It's not an embedded Linux distribution –
It creates a custom one for you.

The Yocto Project Overview and Update

Intel Corporation
February 15, 2012

It’s not an embedded Linux distribution

●

It creates a custom one for you.

Agenda

- **What is the Yocto Project**
  - Overview
  - Community

- **Overview of Poky – the build system**
  - Bitbake
  - Application Developer Kit
  - Hob2 Show & Tell

- **The Yocto Project Update**
  - 1.1 / 1.2 Features
  - Beyond 1.2

Yocto Project Overview

- **The Yocto Project* is an open source collaboration project**
  - Provides templates, tools and methods to help you create custom Linux-based systems for embedded products regardless of hardware architecture.
- **Focused resources for system application developers who need to customize a Linux distribution for a device**
- **Validated and tested BSPs in a common format**
- **Automatically creates an application development SDK customized for each specific device**
- **Supported by embedded industry leaders across multiple architectures (IA, ARM, PowerPC, MIPS, etc)**
- **Is a great starting point for “roll your own” embedded developers and commercial distribution vendors.**
- **Enables easy transition from Proof of Concept (POC) to supported Commercial Linux with no loss of optimizations, code or design**
- **Proprietary code can be included in build structure within a separate layer, which can be kept private. (security)**
- **Project hosted by the Linux* Foundation**

Participating Organizations

Silicon Vendors

TILERA®

freescale™ semiconductor

LSI™

intel®

CAVIUM NETWORKS

TEXAS INSTRUMENTS

MINDSPEED BUILD IT FIRST™

OSVs

WIND RIVER

timesys®

montavista™

Mentor Graphics®

Embedded Tools, Consulting Services, Users...

RidgeRun Embedded Solutions

Secret Lab

SAKOMAN INC

Panasonic ideas for life USA

DELL

openembedded

THE LINUX FOUNDATION

Contact the Linux Foundation if you are interested in becoming a participating organization.
(Take part in Governance, Advisory Board, Advocacy and Communications)

Benefits of Yocto Project

- **Linux is becoming increasingly popular for Embedded**
- **Non-commercial and commercial embedded Linux has many distros**
  - Developers spend lots of time porting or making build systems
  - Leaves less time/money to develop interesting software features
- **The industry needs a common build system and core technology**
- **Industry leaders have joined together to form the Yocto Project, the benefit of doing so is:**
  - Less time spent on things which don’t add value (build system, core Linux components)
  - Increased ability to enable key silicon features
  - Linux grows more in embedded

What makes up the Yocto Project

- **Embedded tools and a distribution build environment**
  - Eglibc, prelink, pseudo, swabber, along with other tools
- **Support ARM, MIPS, PPC, x86 (32 & 64 bit)**
- **Shares core meta data (OE-core) with Opemembedded community**
- **Complete Embedded Linux OS with meta data**
- **Releases at on a 6 month (or so) cadence**
  - Latest (stable) kernel, toolchain and packages, documentation
  - App Development Tools including Eclipse plugins and emulator
- **BSPs are available from various Vendors**

