LINUTRONIX

LINUX FOR INDUSTRY

LINUTRONIX
LINUX FOR INDUSTRY

elbe

embedded linux build environment

Manuel Traut <manut@linutronix.de>

Chemnitzer Linuxtage 2015

March 21, 2015

overview

1) What is elOe?

overview

1) What is elbe?

2) Example: SDCARD Image for Beaglebone Black:

- Debian RFS (wheezy/armhf)
- customized linux kernel
- bootloader

overview

1) What is elbe?

2) Example: SDCARD Image for Beaglebone Black:

- Debian RFS (wheezy/armhf)
- customized linux kernel
- bootloader

3) What's next?

what is elbe?


```
1   open-embedded
2
3
4 buildroot
5
6           ptxdist
7
8       yocto
```

what is elbe?


```
1    open-embedded
2
3
4  buildroot
5
6              ptxdist
7
8         yocto
```

elbe

uses Debian packages and infrastructure

what is elbe?

highlights

- reproducable images (for embedded targets, virtual machines, PCs)

what is elbe?

highlights

- reproducable images (for embedded targets, virtual machines, PCs)
- security

what is elbe?

highlights

- reproducable images (for embedded targets, virtual machines, PCs)
- security
- licence informations

what is elbe?

highlights

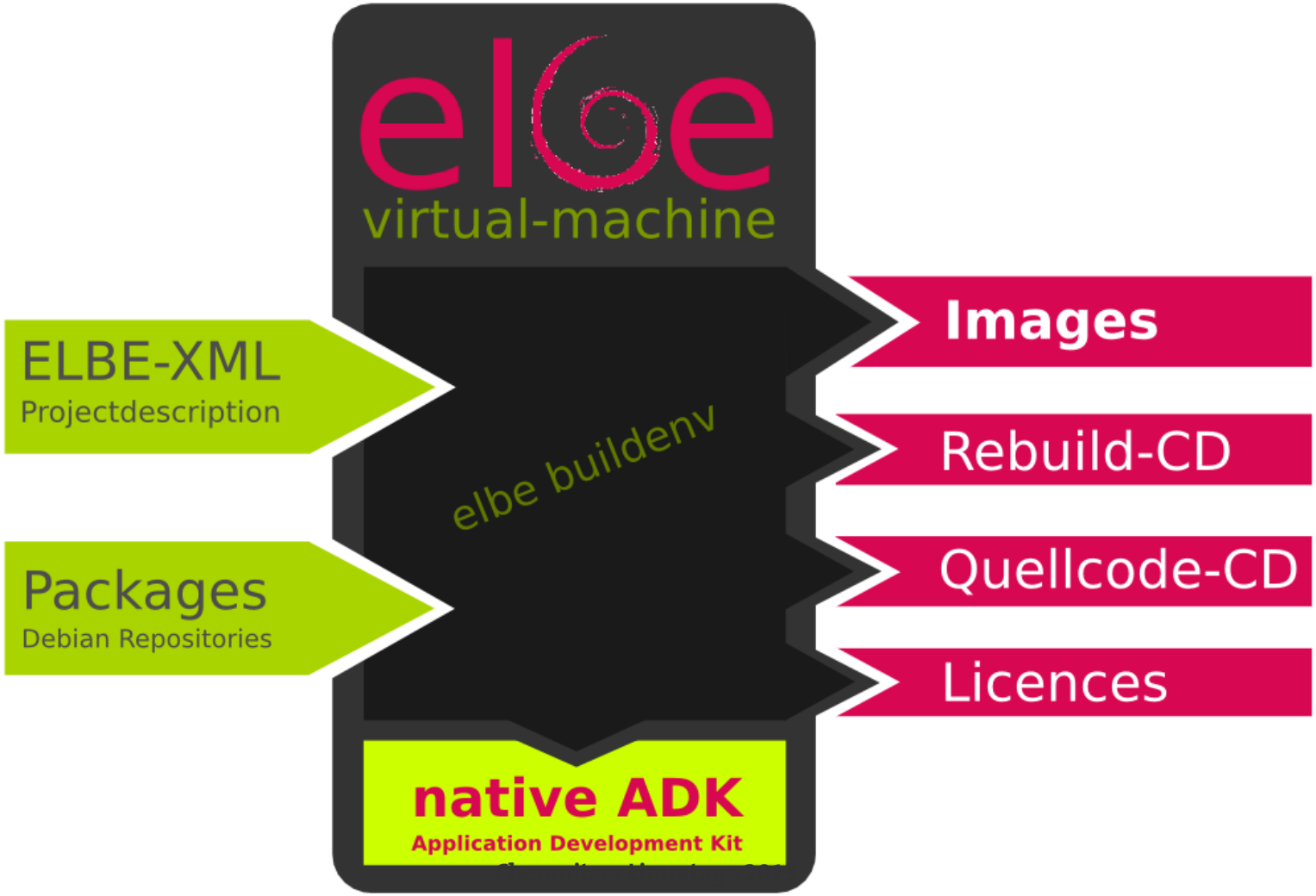
- reproducable images (for embedded targets, virtual machines, PCs)
- security
- licence informations
- source-code cdrom

