A Close Glimpse To MorphOS

.:: Part 1 - System ::.

Note: This document is non-official and highly outdated at the date of availability (04-Feb-2010)!

by
Ulrich Beckers - Genesi SARL Luxembourg

Editors note:
This is the first issue in a series of documents describing the operating system MorphOS. It may give you a first impression about the system - its origin, its architcture, its fascination. This is not a handbook of the system but may help to become familiar with the wonderful world of MorphOS.

Table of content

Important note:

This document is outdated by date of availability in 2010. Hence this version is marked as V1.0SE. It is a rather raw dump of a working draft from 2004/5. The information provided here may be outdated and partially wrong. A few added notes are marked red and italic. Please read the important note at page 15!

Introduction

MorphOS is an entire new operating system for PowerPC microprocessors. Its primary host system are members of the Pegasos PPC computer family by Genesi SARL, Luxembourg.
First versions ran on PPC accelerated Amiga computer systems and it has been tested also on other PowerPC based computers which are run on a IEEE 1275-1996 compatible firmware. Further development is closely coupled to the Open Power Architecture, (OPA - an enhanced open firmware system). Currently it is publically available for the Pegasos and Phase5 PowerUp ppc accelerator cards for Commodore Amiga computers. The full version is publically available free of charge but requires registering, current release is version 1.4.5[1].
There are also versions available for the MCP5200 based Freescale mobileGT board as well as for Genesi's Efika5k2 board, these versions are not public yet.

The operating system is working very efficient: it is fast and has a high responsiveness; it is very scaleable and does not need huge amounts of RAM or big harddrives (in fact it is easy to boot up a fully usable system from CD or an EPROM solution of only a few megabytes) nor wastes too many cpu cycles. The system has an intuitive GUI system build in, but offers also a powerful shell for keyboard control. It is perfectly suited for multimedia applications or gaming, but also for sophisticated office solutions like DTP systems. From the very beginning it supports preemtive multitasking and the foundation of the OS is ready for symmetrical multiprocessing.

MorphOS supports PowerPC processors up to the fourth generation (e.g. Freescale MCP 7447(A/B) or 7448, publically better known as G4) as well as embedded PowerPC processors. The Altivec velocity engine adds an extra benefit to the system as well (upcoming release). Support for 64 bit PowerPC processors (e.g. IBM 970) is easy to do since MorphOS has always been designed to run on 64 bit systems as well.

History

The MorphOS project was started in 1999 by Ralph Schmidt who is still the main owner and first maintainer of MorphOS. In 1999 Ralph Schmidt worked for the German hardware company Phase5 (bankrupt in 2000). Phase5 was specialized in expansion hardware for Amiga computers and Apple MacIntosh computers. In 1996 Phase5 brought the first and only PowerPC acceleration board for Commodore/Escom Amiga computers to the market. After developing the initial PowerPC support for AmigaOS, Ralph Schmidt started his own operating system project – MorphOS - because the official support for the PPC

1 *Note: With MorphOS V2.0 major changes were introduced. Current release is V2.4.*

with the AmigaOS was only poor and then came to complete stagnation.
The intention of MorphOS was to keep the good parts and ideas of the AmigaOS architecture but also build a new and modern operating system, which does not suffer from the limitations of the old AmigaOS.
The name MorphOS is a witness of the background it stems from. This operating system should 'morph' the good aspects of the old AmigaOS from the last century to a really modern operating system design for the 21st century.
The blue butterfly logo shows a tropical *Morpho menelaus*. It was mainly chosen because of the name 'Morpho' but also because butterflies undergo a strong metamorphosis and because of beauty and symphathy - all features which are also true for MorphOS.

The currently (September 2005) most recent public version of MorphOS is V1.4.5 from May 2005 for the Pegasos and V1.4.5 from August 2005 for Phase5 PowerUp boards[2].

Architecture

The foundation of the system is the microkernel Quark. This microkernel is the host for different operating system boxes. Currently there are two boxes available: the ABox which is quite matured already and the QBox which is still in a beginnng phase but is planed to improve later on. The transition towards the QBox will be very seamless. The system is open to host more boxes, but it is unlikely that these will appear in a not too distant future. Before we go into the details let us take a brief view to the differences between microkernel and monolithic kernel systems.

