Testing . . . . .                                                      | 132 |
| <i>B.S. Rao</i>                                                                                                   |     |
| CASE as a Process and Metrics Testbed . . . . .                                                                   | 134 |
| <i>G. Tate</i>                                                                                                    |     |
| Metrics for CASE . . . . .                                                                                        | 135 |
| <i>J. Verner</i>                                                                                                  |     |

## ENABLING TECHNOLOGIES

### CASE '89 Results

|                                                          |     |
|----------------------------------------------------------|-----|
| 1989 Enabling Technologies Track . . . . .               | 138 |
| Visual Programming                                       |     |
| Document Management and Hypertext                        |     |
| Repository Architectures                                 |     |
| Object Management Systems                                |     |
| Impact of the IBM Repository                             |     |
| User Interface                                           |     |
| 1989 Software Factories and Environments Track . . . . . | 145 |
| IPSE                                                     |     |
| IPSE Requirements                                        |     |
| Tool Integration                                         |     |
| IPSE -2                                                  |     |

### CASE Tool Integration

|                                                                                          |     |
|------------------------------------------------------------------------------------------|-----|
| Methods Conformance - Where are the Vendor's Today? . . . . .                            | 150 |
| <i>J.M. Baker</i>                                                                        |     |
| Atmosphere: CASE for System Engineering Support . . . . .                                | 154 |
| <i>A. Dineur</i>                                                                         |     |
| An Introduction to ESF Software Factories . . . . .                                      | 158 |
| <i>C. Fernström, O. Heße, and R.E. Thomas</i>                                            |     |
| Implementing Cooperation and Coordination in Software Engineering Environments . . . . . | 163 |
| <i>J. Garbajosa, J.R. Larre, J. Sánchez, N. Alfaro, and J.J. Galán</i>                   |     |
| How Will We Achieve True Integration? . . . . .                                          | 168 |
| <i>A.J. Symonds</i>                                                                      |     |
| The CASE with OPENvision/C . . . . .                                                     | 173 |
| <i>A.A. Azmi</i>                                                                         |     |
| Integrated Software Engineering Applications . . . . .                                   | 174 |
| <i>P. Brown</i>                                                                          |     |
| Quality Software from IPSEs . . . . .                                                    | 176 |
| <i>A. Earl</i>                                                                           |     |
| Automated Process Definition and Execution . . . . .                                     | 179 |
| <i>J.R. Friedman and G.F. Hoffnagle</i>                                                  |     |
| Foundation for a C++ Programming Environment . . . . .                                   | 181 |
| <i>R.P. Gabriel, N. Bourbaki, M. Devin, P. Dussud, D. Gray, and H. Sexton</i>            |     |
| A CASE Rapid Prototyping System for the 90's . . . . .                                   | 182 |
| <i>A. Guzman and D. Hartman</i>                                                          |     |
| Standardized Protocols for Inter Tool Communication . . . . .                            | 184 |
| <i>D. Hare</i>                                                                           |     |
| Computer-Aided Software Engineering in a Network-based Team Environment . . . . .        | 186 |
| <i>G.-S. Kuo</i>                                                                         |     |
| The Role of a Uniform Language-Based Interface in CASE Integration . . . . .             | 187 |
| <i>C. Lagoze</i>                                                                         |     |
| CASE Tool Integration . . . . .                                                          | 189 |
| <i>H. Majima</i>                                                                         |     |
| The Development of a Multi-Platform CASE Tool Common Development Environment . . . . .   | 190 |
| <i>R.W. Mantha and D. Ridjanovic</i>                                                     |     |
| Mahalo: A General Software Design Support Environment . . . . .                          | 193 |
| <i>I. Miyamoto and G. Shi</i>                                                            |     |

|                                                                                         |     |
|-----------------------------------------------------------------------------------------|-----|
| CASE Integration: The "Market Forces" Model . . . . .                                   | 196 |
| <i>C.D. Sharpe and R.D. Avis</i>                                                        |     |
| Upper Case Meets Lower Case: Tool Integration to Support the Entire Lifecycle . . . . . | 199 |
| <i>M. Sokolsky</i>                                                                      |     |
| Goals and Requirements for Integration Frameworks . . . . .                             | 201 |
| <i>I. Thomas</i>                                                                        |     |
| Achieving Effective Tool Integration in a CASE Environment . . . . .                    | 204 |
| <i>J.R. Venable</i>                                                                     |     |
| Issues for CASE Tool Integration . . . . .                                              | 206 |
| <i>N. Wybolt</i>                                                                        |     |
| CASE Tool Integration and Standardization . . . . .                                     | 208 |
| <i>P.F. Zarrella</i>                                                                    |     |