what is elbe?

highlights

- reproducable images (for embedded targets, virtual machines, PCs)
- security
- licence informations
- source-code cdrom
- no cross compilation needed

what is elbe?


elbe
virtual-machine
ELBE-XML
Projectdescription
Packages
Debian Repositories
elbe buildenv
native ADK
Application Development Kit
Images
Rebuild-CD
Quellcode-CD
Licences

init

create a new elbe virtual-machine:

```
$ elbe init example.xml
```

init

create a new elbe virtual-machine:

```
$ elbe init example.xml
```

- **creates a project directory and Makefile**
- **downloads the elbe-bootstrap package**

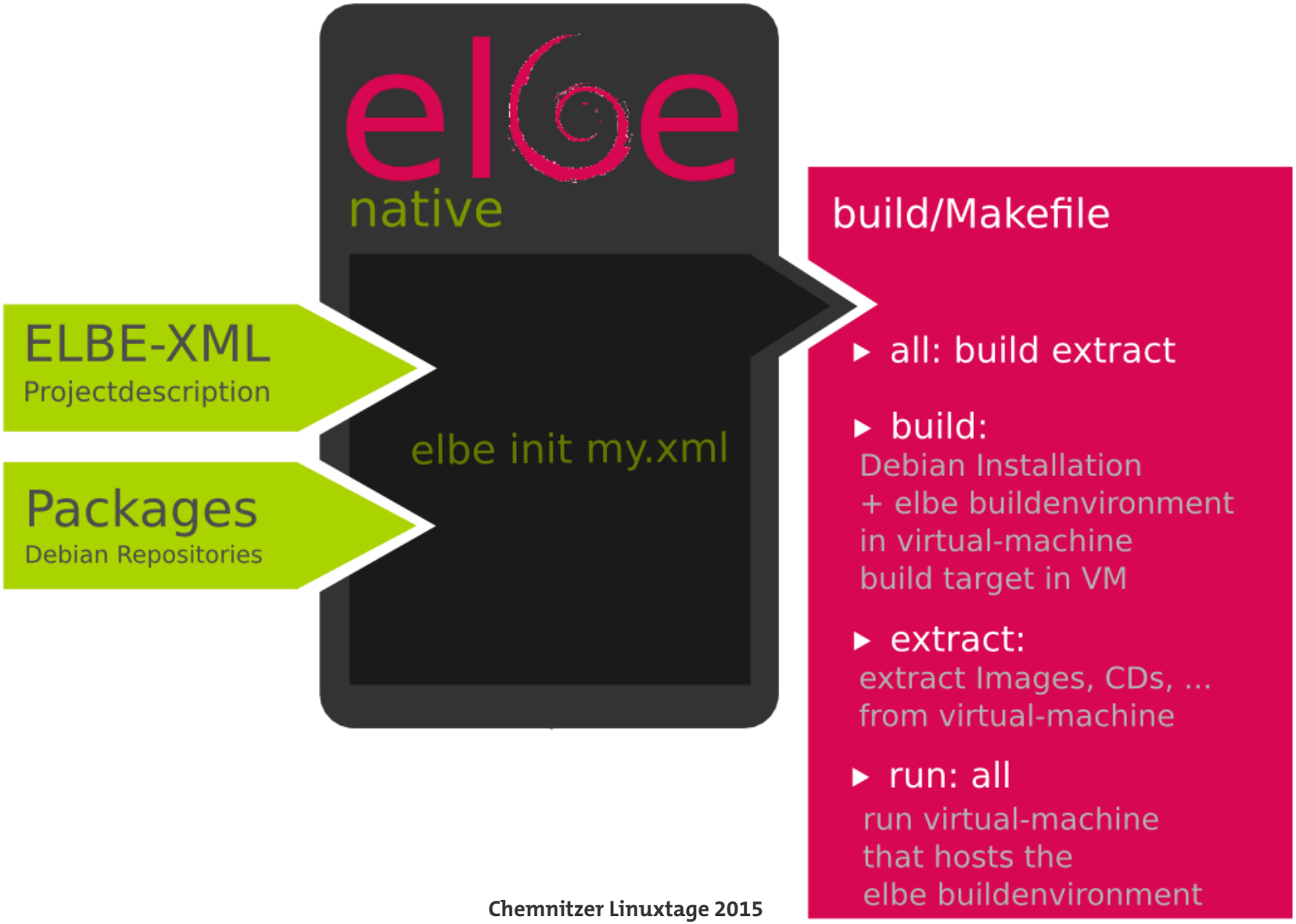
initvm section

```
1 <initvm>
2   <buildtype>amd64</buildtype>
3   <mirror>
4     <primary_host>ftp.tu-chemnitz.de</primary_host>
5     <primary_path>/pub/linux/debian/debian</primary_path>
6     <primary_proto>http</primary_proto>
```