Microkernel & PPC

With microkernels and monolithic kernels it is similar like it is with RISC and CISC microprocessors. Both have advantages and disadvantages and it is not always easy to distinguish between them. With microkernels and monolithic kernels it is like that (in fact there are several hybrid kernel based systems like the WindowsNT based ones which started as a microkernel but added several services into the kernel space). While most common desktop systems, like Linux, Windows or OS X, belong to the group of the monolithic (the latter both are hybrid ones) based systems, the microkernel systems are rather rare here. A monolithic kernel incorporates several services like the graphic or network subsystem within the kernel (or kernel space) while a microkernel only mangages the very root services like task and address scheduler. A microkernel based system is more scaleable (only services which are needed at a given time are running) and solid (e.g. kernel threads cannot overwrite each other's address space) and offers very quick response times. But microkernel systems

2 *Note: With MorphOS V2.0 major changes were introduced. Current release is V2.4.*

need a lot of interprocess communication (IPC) and thus, many context switches. In the terms of computing power these are expensive and lead to speed penalties on many systems. To avoid these speed penalties many operating systems are monolithic or hybrid kernel based but coevally gain the tendency to become bloated with a decreasing responsiveness and high risks for kernel errors (e.g. “blue screens“ on MS Windows based maschines).
How can MorphOS be based on a microkernel but keep being that fast with this extraordinary high responsiveness? Much of this is due to the fact that it runs on PowerPC processors which are suited better for frequent context switches because of the availability of many registers. It comes close to never to situations that a register content must be saved to the stack. It can be assumed that context switches on the PowerPC architecture are several times faster than they are on the IA-32 architecture.

How is that realized? In the following paragraphs we will learn something about the basic architecture of MorphOS.

Quark

Quark is the foundation of MorphOS. It is a newly designed microkernel which starts the system just after the hardware initialization. Quark is very minimalistic but provides memory protection (MP), resource tracking and symmetric multiprocessing (SMP). A hardware abstraction layer (HAL) offers a common interface to different hardwares. Among the services which get started by Quark are the MasterClanServer, AddressServer, ConfigServer, CPUTime-Server and SystemInit.

As mentioned already a major feature of MorphOS is the OS box approach. SystemInit launches the box threads, currently this is mainly the ABox task.

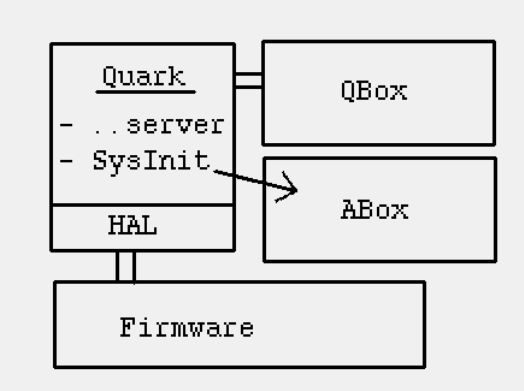

Fig. 1: *Scheme of MorpOS's architecture. On top of the common kernel Quark several OS-boxes are launched.*

ABox

The ABox is the most prominent part of MorphOS currently and it is that part which is commonly referred to as 'MorphOS' (but one should always keep in mind that MorphOS is more than the ABox only - this document describes it).
The ABox is an entire operating system within MorphOS. It offers all services which are required for a today's desktop system. The system design includes a complete PowerPC code based reimplementation of the Amiga API V 3.1 but the OS is heaviliy enhanced and optimized. For legacy compatibility to existing Amiga applications a very fast emulator of the Motorola MC680x0 processor family is included. Even the system is enhanced a lot, there are some limitations for this box due to compatibility reasons - there is no multiuser system and only a limited memory protection included within this box. But remember, these limitation are only valid within the ABox. For future enhancements of MorphOS there is the QBox which will be briefly covered within the next part.

QBox[3]

The QBox is the future part of MorphOS. The QBox is not that sharp to segregate from the Quark kernel as the ABox is. For further development of the QBox the Quark kernel will be enhanced. Put together, Quark and the QBox arise the 'Q' environment. Q will show a lot of similarities to the ABox, several ABox parts were developed with the possibility for a future transition to Q. Thus, the transition to Q will be quite seamless and for a user it will not always be obvious if an application is calling the ABox or Q.
MorphOS using Q will not have those limitations the ABox has and offer much more. The details will have to be evaluated still. First steps will be to put the drivers from the ABox to Q.
But for all MorphOS V1.x versions the development is focused to the ABox which is covered in detail now.

ABox

As mentioned already the ABox is that part of MorphOS a user or programmer currently has to deal with. Now we take a closer look to the ABox.

While booting

After a boot up time of only a few seconds the system is ready to use and one is welcomed by *Ambient* which is the graphical desktop of the system. While the OS is very intuitive to use and easy to understand one might like to know how the system operates behind the scenes.

