ctys-uc-Android(7)
Setup of **Android** on
Qemu/KVM and **VirtualBox**

August 18, 2012

Contents

List of Figures

.

1 General

The current document shows the basic installation of Android, which is a Linux variant.

The following environment is used here:

- Debian-5.0.6 with VirtualBox-3.2.10

- CentOS-5.4 with kvm-83 / Qemu-0.9.1

- eeeDroid-1.6
  The current description is based on the edition for i386 architecture. Download the image:

  ```
  androidx86/eeeDroid_2008-12-20_1843Z.img
  ```

- UnifiedSessionsManager - ctys-01.11.011

.

2 Setup of Host-OS and Hypervisor

The installation for the following variants has to be performed by the appropriate standard setup of the HostOS and , which quite straight forward:

- Debian with VirtualBox
  Install the download version instead of the OSE edition shipped with the distribution.

- CentOS with QEMU/KVM
  Here the standard distribution is installed. Additional packages are vde2-2.2.3 and Qemu-0.12.2, which are build and installed to '/opt'. The vde2-2.2.3 package for network encapsulation requires a symbolic link

  ```
  ln -s /opt/vde2-2.2.3 /opt/vde
  ```

  The wrapper vde may not be required, when the Qemu support option is compiled in, but this is not yet widely the case. Thus vde2 is still utilized as standard.

3 Setup of the UnifiedSessionsManager

3.1 Install tgz BASE-Package + DOC-Package on Debian

1. Unpack the tar-gzip-archive and apply the standard installation procedure, where the call has to be executed by typing the fully qualified absolute path when ambiguity could occur. This is due to automatic usage of consistent libraries for the install procedure.

   ```
   ctys-distribute -F 2 -P UserHomeCopy root@lab02
   ```

2. Open a Remote Shell by call of CLI plugin:

   ```
   ctys -t cli -a create=l:tst137 root@tlab02
   ```

3. Check the plugins states by calling ctys-plugins:

   ```
   ctys-plugins -T all -E
   ```

3.2   Install rpm BASE-Package + DOC-Package on CentOS

The following steps are required for a RPM based setup on CentOS. The installation is relocatable, but located at '/opt', and installed locally by 'ctys-distribute'.

1. Install BASE package.

   ```
   rpm -i ctys-base-01.11.011.noarch.rpm
   ```

2. Now install a a local version, here by copy. The PATH prefix is important here, particularly in case of updates. The path is resolved to it's actual path by eliminating any symbolic link, and used for consistent link of libraries.

   ```
   /opt/ctys-01.11.011/bin/ctys-distribute -F 2 -P UserHomeCopy
   ```

3. Next the menu is setup.

   ```
   ctys-xdg --menu-create
   ```

4. Now the help is available as eihter a Gnome or KDE menu. Alternatively could be called from the commandline.

3.3   Setup of the Gnome Menu

The setup of the Gnome Menu is quite simple, the contained tool **ctys-xdg(1)** sets up a standard menu by the call:

```
ctys-xdg --menu-create
```
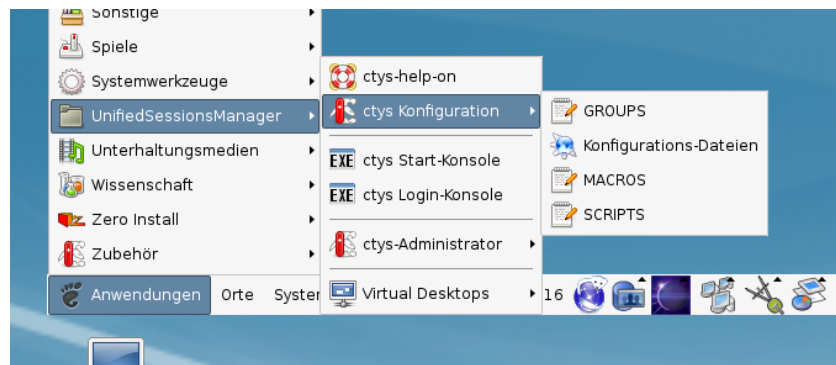

Figure 1: Create Menu

The setup could be targeted either for private menus or shared menus. Both setups are based on a menu template, which is stored in the configuration subdirectory 'xdg.d'. The call

```
ctys-xdg --menu-cancel
```

removes the installed files. For current version no checks for changed files is done. The menues could be edited and extended by the call

```
ctys-xdg --menu-edit
```

which opens the related directories for modification of '*.menu', '*.desktop', and '*.directory' files.

4 Creation and Installation

4.1 Creation and Installation on QEMU/KVM

The demo example VM is here named tst141, this is the hostname of GuestOS too.

1. Login into the machine where QEMU/KVM is installed.

   ```
   ssh -X ap2
   ```

2. Change to the vmpool and create a directory and change into.

   ```
   mkdir tst141
   ```

3. Call the install and configuration utility for VMs. Here some values are set by environment variables, a complete list including the actually assigned values could be displayed by the option **–levo**.

   ```
   ARCH=i386 \
   DIST=Android \
   DISTREL=1.6-r2 \
   OS=Linux \
   OSREL=2.6 \
   ctys-createConfVM -t qemu --label=tst141
   ```

   This call creates a virtual image(hda.img), the call-wrapper(tst140.sh), and the configuration file(tst140.ctys). The files are created from templates by assigning configuration values either from pre-configured default values, or interactive variation.

   The resulting parameters are:

   ```
   Not all values require to be set, some will be requested later
   by dialogue.
   Thus it is not neccessary to have values assigned to the complete
   displayed set.


   Actually used sources for default values:
     no-marker  = Pre-Set value, either from defaults configuration,
                  or by commandline.
     no-value   = Either requested by dialog later, or the defaults
                  of the finally
                  called application are used.
     (c)        = Read from actual configuration file, e.g. vmx-file.
     (d)        = Read from database.
     (g)        = Dynamically generated.
     (h)        = Used from current host as default.
     (m)        = Received from mapping definitions.

   Applicable modifications:
     blue       = By call option, defines dependency for others.
     green      = By environment, 'could be set almost independent'
                  from other values.
     cyan       = By miscellaneous facilities, but is dependent from
                  others.
                  E.g. LABEL defines by convention the network 'hostname',
                  thus the TCP/IP params.
                  This could ..., but should not be altered!

   Most of the missing values will be fetched during actual execution
   ```

of this tool by dynamic evaluation.

```
                 VAR name:Initial Value

            C_SESSIONTYPE:QEMU
                    LABEL:tst141
                      MAC:00:50:56:13:11:69 (m)
                       IP:172.20.2.245 (m)
                   BRIDGE:
                     DHCP:
                  NETMASK:
                      TCP:
                  GATEWAY:

                   EDITOR:acue

                     UUID:ff81f9d8-ba06-4c90-a801-484ad4841b50 (h)

                     DIST:Android
                  DISTREL:1.6-r2
                       OS:Linux
                    OSREL:2.6

                     ARCH:i386
              ACCELERATOR:KVM  (h)
                      SMP:
                  MEMSIZE:512
               KBD_LAYOUT:de

               STARTERCALL:/usr/libexec/qemu-kvm
               WRAPPERCALL:tst141.sh

           DEFAULTBOOTMODE:HDD

         DEFAULTINSTTARGET:/mntn/vmpool/vmpool05/kvm/test/tst-ctys/...
                          ...tst141/hda.img
       HDDBOOTIMAGE_INST_SIZE:8G
  HDDBOOTIMAGE_INST_BLOCKSIZE:256M
 HDDBOOTIMAGE_INST_BLOCKCOUNT:32
    HDDBOOTIMAGE_INST_BALLOON:y

          DEFAULTINSTMODE:CD
             INSTSRCCDROM:/mntn/swpool/UNIXDist/../miscOS/Android/raw/...
                         ...android-x86/android-x86-1.6-r2.iso
         DEFAULTINSTSOURCE:/mntn/swpool/UNIXDist/../miscOS/Android/...
                          ...raw/android-x86/android-x86-1.6-r2.iso
              INST_KERNEL:
              INST_INITRD:

                  VMSTATE:ACTIVE
```

Remember that his is a draft pre-display of current defaults.
No consistency-checks for provided values are performed at this stage.
Some missing values are evaluated at a later stage dynamically.

An alternate call for the installation is the remote execution:

```
ctys -t qemu \
 -a create=l:tst140,id:${TST140}/tst140.ctys,instmode,console:sdl\
 app2
```

This starts the same by transforming to the target host 'app2' and calling the previous wrapper script.

