Inter-Office Memorandum

To	Distribution	Date	July 14, 1977
From	Wendell Shultz	Location	Palo Alto
Subject	**Mesa 3.0**	Organization	SDD/SD

XEROX

Filed on: <Lopez>Mesa3.0.memo

Attached are the work plan and release definition for Mesa 3.0, which is now under development. Please review their content in preparation for a formal approval of the contents of this Mesa release. Questions and comments should be directed to John Wick

As noted in the release definition, several features of old releases are being temporarily retained in order to ease conversion of existing source code. In some cases, compatibility is easily maintained and carries low cost; in others, retaining the old design severely constrains the development of a new feature. We need to determine the period over which compatibility will be maintained, and a general policy on this issue.

A meeting to discuss and approve the definition of Mesa 3.0 has been tentatively scheduled for Wednesday, July 20. Please let me know if this date is convenient.

Attachments

Distribution:

Geschke
Lampson
Lynch
Mitchell
Szelong

c: J. Weaver
J. Wick

Inter-Office Memorandum

To	John Weaver	Date	June 30, 1977
From	John Wick	Location	Palo Alto
Subject	Mesa 3.0 Development Work Plan	Organization	SDD/SD

XEROX

Filed on: <WICK>MESAPLAN30.MEMO

Introduction

This document describes the work plan for Mesa development. It addresses the planning, implementation, and release of the next version of the Mesa system (designated Mesa 3.0). This is a preliminary work plan, which will be updated as the release definition is refined and approved. Longer range planning is discussed in the Mesa Release Procedure.

Mesa 3.0 will include a number of improvements, the foremost of which is an implementation of configurations and a binding scheme for constructing configurations from a set of independently compiled modules. Configurations represent a major extension in all components of the system. The primary emphasis in this plan is on the effects of introducing and converting to configurations.

For the most part, changes to source programs required by Mesa 3.0 are confined to a small number of infrequently used constructs. In addition, most applications contain a single module which loads and initializes the rest of their components; such modules will require more extensive changes. Good documentation will be essential in planning these modifications. Therefore, Mesa 3.0 will not be released until existing documentation is in agreement with the changes and additional features described in the Release Definition. The documentation will be subject to the same alpha test procedures as the software.

Mesa support is included in this plan only to the extent that a fixed level of manpower has been allocated for all support functions. This commitment is known to be insufficient to satisfy the entire support requirement; we anticipate that some resource from outside the development team will be assigned to assist in this task. Mesa support will be described in the Mesa Support Work Plan to be issued when the requirements are better defined and potential manpower sources have been identified.

Mesa 3.0 Task Descriptions

Release Definition and Plan

This task involves the initial selection of the system changes to be included in this release. A companion Release Definition will describe the technical issues relating to the selected changes. As the definition is revised and approved, this work plan will be updated to reflect the decisions reached. The major mileposts given below provide a guideline as to the appropriate scope of change in this version.

Release Development

This activity includes the design, coding, and unit test of each change to the system. For this release, the major activities are construction of a new binder and a major reorganization of the functions performed by the loader. (The old binder and loader will continue to be supported temporarily for compatability.) Other substantial changes are required in the compiler, in the modules which construct image files, in the debugger, and in the system building and bootstrapping software.

An essentially fixed overhead is involved with generating and installing a new release. This includes compiling and building each subsystem, performing consistency checks, storing files on the central file store, constructing backup disk packs, and recording statistics on each major component of the system.

System Test

Initially, functional and regression tests will be performed using components of the Mesa software as test cases. We expect to convert the system and the debugger to the new scheme, and to retain the compiler and system building software as regression tests (this organization is dictated by bootstrapping considerations). Other regression tests will include a number of packages obtained from the Tools Environment and Communications Software. Full alpha testing will be performed by the Tools Section of the Development Environment and Support Group, by the Communications Systems Group, and by the Prototype Software Group.

Documentation

The *Mesa Language Manual* will be revised to reflect features added to support configurations. New sections (or a new document) explaining configuration descriptions and the binding and loading operations will be provided. Of particular importance is a consistent presentation of the effects of all changes so that users can convert to the new system with minimum difficulty. A number of specific change notices and technical bulletins will accompany the release.

