

# Buildroot

**Making embedded Linux easy?  
A real-life example.**

- Introduction
- Buildroot overview
- Br2-external
- Additional infrastructure
- Conclusion

Yann E. MORIN <yann.morin@orange.com>

Embedded Linux Conference Europe 2017



# Introduction

- About me
- Context
- Team
- Build environment

# About me

"Yann E. MORIN" <yann.morin@orange.com>

- Linux
- Embedded
- Security
- Network
- FLOSS

The other "Yann E. MORIN"

- Linux
- Embedded
- Realtime
- Security
- Network
- FLOSS

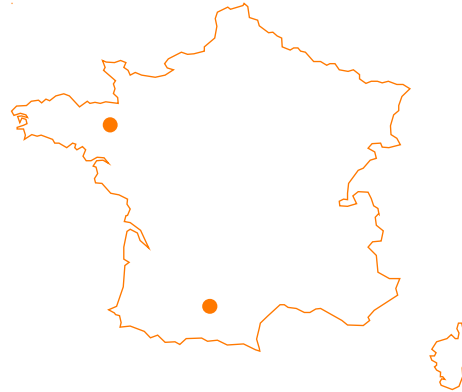
# Context

## Targets: STB, STB-like

- Different generations
  - Performance
  - R&D vs. production
  - Inherit constraints
- Part of the firmware
  - Not the main part
  - Aggregation of multiple third-parties
  - External integrator
- Complete rewrite from scratch

## Team

- Three teams, ~30 developers
- Two geographical locations
- Application developers
  - Not all Linux experts
  - Not all embedded experts



# Choosing a development environment – Late 2013

## Requirements

- Generic buildsystem
  - Target agnostic
  - Run-time agnostic
- Easy to learn and use
- No build-time overhead
- Existing solution
- Not my choice!

## SoftCo SDK

- Dedicated to the production devices
- Very specific, very custom

## OpenEmbedded

- Distribution generator
- Very versatile, highly customisable
- Steep learning curve
- No in-house knowledge

## Buildroot

- Firmware generator
- Flexible, extendable (BR2\_EXTERNAL!)
- Moderate (leaning to easy) learning curve
- In-house knowledge

## Others: OpenWrt, OE-lite, PTXdist...

- Much smaller communities (!OpenWrt)
- Not really investigated, no in-house knowledge

# Buildroot overview



- Quick introduction
- Build sequence
- Package infrastructure
- Br2-external

# Buildroot overview - Quick Introduction

## Buildroot

- Is a simple, efficient and easy-to-use tool to generate embedded Linux systems through cross-compilation
- Is entirely community-driven
- Uses kconfig-driven configuration
- Uses Makefiles
- Has a website <https://buildroot.org/>
- Has a manual <https://buildroot.org/manual.html>
- Is fun!

```
#####  
#  
# lpeg  
#  
#####  
  
LPEG_VERSION = 1.0.1-1  
LPEG_LICENSE = MIT  
  
$(eval $(luarocks-package))  
  
-----  
  
#####  
#  
# fping  
#  
#####  
  
FPING_VERSION = 4.0  
FPING_SITE = http://fping.org/dist  
FPING_LICENSE = BSD-like  
FPING_LICENSE_FILES = COPYING  
  
$(eval $(autotools-package))
```

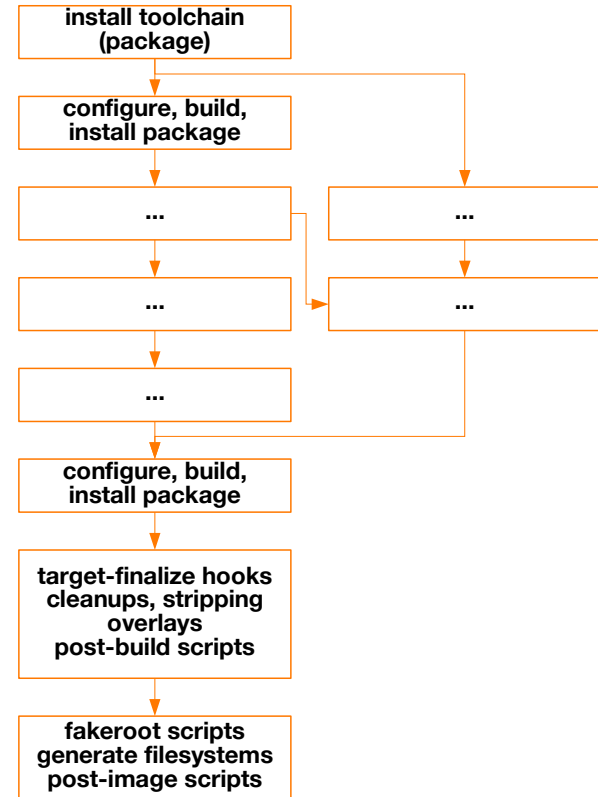
# Buildroot overview – Build sequence

## Buildroot overall build sequence

- Build packages, in sequence
  - Dependency chain
- Finalise target
  - Cleanups: .a .la .h man...
  - Stripping
- Generate the filesystem image(s)
  - A tarball is a filesystem image ;-)