initvm section

```
1 <initvm>
2   <buildtype>amd64</buildtype>
3   <mirror>
4     <primary_host>ftp.tu-chemnitz.de</primary_host>
5     <primary_path>/pub/linux/debian/debian</primary_path>
6     <primary_proto>http</primary_proto>
```

```
1     <!-- elbe-buildenv -->
2     <url-list> <url> <binary>
3       http://debian.linutronix.de/elbe-testing wheezy main
4     </binary> </url> </url-list>
5   </mirror>
6   <suite>wheezy</suite>
7 </initvm>
```


elbe
native
ELBE-XML
Projectdescription
Packages
Debian Repositories
elbe init my.xml
build/Makefile
all: build extract
build:
Debian Installation
+ elbe buildenvironment
in virtual-machine
build target in VM
extract:
extract Images, CDs, ...
from virtual-machine
run: all
run virtual-machine
that hosts the
elbe buildenvironment

buildchroot

build a project:

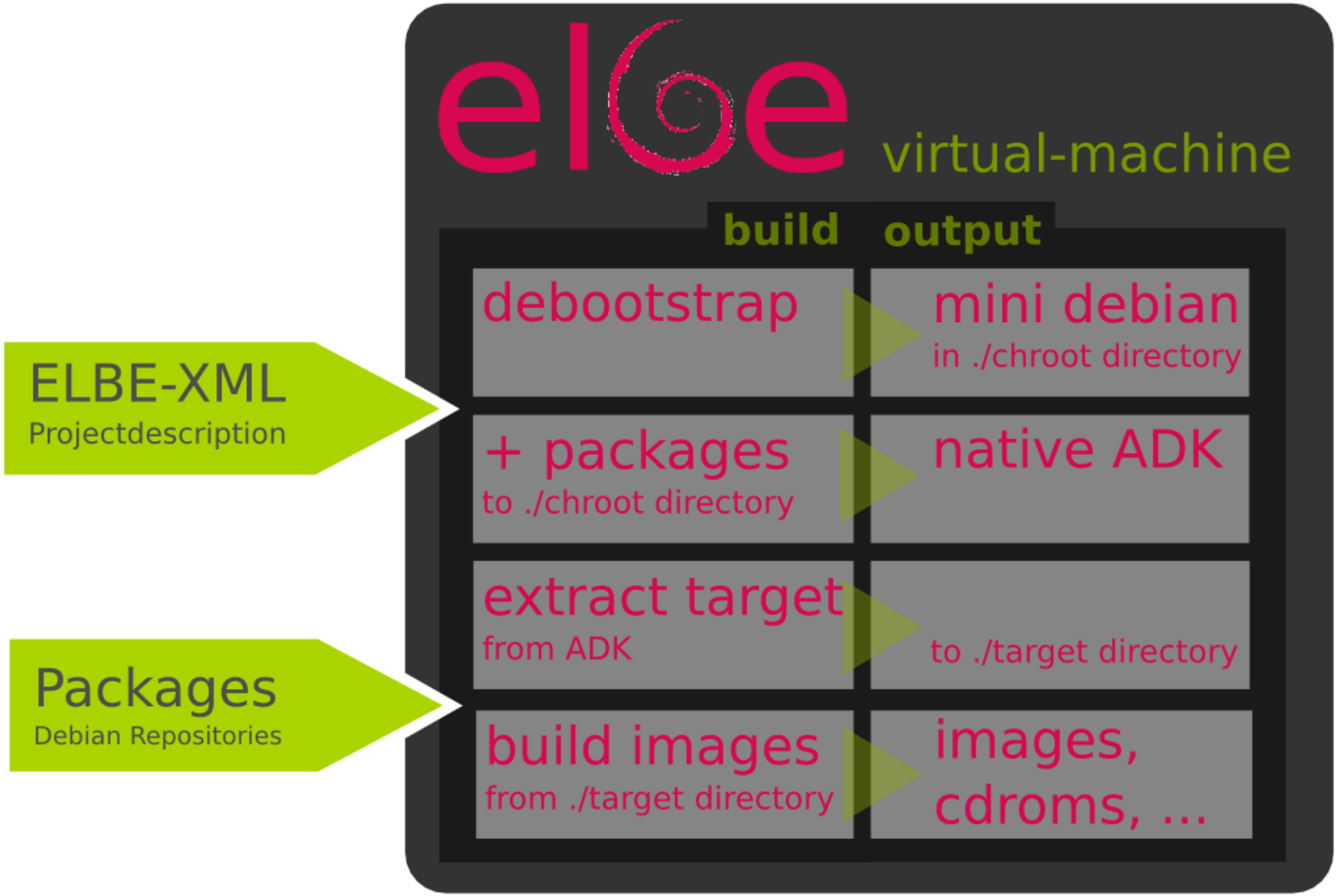
```
$ elbe buildchroot \
  -t /var/cache/elbe/build \
  -o /var/cache/elbe/build.log \
  beaglebone-black.xml
```

buildchroot

build a project:

```
$ elbe buildchroot \
  -t /var/cache/elbe/build \
  -o /var/cache/elbe/build.log \
  beaglebone-black.xml
```

- creates a project directory
- builds the project


elbe virtual-machine
build
output
debootstrap
mini debian
in ./chroot directory
+ packages
to ./chroot directory
native ADK
extract target
from ADK
to ./target directory
build images
from ./target directory
images,
cdroms, ...
ELBE-XML
Projectdescription
Packages
Debian Repositories

project section

```
<project>
  <name>beaglebone-black</name>
  <buildtype>armhf</buildtype>
  <mirror>
    <primary_host>ftp.tu-chemnitz.de</primary_host>
    <primary_path>/pub/linux/debian/debian</primary_path>
    <primary_proto>http</primary_proto>
```

