ARWINSS

The new Windows subsystem for ReactOS / Windows

Outline

- Existing Win32 subsystem overview
  - History
  - Advantages
  - Disadvantages and problems
- Introducing “Win32 subsystem v2.0”
  - Why a new version?
  - The beginning
  - How was it made
- Architecture explained
  - Brief overview
  - The big picture
  - The absence of X Windows dependency
  - The role of Wine
- Arwinss and ReactOS
  - Why is it so important?
  - Benefits
  - Work sharing
- Future of Arwinss
  - Composite desktop support
- Further Information
- Screenshots

Win32 subsystem in trunk. History.

- Appeared as early as in 20th of May, 1999 (win32k), committed by Rex, with early work of Emanuele Aliberti, Eric Kohl, David Welch, Jason Filby, Casper Hornstrup. Nothing really shown on the screen, just the very basics of win32k.
- A lot of work has been done by them in the period of 1999-2001, introducing DCs, window stations, desktops, mouse support, etc.
- GvG joined the 15th of February, 2003 and started working on supporting VMWare Video drivers in win32k along with Richard Campbell who worked on windowing, drawing, hacking it to work.

Win32 subsystem history (cont)

- Gunnar wrote the timer implementation, with a lot of code written by GvG and Richard Campbell in 2003.
- James Tabor joined the 7th of July, 2003 implementing NtUserQueryWindow, along with Royce Mitchell at about the same time. (David Welch was still committing!)
- Thomas Weidenmueller joined the win32k development the 1st of August, 2003 with menu improvements.
- Aleksey joined win32k on 25th of August, 2003 with NtGdiRealizePalette() fixes.

Win32 subsystem history (cont, even more boring)

- Filip Navara made a substantial contribution along with Mark Tempel also in 2003, with occasional commits by Art Yerkes, Andrew Greenwood, Gregor Anich, Mike Nordell, etc.
- Magnus Olsen joined win32k development the 16th of March, 2005 with DirectX kernelmode support and hacking here/hacking there.
- Modern history: Christoph von Wittich, Alex Ionescu, Brandon Turner, Herve Poussineau, Saveliy Tretiakov, Timo Kreuzer (his first commit was actually to win32k! 08.01.2007, rev.25352), Ged Murphy, Dmitry Gorbachev, Gregor Schneider, Stefan Ginsberg, etc.

Win32 subsystem. Advantages.

- 30+ people worked on it over 10 years.
- Targets Windows XP's Win32 subsystem architecture
- Alex, and later, Timo made a big effort on bringing win32k syscalls list to be compatible with the real Windows syscalls.
- Real video drivers are supported (VMWare, some video cards)

Win32 subsystem. Problems.

- 30+ people worked on it over 10 years. Quite enough time and manpower, and still no satisfactory result. Why? Win32 with Windows design is a huge monster, which would need 10x more resources.
- Apps which don’t have win32-related problems could be counted using the fingers of one hand.
- Numerous important bugs (move-mouse-to-download-files, message queue and font rendering issues, etc…) are sitting in bugzilla for YEARS.
- Only very few parts of Win32 subsystem really correspond to Windows XP Win32 subsystem architecture, other parts are incompatible (Wine or ReactOS-specific code).

Win32 subsystem V2.0

- Any good manager should try to target the future instead of the present.
- Hence the decision - do a totally new version of Win32 subsystem:
  - A chance to put forward good design decisions and throw bad out
  - Not worry about trunk breakage
  - Should ultimately be substantially better than existing subsystem

Win32 subsystem V2.0 (cont)

- Historical attempts to do this are known in ReactOS. All failed. To quote Filip Navara “It [Win32 subsystem in ReactOS] should be completely rewritten”.
- A unique solution is needed for this rewrite to become successful. And it is found.