## Extras

- target-finalize hooks
- post-build, fakeroot, post-image scripts





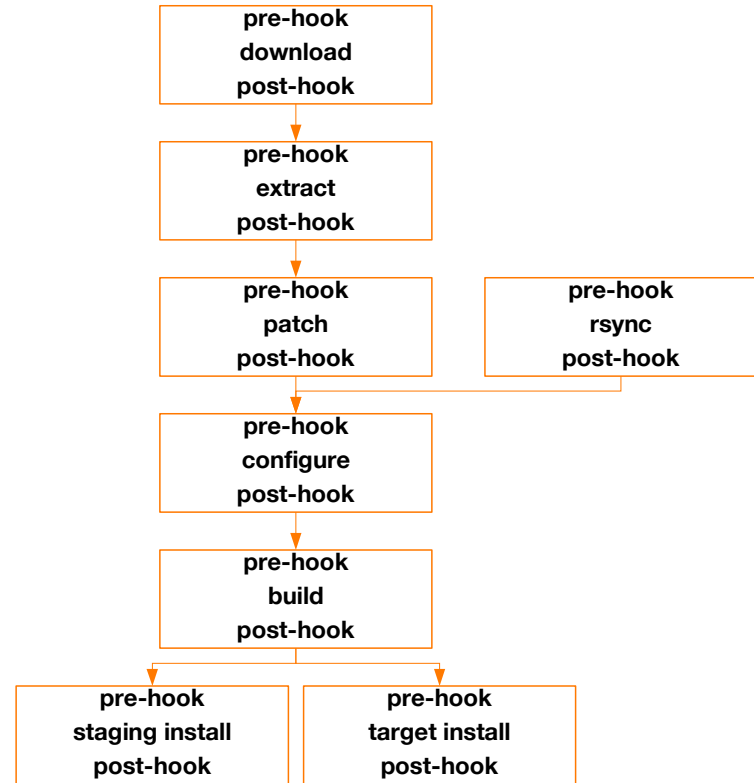
# Buildroot overview – Package infrastructure

## Buildroot package infrastructure

- Download
- Extract
- Patch
- Configure
- Build
- [Install in staging/]
- Install in target/

## Extras

- Local rsync during development
- Pre- and post-hooks for each step



# Buildroot overview - BR2\_EXTERNAL

## Local extensions

- **Without touching the Buildroot tree**
  - For proprietary packages
  - Staging area before upstreaming
- Access to all of Buildroot infrastructures
- Stackable

## Extensibility

- Defconfigs
- Packages
- Filesystems
- Bootloaders
- Board files
- Extra logic

# Br2-external

- Warming-up
- Buildroot
- Defconfigs
- Packages
- Filesystems
- Board files
- Extra logic

# BR2\_EXTERNAL - Warming-up

## Local customisations

```
$ ls -hlF
-rw-rw-r-- 1 ymorin ymorin  0 Sep 12 08:49 Config.in
-rw-rw-r-- 1 ymorin ymorin  0 Sep 12 08:49 external.mk
```

# BR2\_EXTERNAL - Buildroot

## Local customisations

- **Buildroot as a git submodule**

```
$ ls -hlF
drwxrwxr-x 13 ymorin ymorin 4.0K Aug 29 15:57 buildroot/
-rw-rw-r--  1 ymorin ymorin   0 Sep 12 08:49 Config.in
-rw-rw-r--  1 ymorin ymorin   0 Sep 12 08:49 external.mk
```

# BR2\_EXTERNAL - Defconfigs

## Local customisations

- Buildroot as a git submodule
- New configurations

```
$ ls -hlF
drwxrwxr-x 13 ymorin ymorin 4.0K Aug 29 15:57 buildroot/
drwxrwxr-x  2 ymorin ymorin 4.0K Sep 12 08:52 configs/
-rw-rw-r--  1 ymorin ymorin   0 Sep 12 08:49 Config.in
-rw-rw-r--  1 ymorin ymorin   0 Sep 12 08:49 external.mk
```

```
$ ls -hlF configs/
-rw-rw-r--  1 ymorin ymorin 4.0K Sep 12 08:52 board1_dev_defconfig
-rw-rw-r--  1 ymorin ymorin 3.2K Sep 12 08:52 board1_e2e_defconfig
-rw-rw-r--  1 ymorin ymorin 3.0K Sep 12 08:52 board1_prod_defconfig
-rw-rw-r--  1 ymorin ymorin 4.2K Sep 12 08:52 board2_dev_defconfig
-rw-rw-r--  1 ymorin ymorin 3.3K Sep 12 08:52 board2_e2e_defconfig
-rw-rw-r--  1 ymorin ymorin 3.1K Sep 12 08:52 board2_prod_defconfig
```