project section

```
1 <project>
2   <name>beaglebone-black</name>
3   <buildtype>armhf</buildtype>
4   <mirror>
5     <primary_host>ftp.tu-chemnitz.de</primary_host>
6     <primary_path>/pub/linux/debian/debian</primary_path>
7     <primary_proto>http</primary_proto>
```

```
1     <!-- kernel -->
2     <url-list><url><binary>
3       http://debian.linutronix.de/elbe wheezy main
4     </binary></url></url-list>
5   </mirror>
6   <noauth />
7   <suite>wheezy</suite>
8 </project>
```

target section [1/3]

```
1 <target>
2   <hostname>bbb</hostname>
3   <passwd>foo</passwd>
4   <console>ttyS0,115200</console>
```

target section [1/3]

```
<target>
  <hostname>bbb</hostname>
  <passwd>foo</passwd>
  <console>ttyS0,115200</console>
```

```
  <images>
    <msdoshd>
      <name>sdcard.images</name>
      <size>1900MiB</size>
```

target section [1/3]

```
<target>
  <hostname>bbb</hostname>
  <passwd>foo</passwd>
  <console>ttyS0,115200</console>

  <images>
    <msdoshd>
      <name>sdcard.images</name>
      <size>1900MiB</size>

        <partition>
          <size>100MB</size>
          <label>boot</label>
          <bootable />
        </partition>
```

target section [1/3]

```
<target>
  <hostname>bbb</hostname>
  <passwd>foo</passwd>
  <console>ttyS0,115200</console>

  <images>
    <msdoshd>
      <name>sdcard.images</name>
      <size>1900MiB</size>

        <partition>
          <size>100MB</size>
          <label>boot</label>
          <bootable />
        </partition>

        <partition>
          <size>remain</size>
          <label>rfs</label>
        </partition>
    </msdoshd>
  </images>
```

target section [2/3]

```
1  <fstab>
2    <bylabel>
3      <label>boot</label>
4      <mountpoint>/boot</mountpoint>
5      <fs>
6        <type>vfat</type>
7      </fs>
8    </bylabel>
```

target section [2/3]

```
1  <fstab>
2    <bylabel>
3      <label>boot</label>
4      <mountpoint>/boot</mountpoint>
5      <fs>
6        <type>vfat</type>
7      </fs>
8    </bylabel>
```

```
1    <bylabel>
2      <label>rfs</label>
3      <mountpoint>/</mountpoint>
4      <fs>
5        <type>ext4</type>
6        <tune2fs>-i 0</tune2fs>
7      </fs>
8    </bylabel>
9  </fstab>
```

target section [3/3]

```
1  <finetuning>
2    <rm>/var/cache/apt/archives/*.deb</rm>
3    <mv path="/boot/vmlinuz-3.12.4-rt6-00013-ga8fd04d">
4      /boot/zImage
5    </mv>
6  </finetuning>
```

target section [3/3]

```
1   <finetuning>
2     <rm>/var/cache/apt/archives/*.deb</rm>
3     <mv path="/boot/vmlinuz-3.12.4-rt6-00013-ga8fd04d">
4       /boot/zImage
5     </mv>
6   </finetuning>
```

```
1   <pkg-list>
2     <pkg>linux-image-3.12.4-rt6-00013-ga8fd04d</pkg>
3     <pkg>openssh-server</pkg>
4   </pkg-list>
5 </target>
```

target section [3/3]

```
1   <finetuning>
2     <rm>/var/cache/apt/archives/*.deb</rm>
3     <mv path="/boot/vmlinuz-3.12.4-rt6-00013-ga8fd04d">
4       /boot/zImage
5     </mv>
6   </finetuning>
```

```
1   <pkg-list>
2     <pkg>linux-image-3.12.4-rt6-00013-ga8fd04d</pkg>
3     <pkg>openssh-server</pkg>
4   </pkg-list>
5 </target>
```

```
1 <!-- /boot/[MLO, uboot.images] -->
2 <archive>
3 QlpoOTFBWSZTWbr9i8ICOcH//////
4 ...
5 </archive>
```

generated files

```
$ ls
buildenv.img          licence.txt
elbe-report.txt       sdcard.img
bin-cdrom.iso         source.xml
Makefile              validation.txt
```