Support

Support is provided at a fixed level. This represents only the contribution of the Mesa Development Team to the support effort. The details of the overall Mesa support strategy will be specified in the Mesa Support Work Plan.

Other Task Descriptions

External Review and Planning

Because of the close relationship between Mesa, the machine architecture, and the Pilot operating system, a significant amount of time must be devoted to planning and design activities carried on outside the Mesa Group. This includes, for example, a detailed and careful review of the *Pilot Functional Specifications* and the *OIS Processor Principles of Operation.*

Long Range Planning

To ensure a quality product, long range planning on the Mesa language and its implementation is a continuous activity. This task includes, for example, periodic analyses of the instruction set for code density, as well as a semi-permanent Language Working Group. These activities involve Parc assistence.

Projected Sizes

The following estimates are in thousands of new source lines:

Compiler changes	1
Binder	7
System (Loader)	1
System (MakeImage)	1
Debugger changes	1
BootMesa	3
Support	2.5
Total	16.5

These sizes are based on the lines of code written as of 16 June (8.2K source lines) and on the sizes of comparable modules in the current system.

Major Mileposts

Release Definition and Plan	
release for review	10 July
approval	15 July
Development complete	5 August
System test complete	15 August
Release for alpha test	15 August
Alpha test complete	20 September
Documentation	
release for alpha test	15 August
release for review	31 August
approval	20 September
Release Mesa 3.0	30 September

Manpower Plan

	J	J	A	S
Planning	¾	0	0	0
Development	4	3	1	0
System test	0	2	2	0
Alpha test support	0	0	¾	2
Documentation	½	¾	1	½
Support	¼	¼	¼	¼

Issues

The plan for conversion to the Dstar, regardless of its exact nature, will have a heavy and relatively immediate impact on the activities of the Mesa Group. For example, if the conversion plan calls for a D0 instruction set, rather than Alto/Mesa emulation, a new version of the compiler with different code generation modules will be required in the near term. Multiple versions of the system and the debugger will also be required. The resolution of conversion issues has a critical and immediate impact on Mesa planning.

Due to lack of response to the Binder Working Group Report, we are unclear as to how the new binder will be used, what modes of operation should be optimized, and what features should be cut back or extended (with a corresponding adjustment to the release date). This plan assumes that an early release date is preferable to extended features.

Due to the heavy demand for development work and a lack of manpower, the support activity continues to be understaffed.

Assumptions and Risks

The schedule assumes availability of Richard Johnsson to complete the binder subsystem, assist with required debugger changes, and convert the system building and bootstapping processes to the new binder. The mileposts above target completion of these tasks by approximately 1 September. He will retain primary responsibility for the microcode and code generators after that date.

The schedule also assumes that Ed Satterthwaite will be available one-half time during June and July for work on the section of the compiler which implements the Mesa type system; that Jim Mitchell will be available one-half time during July (and somewhat less during August) for work on the language manual; and that Butler Lampson, Jim Mitchell, Chuck Geschke, and Ed Satterthwaite are available in a consulting role for an extended period. The mileposts assume that Mitchell will complete a draft of the revisions to the *Mesa Language Manual* (or an additional document) by 15 August.

The major mileposts assume that the groups designated to perform alpha test will be in a position to test the system during the dates indicated, and that they will make every effort to report problems before attempting to work around them.

The work plan assumes that a maintenance release will not be required before the next functional release, and that the effort expended on system support can remain at its current level.

Another schedule dependency involves the timely review and approval of the release definition, the release documentation, and the Software Release Description.

The release procedures assume that PARC will continue to supply file storage on Maxc. This includes a short-term requirement for a files-only directory equal in capacity to the <MESA> directory during the period that the system is undergoing alpha test.

Distribution:

SD Managers
SD Mesa Group
Geschke, Lampson, Mitchell, Satterthwaite

XEROX

INFORMATION PRODUCTS GROUP
System Development Division
July 11, 1977

To: John Weaver

From: John Wick

Subject: Mesa 3.0 Release Definition

Keywords: Mesa, Release Plan, Work Plan

Filed On: <WICK>MESADEF30.BRAVO

This memo describes those features which are being planned for Mesa 3.0. The first section describes tasks whose implementation is well under way. Section two lists items which are likely to be included if resources are available.