The resulting files in both cases are:

- tst141.ctys
- tst141.sh
- hda.img

4. Once the set of files is created the virtual machine is prepared for startup. For some other systems complete installation routines are available, e.g. debian and CentOS. The current state could be checked now by the following call.

```
./tst141.sh --console=vnc  --vncaccessdisplay=47 --print --check
```

This shows the current resulting call:

```
###########################
#Display call             #
###########################


QEMU_VERSION      = "qemu-0.9.1-kvm-83-maint-snapshot-20090205"
QEMU_MAGIC        = "QEMU_091"
QEMU_ACCELERATOR = "KVM"


ctys-uc-AndroidNAME   = "tst141.sh"
              +->STARTERCALL     = /usr/libexec/qemu-kvm
              +->REALSTARTERCALL = /usr/libexec/qemu-kvm


#The resulting call is:   #
--->
eval "/opt/vde/bin/vdeq  /usr/libexec/qemu-kvm     \
  -net nic,macaddr=00:50:56:13:11:69,model=rtl8139 \
  -net vde,sock=/var/tmp/vde_switch0.acue  \
  -name "tst141" -vga cirrus -localtime  -k de -m 512  -cpu qemu32  \
  -serial mon:unix:/var/tmp/qemumon.tst141.21844.acue,server,nowait \
  -daemonize  -vnc :47 \
  -boot c /mntn/vmpool/vmpool05/kvm/test/tst-ctys/tst141/hda.img"
<---

EXECALL:/opt/vde/bin/vdeq /usr/libexec/qemu-kvm
  -net nic,macaddr=00:50:56:13:11:69,model=rtl8139
  -net vde,sock=/var/tmp/vde_switch0.acue
  -name "tst141"
  -vga cirrus
  -localtime
  -k de
  -m 512
  -cpu qemu32
```

```
-serial mon:unix:/var/tmp/qemumon.tst141.21844.acue,server,nowait
-daemonize
-vnc :47
-boot c
/mntn/vmpool/vmpool05/kvm/test/tst-ctys/tst141/hda.img
```

The installation is slightly different due to boot from install media.

```
./tst141.sh --console=vnc  --vncaccessdisplay=47 --print --instmode --check


###########################
#Display call             #
###########################


QEMU_VERSION      = "qemu-0.9.1-kvm-83-maint-snapshot-20090205"
QEMU_MAGIC        = "QEMU_091"
QEMU_ACCELERATOR = "KVM"


ctys-uc-AndroidNAME   = "tst141.sh"
              +->STARTERCALL     = /usr/libexec/qemu-kvm
              +->REALSTARTERCALL = /usr/libexec/qemu-kvm


#The resulting call is:   #
--->
eval "/opt/vde/bin/vdeq  /usr/libexec/qemu-kvm \
 -net nic,macaddr=00:50:56:13:11:69,model=rtl8139 \
 -net vde,sock=/var/tmp/vde_switch0.acue  \
 -name "tst141" -vga cirrus -localtime -k de -m 512 -cpu qemu32 \
 -serial mon:unix:/var/tmp/qemumon.tst141.23708.acue,server,nowait \
 -daemonize -vnc :47 -boot d \
 -cdrom /mntn/swpool/UNIXDist/../miscOS/Android/raw/...
        ...android-x86/android-x86-1.6-r2.iso \
 -hda /mntn/vmpool/vmpool05/kvm/test/tst-ctys/tst141/hda.img "
<---

EXECALL:/opt/vde/bin/vdeq /usr/libexec/qemu-kvm
  -net nic,macaddr=00:50:56:13:11:69,model=rtl8139
  -net vde,sock=/var/tmp/vde_switch0.acue
  -name "tst141"
  -vga cirrus
  -localtime
  -k de
  -m 512
  -cpu qemu32
  -serial mon:unix:/var/tmp/qemumon.tst141.23708.acue,server,nowait
  -daemonize
  -vnc :47
  -boot d
  -cdrom /mntn/swpool/UNIXDist/../miscOS/Android/raw/...
          ...android-x86/android-x86-1.6-r2.iso
  -hda /mntn/vmpool/vmpool05/kvm/test/tst-ctys/tst141/hda.img
```

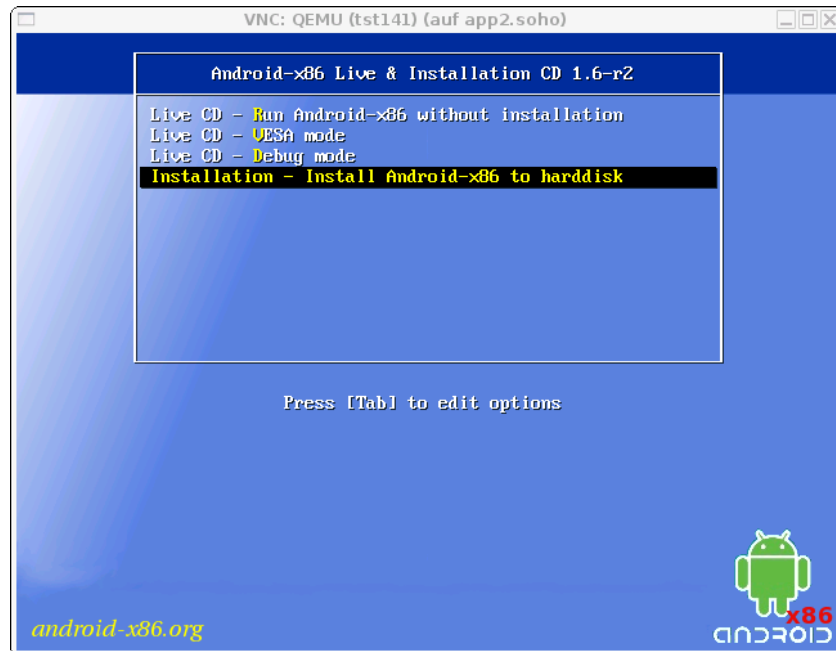The actual call starts the VM and displays the following screen.

Figure 2: Install Menu on QEMU/KVM

The install procedure just installs here a life system on disk, thus proceeds quite fast. After the installation unmount the install media and boot into Android.

5. In order to reboot just shutdown and boot again without the 'instmode' option. The shutdown could be proceeded by the 'quit' command within the monitor. The **monitor mode** is entered e.g. by **Ctrl-Alt-2**. One possible boot call for SDL console is:

```
ctys -t qemu \
    -a create=l:tst141,id:${PWD}/tst141.ctys,console:sdl \
    app2
```

The next starts with VNC console, which is default:

```
ctys -t qemu \
    -a create=l:tst141,id:${PWD}/tst141.ctys,console:vnc \
    app2
```
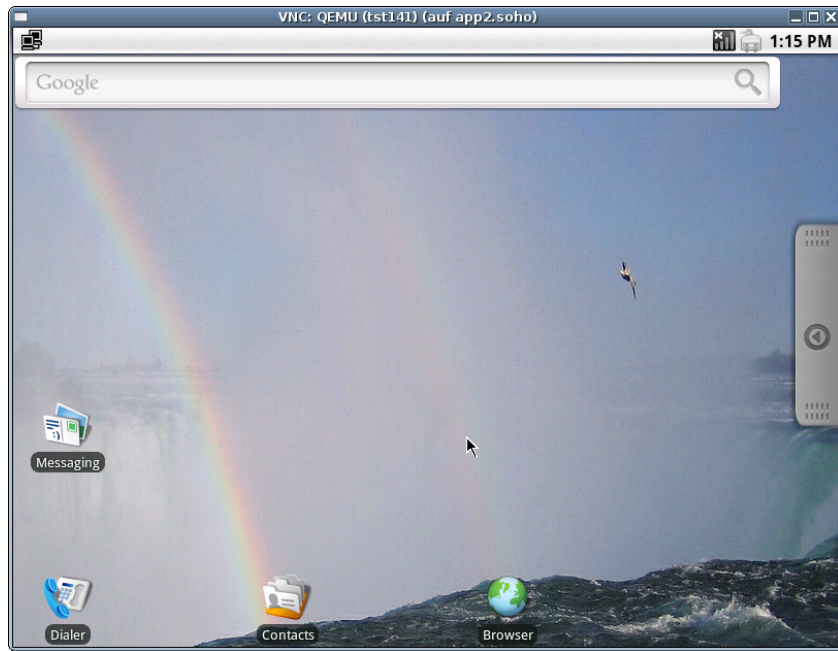
Figure 3: Install Menu on QEMU/KVM

When standard options are used the VM crashes when the screensaver is activated. Two workarounds are possible, first deactivating ACPI, second deactivating the screensaver. Here both are applied.