ebian

current image size

231 MB

current image size

231 MB

common finetuning rules

-44 MB (rm /var/lib/apt/lists/*debian*)
-40 MB (rm /var/cache/apt/*.bin)
-40 MB (rm /usr/share/locale/*)
-13 MB (rm /usr/share/doc)
-08 MB (rm /usr/share/man*)

current image size

231 MB

common finetuning rules

-44 MB (rm /var/lib/apt/lists/*debian*)
-40 MB (rm /var/cache/apt/*.bin)
-40 MB (rm /usr/share/locale/*)
-13 MB (rm /usr/share/doc)
-08 MB (rm /usr/share/man*)

new rfs size

86 MB

modes

can be used to get very small images

modes

can be used to get very small images

setsel
- automatic dependency resolution (can be overridden)
- files from postinst-scripts are on the target
- dpkg and perl needs to be on the target

modes

can be used to get very small images

- setsel
  - automatic dependency resolution (can be overridden)
  - files from postinst-scripts are on the target
  - dpkg and perl needs to be on the target
- diet
  - reverse dependency resolution
  - postinst-scripts are executed (but may fail)

modes

can be used to get very small images

- **setsel**
  - automatic dependency resolution (can be overridden)
  - files from postinst-scripts are on the target
  - dpkg and perl needs to be on the target
- **diet**
  - reverse dependency resolution
  - postinst-scripts are executed (but may fail)
- **tighten**
  - no dependency resolution
  - files from postinst-scripts are not on the target


by: Jaane Wendt

using the native ADK

- **start the virtual-machine:**

```
1 $ make run
```

using the native ADK

- **start the virtual-machine:**

```
$ make run
```

- **enter the native ADK:**

```
$ elbe chroot /var/cache/elbe/build
```

using the native ADK

- **start the virtual-machine:**

```
$ make run
```

- **enter the native ADK:**

```
$ elbe chroot /var/cache/elbe/build
```

- **build your application like on a native Debian machine**

using the native ADK

- start the virtual-machine:

```
$ make run
```

- enter the native ADK:

```
$ elbe chroot /var/cache/elbe/build
```

- build your application like on a native Debian machine
- package own application as debian package (dh_make)

using the native ADK

- start the virtual-machine:

```
$ make run
```

- enter the native ADK:

```
$ elbe chroot /var/cache/elbe/build
```

- build your application like on a native Debian machine
- package own application as debian package (dh_make)
- host debian application in own repository (reprepro)

