# Pegasus II Review, December 1997
# The Nemesis OS Demonstration

Ian Pratt, University of Cambridge

December 2, 1997

## 1 Nemesis on Intel x86 Architecture Systems

The machine used for the demonstration is a uni-processor Pentium II system, equipped with a Matrox Millennium II framebuffer, Digital DE500 100Mb/s Ethernet card, Digital ATMWorks 750 155Mb/s ATM interface, and an NCR810 SCSI-2 controller. Nemesis also supports multi-processor systems, and a variety of other framebuffer and network interface cards.

## 2 Applications

Figure 1 is a screen dump captured from a Nemesis system. The following sections describe the various applications that are running.

### 2.1 QoS Control Panel

*QoSBars* provides a graphical display of applications' resource usage, and allows the resource reservations made to them to be controlled. As well as the CPU resource, *QoSBars* can be used to control other system resources, such as disk and network bandwidth.

However, *QoSBars* is not intended for use by end-users; it is simply a temporary 'engineering interface'. Work is underway to develop a *User Agent* that will be responsible for sharing out system resources amongst applications in accordance with a user prescribed policy.

### 2.2 The Nemesis Window Manager

The Nemesis Window Manager provides a user-interface to the $\mathcal{WS}$ Window System. It provides functionality similar to most X11 window managers, allowing users to manipulate application windows, and perform operations such as move, kill, iconify etc.

## 2.3 WSTERM and Shell

WSTERMs are similar to X11 XTerms. They typically execute a shell, enabling users to launch applications, manipulate environment variables, and perform job control.

The shell allows commands to be passed to a CLANGER interpreter. CLANGER is a powerful language, enabling users to explore the Nemesis name space, and examine and modify system parameters (providing they have suitable privileges to do so).

## 2.4 Animation application

This is a simple 3D animation that demonstrates Nemesis' OpenGL compatible rendering library.

## 2.5 M-JPEG Player application

*JDA* is used to playback pre-recorded M-JPEG video sequences. The host CPU is used to perform the computationally intensive task of decoding and displaying the video images. In this demonstration, two *JDA* applications are running, playing different movies stored on the same SCSI disk. The Nemesis disk device driver ensures that the two applications receive the disk bandwidth they require regardless of the actions of other applications.

*JDA* is an adaptive application, capable of degrading the output image quality and frame rate to best utilize the resources available to it. This is achieved while maintaining the real-time playback rate of the original video sequence.

## 2.6 Network Video application

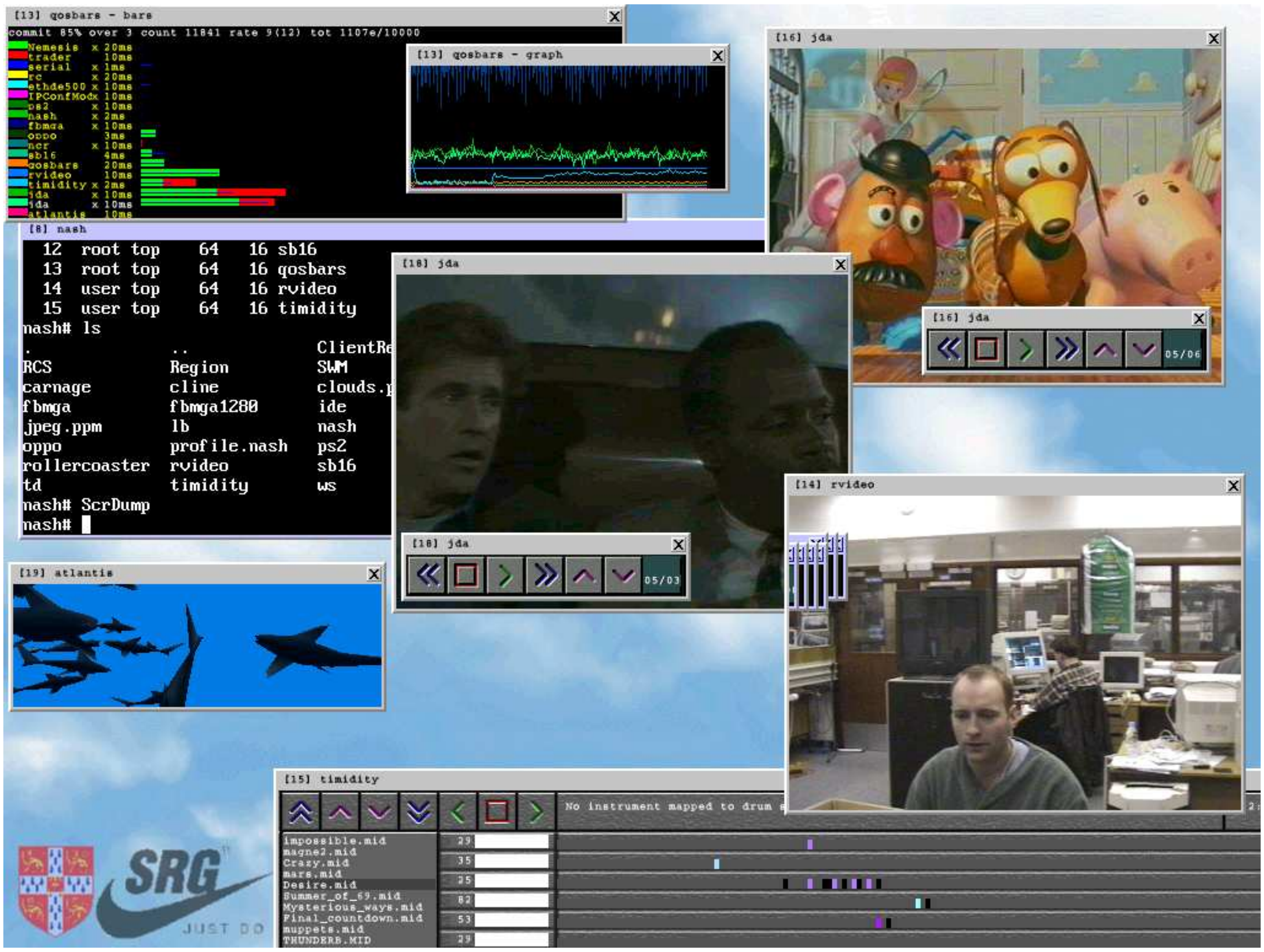
The *RVideo* application displays video data received from the network. In this demonstration, the video is a 384x288 25fps stream (roughly equivalent to Super-VHS video quality) generated by a FORE Systems AVA-300. The 45Mb/s ATM video stream is received by the host using the Digital ATMWorks 750 interface.