3 *Note: Currently the QBox is rather on hold and will - if at all - probably not implemented as described here. The box approach may help to migrate to other processor architectures though.*

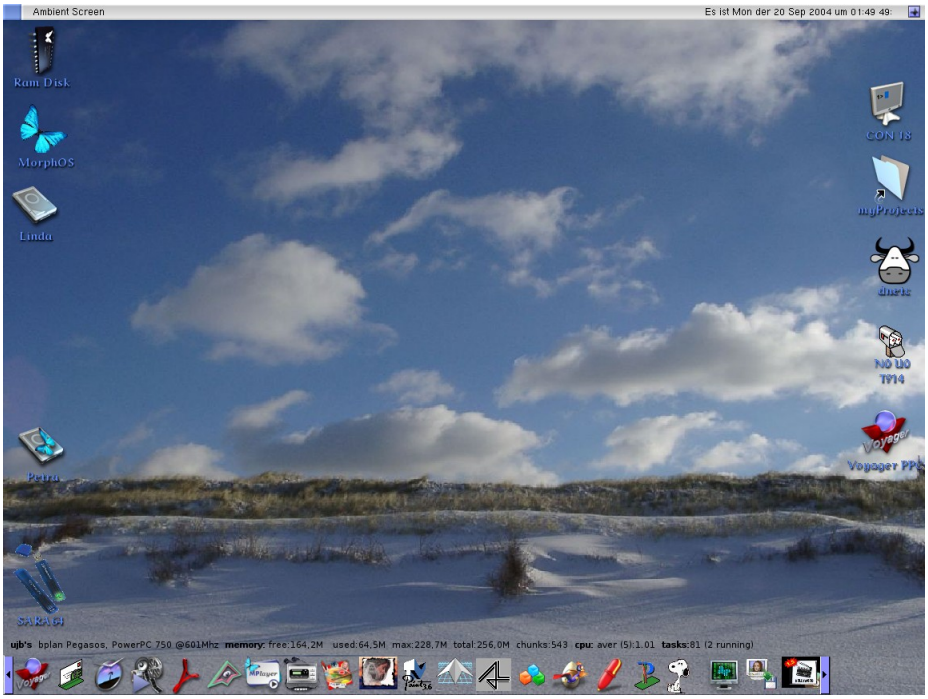

Fig. 2: *The MorphOS desktop 'Ambient' right after booting on an average setup. Here on the left side there are icons for all disk drives present, on the right side some application icons are located while on the bottom an application starter panel resides.*

While booting the ABox the first compound which is executed is the ABox kernel called *exec.* All ABox tasks are scheduled by *exec.* It offers full preemtive multitasking and very fast response times. The design of the ABox is microkernel like. Thus, the core system is not only consisting of *exec* - the system is highly moduar designed and consists of a set of shared libraries for system calls. Beside *exec.library* there are also other core compounds like the *intuition.library* (window and screen operations), *graphics.library* (graphic primitives) or *dos.library* (disk operating system) to name a few of the most important libraries. The basic system install includes some dozens of libraries to provide a vast set of system functions. It is also easy to enhance functionality by adding custom libraries to the system. To avoid system instability and increase clarity the original system files are capsulated, but are never hidden to the user.
When looking to the screen shot in figure 2 one can see icons for several volumes which are present at that system (the icons on the left hand). MorphOS (i.e. the ABox, Quark as well) works device orientated. Several devices are known to MorphOS. A device is basically a sender or receiver of data. It may be either a physical one like a harddisk partition or usb device but also a logical one like a pipe or an assigned destination. The name scheme for devices is *device:* (some name followed by a colon). There are units (e.g. a cd drive) which are named by the system, but still the volumes of these units do have individual names. To keep the cd drive example, the drive may be named *CD0:*, the inserted disc may be named myRandomCD. The according name to access that

specific volume is myRandom*CD: then*. If you access to *CD0:* you get access to the volume currently inserted into the drive, if you access a specific volume the system is looking if this volume is available somewhere (current location of that volume does not matter). While booting, the system generates a ram disk of dynamically scaled size which is usable like every other volume.
A very powerful feature to mention is the assign functionality. By using assigns one can expand and control the search path for volumes. A volume can not only be mounted to a device, but also to some given location. It is possible to specify multiple search pathes for one volume, add and remove entries - all this while actually running the system. The system adds several of these logical volumes while booting, for example the root directory of the system SYS: is assigned to the volume the ABox starts booting from. The assign is very powerful and offers much more than the hardlink/softlink concept of several other operating systems do.