# BR2\_EXTERNAL - Packages

## Local customisations

- Buildroot as a git submodule
- New configurations
- New packages

```
$ ls -hlF
drwxrwxr-x 13 ymorin ymorin 4.0K Aug 29 15:57 buildroot/
drwxrwxr-x  2 ymorin ymorin 4.0K Sep 12 08:52 configs/
drwxrwxr-x  2 ymorin ymorin 4.0K Sep 12 08:52 package/
-rw-rw-r--  1 ymorin ymorin 6.4K Sep 12 08:49 Config.in
-rw-rw-r--  1 ymorin ymorin  113 Sep 12 08:49 external.mk

$ ls -hlF package/
drwxrwxr-x 7 ymorin ymorin 4.0K Aug 29 15:40 applayer/
drwxrwxr-x 5 ymorin ymorin 4.0K Aug 29 15:40 thirdparty/
drwxrwxr-x 9 ymorin ymorin 4.0K Aug 29 15:40 tools/

$ ls -hlF package/applayer/
drwxrwxr-x 7 ymorin ymorin 4.0K Aug 29 15:40 orange-epg/
drwxrwxr-x 5 ymorin ymorin 4.0K Aug 29 15:40 orange-fonts/
drwxrwxr-x 5 ymorin ymorin 4.0K Aug 29 15:40 orange-images/
drwxrwxr-x 5 ymorin ymorin 4.0K Aug 29 15:40 orange-live/
drwxrwxr-x 9 ymorin ymorin 4.0K Aug 29 15:40 orange-libdatamodel/
drwxrwxr-x 9 ymorin ymorin 4.0K Aug 29 15:40 orange-pvr/

$ ls -hlF package/tools/
drwxrwxr-x 7 ymorin ymorin 4.0K Aug 29 15:40 orange-gp-tools/
drwxrwxr-x 7 ymorin ymorin 4.0K Aug 29 15:40 orange-devel-tools/

$ cat external.mk
include $(sort $(wildcard $(BR2_EXTERNAL)/package/*/*/*.mk))

$ cat Config.in
source "$BR2_EXTERNAL/package/applayer/orange-init/Config.in"
source "$BR2_EXTERNAL/package/applayer/orange-live/Config.in"
source "$BR2_EXTERNAL/package/applayer/orange-epg/Config.in"
source "$BR2_EXTERNAL/package/tools/orange-devel-tools/Config.in"
source "$BR2_EXTERNAL/package/thirdparty/foo-blbla/Config.in"
[...]
```

# BR2\_EXTERNAL - Filesystems

## Local customisations

- Buildroot as a git submodule
- New configurations
- New packages
- New filesystems

```
$ ls -hlF
drwxrwxr-x 13 ymorin ymorin 4.0K Aug 29 15:57 buildroot/
drwxrwxr-x  2 ymorin ymorin 4.0K Sep 12 08:52 configs/
drwxrwxr-x  2 ymorin ymorin 4.0K Sep 12 09:11 fs/
drwxrwxr-x  2 ymorin ymorin 4.0K Sep 12 08:52 package/
-rw-rw-r--  1 ymorin ymorin 6.7K Sep 12 08:49 Config.in
-rw-rw-r--  1 ymorin ymorin  183 Sep 12 08:49 external.mk

$ ls -hlF fs/
drwxrwxr-x 2 ymorin ymorin 4.0K Sep  7 15:36 orange-gp/

$ cat fs/orange-gp/orange-gp.mk
ROOTFS_ORANGE_GP_DEPENDENCIES = rootfs-tar orange-gp-tools
define ROOTFS_ORANGE_GP_CMD
mk-gp -r $(BINARIES_DIR)/rootfs.tar -o $(@)
endef
$(eval $(call ROOTFS_TARGET,orange-gp))
```



# BR2\_EXTERNAL - Board files

## Local customisations

- Buildroot as a git submodule
- New configurations
- New packages
- New filesystems
- Board files

```
$ ls -hlF
drwxrwxr-x 13 ymorin ymorin 4.0K Aug 29 15:57 buildroot/
drwxrwxr-x  2 ymorin ymorin 4.0K Sep 12 08:59 boards/
drwxrwxr-x  2 ymorin ymorin 4.0K Sep 12 08:52 configs/
drwxrwxr-x  2 ymorin ymorin 4.0K Sep 12 09:11 fs/
drwxrwxr-x  2 ymorin ymorin 4.0K Sep 12 08:52 package/
-rw-rw-r--  1 ymorin ymorin 6.7K Sep 12 08:49 Config.in
-rw-rw-r--  1 ymorin ymorin  183 Sep 12 08:49 external.mk
```

```
$ ls -hlF boards/common/
drwxrwxr-x  2 ymorin ymorin 4.0K Sep 12 08:59 skeleton/
drwxrwxr-x  2 ymorin ymorin 4.0K Sep 12 08:59 overlay-tests/
-rwxrwxr-x  2 ymorin ymorin 4.0K Sep 12 08:59 post-build.sh
-rwxrwxr-x  2 ymorin ymorin 4.0K Sep 12 08:59 post-build-tests.sh
```

```
$ ls -hlF boards/board1/
drwxrwxr-x  2 ymorin ymorin 4.0K Sep 12 08:59 overlay/
```

```
$ ls -hlF boards/board2/
drwxrwxr-x  2 ymorin ymorin 4.0K Sep 12 08:59 overlay/
```