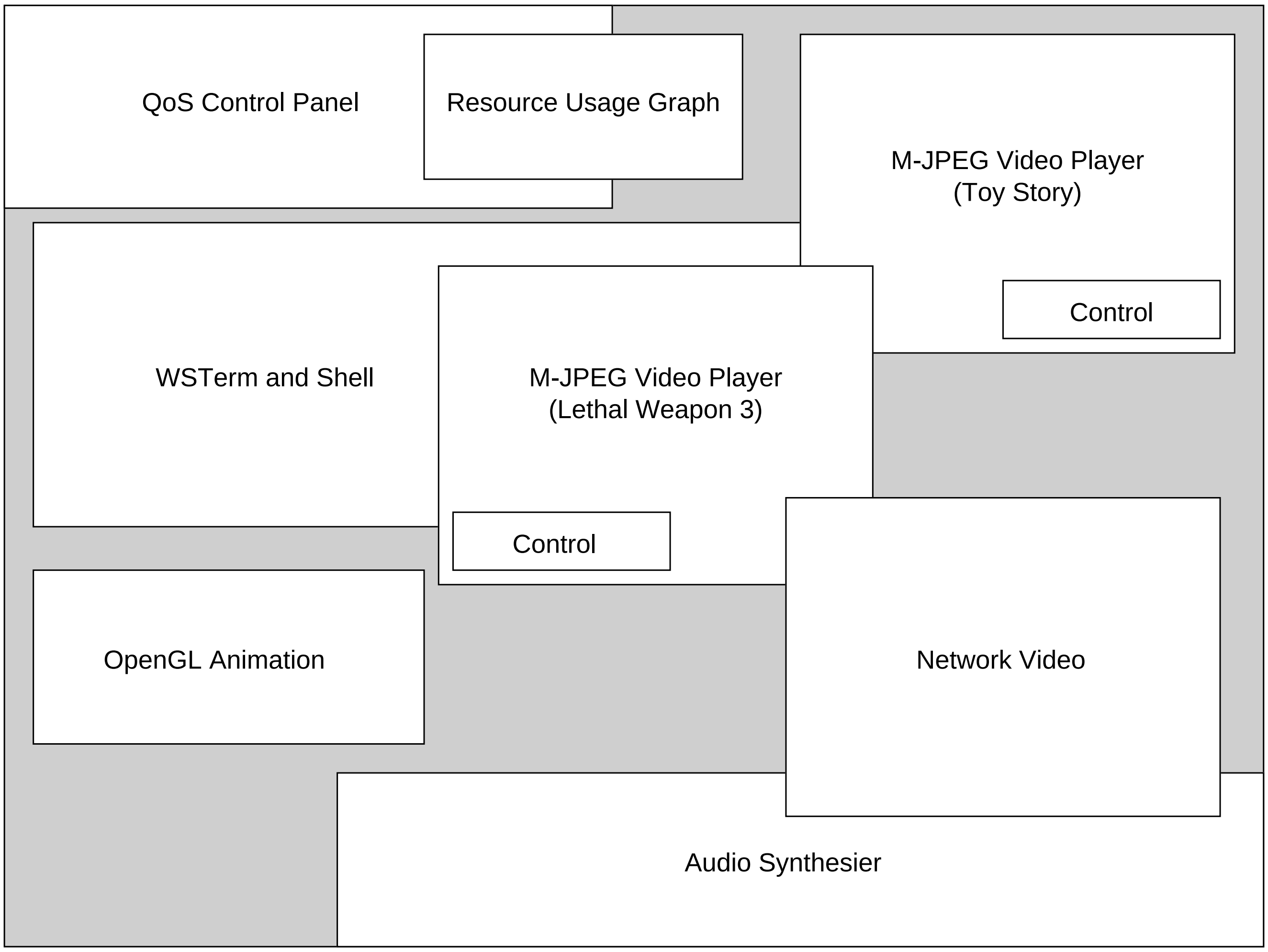
Currently, RVideo makes no attempt to adapt its resource requirements, and thus the video quality degrades rapidly if it is not delivered the resources it requires.