t’s not an embedded Linux distribution – it creates a custom one for yo

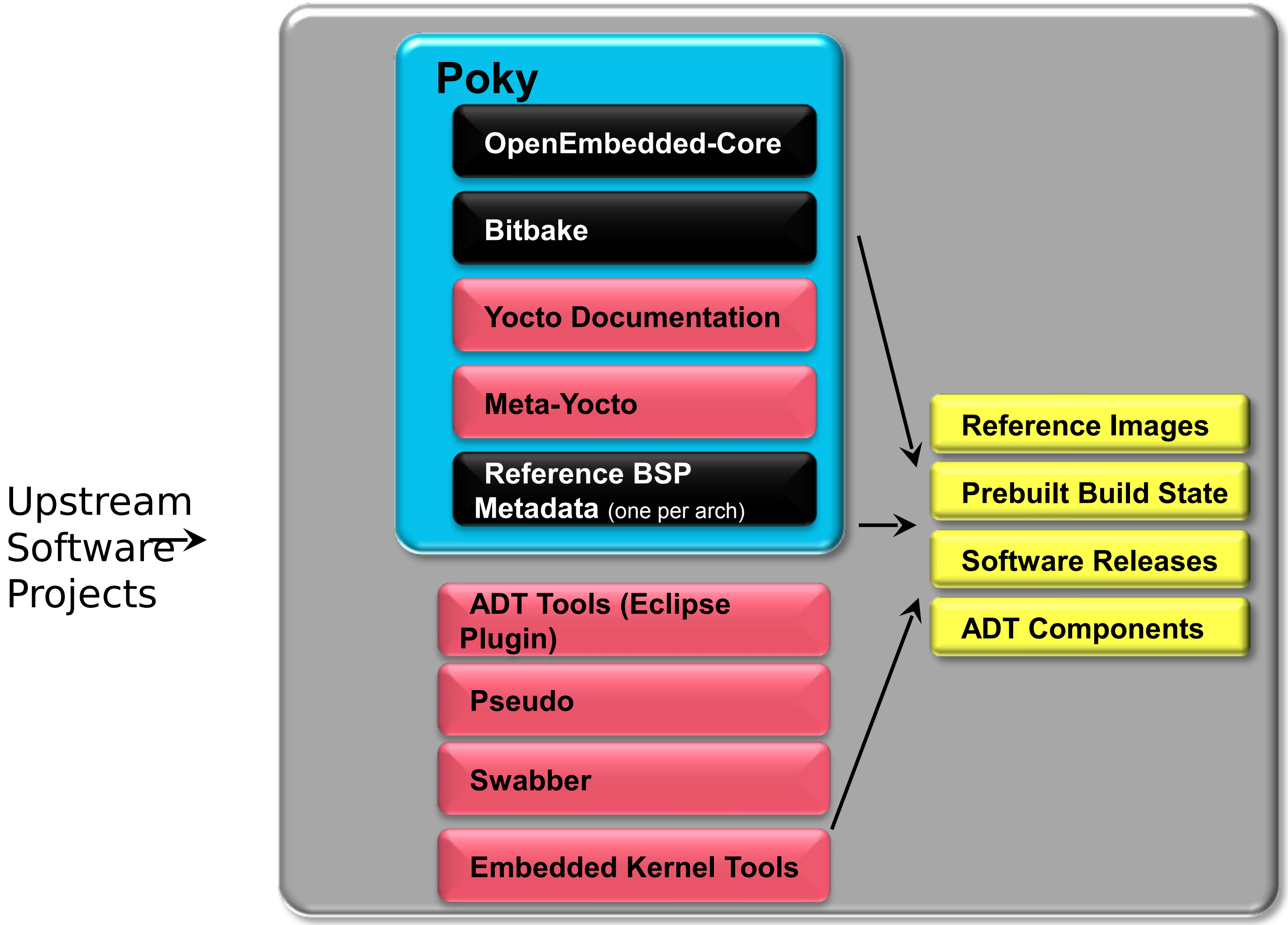
Why does it matter to you?

- **Start with a know validated set of packages**
  - Toolchain, kernel and userspace
- **Bootable Embedded Linux Image in about 1 hour**
- **Excellent Application & System Developer tools**
  - Eclipse, performance, debug
  - Tracing, power analysis
- **Flexible kernel development tools**
  - Configuration and patch management
- **Porting to new hardware is easy**
  - Change or create a new config file and rebuild
- **Easy path to the commercial Embedded Linux Market**
  - Mentor Graphics, Montavista Timesys, Wind River

Agenda

- **What is the Yocto Project**
  - Overview
  - Community
- **Overview of Poky – the build system**
  - Bitbake
  - Application Developer Kit
  - Hob2 Show & Tell
- **The Yocto Project Update**
  - 1.1 / 1.2 Features
  - Beyond 1.2

Yocto Project = Poky + Tools + Upstreams



Upstream Software Projects

Poky

OpenEmbedded-Core

Bitbake

Yocto Documentation

Meta-Yocto

Reference BSP Metadata (one per arch)

ADT Tools (Eclipse Plugin)