# BR2\_EXTERNAL - Extra logic

## Local customisations

- Buildroot as a git submodule
- New configurations
- New packages
- New filesystems
- Board files
- Extra logic - Additional infrastructure

```
$ ls -hlF
drwxrwxr-x 13 ymorin ymorin 4.0K Aug 29 15:57 buildroot/
drwxrwxr-x  2 ymorin ymorin 4.0K Sep 12 08:59 boards/
drwxrwxr-x  2 ymorin ymorin 4.0K Sep 12 08:52 configs/
drwxrwxr-x  2 ymorin ymorin 4.0K Sep 12 08:57 core/
drwxrwxr-x  2 ymorin ymorin 4.0K Sep 12 09:11 fs/
drwxrwxr-x  2 ymorin ymorin 4.0K Sep 12 08:52 package/
-rw-rw-r--  1 ymorin ymorin 6.7K Sep 12 08:49 Config.in
-rw-rw-r--  1 ymorin ymorin  237 Sep 12 08:49 external.mk

$ ls -hlF core/
-rw-rw-r--  1 ymorin ymorin 5.7K Sep 12 08:49 pkg-orange.apparmor.mk
-rw-rw-r--  1 ymorin ymorin 2.3K Aug 29 15:40 pkg-orange.config.mk
-rw-rw-r--  1 ymorin ymorin 2.3K Aug 29 15:40 pkg-orange.dbus.mk
-rw-rw-r--  1 ymorin ymorin 611 Aug 29 15:40 pkg-orange.doc.mk
-rw-rw-r--  1 ymorin ymorin 1.4K Sep  8 16:22 pkg-orange.links-list.mk
-rw-rw-r--  1 ymorin ymorin 1.6K Sep  7 15:36 pkg-orange.mk
-rw-rw-r--  1 ymorin ymorin 553 Aug 29 15:40 pkg-orange.services.mk
-rw-rw-r--  1 ymorin ymorin 783 Sep 12 08:49 pkg-orange.translation.mk
-rw-rw-r--  1 ymorin ymorin 1.2K Aug 29 15:40 pkg-orange.users.mk
-rw-rw-r--  1 ymorin ymorin 2.9K Sep 11 14:25 system.mk
-rw-rw-r--  1 ymorin ymorin 7,5K Sep 11 14:47 system-extra.mk

$ cat external.mk
include $(BR2_EXTERNAL)/core/pkg-orange.mk
include $(BR2_EXTERNAL)/core/system.mk
include $(BR2_EXTERNAL)/core/system-extra.mk

$ cat core/pkg-orange.mk
# Define variable so it is recursively expanded
orange-package =
include $(BR2_EXTERNAL)/core/pkg-orange.users.mk
include $(BR2_EXTERNAL)/core/pkg-orange.doc.mk
include $(BR2_EXTERNAL)/core/pkg-orange.dbus.mk
include $(BR2_EXTERNAL)/core/pkg-orange.translation.mk
include $(BR2_EXTERNAL)/core/pkg-orange.config.mk
include $(BR2_EXTERNAL)/core/pkg-orange.services.mk
include $(BR2_EXTERNAL)/core/pkg-orange.apparmor.mk
include $(BR2_EXTERNAL)/core/pkg-orange.links-list.mk
```

# Additional infrastructure

# Additional infrastructure – Raw make rules

## Raw Makefile code

- Can be anything
  - As long as it does not clash with existing infrastructure!
- Example: check dependencies
  - Ensure packages define a valid dependency chain
  - But does not catch inherited dependencies like A -> C when A -> B and B -> C

```
$ cat core/system-extra.mk
# Check that packages correctly declare their dependencies
ifeq ($(CHECK_PACKAGES),)
CHECK_PACKAGES := orange-% host-orange-%
endif
check-deps:
@printf "Checking packages one by one; can take many hours...\n"
@for p in $(filter $(CHECK_PACKAGES),$(PACKAGES)); do \
printf "Checking %s.." "$${p}"; \
make -C $(CONFIG_DIR) clean >/dev/null 2>&1; \
if make -C $(CONFIG_DIR) "$${p}" >"$(CONFIG_DIR)/$${p}.log" 2>&1; then \
if grep -E '.*Circular (.+ <- .+) dependency dropped\.$$$' \
"$(CONFIG_DIR)/$${p}.log" >/dev/null 2>&1; \
then \
printf " FAILED: circular dependency\n"; \
else \
printf " OK\n"; \
fi; \
else \
printf " FAILED: missing dependency?\n"; \
fi; \
done
```

# Additional infrastructure – target-finalize

## Elf sanitising

- **Target-finalize hook**
  - **Remove symlinks to shared libraries**
  - **Rename shared libraries to their SONAME**
- **Offload to a helper script (shell, python, perl, whatever you fancy)**

```
$ cat core/system.mk
define ORANGE_TARGET_FINALIZE_CLEAN_ELF_LIBS
    $(BR2_EXTERNAL)/tools/clean-elf-libs $(TARGET_DIR) $(TARGET_READELF)
endef
TARGET_FINALIZE_HOOKS += ORANGE_TARGET_FINALIZE_CLEAN_ELF_LIBS
```