## Repositories

|                                                                                             |     |
|---------------------------------------------------------------------------------------------|-----|
| Making a Repository Useful: Standard Information Models . . . . .                           | 212 |
| <i>S.A. Borkin</i>                                                                          |     |
| Introduction and Acceptance of Open CASE Data Repositories . . . . .                        | 213 |
| <i>H. Davis</i>                                                                             |     |
| About CASE Object Management Systems . . . . .                                              | 214 |
| <i>G. Godart</i>                                                                            |     |
| The Hypernode Model as the Basis for a CASE Information Repository . . . . .                | 216 |
| <i>M. Levene, A. Poulouvassilis, and R. Offen</i>                                           |     |
| Cooperation and CASE Databases . . . . .                                                    | 219 |
| <i>L. Miguel</i>                                                                            |     |
| Repositories: Are They Really Necessary for CASE Tool Integration? . . . . .                | 221 |
| <i>N. Wybolt</i>                                                                            |     |
| Graph Calculus as a Query Language for Hypergraph-Based Software Object Databases . . . . . | 223 |
| <i>Y. Yang and F. Hunt</i>                                                                  |     |

## MetaCASE

|                                                                                                                           |     |
|---------------------------------------------------------------------------------------------------------------------------|-----|
| ALF: A Generic Software Process-Centered CASE Environment . . . . .                                                       | 228 |
| <i>F. Oquendo, J.-D. Zucker, and P. Griffiths</i>                                                                         |     |
| A Visual Software Process Language . . . . .                                                                              | 232 |
| <i>C.D. Shepard, S.C. Sibbald, J.J.P. Tremblay, and C.M. Wortley</i>                                                      |     |
| Usability as a Design Perspective for CASE . . . . .                                                                      | 237 |
| <i>A.S. Duncan</i>                                                                                                        |     |
| Methodologies and CASE Tools: The Missing Link (A Meta-CASE Model for<br>Designing and Applying Methodologies) . . . . .  | 239 |
| <i>R. Goldstein</i>                                                                                                       |     |
| Software Artifacts Laboratory . . . . .                                                                                   | 241 |
| <i>A. Lakhota</i>                                                                                                         |     |
| Modeling the Software Development Process and the Benefits for CASE . . . . .                                             | 243 |
| <i>P.V. O'Kane</i>                                                                                                        |     |
| Organizational and Process Modelling Can Support Technical and Management<br>Innovation in Software Development . . . . . | 245 |
| <i>R.J. Offen</i>                                                                                                         |     |
| Software Process Models: Establishing a Sound Approach to the Software<br>Development Process . . . . .                   | 246 |
| <i>K. Proudlock</i>                                                                                                       |     |
| One View of the Transformation of JSD to SA . . . . .                                                                     | 248 |
| <i>R. Rask</i>                                                                                                            |     |

|                                                     |     |
|-----------------------------------------------------|-----|
| CASE Tool Integration and Process Control . . . . . | 250 |
| <i>C.D. Sharpe</i>                                  |     |
| Customizing CASE Tools . . . . .                    | 252 |
| <i>B. Tagg</i>                                      |     |

## METHOD AND TOOLS

### CASE '89 Results

|                                              |     |
|----------------------------------------------|-----|
| 1989 Methods and Tools Track . . . . .       | 254 |
| Executable Models                            |     |
| Configuration Management                     |     |
| Real-Time Systems and Concurrency            |     |
| Methods and Tool Development                 |     |
| Tools: State of the Art                      |     |
| Software Re-usability                        |     |
| Capture of Human Understanding               |     |
| CASE for Knowledge-Based Systems             |     |
| Object-Oriented Design and CASE              |     |
| Maintenance and Reverse Engineering: Methods |     |

