



TABLE OF CONTENTS

| CHAPTER | TITLE | PAGE |
|---------|--------------------------------------|------|
| 1. | The VSE Paradigm | 01 |
| 2. | VSE Basics | 03 |
| 3. | The VSE Language | 10 |
| 4. | Bubble Sort | 17 |
| 5. | Intertask Communications And Control | 22 |
| 6. | Run-Time Graphics - Basics | 25 |
| 7. | Run-Time Graphics – Event Handler | 35 |
| 8. | The Panel Library Manager | 43 |
| 9. | Airport Status | 54 |
| 10. | Pepsi Challenge | 65 |

May 31, 2001

This book was produced by Henry Ledgard. Major contributions were made by Rashmi Tambe, Marty Chintakindi, Randi Appitika, and Shawn Wang.

CHAPTER 1: THE VSE PARADIGM

VSE is a total environment for building large scale software. VSE uses a unique graphical approach to building and storing large quantities of reusable software components. VSE is based on a very high-level software architecture that enables the user to easily define components of an application.

VSE supports development of software using drawings that contain hierarchical models. The user creates drawings using a library of symbols. These symbols, and the lines that interconnect them, can have different colors and styles that have meaning to the system.

ARCHITECTURE ENVIRONMENT

VSE features a unique separation of the architectural design of a system from its detailed implementation. This can only occur when the drawings (architecture) provide a one-to-one mapping into the language (code). This one-to-one mapping can only be accomplished when data is separated from instructions. It is the separation of data from instructions that provides a framework for software visualization.

To support these requirements, the Architecture Environment provides facilities for completing the detailed architectural design, and producing engineering drawings. This includes the design of models to the primitive layer, identifying the contents of resources and processes. The most significant part of the architectural process is determining how the resources are dedicated to, or shared by, processes and models. This determines the understandability and independence of models, the critical components if they are to be reusable.

The drawings contain several intuitive system symbols. Developers can pop the covers of symbols to see the next layer down. The resources (data structures) and processes (executable instructions) are found at the lowest level of the hierarchy. Users can edit the data and instructions they contain - directly on the drawing.

LANGUAGE ENVIRONMENT

Having completed the program architecture, the programmer proceeds to design the detailed attribute and rule structures for the processes and resources that are defined in the architecture. Individual models can be prepared in the Language Environment to resolve conflicts and unresolved references. This implies the preparation of all

constituent resources and processes. One also writes a control specification in the Language Environment.

The concepts are quite simple, once one realizes the importance of separating data from instructions, i.e., it is the key to visualization of software. Software is made up of data and instructions. In VSE, data is stored in resources, and instructions are stored in processes. These are grouped together to form elementary models. Elementary models can then be grouped with other models to form hierarchical models. The use of VSE is described starting in the next chapter of this manual.

RUN-TIME GRAPHICS

The Run-Time Graphics (RTG) system is an extension to VSE that provides the developer with a development environment that makes it easy to build dynamic interactive graphics. This allows the end user to interact with a program, while it is running, using a mouse and keyboard at a graphics workstation.

ICON LIBRARY MANAGER (ILM)

The Icon Library Management (ILM) facility is a subsystem of the GSS Run-Time Graphics (RTG) system. The facility provides the ability to create and manage icons to be used in RTG. The facility includes extensive capabilities for creating icon images from standard graphics primitives. To support the development of icons, the ILM consists of 2 subsystems. The first subsystem is the elementary icon builder that provides the ability to create elementary icons. The other subsystem is the hierarchical icon builder that provides the ability to create icon hierarchies from elementary icons.

PANEL LIBRARY MANAGER (PLM)

The Panel Library Management (PLM) facility is a subsystem of the GSS/VSE Run-Time Graphics (RTG) system. The facility provides the ability to create and manage panels for prompted input and output at run-time.

THE GENERAL SIMULATION SYSTEM (GSS)

The General Simulation System (GSS) provides a complete facility for building models and running simulations of general dynamic systems. GSS is specifically designed to handle the complexities of very large scale simulations, including those that must be run in real-time. Applications include secure, mobile communications systems, real time control systems, decision support systems, manufacturing systems, banking systems, etc. GSS has the ability to model complete physical systems, including models of the environments in which they are embedded. For example, GSS allows for specification of

vehicle motion, dynamic communications links, and environmental factors, such as terrain, foliage, and radio signal interference.