An Introduction To MOSES

Version 1.0

The MOSES Project:

Meta Operating System And Entity Shell

Daniel J. Pezely

22 May 1991

1 Overview

The MOSES Project goal is to develop a solid foundation for virtual environments. Virtual environments require distributed systems, and the underlying platform resemble operating systems. MOSES, the Meta Operating System and Entity Shell, provides the high-level operating environment needed as a foundation and may be either a native operating system itself or may be an application upon another operating system.

There are two fundamental design elements which makes MOSES suitable for handling arbitrary lists of information in a distributed environment: Entities and the DataSpace. The notion of an entity comes directly from William Bricken's VEOS Project, but the DataSpace structure and organization is unique to MOSES.

2 Executive Summary

Further analysis of the operating environment design requires a full implementation of the core programs and libraries. The design analysis and the-

ory has shown that the MOSES System provides for a very stable platform to build virtual environments upon. This platform design permits multiple users, potentially distributed over a wide area, to share information (both functions and data). Such a design quest was the primary reason for initiating the MOSES Project.

One design point to remember is: The MOSES System kernel (core routines and libraries) *is* an operating system *and* a distributed system. Our kernel requires none of the low-level internals of typical operating systems but contains the same general design characteristics. Such low-level operating system internals may, of course, be added later to make MOSES stand-alone.

3 Recommendations

We recommend the MOSES Project continues with a full kernel implementation. This kernel resides upon existing Unix platforms, leaving our developers free from re-implementing a whole operating system.

As the chief designer of MOSES, Daniel Pezely should lead the development of the kernel under William Bricken, but tools should be controlled by someone focusing primarily on tool development.

The kernel elements may be added to existing programming language libraries as new language primitives for the tool builders to use. All elements of the meta operating system may be implemented using these primitives and existing programming libraries.

With Daniel Pezely leading the System development, he should also develop the kernel communications element. One other person may be necessary to develop and maintain the DataSpace elements for rapid Project development.

All tools should be under another person's management for distribution of human resources, as is planned for the 1991 Summer Intern Program at HITL.

Tool primitives and interfaces need standardization within the Lab. The standard should be decided upon as a group effort of the kernel implementors and the tool builders. Such design discussions should not consume more than one full day, if we use last summer's design discussions as a basis for estimating time.

4 Summary of MOSES Features

MOSES is a very high-level operating system providing users with a system independent platform to build upon. We call this system an *operating environment* due to its high level features. Users—client programmers and application participants alike—need not worry about otherwise complex routines as memory/storage access, function/task control, and communication/transportation, since the operating environment provides such routines as features. The operating environment also provides a wide range of interfaces to these features:

- high-level routines with intuitive access methods
- mid-level routines designed for ease of use but geared towards efficiency
- low-level routines for optimized high-speed transactions between precompiled tools and the kernel.

Although we do not design for maximum efficiency, we do not ignore such issues. We designed for known and unknown systems technology yet to come but should also be effective on today's systems.

Implementing this operating environment completely using existing platforms, such as Unix operating systems and research-oriented distributed environments, is possible today.