Win32 subsystem V2.0 (cont)

- A summary of “Why?”, as in “Why a new one, why not improve existing one?”
  - Why not improve Linux instead of doing ReactOS? ☺
  - Current version of win32ss is a mix of old Wine code, old ReactOS specific code, and some good new code.
  - Days and months spent to marking Wine’s code in ReactOS’s win32ss and trying to sync it are wasted.
  - ReactOS needs a perfectly working win32ss ASAP. It can’t wait another 10 years.

Win32 subsystem V2.0 (cont)

- How? Writing a new win32ss would require years of work?!
  - Writing it from scratch – yes, it would take years. In fact, that's why all previous rewrites failed
  - A new, radical solution becomes possible: Reuse win32ss code from Wine project as much as possible.

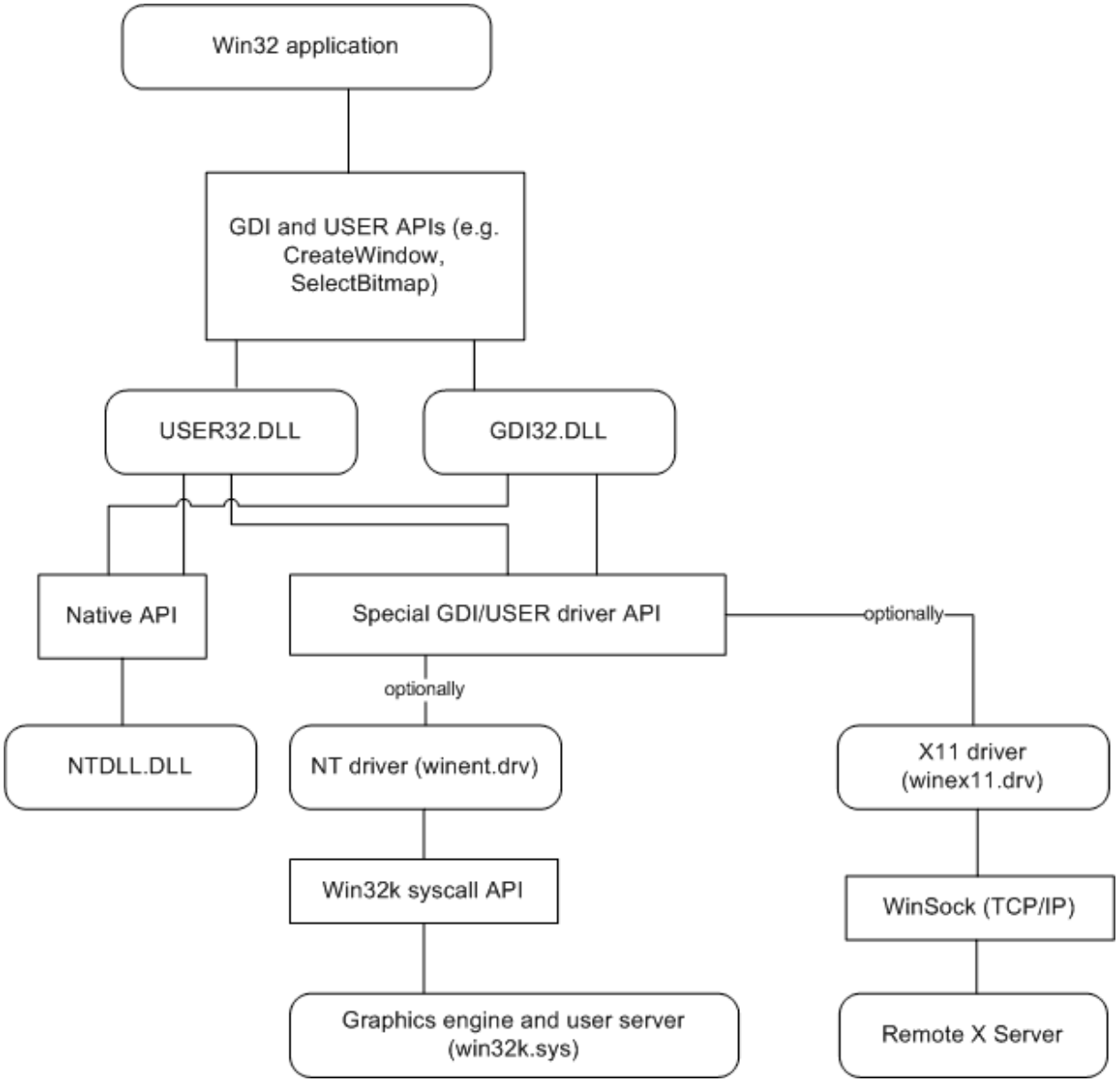
ARWINSS architecture

- Implements APIs exposed via USER32 and GDI32 libraries.
- Bases on Wine source code
  - Windowing and GDI code is fully isolated from other parts of Wine, forming kind of a library
  - Windows server code is also isolated from the rest (most) of unnecessary Wineserver code

ARWINSS architecture (cont)

- USER32.DLL and GDI32.DLL are nearly unchanged Wine source code
- WINENT.DRV – a custom, ReactOS-specific user/gdi driver for fast graphics and windowing operations
- WIN32K.SYS – low-level graphics support, user server implementation, minimal Win32 support for the kernel
- WINEX11.DRV – an **optional** user/gdi driver allowing to use remote X Server instead of a local display

ARWINSS architecture, the big picture



The absence of X Server

- A thorough reader would definitely ask: Wine is tied to X Server, how can you avoid that?
- Easy answer: we don’t.
- Complex answer: Wine implements a special layer of abstraction, called user/gdi driver, which abstracts all X11 specific details in a standalone library. ARWINSS implements its own fast user/gdi driver.