The following deactivates the screensaver - here called 'Screen timeout'. The menu order is:

```
Settings -> Sound&display -> Screen timeout -> Never timeout
```
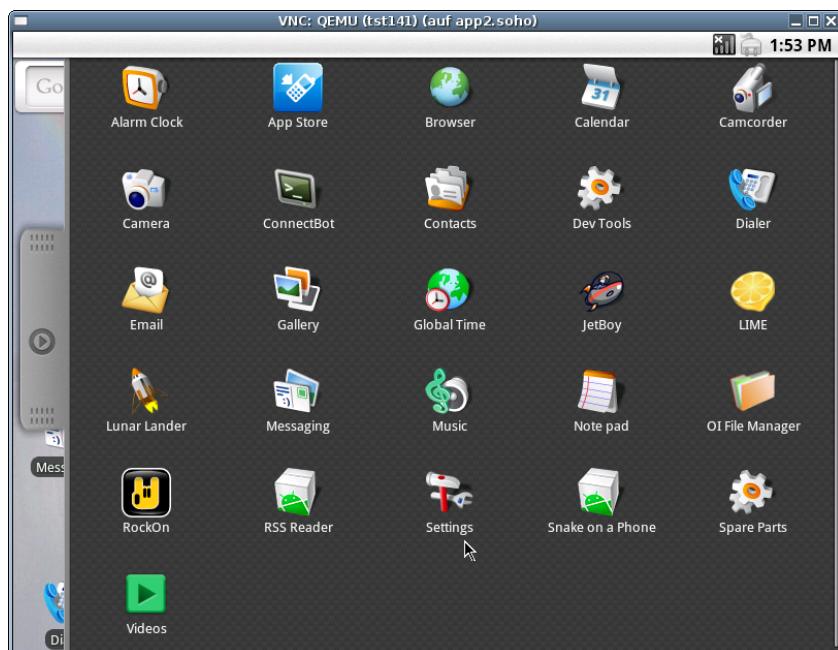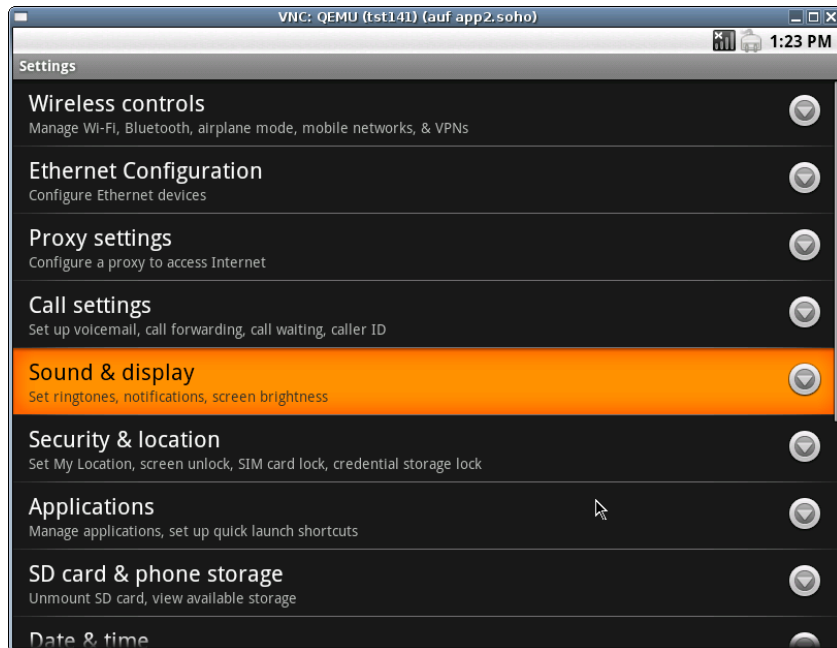

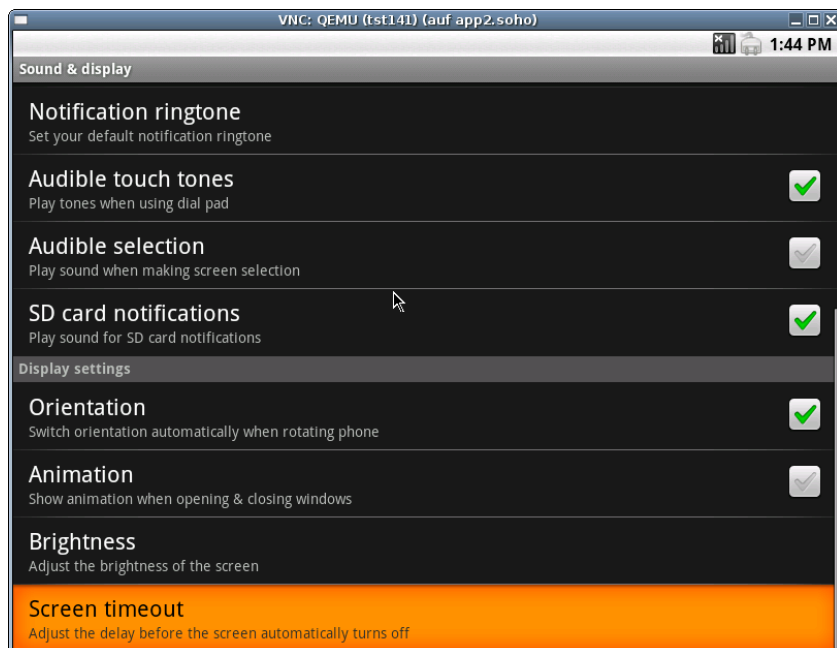Figure 4: Deactivate screensaver - 01

Figure 5: Deactivate screensaver - 02
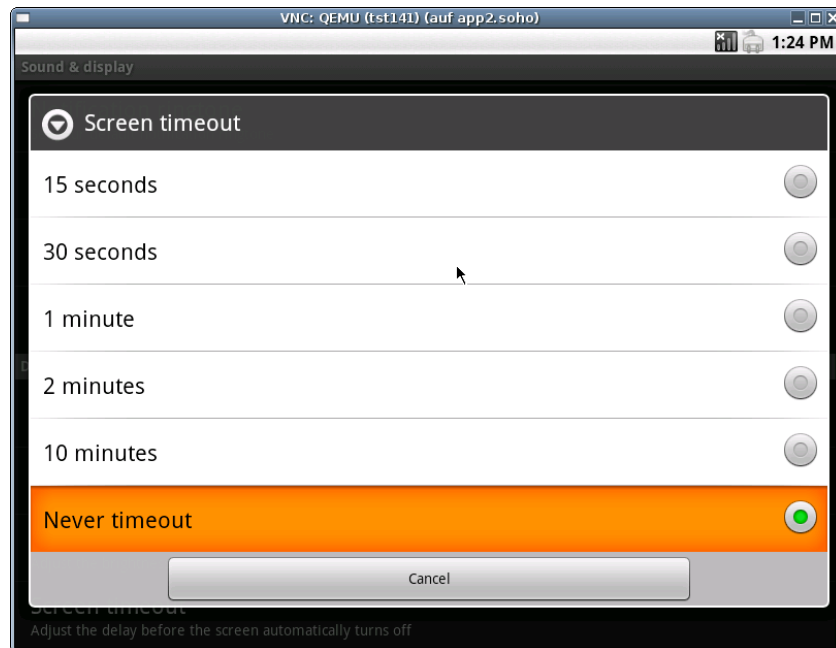

Figure 6: Deactivate screensaver - 03

Figure 7: Deactivate screensaver - 04

For stable operations the following variation of predefined settings are applied maunally within the file 'tst141.ctys':

