THE SHAPE OF THINGS TO COME

By

P. A. Zelenski
The MacNeal-Schwendler Corporation
Los Angeles, California

The Shape of Things to Come

Paul A. Zelenski, Dan J. Bryce, Jim I. Walker
The MacNeal-Schwendler Corporation

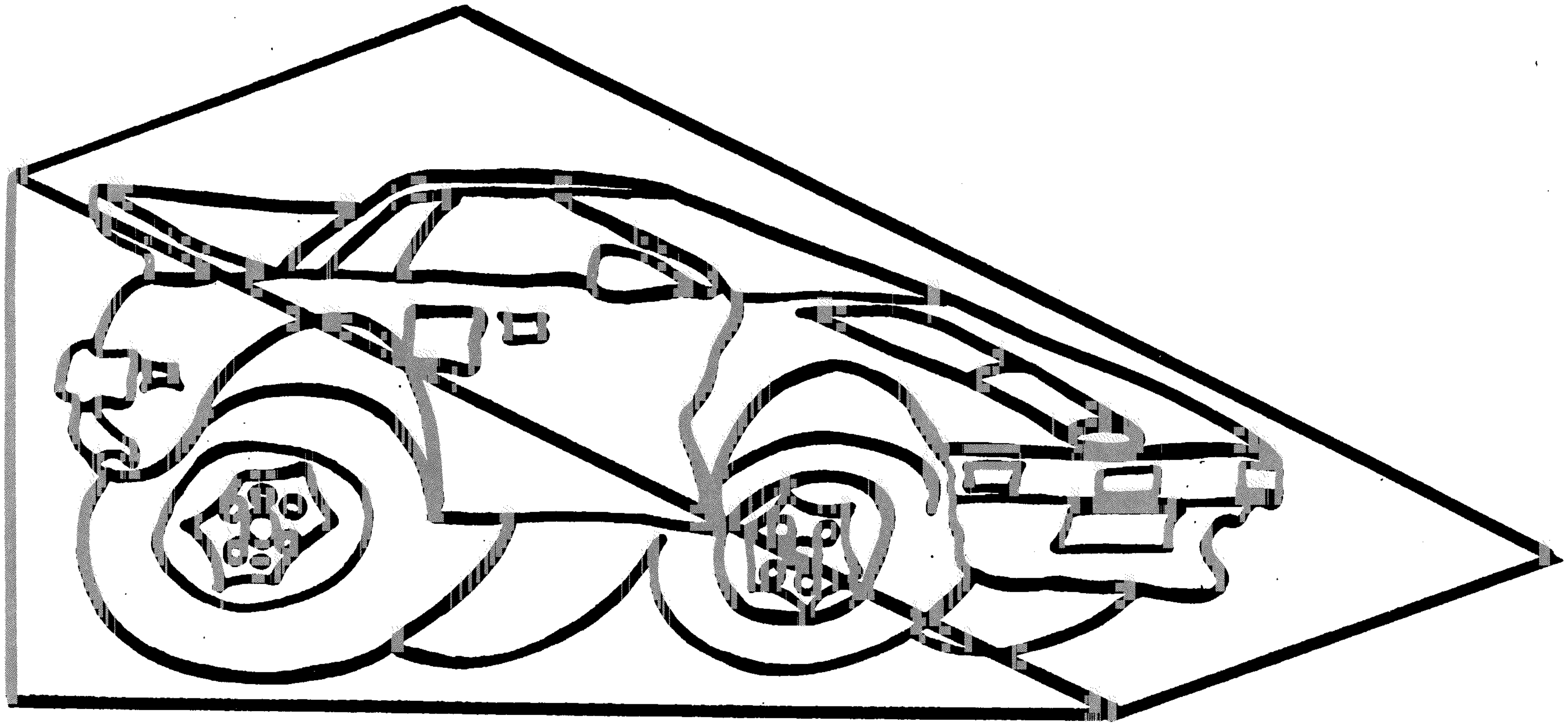
ABSTRACT

The development of engineering software tools is a very specialized and complex field of endeavor. MSC has long been a leader in the development of Engineering analysis tools and has now embarked on a growth path to include an integrated interactive graphics system into its successful and capable repertoire of software products.

This paper suggests the principles involved in the design and development of such a system. The concept of "form follows function" leads to a discussion of the attributes that good engineering software ought to have and the mechanisms that software developers may use to build those attributes into the software.

Programming techniques involving rapid prototyping, artificial intelligence, new programming languages and the "object oriented" approach are discussed.

The conclusion of this presentation is the realistic and critical observation that all stages of the development activity must be done with the end user in mind. All capabilities added to the system and all techniques used in its development must bring out the primary premise that "the User is King."