The role of Wine

- ARWINSS takes the best from Wine:
  - “Cheap” syncs of work done by hundreds of developers for every new version (takes ~30 minutes to merge and test)
  - At least 13495 apps from appdb.winehq.org become supported, plus support of those apps which Wine can’t run by design (hardware protection, drivers, etc)
  - Good, proven, regression tested source code

The role of Wine (cont)

- …and leaves the worst:
  - Ugly emulation of NT kernel
  - Incorrect call chains in kernel32/ntdll
  - ntoskrnl.exe being just another service
  - Very slow communication with Wineserver
  - Wineserver as a nightmare
  - UNIX dependencies
  - …

ARWINSS and ReactOS

- Why is it so important?
  - A smooth development strategy (testing in Wine is always possible, comparing debug logs, finding what goes wrong and where)
  - Rapid development speed (a working version was made mainly by one person within two months, compare that with 10 years of work)
  - Fixes huge amount of bugs at once (see “Potential benefits” in the Arwinss wiki page [1])
  - Real development plan, including switching ReactOS to beta stage

ARWINSS and ReactOS

- We need ReactOS to be compatible, and we need it now
- Fastest path to real world usage
- Development resources optimized requiring less efforts to keep user-mode components up to date
- It can easily bring usual features which took ReactOS years to implement and they are still not done. For example, printing support which is a must for any real world usage.

Future of ARWINSS

- ARWINSS doesn’t provide just stability and compatibility, it provides an extensibility for implementing new features
- One example: composite desktop as seen in Windows 7
- Another one: a terminal server
- We can try to win the race only when we have the car, ARWINSS needs us and we need it to move forward.