using the native ADK

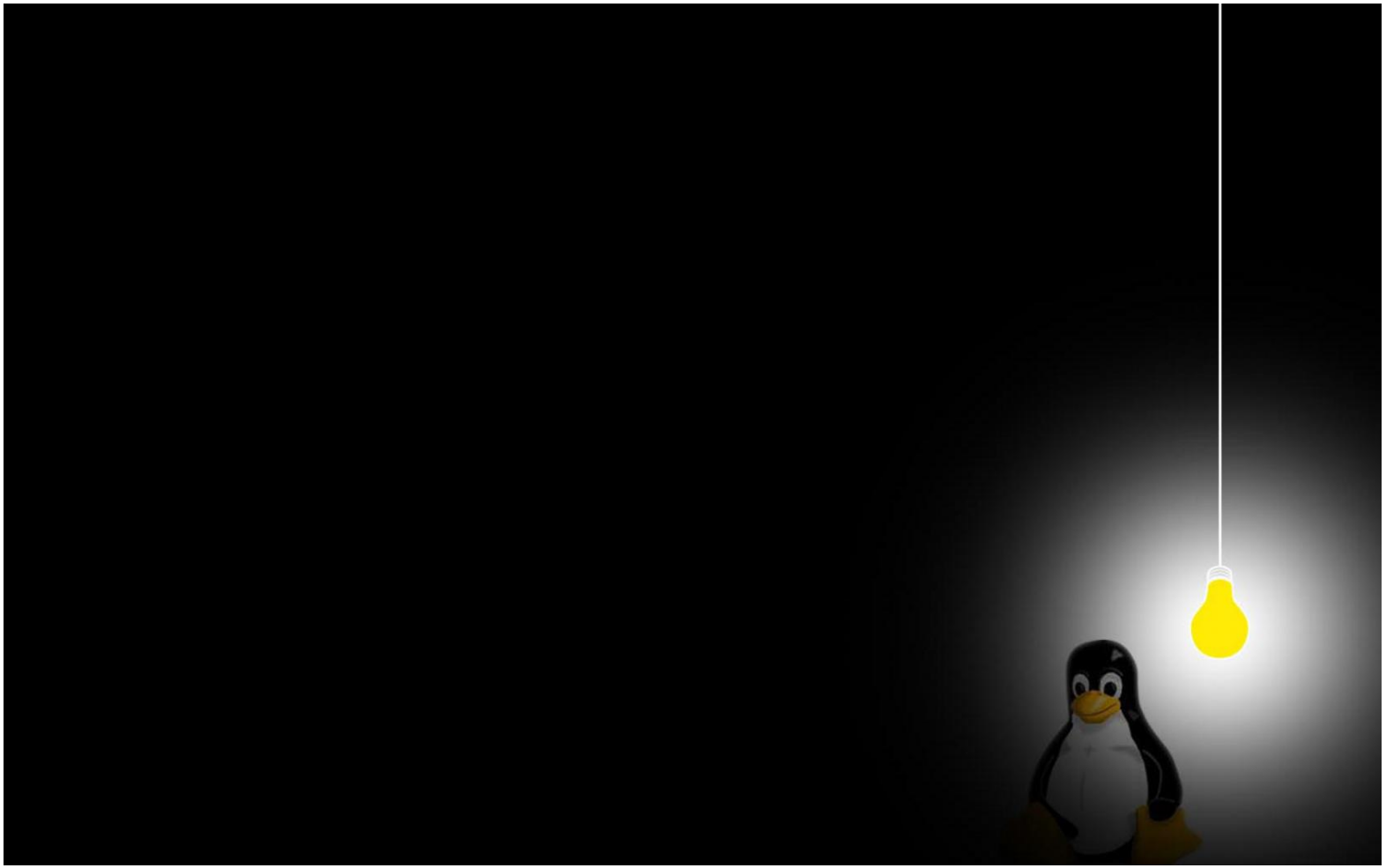
- start the virtual-machine:

```
$ make run
```

- enter the native ADK:

```
$ elbe chroot /var/cache/elbe/build
```

- build your application like on a native Debian machine
- package own application as debian package (dh_make)
- host debian application in own repository (reprepro)
- add own packages to XML description, rerun buildchroot