### Requirements Elicitation and Traceability

|                                                                                                                                   |     |
|-----------------------------------------------------------------------------------------------------------------------------------|-----|
| Using Analogies During Software Maintenance . . . . .                                                                             | 260 |
| <i>V.S. Murthy and S.A. Raghavan</i>                                                                                              |     |
| Towards Automated Requirements Analysis . . . . .                                                                                 | 269 |
| <i>S.A. Raghavan and S. Badlani</i>                                                                                               |     |
| Marcel: A Requirements Elicitation Tool Utilizing Scenarios . . . . .                                                             | 279 |
| <i>S.M. Thebaut, M.F. Interrante, and T.F. Burch</i>                                                                              |     |
| A Support Tool for Integrated Dialogue Specification . . . . .                                                                    | 290 |
| <i>E.J.T. van Dinter, M.P.W. Martens, A.H.M. ter Hofstede, and S. Brinkkemper</i>                                                 |     |
| A CASE-Tool Shell to Facilitate the Customization of Requirements Engineering Support . . . . .                                   | 302 |
| <i>S. Brinkkemper, A.H.M. ter Hofstede, V. Verhoef, and G.M. Wijers</i>                                                           |     |
| Automating User Requirements Elicitation . . . . .                                                                                | 304 |
| <i>E. Carmel, J.F. George, and J.F. Nunamaker, Jr.</i>                                                                            |     |
| Requirements Elicitation and Traceability . . . . .                                                                               | 306 |
| <i>M. DeMeis</i>                                                                                                                  |     |
| ACTESS: Support for Validation of System Specifications . . . . .                                                                 | 307 |
| <i>J. Györkös, I. Rozman, and T. Welzer</i>                                                                                       |     |
| A Requirements Traceability and Management System (ARTMS) . . . . .                                                               | 310 |
| <i>A.A. Kamran</i>                                                                                                                |     |
| A Knowledge Representation for Requirements Elicitation . . . . .                                                                 | 312 |
| <i>J.R. Lima and H.B. Rego</i>                                                                                                    |     |
| A User-Centered CASE Tool that Enforces Completeness, Consistency and Correctness of Information Systems Specifications . . . . . | 315 |
| <i>R. Montealegre and F. Paz</i>                                                                                                  |     |
| CASE Support for Requirements Analysis . . . . .                                                                                  | 318 |
| <i>S.A. Raghavan and S. Badlani</i>                                                                                               |     |
| Knowledge Elicitation and Representation . . . . .                                                                                | 321 |
| <i>H. B. Rego and J.R. Lima</i>                                                                                                   |     |
| Integrating Front-End CASE with Requirements Management . . . . .                                                                 | 323 |
| <i>J. Thompson</i>                                                                                                                |     |
| FASTER (A FAST State machine ExecutoR) . . . . .                                                                                  | 326 |
| <i>P.L. Treadway</i>                                                                                                              |     |

## Reuse

|                                                                                                       |     |
|-------------------------------------------------------------------------------------------------------|-----|
| Exploiting Reusable Specifications Through Analogy . . . . .                                          | 328 |
| <i>N. Maiden and A. Sutcliffe</i>                                                                     |     |
| Cooperative Production of Reusable Design Units . . . . .                                             | 343 |
| <i>M.G. Fugini and B. Pernici</i>                                                                     |     |
| Towards Design Reuse in CASE Tools . . . . .                                                          | 347 |
| <i>H.-Y. Lee</i>                                                                                      |     |
| Truly Intelligent Analytic Support in CASE Tools . . . . .                                            | 348 |
| <i>N. Maiden and A. Sutcliffe</i>                                                                     |     |
| Reuse of Software Components (Alternatives in Characteristics, Organization and Tool Support) . . . . | 350 |
| <i>W. Rossak, R. Mittermeir, and E. Hochmüller</i>                                                    |     |
| Component Understanding Within the Software Reuse Process . . . . .                                   | 352 |
| <i>R. Thomson</i>                                                                                     |     |