The Shape of Things to Come

Paul A. Zelenski, Dan J. Bryce, Jim I. Walker

The MacNeal-Schwendler Corporation

(THIS VIEWGRAPH WAS INTENTIONALLY LEFT BLANK)

1. Open Architecture
2. Intelligent Defaults
3. Near-Infinite Flexibility
4. Bridges to Other Systems
5. Documentation
6. Close Coupling with Interfacing Software
7. User Extensible
8. Non-Static User Interface
9. Recognize Historical Trends
10. Bias for Action
11. Rapid Prototyping
12. User Comes First
13. Present Capabilities From Users Point of View
14. Productivity and Quality
15. Simple is Powerful
16. Encapsulation for Simplicity and Control
17. Object Oriented Approach
18. Easily and Purposefully Extensible
19. Expandable and Powerful Data Structures
20. Consistent Interface
21. Easy to Learn
22. Concise
23. Natural
24. Familiar and Predictable
25. Context Sensitive Help
26. Scaffolding (Checkpointing and Resuming)
27. System Response
28. High Information Content Per Frame
29. Flexible Visual Rearrangement
30. Orientation Tools (you are here)
31. Refinement of Masses of Information
32. Entertaining and Pleasurable
33. Encourage Exploration and Progress
34. Levels of "Hand Holding"
35. Learning by Experimentation
36. Adapt Modern Tools to the Task
37. Design to use Standards
38. Design to use Hardware Features
39. Design to use Personal Workstation Features
40. Uncompromising Quality
41. Attention to Detail
42. Stick to What You Know
43. Good Standards in Programming Practices
44. Consistent Environment Across Many Platforms

1. Open Architecture

2. Intelligent Defaults

3. Near-Infinite Flexibility

4. Bridges to Other Systems

5. Documentation

6. Close Coupling with Interfacing Software

7. User Extensible

8. Non-Static User Interface

9. Recognize Historical Trends

10. Bias for Action

11. **Rapid Prototyping**

12. User Comes First

13. Present Capabilities From Users Point of View

14. Productivity and Quality

15. Simple is Powerful

16. Encapsulation for Simplicity and Control

17. Object Oriented Approach

18. Easily and Purposefully Extensible

19. Expandable and Powerful Data Structures

20. Consistent Interface

21. Easy to Learn
22. Concise
23. Natural
24. Familiar and Predictable
25. Context Sensitive Help
26. Scaffolding (Checkpointing and Resuming)
27. Response Response
28. High Information Content Per Frame
29. Flexible Visual Rearrangement
30. Orientation Tools (You Are Here)
31. Refinement of Masses of Information
32. Entertaining and Pleasurable
33. Encourage Exploration and Progress
34. Levels of "Hand Holding"
35. Learning by Experimentation