# Additional infrastructure – Basics

## Orange package

- Is a standard Buildroot package
- With a little addition
- Mimicking the existing infrastructures
- Using declarative statements

```
$ cat external.mk
ORANGE_GIT_BASE_URL = git-user@forge.orange.com:the-project
```

---

```
$ cat package/applayer/orange-live/orange-live.mk
#####
#
# Orange LIVE
#
#####

ORANGE_LIVE_VERSION = 27fcb725d6f3ca7ea5d684cfdaf4124515379552
ORANGE_LIVE_SITE = $(ORANGE_GIT_BASE_URL)/src/applayer/apps/live.git
ORANGE_LIVE_SITE_METHOD = git
ORANGE_LIVE_LICENSE = Proprietary
ORANGE_LIVE_REDISTRIBUTE = NO
ORANGE_LIVE_INSTALL_STAGING = YES

ORANGE_LIVE_DEPENDENCIES = \
    orange-libdatamodel \
    orange-libtracer \
    orange-backends

$(eval $(orange-package))
$(eval $(cmake-package))
```

# Additional infrastructure – Documentation

## Documentation

- Let packages handle the documentation how they want: static, or generated with doxygen, docstring...
- Build the documentation
  - Post-build hook
- Install the documentation
  - Not a target content, not a staging content, not an image either...
  - Divert to a doc/ sub-directory in the standard Buildroot output directory
  - Hijack post-install-target hook
- One macro for each package (bad!)
  - Too-many dollars problem

```
$ cat core/pkg-orange.doc.mk
define orange-build-doc-inner

ifeq ($$(($(1)_HAS_DOC),YES)

define $(1)_BUILD_DOC
  $$$(MAKE) -C $$$(@D)/doc
endif
$(1)_POST_BUILD_HOOKS += $(1)_BUILD_DOC

define $(1)_INSTALL_DOC
  $$$(MAKE) -C $$$(@D)/doc install DESTDIR="$$$(BASE_DIR)/doc"
endif
$(1)_POST_INSTALL_TARGET_HOOKS += $(1)_INSTALL_DOC

endif

endif

orange-package += $(call orange-build-doc-inner,$(call UPPERCASE,$(pkgname)))
```

---

```
$ cat package/applayer/orange-live/orange-live.mk
ORANGE_LIVE_VERSION = 27fcb725d6f3ca7ea5d684cfdaf4124515379552
[...]
```

```
ORANGE_LIVE_HAS_DOC = YES
```

```
$(eval $(orange-package))
$(eval $(cmake-package))
```

# Additional infrastructure – Translations

## All based on Qt

- Build translation as post-build hook
- Install translations as post-install-target hook
- Automatically inherit dependency on qt5tools
  - Translations disabled if qt5tools-linguist not enabled
- One macro for all packages (good!)

```
$ cat core/pkg-orange.dbus.mk
define ORANGE_BUILD_TRANSLATION_FILES
    $(HOST_DIR)/usr/bin/lrelease $(@D)/translations/*.ts
endef
define ORANGE_INSTALL_TRANSLATION_FILES
    mkdir -p $(TARGET_DIR)/usr/share/orange/translations
    $(INSTALL) -m 0644 $(@D)/translations/*.qm $(TARGET_DIR)/usr/share/orange/translations/
endef

define orange-install-translation-inner
ifeq ($$(BR2_PACKAGE_QT5TOOLS_LINGUIST_TOOLS),y)
ifeq ($$(1)_HAS_TRANSLATIONS),YES)
$(1)_POST_BUILD_HOOKS += ORANGE_BUILD_TRANSLATION_FILES
$(1)_POST_INSTALL_TARGET_HOOKS += ORANGE_INSTALL_TRANSLATION_FILES
$(1)_DEPENDENCIES += qt5tools
endif
endif

orange-package += $(call orange-install-translation-inner,$(call UPPERCASE,$(pkgname)))
```

---

```
$ cat package/applayer/orange-live/orange-live.mk
ORANGE_LIVE_VERSION = 27fcb725d6f3ca7ea5d684cfdaf4124515379552
[...]
```

ORANGE\_LIVE\_HAS\_TRANSLATIONS = YES

```
$(eval $(orange-package))
$(eval $(cmake-package))
```



# Additional infrastructure – Configuration 1/2

## Configuration files

- Needed at runtime
  - Simple copy as a post-install-target hook

```
$ cat core/pkg-orange.config.mk
define ORANGE_INSTALL_CONFIG_FILES
    $(foreach cfg,$($(PKG)_CONFIG_FILES),\
        $(INSTALL) -m 0644 -D $$($(PKG)_PKGDIR)/config/$(cfg) \
            $(TARGET_DIR)/usr/share/orange/$(cfg)$(sep))
endef

define orange-install-config-inner
$(1)_POST_INSTALL_TARGET_HOOKS += ORANGE_INSTALL_CONFIG_FILES
endef

orange-package += $(call orange-install-config-inner,$(call UPPERCASE,$(pkgname)))
```