Future of ARWINSS

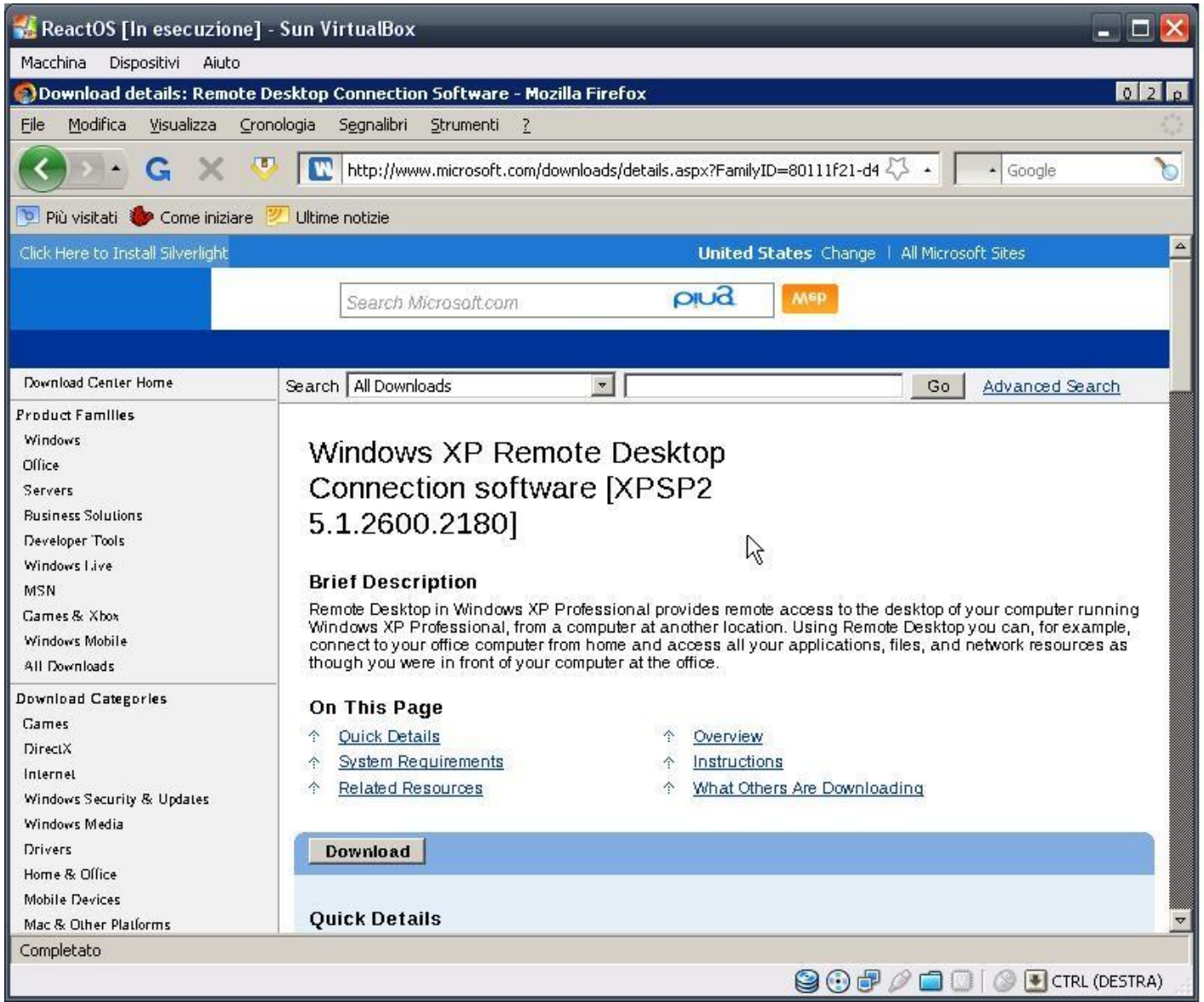
With your help we can take ReactOS to the place it deserves, reaching end users in a good shape and still growing…

Let’s look to the future, it will pay us back.