- Activate: NIC=$NIC:-pcnet
- Add: ARGSADD=" -no-acpi ";
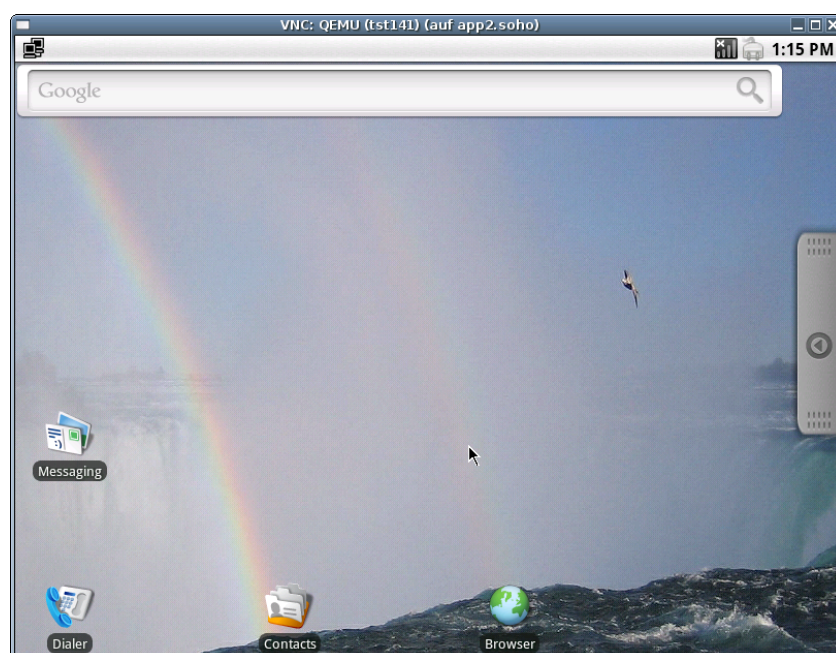- Eventually activate: VGADRIVER="-vga std"

Figure 8: Android on QEMU/KVM

4.2 Creation and Installation on VirtualBox

The creation of the raw VM is the first step to be executed at the host operating system. This could be either performed locally or remote and requires the usage of the provided tools by VirtualBox(TM).

1. Login into the machine where VirtualBox is installed.

   ```
   ssh -X lab02
   ```

2. Execute the VirtualBox(TM) console.

   ```
   VirtualBox
   ```

3. Create the VM, the machine is called here 'tst140'. The OS is 'Linux', the version is 'Linux 2.6'.

Figure 9: Create Virtual Machine

4. Set RAM to 512MByte.

Figure 10: Set virtual RAM

5. Create a virtual HDD, here 8GByte is choosen. When finished the raw VM is present and could be used as required, for basic functions of ctys no additional configuration is required.
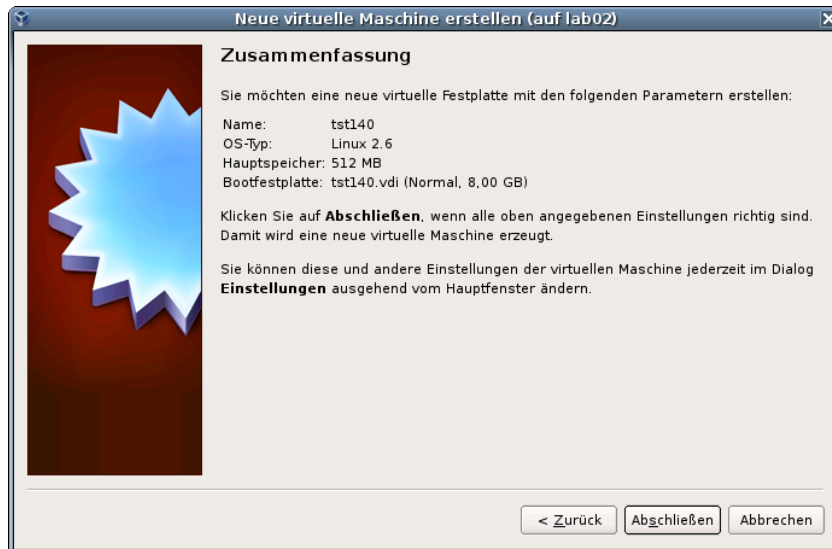

Figure 11: Create Virtual HDD

Figure 12: Check HDD image file

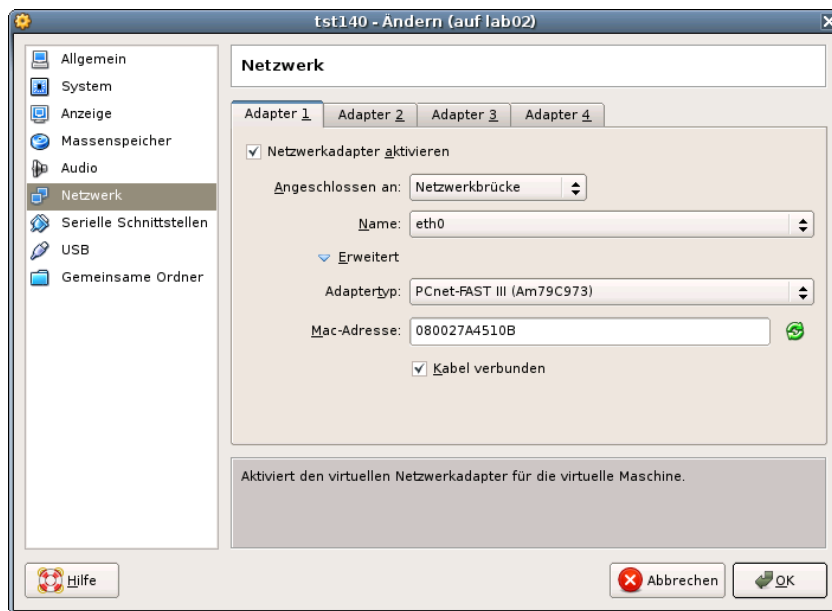6. The network device should be set to 'PCnet-Fast III' with DHCP, either NAT or bridged.

Figure 13: Network device

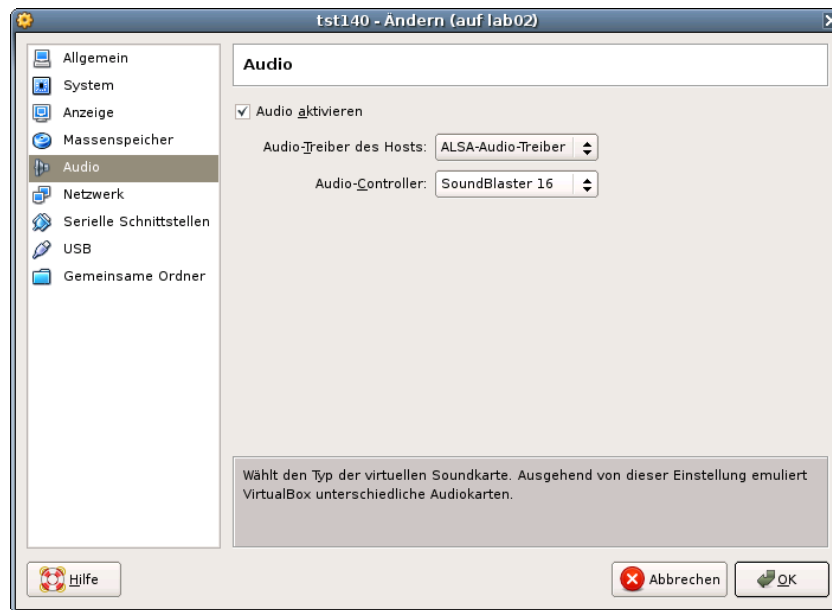7. The audio card has to be set to 'Sound Blaster 16'.

Figure 14: Audio device

8. When additional information should be stored coallocated to the VM and scanned automatically into a database, than the tool **ctys-createConfVM(1)** should be applied. This generates additional detailed information related to the specific VM and the inherent guest OS. The call could be executed either interactive or automatic.

Call within the same directory for first inspection:

```
ARCH=i386 \
DIST=Android \
DISTREL=1.6-r2 \
OS=Linux OSREL=2.6 \
ctys-createConfVM -t vbox --label=tst140 --levo
```

This lists some defaults for the specific hypervisor. These could be preconfigured by specific template files within the configuration directory **ctys-createCOnfVM.d**. The result should look like the following:

```
Not all values require to be set, some will be requested later by
dialogue.
Thus it is not neccessary to have values assigned to the complete
displayed set.


Actually used sources for default values:
  no-marker  = Pre-Set value, either from defaults configuration,
               or by commandline.
  no-value   = Either requested by dialog later, or the defaults
               of the finally called application are used.
  (c)        = Read from actual configuration file, e.g. vmx-file.
  (d)        = Read from database.
  (g)        = Dynamically generated.
  (h)        = Used from current host as default.
  (m)        = Received from mapping definitions.
```

```
Applicable modifications:
  blue        = By call option, defines dependency for others.
  green       = By environment, 'could be set almost independent'
                from other values.
  cyan        = By miscellaneous facilities, but is dependent from
                others.
                E.g. LABEL defines by convention the network 'hostname',
                thus the TCP/IP params.
                This could ..., but should not be altered!
```

Most of the missing values will be fetched during actual execution of
this tool by dynamic evaluation.

```
                   VAR name:Initial Value

            C_SESSIONTYPE:VBOX
                    LABEL:tst140
                      MAC:08:00:27:A4:51:0B (c)
                       IP:
                   BRIDGE:
                     DHCP:
                  NETMASK:
                      TCP:
                  GATEWAY:

                   EDITOR:root

                     UUID:97d5a071-1914-477c-89c4-d47dd7adac74 (c)

                     DIST:Android
                  DISTREL:1.6-r2
                       OS:Linux
                    OSREL:2.6

                     ARCH:i386
              ACCELERATOR:HVM  (c)
                      SMP:1 (c)
                  MEMSIZE:768 (c)
              KBD_LAYOUT:de

              STARTERCALL:/usr/bin/VirtualBox

          DEFAULTBOOTMODE:HDD

        DEFAULTINSTTARGET:/mntn/vmpool/vmpool05/vbox/test/...
                          ...tst-ctys/tst140/tst140.vdi
    HDDBOOTIMAGE_INST_SIZE:8192M

                  VMSTATE:ACTIVE
```

Remember that his is a draft pre-display of current defaults.
No consistency-checks for provided values are performed at this stage.
Some missing values are evaluated at a later stage dynamically.