## Object-Oriented Methods

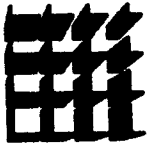
|                                                                                                        |     |
|--------------------------------------------------------------------------------------------------------|-----|
| A Functional and Object-Oriented Design Approach . . . . .                                             | 356 |
| <i>G. Dupont, P. Heringer, and C. Villerman</i>                                                        |     |
| Combining the CASE Tool: StP with the Object-Oriented-Active<br>Knowledge Base System: Mokum . . . . . | 373 |
| <i>R.P. van de Riet</i>                                                                                |     |
| Object Oriented Specification and CASE Support of Information System Development . . . . .             | 387 |
| <i>O. Barros</i>                                                                                       |     |
| What is the Best Input for Object-Oriented Design? . . . . .                                           | 388 |
| <i>G. Dupont</i>                                                                                       |     |
| Position on State-of-the-Art, Relating to Object-Oriented Methods . . . . .                            | 390 |
| <i>J. Norberg</i>                                                                                      |     |

## Formal Methods

|                                                                                                                            |     |
|----------------------------------------------------------------------------------------------------------------------------|-----|
| Declarative Factoring . . . . .                                                                                            | 394 |
| <i>R.J. Abbot</i>                                                                                                          |     |
| Towards a Domain Description Grammar: An Application of Linguistic Semantics . . . . .                                     | 410 |
| <i>R.P. Carasik, S.M. Johnson, D.A. Patterson, and G.A. Von Glahn</i>                                                      |     |
| Destination "Moon:" Normalized Object-Oriented Method . . . . .                                                            | 420 |
| <i>H. Habrias</i>                                                                                                          |     |
| CASE in the 90's . . . . .                                                                                                 | 445 |
| <i>R.W. Schutten</i>                                                                                                       |     |
| Test Case Generation from Specifications Written in PROLOG: A Meta-Interpretive Approach . . . .                           | 449 |
| <i>R. Denney</i>                                                                                                           |     |
| A Formalized Program Specification Method . . . . .                                                                        | 453 |
| <i>H.M. Edwards, J.B. Thompson, and P. Smith</i>                                                                           |     |
| On the Formal Methods for Proving System Correctness of Concurrent Processes<br>Using Asynchronous Communication . . . . . | 455 |
| <i>G. Hemdal and T. Dexter</i>                                                                                             |     |
| Methods and Tools . . . . .                                                                                                | 457 |
| <i>H.-H. Kaaber</i>                                                                                                        |     |
| USDM: An Integrated Methodology for Realtime Distributed Programming . . . . .                                             | 459 |
| <i>K. Steer</i>                                                                                                            |     |
| A Formal Basis with Tool Support for Jackson System Development . . . . .                                                  | 461 |
| <i>W.L. Yeung</i>                                                                                                          |     |