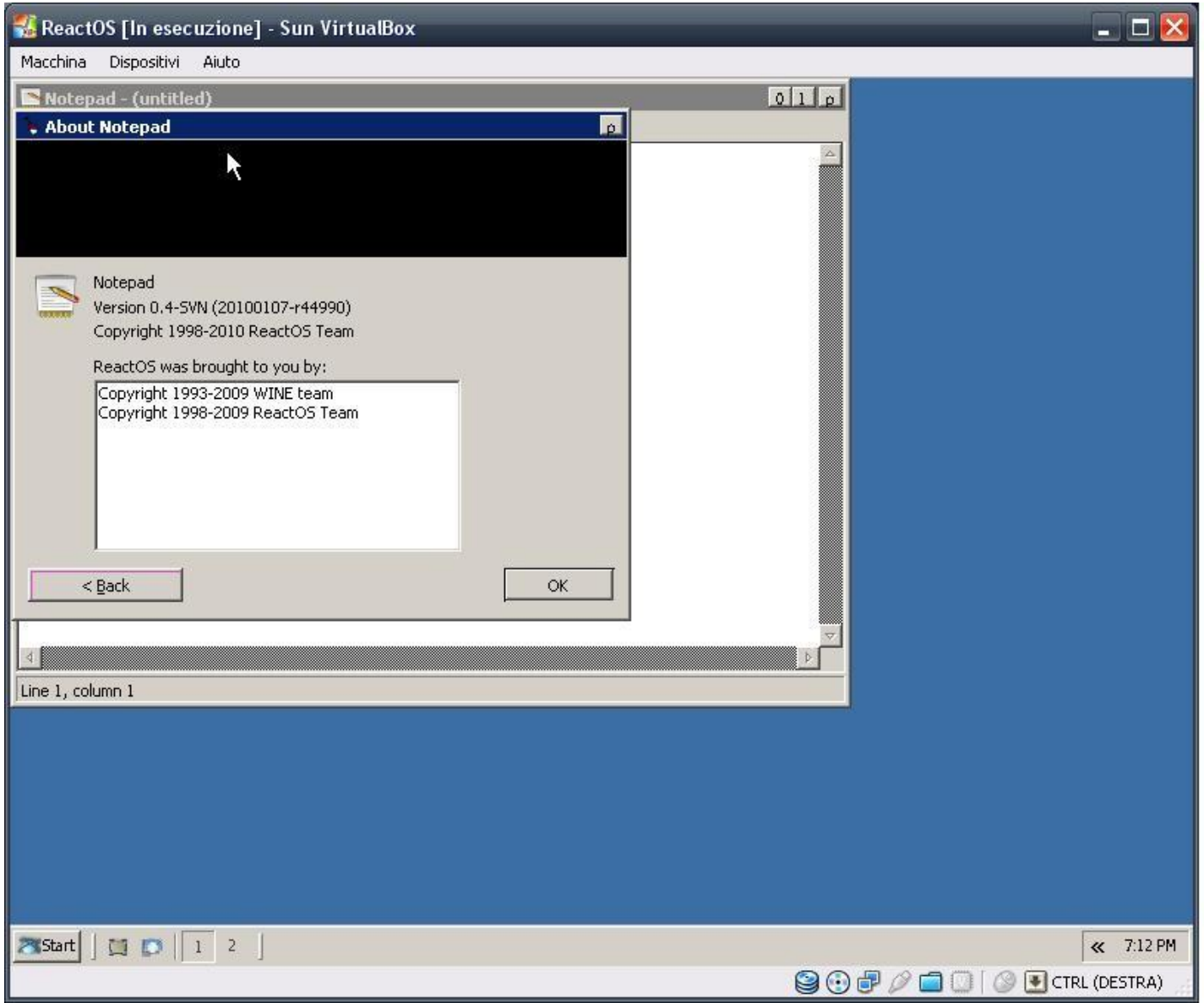
Further information

1. http://www.reactos.org/wiki/Arwinss - wiki page with installing, testing and debugging HOWTOs.
2. http://www.reactos.org/wiki/Arwinss_technical - Technical information, a starting point for Arwinss hacking.
3. http://www.assembla.com/spaces/reactos/tickets - Assembla ReactOS space, which should be used for tracking Arwinss bugs and tasks.
4. http://svn.reactos.org/svn/reactos/branches/arwinss/reactos/ - Web interface to the source code
5. svn://svn.reactos.org/reactos/branches/arwinss/reactos/ - SVN checkout url