The shell

Best to understand the concept of devices and volumes is by using MorphOS's shell (see figure 3). The shell in fully integrated into the graphical environment of MorphOS, it is not a seperate layer as it is in several other operating systems.

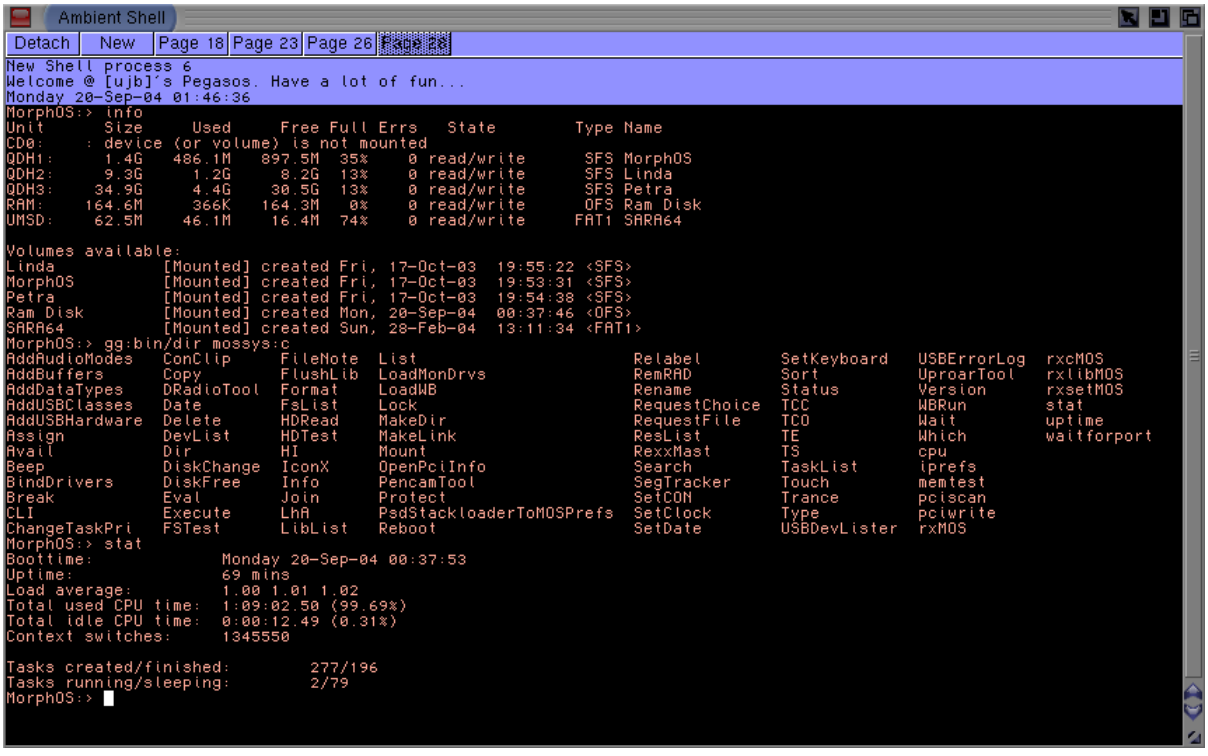

Fig. 3: *The shell showing some information about the system .*

It is in many regards similar to some shell systems known from the *nix world, but has its own set of commands and style. It is very powerful but also quite easy to conrol. The shell supports wildcards, dialogues, pipes, scripting,

command history, logging, tab command completition, control-keys and escape sequences; multiple instances of a shell can be started (even within one single window).
One does not need the shell to control the system, but if working with the shell it will be loved for its power, options and clear structure. A detailed introduction to the Shell and MorphOS DOS is to obtain from a coming publication.

Datatypes

Another useful feature are the *datatypes*. By datatypes the system provides global routines to read and/or write file formats. Applications which utilize the datatype concept are able to handle all fileformats the system provides datatypes for. It is to emphasize that the system is easily expandable with additional datatypes. Thus, applications using them gain more usability and will stay future compatible even if an application is not maintained any longer.

Icon system

The icon system is very transparent to the user. All icon information is saved within a file with the same name but extended by '.info'. An icon may be any png file, but MOS can also handle several legacy icon file types known from the AmigaOS. The '.info' files consist of graphical information and additional information. This may contain information like the place where the icon should appear or about a default application to launch with that file (e.g. a divx movie file might contain information which very movie player it likes to open). The settings stored here are not global but file specific.

a)

b)

Fig. 4: *a) Window showing properties associated to a specific icon b) Different icons from MorphOS - top left shows a (legacy) 'glow icon' (selected mode), bottom left a (legacy) 'New Icon', both other icons are MorphOS native png icons.*

In the upcoming version of MorphOS there will be an automatic file type recognition[4]. It is based on mime type recognition and not on file name extensions. The system will automatically add a predefined action (e.g open a picture viewer if clicked on a picture file) to a file which has no individual icon information available.