When the call is finished without the '–levo' option the file 'tst140.ctys' with additional configuration information information is stored.

9. The start of the VM could be proceeded either by calling VirtualBox, or by the VBOX plugin. Both require in current version the pre-configuration of the appropriate install procedure e.g. by attaching the install media. Here the boot image 'android-x86-1.6-r2.iso' is required.
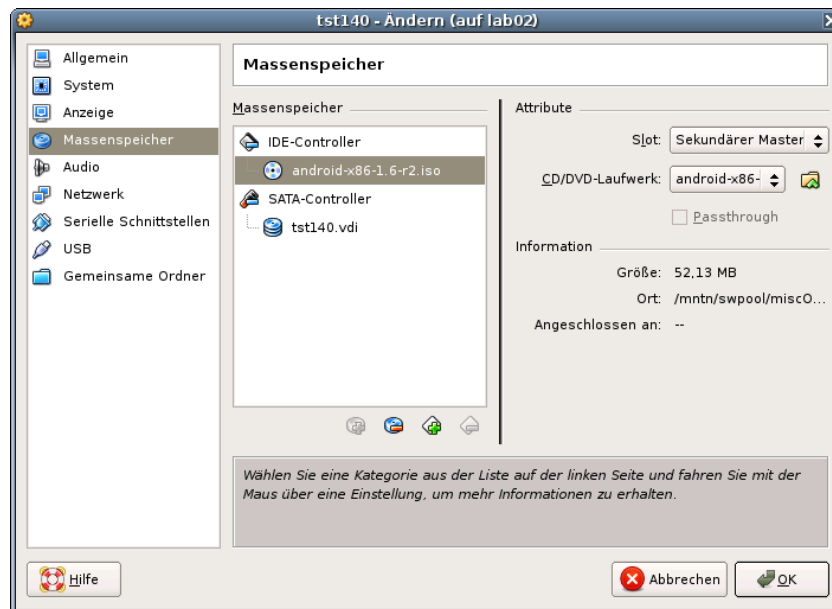

Figure 15: Install media

The following call starts the VirtualBox console.

```
VirtualBox
```

The following call variant starts the remote VM with a VirtualBox console:

```
ctys -t vbox \
 -a create=l:tst140,id:${TST140}/tst140.ctys,console:vbox\
 app2
```

10. Now boot the VM and choose 'Installation Only' to start the installation.
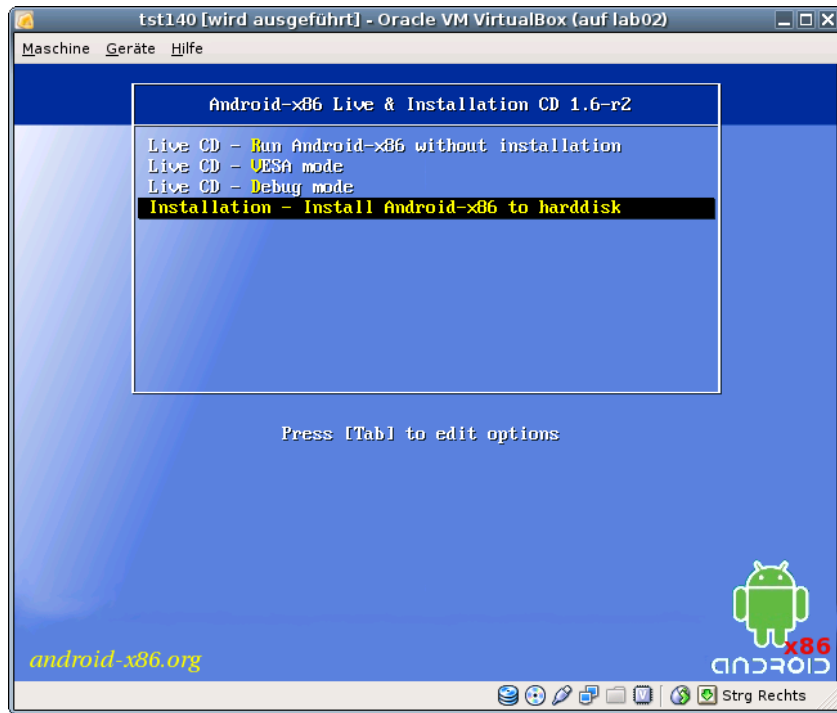
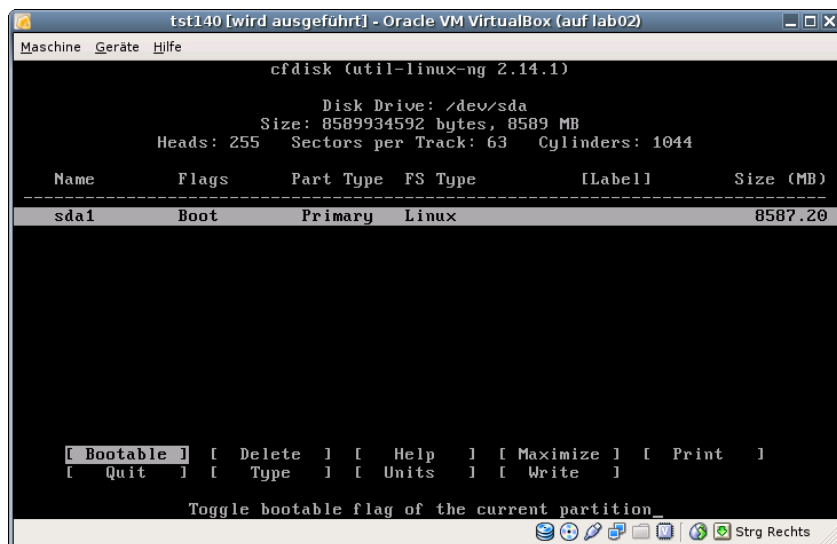Figure 16: Install menue

11. HDD partitioning.

Figure 17: Format vHDD

12. After the installation unmount the install media and boot into Android. In case of a first start the call could look like:

```
ctys -t vbox \
    -a create=l:tst140,id:${PWD}/tst140.ctys,console:vbox \
```
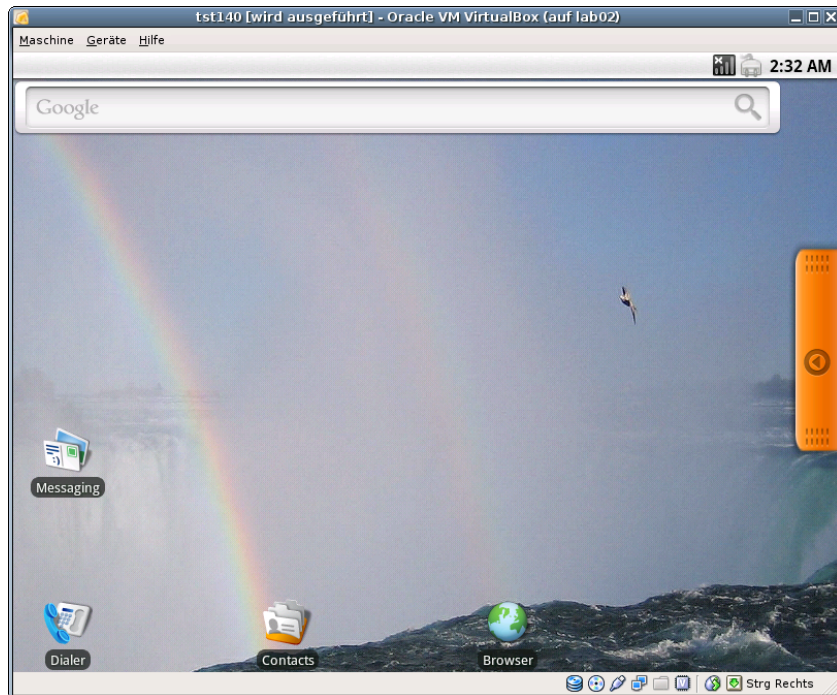
```
app2
```

The default console is here RDP.