## Verification and Validation Tools

|                                                                                                                    |     |
|--------------------------------------------------------------------------------------------------------------------|-----|
| Transferring Methods, Tools and Experience Between Electronic Design Automation (EDA) Domain and CASE . . . . .    | 464 |
| <i>R.N. Mudry</i>                                                                                                  |     |
| <b>The Effect of CASE on Software Engineering Practices</b>                                                        |     |
| WINTRODUCE: A Tool for Object-Oriented MS-Windows Code Generation . . . . .                                        | 468 |
| <i>K. Coninx, P. Vandoren, and E. Flerackers</i>                                                                   |     |
| A Graphical Entity-Relationship Based Database Management System Workbench . . . . .                               | 480 |
| <i>T.W. Ling and M.L. Lee</i>                                                                                      |     |
| Effects of CASE on Software Engineering Practices . . . . .                                                        | 496 |
| <i>S.K. Majumdar</i>                                                                                               |     |
| A Case for Influencing Systems Development Methodology . . . . .                                                   | 502 |
| <i>P.K. Pelly</i>                                                                                                  |     |
| Process Maturity as a Framework for CASE Tool Selection . . . . .                                                  | 519 |
| <i>S.L. Pfleeger</i>                                                                                               |     |
| CASE Discourages the High-Level Iteration Most Practices Require . . . . .                                         | 529 |
| <i>R.C. Cohen</i>                                                                                                  |     |
| Micro-Computer-Based Development of Mainframe Applications . . . . .                                               | 531 |
| <i>J.M. Grochow</i>                                                                                                |     |
| Managing the Transition - Technology Transfer or Methods & Tools to Support the Transitions . . . . .              | 532 |
| <i>P. Jorgensen</i>                                                                                                |     |
| CASE Tools for Structural Design . . . . .                                                                         | 533 |
| <i>T. Matsubara</i>                                                                                                |     |
| Three-Step Procedure to Accomodate Multiple Methodologies . . . . .                                                | 535 |
| <i>H.H. Sayani</i>                                                                                                 |     |
| CASE Tools for Evolving Multiprocessor Architectures . . . . .                                                     | 536 |
| <i>S.R. Schach</i>                                                                                                 |     |
| The Effect of CASE Tools on Software Development Practices within Commercial Data Processing Departments . . . . . | 538 |
| <i>J.B. Thompson, S.C. Stobart, and P. Smith</i>                                                                   |     |
| Report on the CASE '89 International Standards Coordination Meeting . . . . .                                      | 542 |
| <i>P. Eirich, Chairman</i>                                                                                         |     |
| <i>H. Downs, M. Morron, W. Page, D. Sharon, H. Taraji, and G. Tatge</i>                                            |     |
| Keyword Index by Paper . . . . .                                                                                   | 550 |
| Author Index . . . . .                                                                                             | 552 |



**CASE '90**

*Fourth International Workshop on  
Computer-Aided Software Engineering*

*Co-sponsored by the Int'l. Workshop on  
CASE Inc. and the IEEE Computer Society*

# **Existing Systems**

**CASE '89 RESULTS**

# **MAINTENANCE AND REVERSE ENGINEERING**

- **GOALS**
  - **APPROACHES**
  - **DEFINITIONS**
  - **OPEN ISSUES**
  - **RESEARCH PROJECTS**
  - **RESEARCH NEEDED**
- 
- **CHANGE FROM 1988:    RECOGNITION**  
**BUZZWORD STATUS AND HYPE**  
**MORE PEOPLE DOING MORE THINGS**

## **GOALS**

- \* Deal with complexity
- \* Deal with volume
- \* Lost information
- \* Migrate to some other representation (better?)
- \* Provide a level of understanding
- \* Having the information available
- \* Moving to a higher-level abstraction
- \* Detect re-use candidates

## **APPROACHES USED FOR REVERSE ENGINEERING**

1. Code Scanning
2. Analysis of Program Structures
3. Reshaping Structure
4. Analysis of Data Structure
5. Sys/Command Level Scanning e.g. make files
6. Domain Analysis
7. Design Recovery
8. Data Name Standardization
9. Language Translators
10. Program Optimization
11. Abstraction of Design
12. Abstraction of Data
13. Identification of Reusable Portions
14. Identification of Misused Items
15. Presentation/Other Forms of Representation

## **DEFINITIONS**

**REVERSE** - Process of extracting/abstracting design artifacts  
Process of building abstractions which are less  
implementation dependent  
-> information gained

**RE-ENGINEERING** - Process of recasting s/w to another form  
Modification •  
-> results in executable product

**RE-STRUCTURING** - Altering of code for improved str.

**REUSE** -      1 - ID stable components that can be re-used  
                  2 - MECH. that support re-use

**"SOFTWARE PERESTROIKA"**

**RE-DOCUMENTATION**

## ISSUES

- \* Abstraction/Summarization without domain knowledge
- \* Cross-level abstraction vs. utility at same level
- \* "Most vendors appear not to be doing process re-engineering"
- \* Likely not to be a fully automated process
- \* What to capture/analyze?
- \* Nature of R.E. process? -Methodology
- \* The code has lost most real-world requirements information
- \* Conceptual understanding of code/psychology process
- \* Inclusion of concepts from run-time measurement; simulation
- \* "Legal Definition"
- \* Understandability of re-structured code