Screens and windows

MorphOS's graphical subsystem *intuition* is screen and window orientated. Screens are the higher order and the system can open as many screens as the available free memory allows. Every screen can have its own preferences in resolution and visual appearance. Switching between screens is very fast and is handled by a depth gadget and/or key shortcuts. To contribute a better overview there is also a thumbnail view of all open screens available.

Every window resides on a screen, many applications allow to open their windows on a defined screen and even can jump to any open screen during runtime. Windows are freely adjustable in depth, i.e. an active window can stay in the background or come to front, just as one likes. Windows can be moved out of the viewable area of a screen. There is opaque (i.e. window is redrawn in real time) moving and resizing for windows available.

Fig. 5: *Application running on a custom screen of low resolution. One of the application windows is moved out of the screen area. The screen is skinned individually.*

Preferences

The system settings are managed by a central preferences interface. Settings are numerous - from screen settings, usb settings or printers to time, localization or network and more. Hotkeys for any possible purpose may be defined within the preferences.

4 *Note: The mime type recognition has been added to the system for a long while now.*

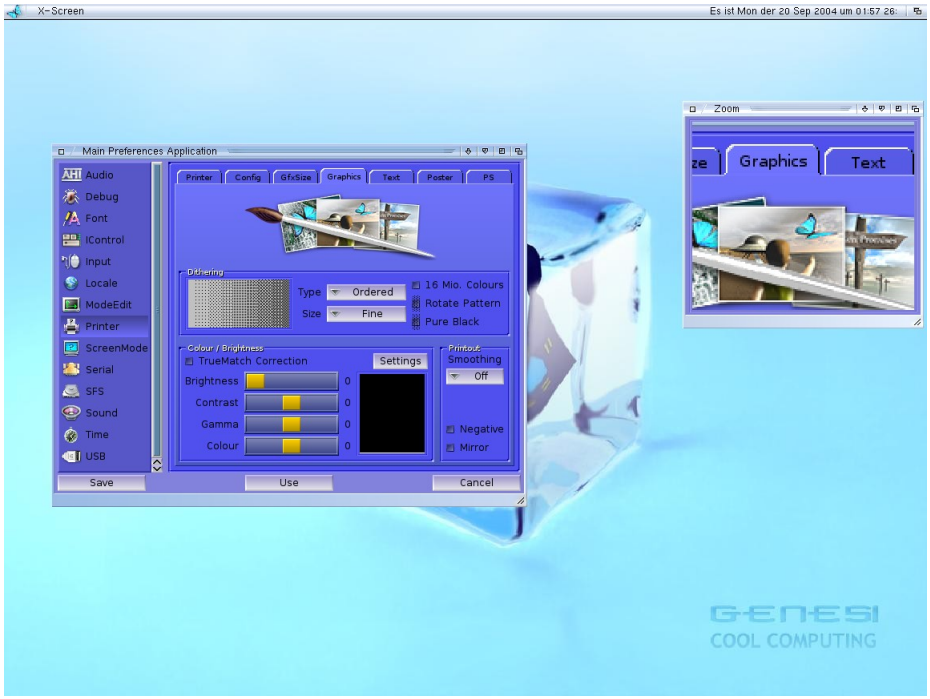

Fig. 6: *System preferences residing on some own screen. Also the zoom tool, a reatime virtual magnifying glass, is shown.*

Appearance of any screen can be skined and changed on the fly. Nearly all settings can be modified instantaneous without the need of a reboot, only very few settings cannot be changed realtime. There are also global preferences for the Ambient desktop and for the magic user interface (MUI).Multiple preferences profiles may be saved.