Figure 18: Android

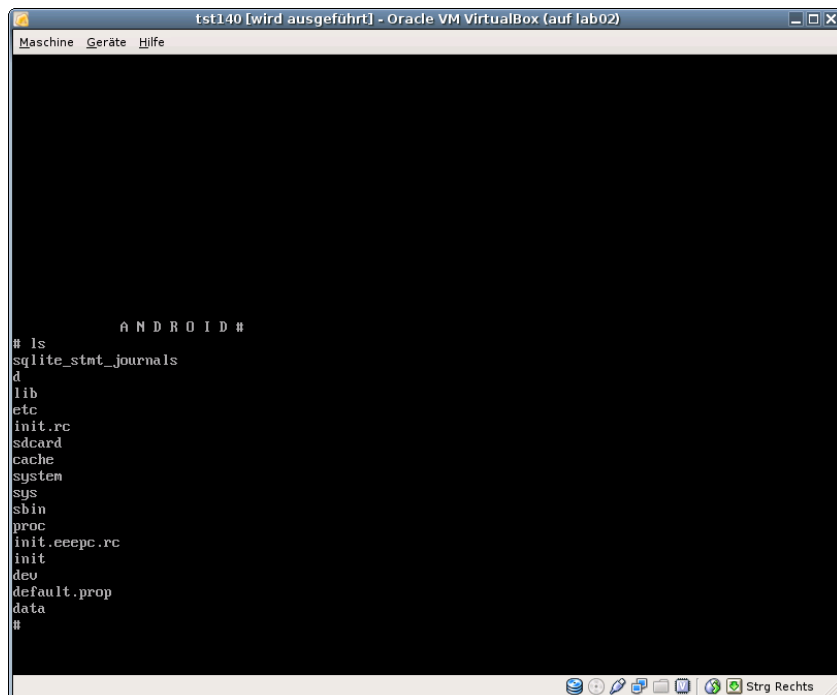Change into console with **Alt-F1**, **Alt-F7** returns to graphical display.

Figure 19: Android ASC-II Console

5 Creation of the Inventory - cacheDB

In case of a common mounted NFS filesystem for the pool VMs for simplicity just change into the directory of the VM on any machine. Call for the first check **ctys-vdbgenVM(1)** with the **–stdio** option for display only.

```
cd /mntn/vmpool/vmpool05/vbox/test/tst-ctys/tst140
ctys-vdbgen --append --base=$PWD --stdio -- root@lab02
cd /mntn/vmpool/vmpool05/kvm/test/tst-ctys/tst141
ctys-vdbgen --append --base=$PWD --stdio -- app2
```

When the result is displyed correctly just call without the '–stdio' option.

```
cd /mntn/vmpool/vmpool05/vbox/test/tst-ctys/tst140
ctys-vdbgen --append --base=$PWD  -- root@lab02
```

The following output should be displayed:

```
Prepare execution-call:

Require DB-PATH,        USE: DEFAULT_DBPATHLST="/homen/acue/.ctys/db/default"
Require DB-PATH,        USE: -o => "/homen/acue/.ctys/db/default"
APPEND mode              : ON(1)
STDIO mode off           : OFF(0)
Set TYPE scope         ADD: DEFAULT="-t ALL"
Preload TYPE set       ADD: DEFAULT="-T ALL"
For splitted operations ADD: DEFAULT="-b sync,seq "
Nameservice cache      OFF: DEFAULT="-c off "
Data cache             OFF: DEFAULT="-C off "

Resulting ENUMERATE    ADD: DEFAULT="-a enumerate=matchvstat:...
   ...active%disabled%empty,machine,b:/mntn/vmpool/vmpool05/vbox/...
   ...test/tst-ctys/tst140 -C off  -c off  -T ALL  "

-> generate DB(may take a while)...
-----------------------------------
START:14:55:11
------


------
END:14:55:38
DURATION:00:00:27
-----------------------------------
RET=0
-----------------------------------

Cached data:

  Mode:                 APPEND
  Pre-Appended:         835 records
```

```
 Appended:              1 records
 Fetched Records Raw:    records
 Fetched Records Unique:  records
 Final:                 836 records


------------------------------------
   ...finished.
```

The QEMU/KVM scan by:

```
cd /mntn/vmpool/vmpool05/kvm/test/tst-ctys/tst141
ctys-vdbgen --append --base=$PWD  -- app2
```

Should display:

```
Prepare execution-call:

Require DB-PATH,      USE: DEFAULT_DBPATHLST="/homen/acue/.ctys/db/default"
Require DB-PATH,      USE: -o => "/homen/acue/.ctys/db/default"
APPEND mode              : ON(1)
STDIO mode off           : OFF(0)
Set TYPE scope        ADD: DEFAULT="-t ALL"
Preload TYPE set      ADD: DEFAULT="-T ALL"
For splitted operations ADD: DEFAULT="-b sync,seq "
Nameservice cache     OFF: DEFAULT="-c off "
Data cache            OFF: DEFAULT="-C off "

Resulting ENUMERATE    ADD: DEFAULT="-a enumerate=matchvstat:active%...
   ...disabled%empty,machine,b:/mntn/vmpool/vmpool05/kvm/test/tst-ctys/tst141
   -C off  -c off  -T ALL  "

-> generate DB(may take a while)...
------------------------------------
START:14:55:40
------



------
END:14:56:29
DURATION:00:00:49
------------------------------------
RET=0
------------------------------------

Cached data:

  Mode:                 APPEND
  Pre-Appended:         836 records
  Appended:             1 records
  Fetched Records Raw:    records
  Fetched Records Unique:  records
  Final:                837 records


------------------------------------
```

```
   ...finished.
```

This shows that only two(1+1) entries are appended to the existing database with 835 VM-Entries. Now check the database entry by calling:

`ctys-vhost tst14`

The following result should be displayed when the regular expression 'tst14.*' matches only twice:

```
label |stype|accel|distro |distrorel|os     |osrel|PM      |if|TCP
------+-----+-----+-------+---------+-------+-----+--------+--+------------
tst141|QEMU |KVM  |Android|1.6-r2   |Linux  |2.6  |app2.soho|0 |172.20.2.245
tst140|VBOX |HVM  |       |         |Linux26|     |lab02    |  |
```

6   Graphical Start of the Virtual Machine

This chapter demostrates the seamless integration of the hypevisors QEMU(emulation), QEMU/KVM, and VirtualBox(TM). The fully automatic generated database is synchronous with the graphical starter and offers the same and one user interface. This is the case for all supported plugins, due to missing native plugins for Android the LOGIN could not be demostrated for this special case.

6.1 Graphical Start of the Virtual Machine by QEMU/KVM

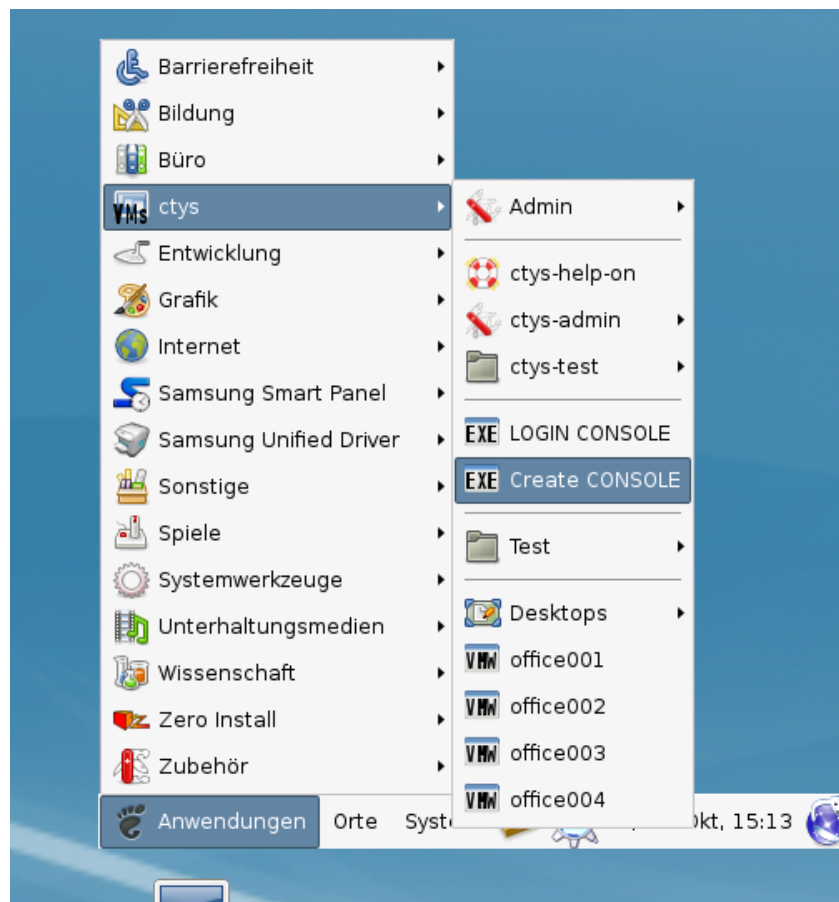Now call the menue item for start of the VM 'tst141'.