Pseudo

Swabber

Embedded Kernel Tools

Reference Images

Prebuilt Build State

Software Releases

ADT Components

⬛ Build system upstream components

🟦 Poky

🟥 Yocto Project components

⬜ Yocto Project

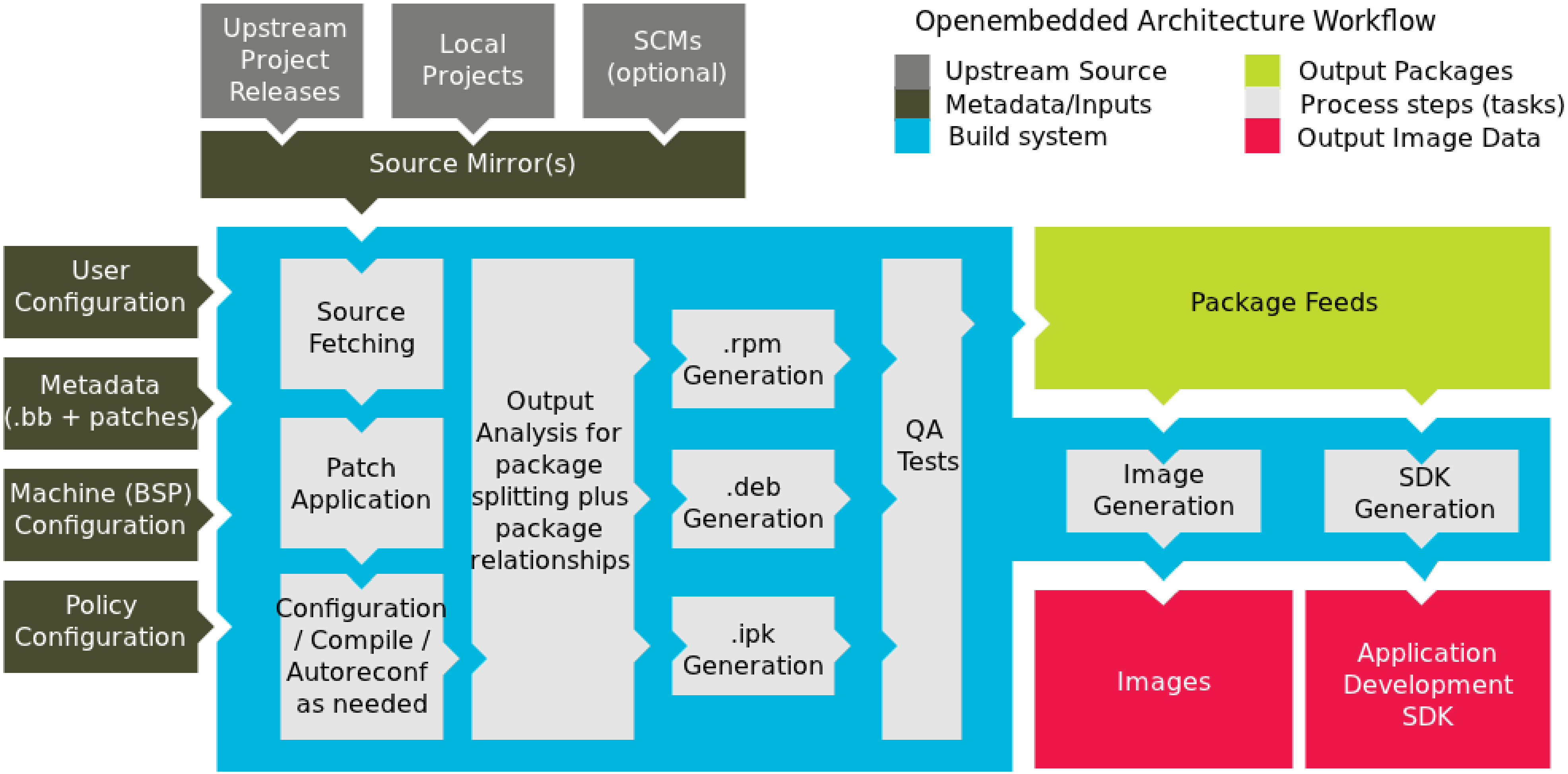
🟨 Yocto Project Output

Yocto Project provides best of upstream for a stable base

What's OE-Core

- **OpenEmbedded (oe-classic)**
  - Large Community (founded in 2005)
  - Lots of recipes, machines & distros
  - Uses bitbake as the task executor
- **Poky**
  - One of a number of commercially supported distros
  - Limited number of current recipes for base construction
  - Tested and updated regularly
- **Layered approach (more later)**
  - OE-Core as base – Machine / Distro neutral
  - Others can build on it

Devils in the Details



Openembedded Architecture Workflow

⬛ Upstream Source
⬛ Metadata/Inputs
🟦 Build system
🟩 Output Packages
⬜ Process steps (tasks)
🟥 Output Image Data

Upstream Project Releases

Local Projects

SCMs (optional)

Source Mirror(s)