---

```
$ cat package/applayer/orange-live/orange-live.mk
ORANGE_LIVE_VERSION = 27fcb725d6f3ca7ea5d684cfdaf4124515379552
[...]
```

```
ORANGE_LIVE_CONFIG_FILES = \
    applicationmanager/apps/live.json \
    orwin/live.json
```

```
ORANGE_LIVE_INIT_APPLICATIONS = live
```

```
$(eval $(orange-package))
$(eval $(cmake-package))
```

---

```
$ cat package/applayer/services/orange-record/orange-record.mk
ORANGE_RECORD_VERSION = a9541f7764d423907dcfe033dec4c23e0e73734c
[...]
```

```
ORANGE_RECORD_INIT_SERVICES = record
ORANGE_RECORD_INIT_SYSTEM = record-backend
```

# Additional infrastructure – Configuration 2/2

## Registering applications

- Variables expanded in a post-hook
  - Not all packages already parsed
  - Evaluated later
- Four types of applications
  - System, started earliest
  - Services, started early
  - Init, started last
  - Standard, started on-demand

```
$ cat core/pkg-orange.services.mk
define orange-init-services-inner
ORANGE_INIT_SYSTEM      += $(if $$$$(1)_KCONFIG_VAR),$$$$(1)_INIT_SYSTEM)
ORANGE_INIT_SERVICES    += $(if $$$$(1)_KCONFIG_VAR),$$$$(1)_INIT_SERVICES)
ORANGE_INIT_APPLICATIONS += $(if $$$$(1)_KCONFIG_VAR),$$$$(1)_INIT_APPLICATIONS)
endef

orange-package += $(call orange-init-services-inner,$(call UPPERCASE,$(pkgname)))
```

---

```
$ cat package/applayer/orange-appenv/orange-appenv.mk
ORANGE_APPENV_VERSION = 266a8243e85d9df6b8694f73e488bedec4719e49
[...]

define ORANGE_APPENV_INIT_SERVICES_JSON
( \
  echo '{'; \
  echo '  "system":['; \
  (:; $(foreach s,$(ORANGE_INIT_SYSTEM),echo '"$(s)",'; )) |sed -r -e '$$$/,,$$//'; \
  echo '  ]'; \
  echo '  "services":['; \
  (:; $(foreach s,$(ORANGE_INIT_SERVICES),echo '"$(s)",'; )) |sed -r -e '$$$/,,$$//'; \
  echo '  ]'; \
  echo '  "applications":['; \
  (:; $(foreach s,$(ORANGE_INIT_APPLICATIONS),echo '"$(s)",'; )) |sed -r -e '$$$/,,$$//'; \
  echo '  ]'; \
  echo '}; \
) >$(@)/init-services.json
endef
ORANGE_APPENV_POST_BUILD_HOOKS += ORANGE_APPENV_INIT_SERVICES_JSON

define ORANGE_APPENV_INSTALL_INIT_SERVICES_JSON
$(INSTALL) -D -m 0644 $(@D)/init-services.json \
  $(TARGET_DIR)/usr/share/orange/init-services.json
endef
ORANGE_APPENV_POST_INSTALL_TARGET_HOOKS += ORANGE_APPENV_INSTALL_INIT_SERVICES_JSON

$(eval $(orange-package))
$(eval $(cmake-package))
```

# Additional infrastructure – Users

## Defining users

- **Standard Buildroot mechanism**
- **With additional checks for validity**
  - **Work on variables**
  - **Known at time of parsing**

```
$ cat core/pkg-orange.users.mk
define orange-users-inner

ifneq ($$(1)_USERS),)
$(1)_USER   = $$$(word 1,$$$(1)_USERS))
$(1)_UID    = $$$(word 2,$$$(1)_USERS))
$(1)_GID    = $$$(word 4,$$$(1)_USERS))

ifeq ($$(1)_UID,-1)
# allow group-only definition
ifneq ($$(1)_USER,-)
$$$(error UID for $(pkgname) must be explicit)
endif
endif
ifeq ($$(1)_GID,-1)
$$$(error GID for $(pkgname) must be explicit)
endif
ORANGE_ALL_USERS += $$$(1)_USERS
endif

endif

orange-package += $(call orange-users-inner,$(call UPPERCASE,$(pkgname)))
```

---

```
$ cat package/applayer/orange-live/orange-live.mk
ORANGE_LIVE_VERSION = 27fcb725d6f3ca7ea5d684cfdaf4124515379552
[...]

define ORANGE_LIVE_USERS
    live 10042 live 10042 * - - orange Orange Live application
endef

$(eval $(orange-package))
$(eval $(cmake-package))
```

