

OpenEmbedded and the Yocto Project: Working together on a common Core

FOSDEM 2012

Paul Eggleton
Intel Open Source Technology Centre

how we work together
Let's start with a little history...



openembedded

OpenEmbedded:

Embedded Linux build system

Uses "bitbake" and recipes to build packages and images

Founded in 2005

Grown since then to

> 7,500 recipes, 300 machines, 20 distros

A number of forks

Produced by small consulting firms and larger OSVs

Molding OE into something commercially supportable

Included Poky developed by OpenedHand

OpenedHand was acquired by Intel and then in Oct 2010...

yocto .

PROJECT

Yocto Project

Linux Foundation project w/ support from chip vendors and OSVs

Main project is the Poky build system

Other projects under the Yocto umbrella e.g.

- pseudo

- swabber

Trying to make it easier to build embedded Linux

Late 2010 - looking for a way to work more closely with OE



New OpenEmbedded structure

Split up into layers

A machine/distro neutral base to build on - OE-Core

Various other layers to enable machines, software and policies



Why split the metadata?

Just the recipes you need for your project
Customisations more visible & easier to manage
Each layer can be focused
 Smaller, reusable units
 Less stale metadata cruft mixed in
 Easy to see how well maintained it is
 Avoid mixing in machine-specific overrides

The image shows a vast quantity of small, dark, circular metal rings, possibly made of steel or a similar alloy. These rings are densely packed and scattered across the entire frame, creating a complex, textured background. The lighting is somewhat uneven, with some areas appearing brighter than others, highlighting the metallic sheen and the circular shape of the components. In the center of the image, the text "OE-Core" is overlaid in a clean, white, sans-serif font, enclosed within a semi-transparent dark rectangular box. The overall composition is a close-up, top-down view of these industrial parts.

OE-Core created from Poky with machines removed, rename poky->core

Archs: ARM, x86, x86-64, MIPS and PowerPC (+ PowerPC64)

Only QEMU emulated machines

Distro-less (some default policy)

One X-based UI (Sato) for testing

Mostly one version of each recipe (some exceptions, e.g. for GPLv2/v3)

Can build working system using just OE-Core (and bitbake) and nothing else

Pull model vs. push model of classic OE

Patches sent to the mailing list, reviewed and merged from there

Yocto Project contributes directly to this core

and then pulls changes into Poky from there

same for bitbake

The basis of the collaboration between OE and Yocto



Layers

Types of layers - machine layer (BSP), software layer, distro layer

Overlaying recipes

Can be done, but leads to maintenance problems

bbappends

Add/change just the variables you need to

Some examples of common tasks via bbappend:

```
DESCRIPTION = "My package with special option enabled"
```

```
EXTRA_OECONF += "--enable-option"
```


Custom /etc/network/interfaces:


1) Add recipes-core/netbase_4.47.bbappend:

```
----- snip -----  
FILESEXTRAPATHS_prepend := "${THISDIR}/${PN}:"  
----- snip -----
```

2) Add recipes-core/netbase/netbase/interfaces

```
FILESEXTRAPATHS_prepend := "${THISDIR}/${PN}:"
```

```
SRC_URI += "custom-changes.patch"
```



Getting started - what should you do?

Create a customisation layer

Look for existing layers / recipes before starting your own



Creating a new layer

Structure:
conf/layer.conf
recipes-*/**/*
README

patches etc.

```
# We have a conf and classes directory
BBPATH := "${BBPATH}:${LAYERDIR}"

# We have recipes-* directories, add to BBFILES
BBFILES := "${BBFILES}
           ${LAYERDIR}/recipes-*/*/*.bb
           ${LAYERDIR}/recipes-*/*/*.bbappend"

BBFILE_COLLECTIONS += "layername"
BBFILE_PATTERN_layername := "^${LAYERDIR}/"

BBFILE_PRIORITY_layername = "5"
```


conf/layer.conf

See Yocto Project developer's guide




Layer tools

Managing metadata across multiple layers can be tricky
Yocto Project is working on tools to do this

A close-up photograph of a metal tool, possibly a wrench or a similar fastener, with a central text overlay. The tool is metallic and shows signs of use, with some wear and discoloration. The background is dark and out of focus. The text "bitbake-layers" is written in a white, sans-serif font on a black rectangular background.

bitbake-layers



combo-layer



Current status

OE Layer index:

- 16 BSP layers

- 12 software layers (e.g. EFL, XFCE)

- 5 distro layers

meta-openembedded

New layers popping up all the time

What's next?

Yocto: Enhance layer tools further

- Start looking at what each layer does down at the variable level in bitbake-layers

- Web-based layer index (searchable)

- Recipe maintenance tools

OE:

- Improve OE documentation

- Bring more metadata over from OE-Classic (need maintainers!)


References

<http://www.yoctoproject.org>

<http://www.openembedded.org>

<http://www.openembedded.org/wiki/LayerIndex>

IRC (freenode): #yocto, #oe



A successful tool is one that was used to do something undreamed of by its author.

Questions?

Photo credits:

* "Structure of the eye" by tompagenet

<http://www.flickr.com/photos/tompagenet/95737053/>

* "Cores" by Marcin Wichary

<http://www.flickr.com/photos/mwichary/3209186260/>

* "Lego Bits Box #1" by jemsweb

<http://www.flickr.com/photos/jemsweb/4363545741/>

* "Layer Cake" by OctopusHat

<http://www.flickr.com/photos/octopushat/1433976199/>

* "Cat eats cake" by kitty.green66

<http://www.flickr.com/photos/53887959@N07/4985430800/>

* "A Couple Layers" by Martin Cathrae

<http://www.flickr.com/photos/suckamc/2882176630/>

* "A successful tool is one that was used to do something undreamed of by its author." by katerha

<http://www.flickr.com/photos/katerha/5746905652/>

* "Sunset" by NeilsPhotography

<http://www.flickr.com/photos/neilspicys/2349801988/>

Talk contents © 2012 Intel Corporation

CC-By-SA

Any opinions stated in this talk are my own and not necessarily those of my employer (or anyone else).
All trademarks belong to their respective owners.