Items are organized alphabetically by major component. References are enclosed in square brackets. To distinguish among multiple works by the same person, a terse abbreviation for the title of the reference has been appended to the author's name (e.g., [LampsonCD*]).

Section I

The following changes will be included in Mesa 3.0.

Language

Boolean Enumerated Type

Defining the built-in type BOOLEAN as an enumerated type will allow it to be used as a variant record tag. [WickLWG]

Data Modules

The destinction between DATA and PROGRAM modules is no longer meaningful, since all modules are started only once under the new scheme proposed for the binder. *Data modules will be retained temporarily for compatability.* [Sweet]

Directory Names

In conjunction with other changes which improve type checking (primarily export checking), the compiler will now verify that the name in the DIRECTORY statement matches the one in the corresponding definitions or program module. The binder will also make this check. [Maybury]

Empty Intervals

Allowing empty intervals (especially for array index sets) in type definitions will provide some help in implementing sequences until they are incorporated into the language. [WickLWG]

External Declarations

The new binding scheme requires that all references to types defined outside a module must be obtained from included modules. The EXTERNAL attribute will therefore be removed from the language. *It will be retained temporarily for compatability.* [Binder, Sweet]

Implementing/Sharing

The IMPLEMENTING construct will be replaced by EXPORTS. For compatability with IMPORTS and EXPORTS, SHARING will be replaced by SHARES. *Both will be retained temporarily for compatability.* [Binder, Sweet]

Implicit Stop

The implicit STOP statement inserted by the compiler between the initialization code and the module's main body will be removed. This implies that all modules need be started only once, regardless of the presence of main body code. [Maybury]

Imports/Exports

The IMPORTS and EXPORTS lists will be added to the program header. The compiler will now check that a module's procedures match the corresponding definitions in the exported interface. [Binder]

Module Instantiation

Parameter passing will move from NEW to START, allowing a module to be instantiated without running any of its code. This permits the binder to run as a preprocessor. The RESTART statement will be added to absorb the current uses of START without parameters. [Binder]

Module Instantiation - Copy

A form of NEW which copies the bindings from an existing module instance will greatly speed this operation in what is thought to be a common case. [WickUMI]

Packed Arrays

Since many applications cannot afford the storage overhead associated with strings, packing array components (at least with byte granularity) seems necessary. The PACKED attribute will be allowed in array declarations (automatic packing is potentially incompatable with existing software). [GeschkeS]

Compiler

Module Format

The format of an object module must be expanded to contain the information needed by the new binder. *This will require recompilation of all existing Mesa programs with the next release.* [Binder]

Program Parameters

The next release of the compiler will type check module parameters (now passed by START). Passing more than five parameters will also be supported. [Maybury]

Signal Descriptors

In the next release, signal values will be structured like procedure descriptors, with a module-unique signal index replacing the entry point number. Like procedure descriptors, signal descriptors can be constructed on the fly, eliminating the need to store the signal value in the global frame. [JohnssonMRI]

Source/Object Mapping

The table produced by the compiler which records source text/object code correspondence will be expanded to include the beginning of each source statement, rather than the beginning of each source line. This will allow finer contol over breakpoint setting. [SatterthwaiteMI, GeschkeMDI]

Static Link

Catch phrases and nested procedures could achieve a better instruction mix if their static links pointed to the first local variable of the enclosing context instead of to the beginning of that frame. [GeschkeMop]

Unwind

In the next release, UNWIND will be a predefined symbol, rather than an external signal. Binding to this symbol will no longer be required.

System Building

Binding

The current binder is being replaced by a preprocessor designed to be run before any loading takes place. It inputs a Configuration Description and combines all the modules it specifies into a single package -- a Binary Configuration Description or BCD -- that can be loaded as a unit. *To ease conversion to the new scheme, the old binder will be retained temporarily.* [Binder]

Loading

A new loader is required which understands Binary Configuration Descriptions. Eventually, it should be capable of loading (and unloading) an arbitrary number of BCDS into a running system. The initial implementation may be limited to a single user BCD. [Binder]

SubSystems

The procedures which save the state of a running system in an Image file must be redesigned to process BCDS. [Binder]

Version Checking

The increase in packaging options made possible with configurations requires more thorough version checking to insure that all of the BCDS, the runtime system, and the debugger are consistent. Version identification must be propagated not only among compiled modules, but BCDs and Image files as well. [Binder]

Architecture

Allocation Trap

The allocation trap microcode will be changed to pass a single parameter (the original destination), in order to avoid a possible stack overflow when a frame trap occurs while processing some other trap.