# Additional infrastructure – D-Bus 1/2

## D-Bus authorisation

- **D-Bus APIs installed as XML descriptions**
  - **Generate code with qdbusxml2cpp**
- **Use D-Bus via QDBus**
  - **Well-known #include <ComOrangeAPIxxx.h>**
  - **Can scan code for those patterns**
  - **Post-build hook**
- **Two buses**
  - **Post-target-install hook for session bus**
  - **Post-staging-install hook for system bus**

```
$ cat core/pkg-orange.dbus.mk
define orange-dbus-inner

# Include interfaces defined by our dependencies
$(1)_INHERITED_DBUS_INTF = $$ (strip \
    $$ (foreach d, $$ ($(1)_DEPENDENCIES), \
        $$$ (call UPPERCASE, $$ (d))_EXPORT_DBUS_INTF))

ifeq ($$( $(1)_HAS_SYSTEM_DBUS_API), YES)

# Even if just for the dbus config files...
$(1)_INSTALL_STAGING = YES

$(1)_EXTRA_DBUS_INTF += $$$ ($(1)_INHERITED_DBUS_INTF)

define $(1)_GEN_DBUS_CONFIG
    $(BR2_EXTERNAL)/tools/gen-dbus-conf \
        -s $$ (@D) \
        -u $$$ ($(1)_UID) \
        $$ (foreach api, $$$ ($(1)_EXTRA_DBUS_INTF), -d $$ (api)) \
        -o $$ (@D)/system-dbus.conf
endif
$(1)_POST_BUILD_HOOKS += $(1)_GEN_DBUS_CONFIG

define $(1)_INSTALL_DBUS_CONFIG
    $$ (INSTALL) -m 0644 -D $$ (@D)/system-dbus.conf \
        $$$ (STAGING_DIR)/etc/dbus-1/system.d/orange-$$$ ($(1)_USER).conf
endif
$(1)_POST_INSTALL_STAGING_HOOKS += $(1)_INSTALL_DBUS_CONFIG

ORANGE_SYSTEM_DBUS_PACKAGES += $$$ (if $$$ ($(1)_KCONFIG_VAR), $(1))

endif

endif

orange-package += $(call orange-dbus-inner, $(call UPPERCASE, $(pkgname)))
```

# Additional infrastructure – D-Bus 2/2

## D-Bus authorisation

- Inherit interfaces exported by dependencies
- Provide extra, non-QDBus APIs
  - When two applications want to talk to each other, chicken-n-egg problem

```
$ cat package/applayer/orange-libdatamodel/orange-libdatamodel.mk  
ORANGE_LIBDATAMODEL_VERSION = 165179e7acbe7a56ee7fb8c66b390228ed2571bd  
[...]
```

```
ORANGE_LIBDATAMODEL_EXPORT_DBUS_INTF = com.orange.datamodel
```

```
$(eval $(orange-package))  
$(eval $(cmake-package))
```

---

```
$ cat package/applayer/orange-live/orange-live.mk  
ORANGE_LIVE_VERSION = 27fcb725d6f3ca7ea5d684cfdaf4124515379552  
[...]
```

```
ORANGE_LIVE_DEPENDENCIES = orange-libdatamodel
```

```
ORANGE_LIVE_HAS_SYSTEM_DBUS_API = YES  
ORANGE_LIVE_EXTRA_DBUS_INTF = com.orange.pvr
```

```
$(eval $(orange-package))  
$(eval $(cmake-package))
```

# Additional infrastructure – AppArmor 1/3

## AppArmor profiles

- Tedious task
- Bit-rot
  - But: security!
- Automatically generate profiles
  - Functionality-centric
  - Developer-provided
  - Build- and data-dependencies tracking
  - Code-scanning

```
$ cat package/applayer/orange-fonts/orange-fonts.mk
ORANGE_FONTS_VERSION = 85ff751c546166896e0899c88c2234abaa684ebb
[...]
ORANGE_FONTS_APPARMOR_DATA = YES
```

---

```
$ cat package/applayer/orange-fonts/apparmor/orange-fonts.aa
/usr/share/fonts/*.ttf r,
```

---

```
$ cat package/applayer/orange-libdatamodel/orange-libdatamodel.mk
ORANGE_LIBDATAMODEL_VERSION = 165179e7acbe7a56ee7fb8c66b390228ed2571bd
[...]
ORANGE_LIBDATAMODEL_APPARMOR_LIBS = /usr/lib/libdatamodel.so
```

---

```
$ cat package/applayer/orange-libdatamodel/apparmor/libdatamodel.so.aa
/etc/orange/datamodel.cfg r,
/run/datamodel.socket rw,
```

---

```
$ cat package/applayer/orange-live/orange-live.mk
ORANGE_LIVE_VERSION = 27fcb725d6f3ca7ea5d684cfdaf4124515379552
[...]
ORANGE_LIVE_DEPENDENCIES = orange-libdatamodel
ORANGE_LIVE_DATA_DEPENDENCIES = orange-fonts

ORANGE_LIVE_APPARMOR_EXECS = /usr/bin/live
```

---

```
$ cat package/applayer/orange-live/apparmor/live.aa
/etc/orange/live.cfg r,
/var/data/live.db rwk,
```