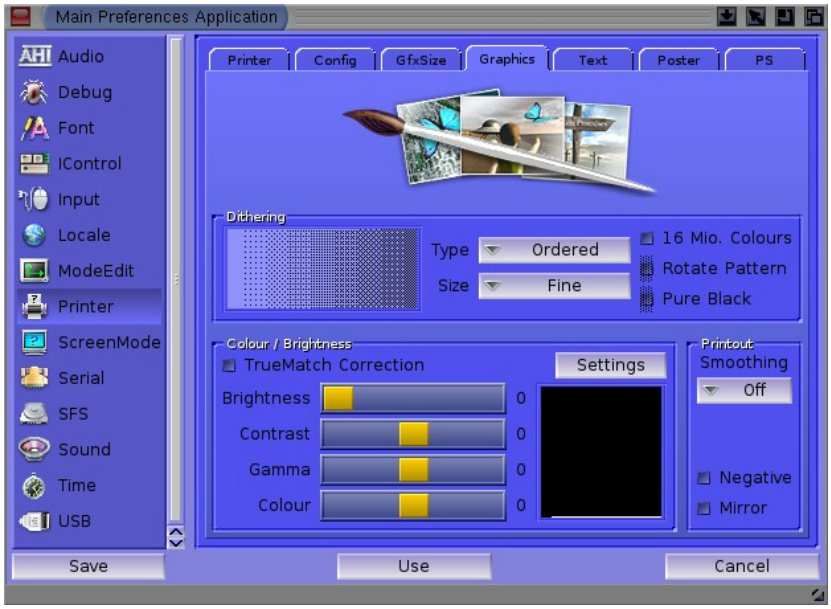

Fig. 7: *Same preferences window as shown in fig. 6, but with a different skin after real time jump to the Ambient screen (screen not shown here, see figure 1)*

Magic User Interface (MUI)

The magic user interface (MUI) is the original application tool kit for MorphOS. Most applications are using MUI right now and the MorphOS style guides

recommend to develop applications using MUI. It offers many build in common classes and saves a lot of programming work while writing applications. It is also possible to add new (external) custom classes which are public to all applications using MUI.
For the user it provides an unreached freedom to customize every application. MUI applications are fully skinable, iconifiable, can jump to any available screen and much more. It also helps getting familiar with every new application by providing a integrative UI. All settings are easy to maintain by a common preferences interface. Every application can have its private MUI settings, but one can also define global MUI settings which are used whenever there is not an individual setting available.