Figure 20: Android Start Menu

The created cacheDB record for thr VM 'tst140' is now automatically visible in the list of startable virtual machines.
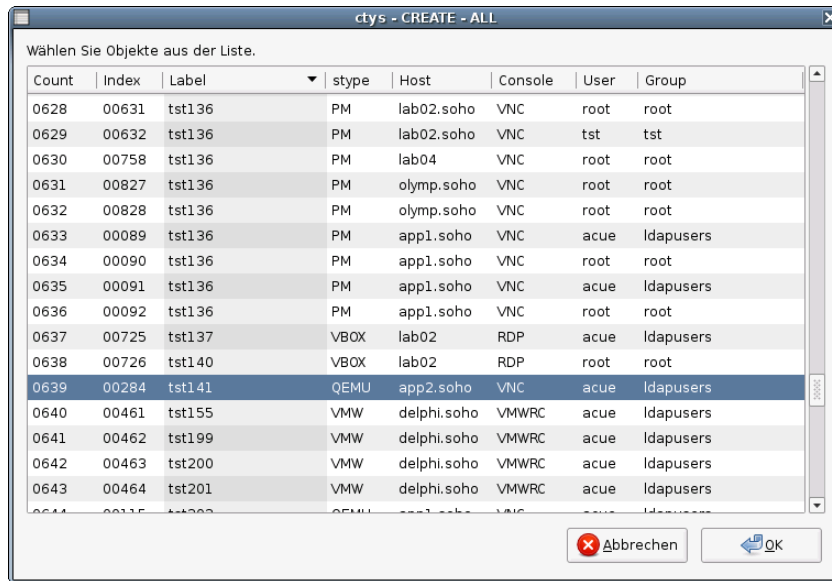
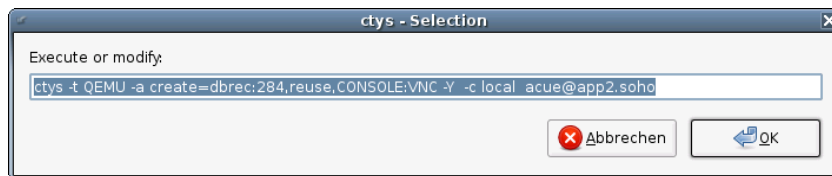Figure 21: Android VM Selection

Confirm the selected entry.

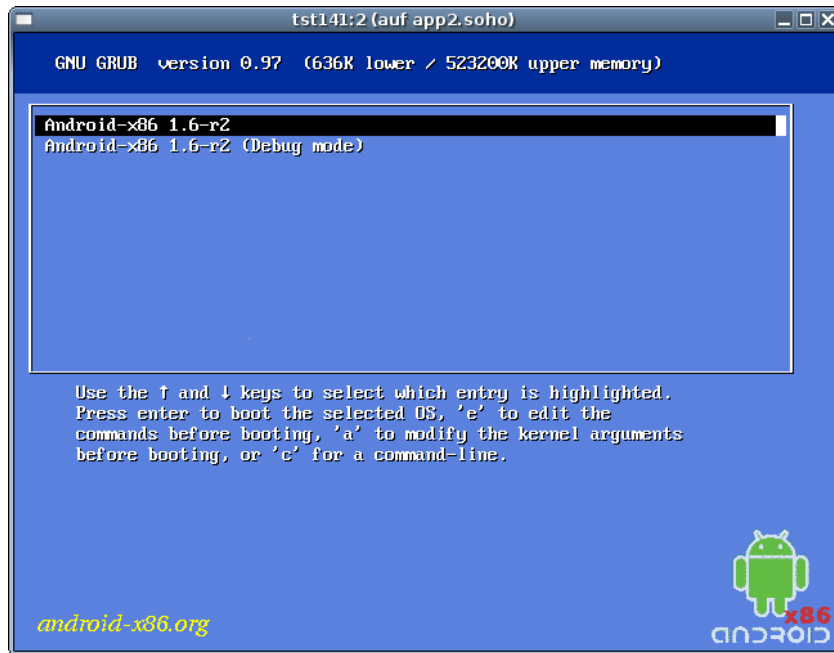Figure 22: Android Call Confirmation

Boot...

Figure 23: Boot Android
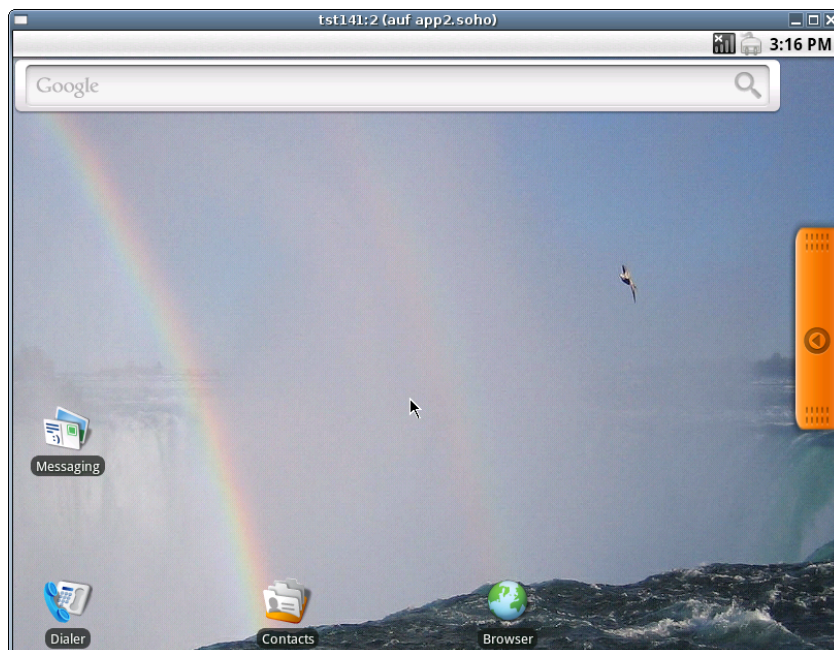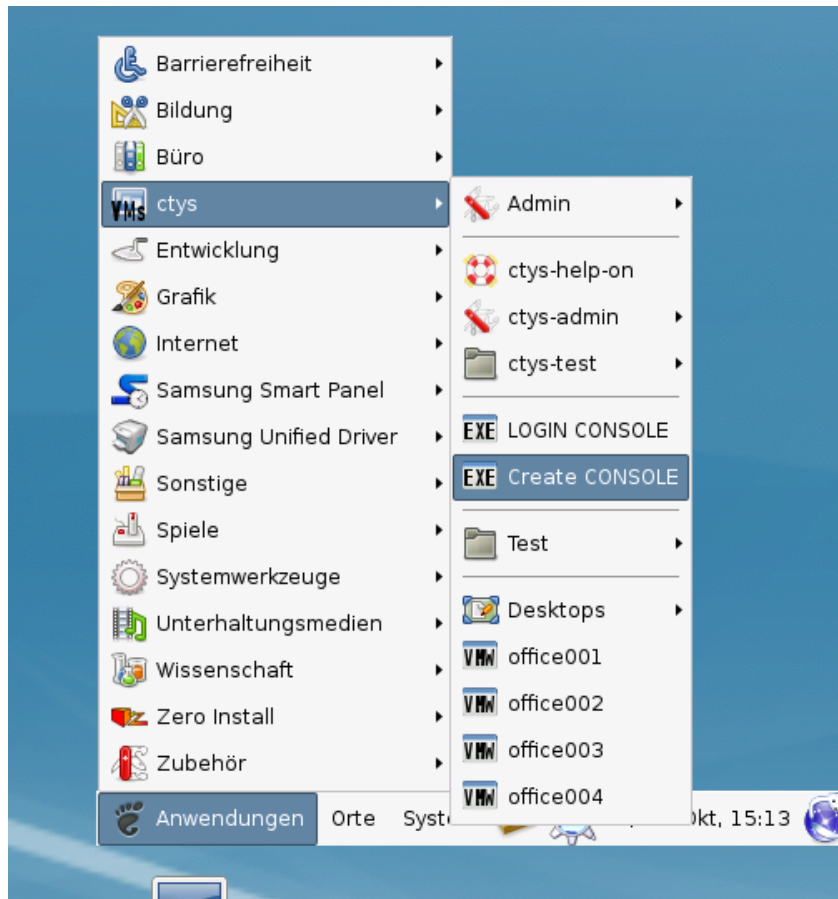
...and enjoy Android.