User Configuration

Metadata (.bb + patches)

Machine (BSP) Configuration

Policy Configuration

Source Fetching

Patch Application

Configuration / Compile / Autoreconf as needed

Output Analysis for package splitting plus package relationships

.rpm Generation

.deb Generation

.ipk Generation

QA Tests

Package Feeds

Image Generation

SDK Generation

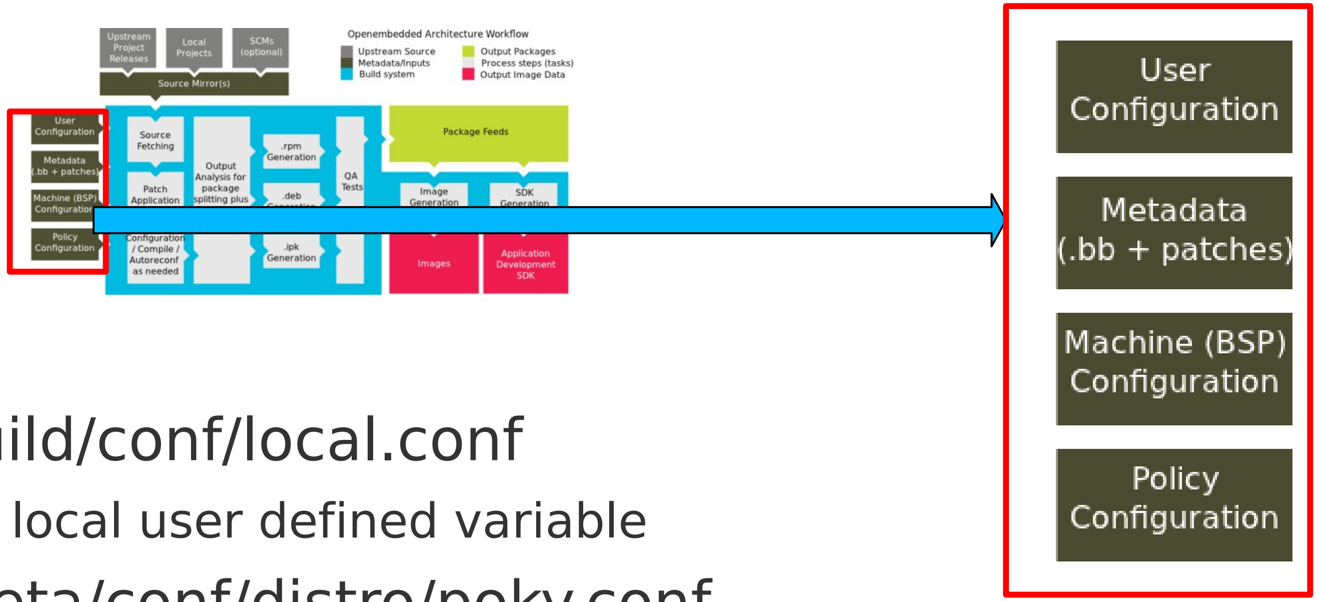
Images

Application Development SDK

Slides and video: http://bit.ly/it9rkB