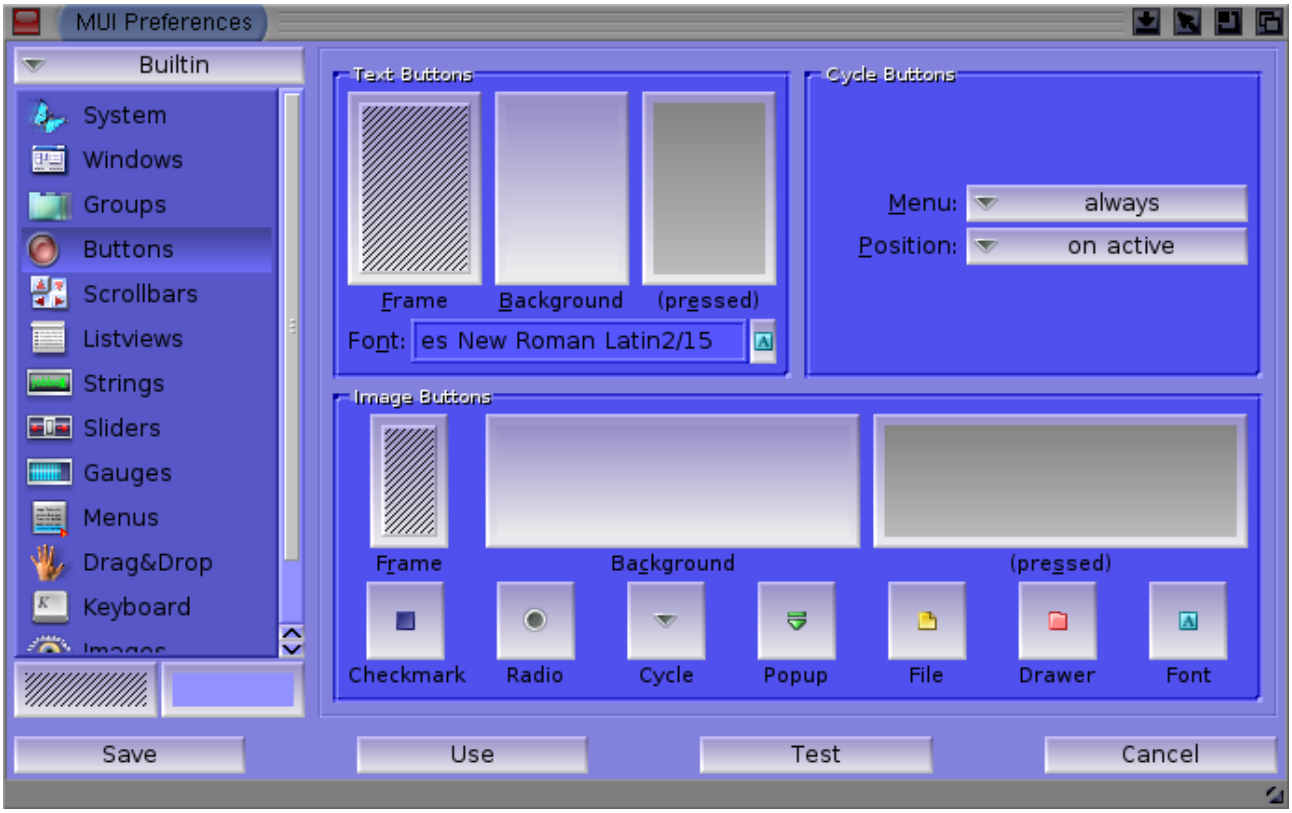

Fig. 8: *Snapshot of the MUI preferences. Every application may have its own preferences. This window shows the global MUI settings.*

Ambient

Ambient is the desktop system of MorphOS. Ambient was put under the GNU public license (GPL) and gets continuously developed since then. For more information about the Ambient GPL project please visit:
http://sourceforge.net/projects/morphosambient/

Ambient is a multithreaded application and based on MUI and thus, very customizable. There are several icon based file operaion options, the right

mouse button offers a context sensitive menu for available operations.
An integrated panel is used as a rapid application starter, the number of panels is unlimited. With *exchange* one can manage running and hidden applications.

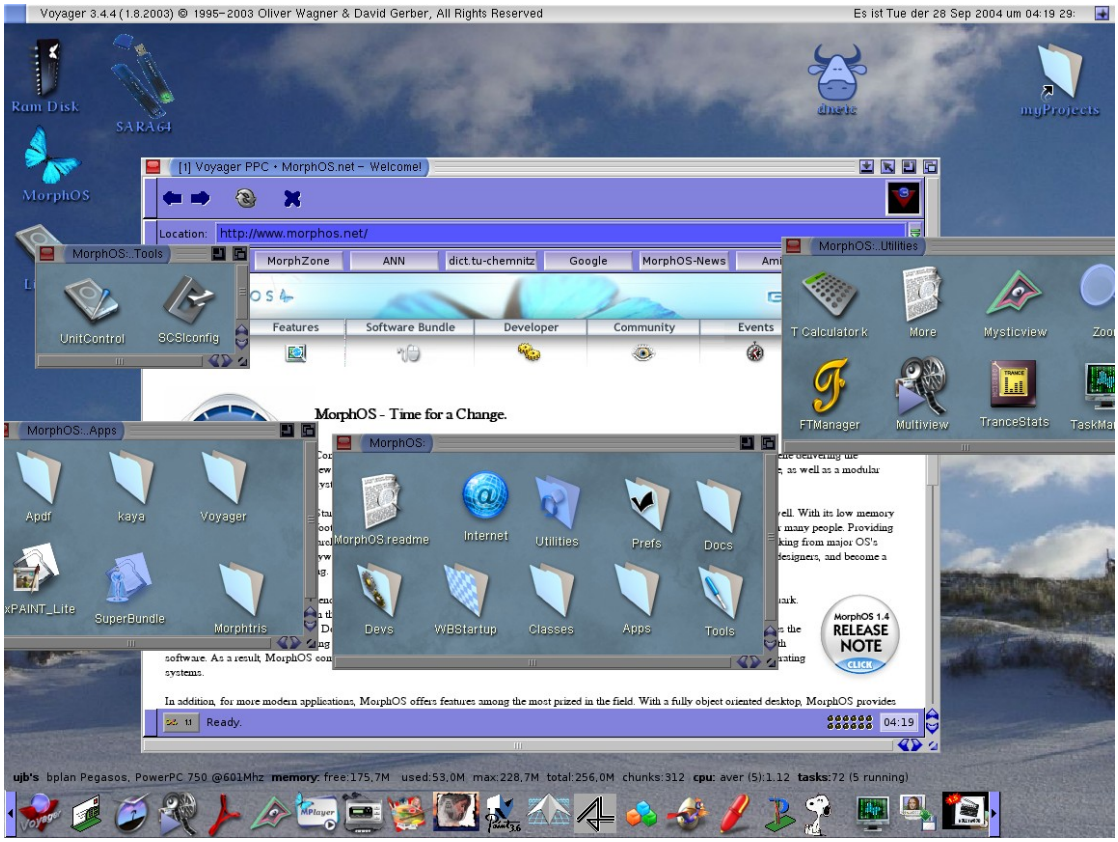

Fig. 9: *Ambient screen with several open windows. Note that the browser window in the background is the active window here.*