## 2.7 Audio Synthesizer

*Timidity* is a digital audio synthesiser. It plays standard MIDI files, using the host CPU to synthesize a CD quality audio stream containing a number of different instruments and effects.

Figure 1: Screen dump of demo and explanatory legend [following page]


[13] qosbars - bars
commit 85% over 3 count 11841 rate 9(12) tot 1107e/10000
[13] qosbars - graph
[16] jda
[8] nash
12 root top 64 16 sb16
13 root top 64 16 qosbars
14 user top 64 16 rvideo
15 user top 64 16 timidity
nash# ls
. .. ClientRe
RCS Region SWM
carnage cline clouds.p
fbmga fbmga1280 ide
jpeg.ppm lb nash
oppo profile.nash ps2
rollercoaster rvideo sb16
td timidity ws
nash# ScrDump
nash#
[18] jda
[16] jda
05/06
[18] jda
05/03
[14] rvideo
[19] atlantis
[15] timidity
No instrument mapped to drum
impossible.mid
magne2.mid
Crazy.mid
mars.mid
Desire.mid
Summer_of_69.mid
Mysterious_ways.mid
Final_countdown.mid
muppets.mid
THUNDERB.MID
SRG
JUST DO


QoS Control Panel
Resource Usage Graph
M-JPEG Video Player
(Toy Story)
Control
WSTerm and Shell
M-JPEG Video Player
(Lethal Weapon 3)
Control
OpenGL Animation
Network Video
Audio Synthesier

Timidity has significant real-time constraints. It communicates with the audio device driver via a 1ms 'play-out' buffer,[1] so must receive the CPU resources it requires at quite a smooth granularity. The level of resource required changes dynamically, depending on the types of instruments and effects that are active within the current musical passage—hence implying a need for relatively frequent resource re-negotiation.

## 3 Points to Note

- The Nemesis OS is now fully functional on Intel x86 architecture platforms. Although not ideal for running Nemesis, a relatively efficient implementation has been achieved.
- Despite the large quantity of I/O in progress on the demonstration system (network, disk, video and audio), only minimal resources are consumed by system domains such as device drivers: Most resources consumed during I/O operations are successfully accounted to the application responsible.
- Nemesis attempts to provide applications with QoS guarantees for all resources—not just the CPU. *QoSBars* can also be used to monitor and make guarantees for other resources, such as disk and network bandwidth.
- Providing applications have been guaranteed the resources they require, they will be unaffected by the actions of other applications. For example, using the window manager to pick-up and move a window will not starve CPU from other applications, hence the output quality of the audio synthesizer will be maintained.
- Although Nemesis allows applications low-level access to I/O devices, this is achieved without sacrificing the protection provided by conventional operating systems. For example, the Nemesis window system ensures that all rendering operations are clipped to the boundaries of the application window.
- Applications running under Nemesis can know what resources will be available to them, thus enabling them to make appropriate application specific fidelity tradeoffs. Such adaption is is demonstrated by the M-JPEG video player applications.

[1] Timidity could choose to operate with a larger play-out buffer, hence relaxing its real-time constraints. However, many similar applications operating on 'live' (latency sensitive) data do not have this luxury, hence Timidity demonstrates a useful property of Nemesis.