Byte Arrays

The present read and write byte instructions (RSTR and WSTR) have the Mesa string format built into them. To support packed arrays, this offset needs to be specified in an alpha instruction field. [GeschkeMop]

Interrupt Handling

The wakeup disable counter instructions of the Dstar (IWDC and DWDC) will be implemented in microcode. [JohnssonRHI, Thacker]

Procedure Descriptor Format

The next release of the system will contain a new procedure descriptor format (the global frame and entry point indexes will be interchanged, in anticipation of conversion to the Dstar instruction set). Complete compatability with the Ds may be included, depending on available microstore. [GeschkeMop, GeschkeEV, Thacker]

Unsigned Compare

To make room for other microcode changes, the unsigned compare instruction (USC) will be removed. A procedure which emulates it will be provided for compatability. [JohnssonRHI]

System

Code Segment Prefix

The format of the code segment prefix has been revised to include the offset and length of the linkage vector (including external signals) in the global frame. *This format restricts the number of global frame table entries per module to a maximum of four, and therefore the number of entry points per module to 128.* These restrictions are consistent with the Dstar design. [Thacker]

Deleting Modules

A kernal function which deleted a module instance will enable applications to move once-only code into separate modules, allowing the space to be reclaimed after initialization. Dangling references to deleted modules might cause a problem; a check could be made, but it involves swapping in every code segment in the system. [JohnssonMRI]

Disk Stream Extensions

A number of more convenient procedures have been proposed, including one which creates a stream from a file name instead of a file handle, as well as a procedure for backing up a stream index.

Free Storage Package

The free storage package has been modified to allow pruning the heap and returning unused storage to the system.

Global Frame Breakage

The packaging of several modules into a single configuration will allow the loader to allocate all the global frames in a single segment. Provided modules are combine into configurations prior to loading, most breakage will be eliminated. [JohnssonMRI]

Interrupt Handling

The inefficiency of disabling/enabling interrupts has been a major problem for the Mesa FTP and Juniper projects. The complexity of the interface (with the Nova emulator) has been a major cause of system bugs. This revision will implement these operations in microcode. [JohnssonRHI]

Kernal Function Calls

A number of proposals involve additions (and deletions) to the system dispatch table (see *Module Instantiation - Copy, Deleting Modules, Start Trap*).

Reorganized Interfaces

Because the new binding scheme requires all external procedure and signal definitions to be obtained from included modules, some reorganization of the system definitions files (mostly additions) was required.

Signaller Improvements

Because the signaller was one of the first modules implemented in Mesa, it makes rather poor use of the facilities which are now available. Several modifications have also been made to improve its efficiency.

Debugger

Global Frame Table

A command which lists all instances of all modules by examining the global frame table will be added.

Module Format

The debugger must be modified to accept the new module format, and perform some minimal processing on the BCDs loaded into the user's core image. *For compatability, the old owner link, binding entry, binding path, and symbol table pointer in the global frame will be retained, allowing the debugger to be converted gradually.* [Binder, Sweet]

Source/Object Mapping

The breakpoint facilities will be extended to make use of the additional statement boundaries now provided by the compiler's output. [GeschkeMDI]

String Display

A number of commands for displaying strings and arrays of characters will be added. Support for packed arrays must also be added.

Operations

Version Identification

In conjunction with the new release policy, a set of version numbers will be assigned to all released software.

Section II

The following changes will be included in Mesa 3.0 as resources permit. The relative priority of each item will be determined as feedback is received from the Mesa community. The release definition will be updated to reflect the decisions reached.