architecture

technologies

- python

architecture

technologies

- python
- qemu-user

architecture

technologies

- python
- qemu-user
- python-apt

architecture

technologies

- python
- qemu-user
- python-apt
- python-mako

architecture

technologies

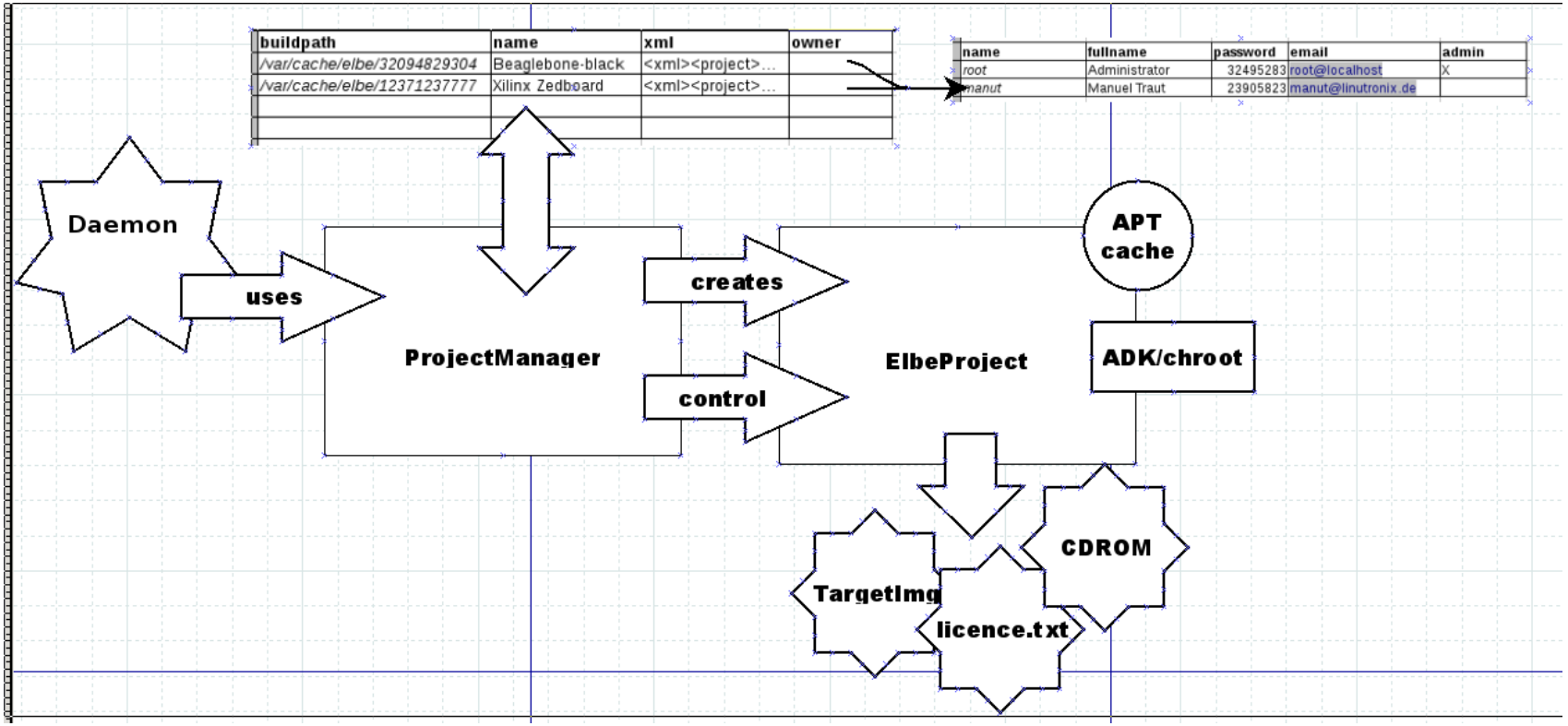
- python
- qemu-user
- python-apt
- python-mako
- python-parted

architecture

technologies

- python
- qemu-user
- python-apt
- python-mako
- python-parted
- sqlalchemy

architecture



SOAP interface / automated testing

open issues for 1.0 release

- CDROM generation

open issues for 1.0 release

- CDROM generation
- migrate examples to Debian/jessie (jessie/systemd support is done)

install

```
# on a Debian based system
$ echo 'deb http://debian.linutronix.de/elbe-testing \
        wheezy main' >> /etc/apt/sources.list
$ aptitude install elbe
```

resources

website http://elbe-rfs.org/

resources

website http://elbe-rfs.org/

mailing-list https://linutronix.de/mailman/listinfo/elbe-devel

resources

website	http://elbe-rfs.org/
mailing-list	https://linutronix.de/mailman/listinfo/elbe-devel
IRC	#elbe @freenode

resources

website	http://elbe-rfs.org/
mailing-list	https://linutronix.de/mailman/listinfo/elbe-devel
IRC	#elbe @freenode
github	https://github.com/linutronix/elbe

resources

website	http://elbe-rfs.org/
mailing-list	https://linutronix.de/mailman/listinfo/elbe-devel
IRC	#elbe @freenode
github	https://github.com/linutronix/elbe

```
git clone https://github.com/Linutronix/elbe.git
git checkout -b devel/elbe-1.0 -t devel/elbe-1.0
```