# Additional infrastructure – AppArmor 2/3

## AppArmor profiles

- **Post-install-staging hook**
  - Install libraries snippets
  - Install data snippets
- **Post-install-target hook**
  - Install ELF executable snippets
- **Register all executables**
  - Data may not already be installed
  - Can only be scanned at target-finalize
- **Handles static data**
  - Translations
  - ...

```
$ cat core/pkg-orange.apparmor.mk
define ORANGE_APPARMOR_INSTALL_EXECS_PROFILES
    $(foreach x,$($(PKG)_APPARMOR_EXECS),\
        $(INSTALL) -D -m 0644 \
            $($ (PKG)_PKGDIR)/apparmor/$(notdir $(x)).aa \
            $(TARGET_DIR)/$(x).aa
    )
endef

define ORANGE_APPARMOR_INSTALL_LIBS_PROFILES
    $(foreach l,$($(PKG)_APPARMOR_LIBS),\
        $(INSTALL) -D -m 0644 \
            $($ (PKG)_PKGDIR)/apparmor/$(notdir $(l)).aa \
            $(STAGING_DIR)/$(l).aa
    )
endef

define ORANGE_APPARMOR_INSTALL_DATA_PROFILES
    $(INSTALL) -D -m 0644 $($ (PKG)_PKGDIR)/apparmor/$( $(PKG)_RAWNAME ).aa \
        $(STAGING_DIR)/usr/lib/apparmor-data.d/$( $(PKG)_RAWNAME ).aa
endef

define orange-apparmor-inner

$(1)_POST_INSTALL_TARGET_HOOKS += \
    ORANGE_APPARMOR_INSTALL_EXECS_PROFILES

$(1)_POST_INSTALL_STAGING_HOOKS += \
    ORANGE_APPARMOR_INSTALL_LIBS_PROFILES \
    ORANGE_APPARMOR_INSTALL_DATA_PROFILES

ORANGE_APPARMOR_EXECS += $$ (if $$$ ($(1)_KCONFIG_VAR),$$ ($(1)_APPARMOR_EXECS))

endef

orange-package += $(call orange-apparmor-inner,$(call UPPERCASE,$(pkgnam)),$(pkgnam))
```

# Additional infrastructure – AppArmor 3/3

## AppArmor profiles

- Target-finalize hook
  - Scan ELF executables
  - What about scripts?

```
$ cat core/pkg-orange.apparmor.mk
define ORANGE_APPARMOR_GEN_EXECS_PROFILES
  $(foreach x,$(ORANGE_APPARMOR_EXECS),\
    $(Q)$ (AA_SCAN) \
      --root-dir $(TARGET_DIR) \
      --staging-dir $(STAGING_DIR) \
      --output-file $(STAGING_DIR)/etc/apparmor.d/orange$(subst /,.,$(x)) \
      --elf-self-read \
      --elf-cross-prefix $(TARGET_CROSS) \
      $(x)
  )
endif
ORANGE_PACKAGE_TARGET_FINALIZE_HOOKS += ORANGE_APPARMOR_GEN_EXECS_PROFILES
```



# Additional infrastructure – post-build

## Post-build scripts

- Not part of a package
- Generate blurbs
- Remove files
- Conditional tweaks

```
$ cat boards/common/post-build.sh
#!/bin/bash
TARGET_DIR="${1}"

# Store git info in an FHS-compliant version file
VERSION="$( cd "${BR2_EXTERNAL}"; git describe --abbrev=64 --dirty 2>/dev/null )"
VER_ID="$( cd "${BR2_EXTERNAL}"; git describe --abbrev=0 2>/dev/null )"
cat <<_EOF_ >"${TARGET_DIR}"/etc/os-release
    NAME='orange'
    VERSION='${VERSION}'
    ID='orange'
    VERSION_ID='${VER_ID}'
    PRETTY_NAME='Orange'
_EOF_

# Dbus interface xml files useless at runtime
rm -rf "${TARGET_DIR}"/usr/share/dbus-1/interfaces

# Cleanup the .empty placeholders
find "${TARGET_DIR}" -type f -name .empty -delete
```

---

```
$ cat boards/common/post-build-tests.sh
#!/bin/bash
TARGET_DIR="${1}"

# Open Dbus on TCP for tests
if ! grep ',port=55554,' "${TARGET_DIR}"/etc/dbus-1/orange.conf >/dev/null; then
    sed_subst='<listen>'
    sed_subst+='tcp:host=localhost,bind=*,port=55554,family=ipv4'
    sed_subst+='</listen>'
    sed -i 's/(\.*<listen>.*</listen>)/"${sed_subst}"\n1/ \
        "${TARGET_DIR}"/etc/dbus-1/orange.conf
fi
```

# Conclusion

- Infrastructure
- Whish-list

# Conclusion

## Infrastructure

- Automatic
- Systematic
- Reproducible
- Maintainable
- Extendable

## Whish-list

- ?

# Thanks

- Questions?

**Yann E. MORIN <[yann.morin@orange.com](mailto:yann.morin@orange.com)>**

**Embedded Linux Conference Europe 2017**