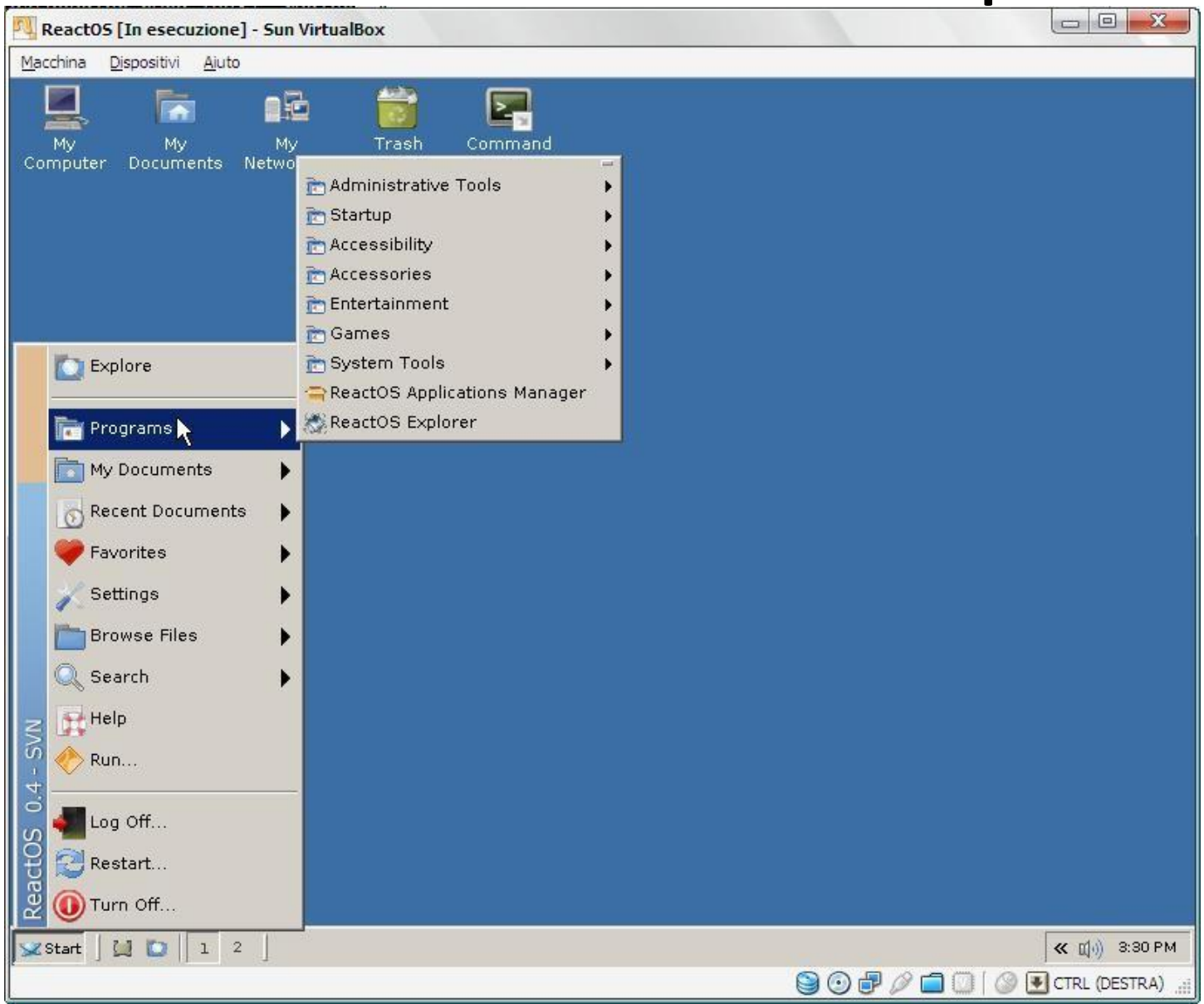
Screenshots. Firefox 3.5



Screenshots. Notepad



Screenshots. Desktop



Screenshots. Arwinss in Windows 2003 using X Windows driver and ReactOS Winlogon