36. Adapt Modern Tools to the Task

37. Design to use Standards

38. Design to use Hardware Features

39. Design to use Personal Workstation Features

<u>40. Uncompromising Quality</u>

41. **Attention to Detail**

42. **Stick to What You Know**

43. Good Standards in Programming Practices

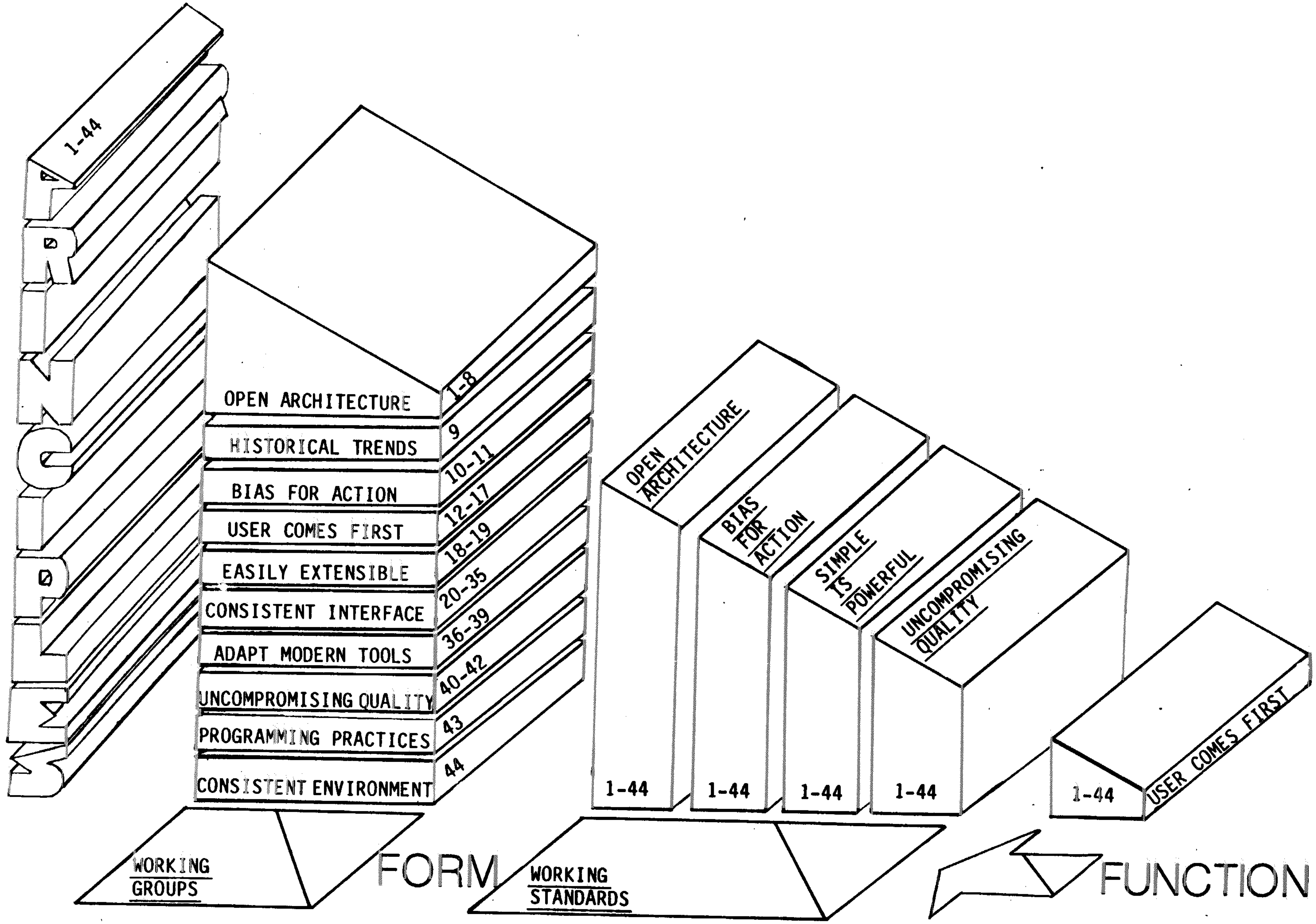
44. Consistent Environment

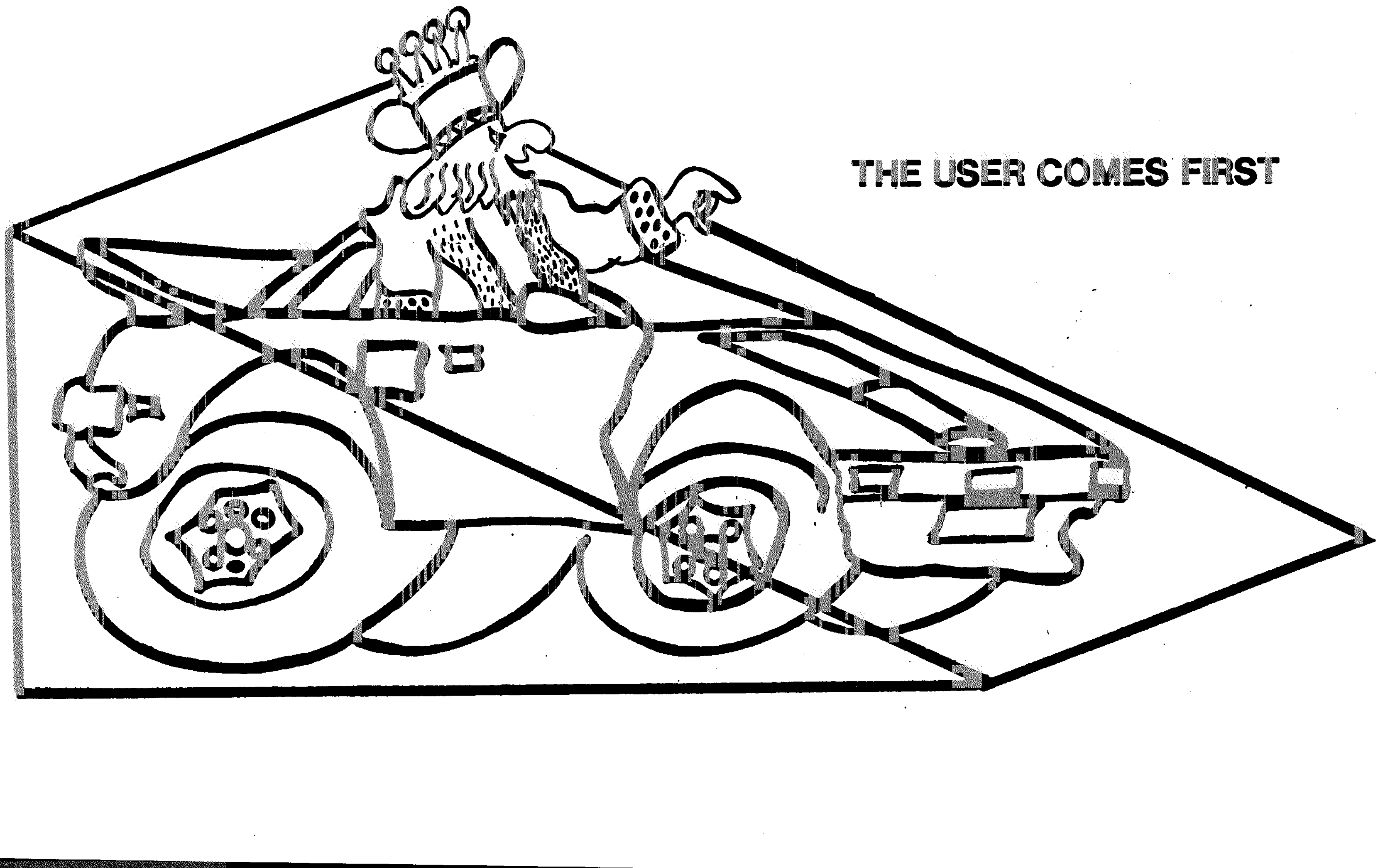
OPEN ARCHITECTURE

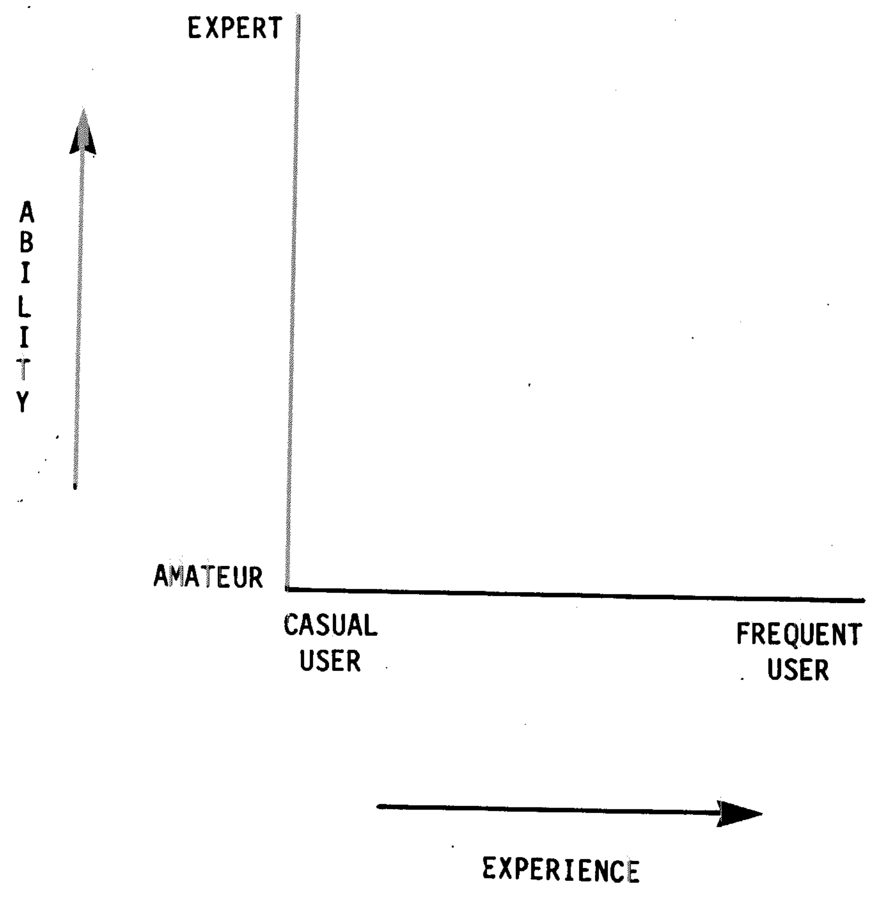
BIAS FOR ACTION

SIMPLE IS POWERFUL

UNCOMPROMISING QUALITY


PRINCIPLES
1-44
OPEN ARCHITECTURE 1-8
HISTORICAL TRENDS 9
BIAS FOR ACTION 10-11
USER COMES FIRST 12-17
EASILY EXTENSIBLE 18-19
CONSISTENT INTERFACE 20-35
ADAPT MODERN TOOLS 36-39
UNCOMPROMISING QUALITY 40-42
PROGRAMMING PRACTICES 43
CONSISTENT ENVIRONMENT 44
WORKING GROUPS
FORM
OPEN ARCHITECTURE 1-44
BIAS FOR ACTION 1-44
SIMPLE IS POWERFUL 1-44
UNCOMPROMISING QUALITY 1-44
USER COMES FIRST 1-44
WORKING STANDARDS
FUNCTION


THE USER COMES FIRST


EXPERT
ABILITY
AMATEUR
CASUAL USER
FREQUENT USER
EXPERIENCE