Language

Included Identifiers

This proposal would optionally allow a list of identifiers to be included from each module in the DIRECTORY statement. If the list is present, any attempt to reference an identifier from that module which is not in the list would generate an error. [LampsonMI]

Machine Dependent Records

Euclid's notation for defining machine dependent records (by specifying bit positions) allows more control over the placement of fields than Mesa's positional notation. This technique would also remove the restrictions on the location of variant record tags. In any case, the compiler should check all forms of machine dependent records for empty holes and issue an error. [LampsonE]

Mutable Variant Records

Several breeches of the type system could be caught by the compiler if it knew whether the type of a variant record were constant over its lifetime, or whether it could change "on the fly". The VARYING attribute has been proposed to indicate the latter. [WickLWG, SatterthwaiteVRC]

Overlayed Variant Records

The constructions for referencing computed variant records could be simplified considerably by allowing access without descrimination to those fields which have unique names. [SimonyiMR]

String Bodies

Some housecleaning on STRINGS is needed, especially if sequences are implemented; at the least, a name should be attached to the string body and text field. The body name could also be used to make clear the pointerness of strings during initialization, a common source of trouble among users. [GeschkeS]

Compiler

Compiling Options

A number of compiling options have been proposed (optimization, cross reference data, warning messages); some user interface for specifying options needs to be designed. The primitive Bcpl switch mechanism could be used. Some options might be included in the source text.

Constant Initialization

In general, constants are copied from the code segment when needed, often using block transfer instructions. As users make more use of processes, the inline code will have to guard against preemption. [SweetMLC]

Cross Reference Data

The routines which produce the cross reference dribble file need to be incorporated into the early passes of the compiler. Some way of enabling this option needs to be specified.

Variant Record Packing

The current algorithm for packing fields in variant records can leave unused bits in some variants, forcing the compiler to disallow comparisons (the unused bits contain garbage). Removing the holes would make some variants one word longer. [SatterthwaiteVRC]

Warning Messages

A number of potential errors could be recognized by the compiler and translated into warning messages ("variable not referenced", for example). Some compiler option is needed to specify whether these messages should be ignored or printed. Each condition needs to be analysed carefully; warnings do no good if they are routinely ignored. [SweetMLC]

Architecture

New Procedure Descriptor Format

A change in the encoding of unbound procedure descriptors would allow a more attractive tradeoff between the maximum number of modules (in an MDS) and the maximum number of entry points per module. [LampsonOISP, GeschkeEV]

System

BitBlt

Now that all Altos have been converted to version 23 microcode, support for software BitBlt can be removed from the system.

Executive User Interface

If the Alto Executive could be extended to load Mesa image files, the user interface for invoking Mesa could be simplified, and a command file processing package could be provided which is consistent with the current Alto conventions.

Start Trap

To avoid a flurry of code swapping (to run initialization code) when systems are first loaded, a trap could be generated when a call to a module which has not been initalized is attempted. A kernal function could clean things up and retry the transfer. [Binder]

Debugger

Configuration Descriptions

A number of debugger commands need to be rethought and reimplemented based on the information now contained in configuration descriptions. The debugger must deal with several new data structures built by the binder and the loader. [GeschkeMDI, Koalkin]

References

Binder Working Group. *Binding in Mesa.* February 18, 1977.
Geschke. *Mesa debugger issues.* January 10, 1977.
_______. *Sequences.* January 28, 1977.
Geschke, Johnsson. *Modifications to Mesa opcodes.* February 28, 1977.
Geschke, Johnsson, Sweet. *Enrty vectors and control links.* March 25, 1977.
Johnsson. *Mesa runtime issues.* January 11, 1977.
_______. *Revised handling of interrupts in Alto/Mesa.* May 26, 1977.
Lampson. *Mesa issues.* January 6, 1977.
_______. *Proposed OIS processor changes.* January 14, 1977.
Lampson, et all. *Report on the Programming Language Euclid.* July, 1976.
Maybury, Mitchell. *Mesa Language Manual.* October 1, 1976.
Satterthwaite. *Mesa issues.* January 10, 1977.
Satterthwaite, Wick. *Variant record changes.* July 7, 1977.
Simonyi. *Mesa requests.* February 18, 1977.
Sweet. *Binder working group minutes.* January-February, 1977.
_______. *Mesa language committee minutes.* July 6, 1977.
Thacker. *OIS Processor Principles of Operation.* April, 1977.
Wick. *Unresolved Mesa issues -- binding.* January 11, 1977.
_______. *Mesa language working group minutes.* January-February, 1977.

Distribution

SD Managers
Mesa Users
Mesa Group