Figure 24: Enjoy Android

6.2 Graphical Start of the Virtual Machine by VBOX

Now call the menue item for start of the VM 'tst140'.

Figure 25: Android Start Menu

The created cacheDB record for thr VM 'tst140' is now automatically visible in the list of startable virtual machines.
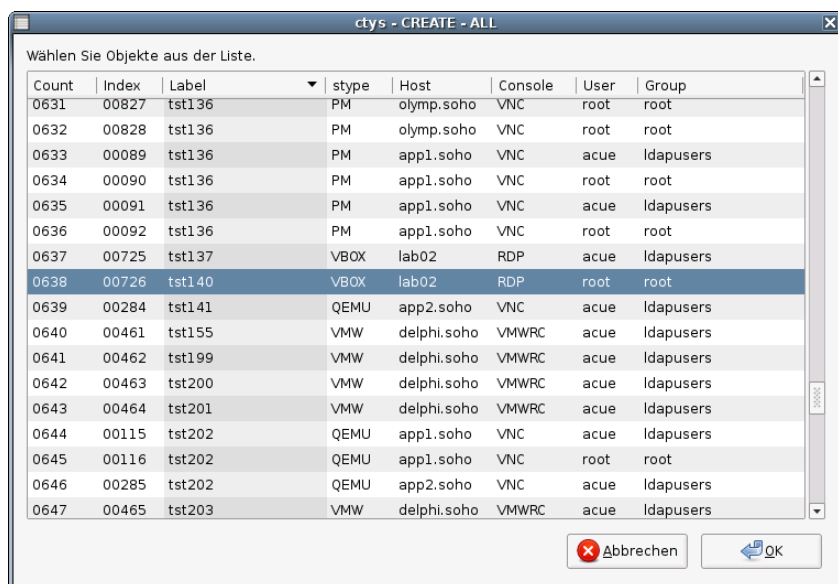

Figure 26: Android VM Selection
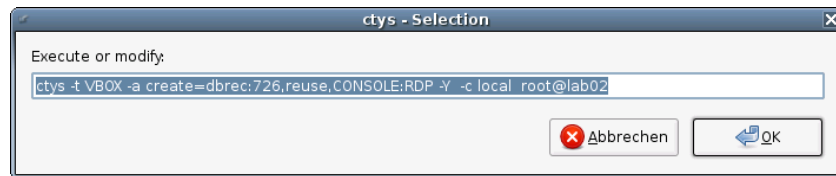
Confirm the selected entry.
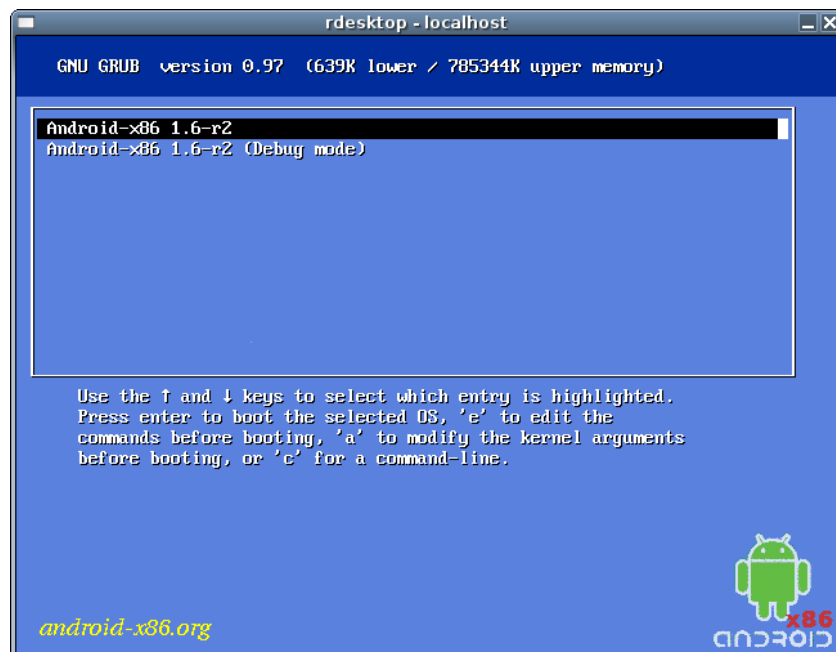

Figure 27: Android Call Confirmation

Boot ...

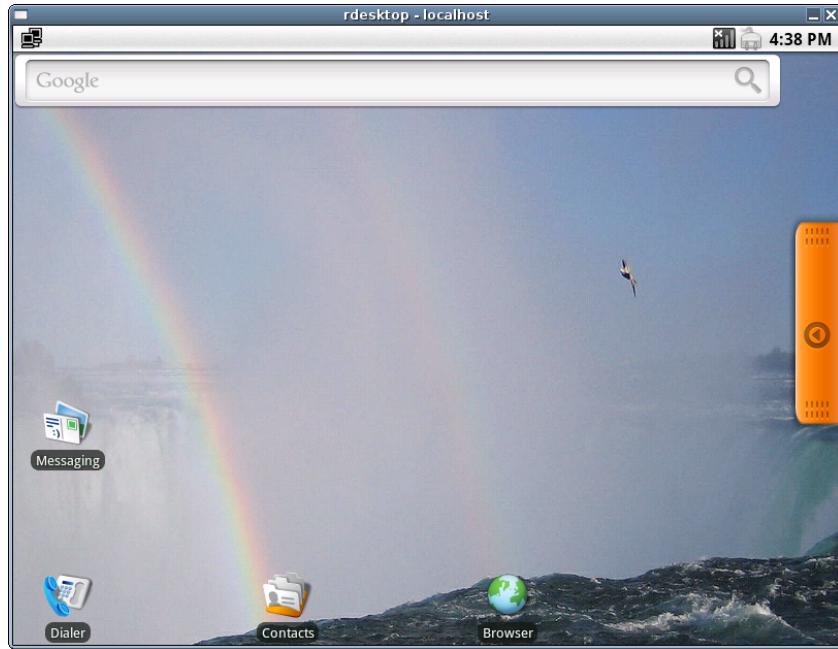Figure 28: Boot Android

...and enjoy Android.

Figure 29: Enjoy Android

7   Manage the VM

For now no native plugin for Android is supported.

8 SEE ALSO

ctys(1) , *ctys-configuration-QEMU(7)* , *ctys-configuration-VBOX(7)* , *ctys-createConfVM(1)* , *ctys-QEMU(1)* , *ctys-uc-QEMU(7)* , *ctys-uc-VBOX(7)* , *ctys-VBOX(1)* , *ctys-vhost(1)*
For System Tools:
Android: [http://www.android.com]
Android-x86: [http://www.android-x86.org]

9 AUTHOR

Maintenance:	<acue_sf1@sourceforge.net>
Homepage:	<http://www.UnifiedSessionsManager.org>
Sourceforge.net:	<http://sourceforge.net/projects/ctys>
Berlios.de:	<http://ctys.berlios.de>
Commercial:	<http://www.i4p.com>

10 COPYRIGHT

Copyright (C) 2008, 2009, 2010 Ingenieurbuero Arno-Can Uestuensoez
For BASE package following licenses apply,

for software see GPL3 for license conditions,

for documents see GFDL-1.3 with invariant sections for license conditions,
This document is part of the **DOC package**,

- for documents and contents from DOC package see

  'Creative-Common-Licence-3.0 - Attrib: Non-Commercial, Non-Deriv'

  with optional extensions for license conditions.

For additional information refer to enclosed Releasenotes and License files.