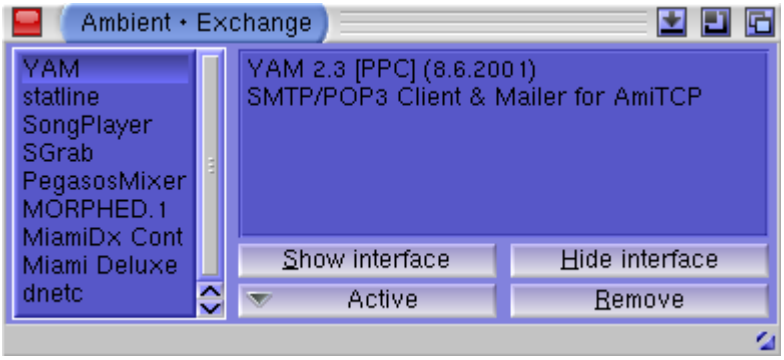

Fig. 10: *The tool 'exchange' for application control*

System utilities and tools

There is a set of utilies and tools coming along with MorphOS. This includes several viewer programms like the multifunction viewer *Multiview* (uses datatypes), the slide show programme *Mystic view*, the text viewer *more*, or the

pdf viewer *Apdf*. There is also the mp3/ogg player *Kaya* or tools like a calculator, a zoom, a task or font manager included. There are several more tools and utilies, some may be changed or added with the next release[5].

ARexx

ARexx is a mighty scripting language, but also a whole programming language. Even for beginners the syntax is easy to understand. It is possible to do quick custom solutions with ARexx as well as more sophisticated applications. For every day use the most important benefit of ARexx is the really powerful scripting capability. Most applications offer the facility to be be controlled by ARexx commands or to trigger some ARexx events. Thus, automated and complex (or simple) interactions of independent applications become possible. The ARexx implementaion is not finished yet and will be completed with a later release.

GeekGadgets and SDL

The GeekGadgets are an optional package which enhance the system with a vast compatibility to the POSIX standard. There are many POSIX applications which utilize the GeekGadgets environment, such as the famous GNU C/C++ compiler GCC/G++. For many applications adopted from the *nix world it is not necessary to host the entire GeekGadgets set, but only the *ixemul.library* which is part of the standard system installation.
The GeekGadgets and development tools are described in another publication of this series.
The famous SDL libraries (including hardware accelerated gfx (3D) support) are available for MorphOS. Many SDL based applications need only a simple recompilation to run on MorphOS.

Legacy support

The ABox API is compatible with the AmigaOS 3.1 API. With the included fast JIT 68K emulator most Amiga applications can be run on the Pegasos with MorphOS. Those applications which are directly using the old Commodore Amiga custom chips will not work. These are mostly very old applications and games that can be run with the help of the Amiga emulator UAE which is also available for MorphOS. The overall compatibility is surprisingly high and will be even higher with the upcoming version of MorphOS. MorphOS is the only solution which is able to run not only the ABox native binaries but also Amiga 68k binaries as well as Amiga PowerPC binaries using the PowerUp or WarpOS enhancements. There is no speed penalty for the latter both, the 68k emulation has an average speed loss of about 1/3 to 1/2 (depends on application, etc.) to native comparable versions.

5 *Note: The collection of tools and utilities, as well as other provided applications underwent serious changes since release V1.4.5.*

Personal notes

This page is intended for your personal notes.

Addional notes

All trademarks used in this document are property of their current owners.
No warranty is given.
For questions or assistance regarding this publication, do not hesitate to contact the the author by email to ubeckers@de.genesi.lu.
Further information regarding MorphOS is obtainable at www.morphosppc.com and with coming additional publications.

Document revision history

Version	***Date***	***Changes***
V1.0	01. Oct. 2004	initial version
V1.0a	28. Jan. 2005	minor changes
V1.0 SE	04. Feb. 2010	special edition of V1.0a with a few recent notes.

Important note:

This document is outdated by the date of availability in 2010. Hence this version is marked as V1.0SE. It is a rather raw dump of a working draft from 2004/5 that never got released publically and/or expanded back then for several reasons. I recently found it within some back up files of an old laptop and eventually decided to make it publically available. Note though, that the information provided here is outdated and may be partially wrong. A few added notes are marked red and italic.