Configuration

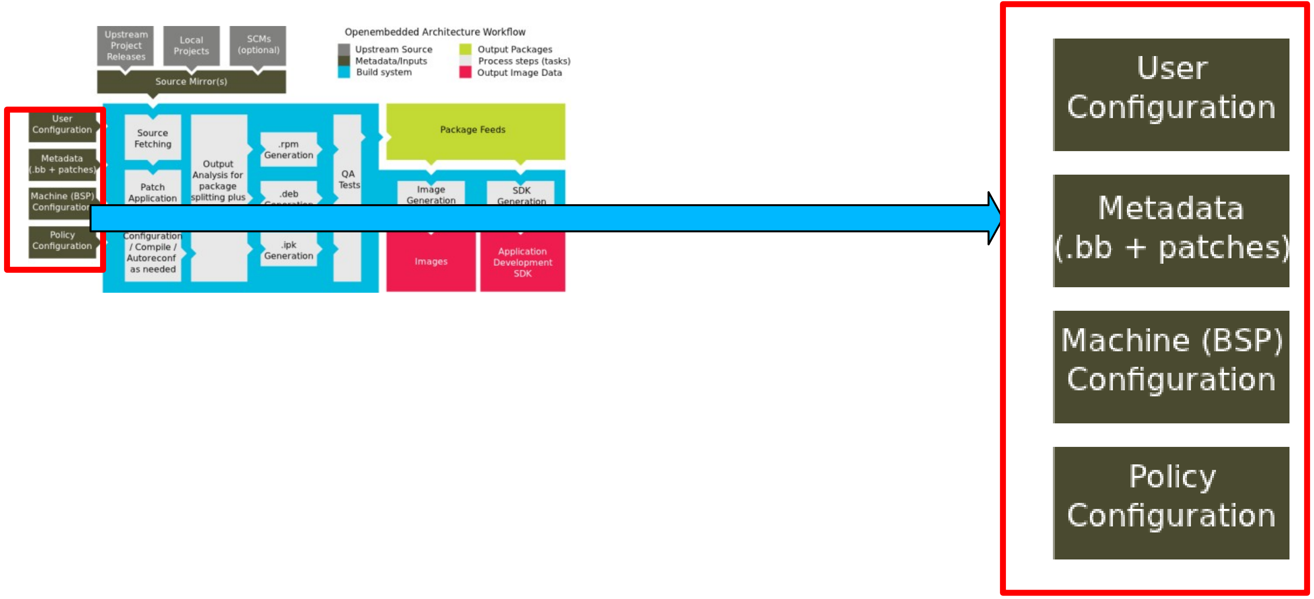
- **Configuration files (*.conf) – global definition of variable**



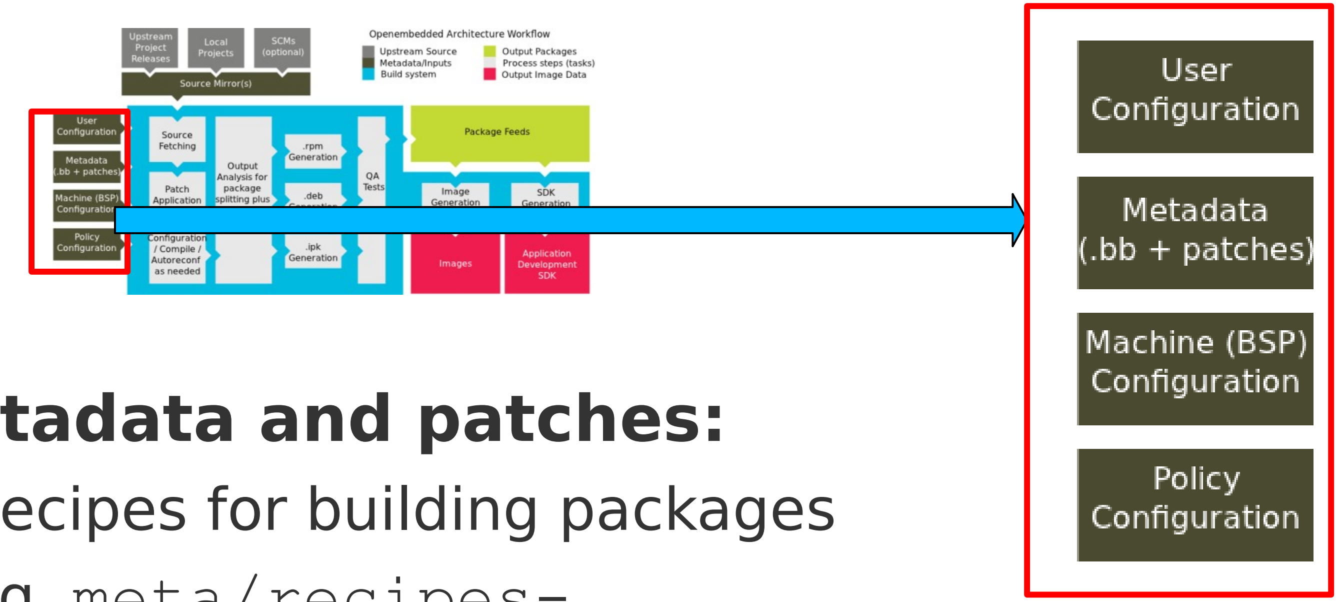
- build/conf/local.conf
  - local user defined variable
- meta/conf/distro/poky.conf
  - Poky policy config variables
- Meta/machine/routerstationpro.conf
  - Machine specific variables

User Configuration

- **User configuration:**
  - conf/local.conf – some things to set:
    - Set BB_NUMBER_THREADS and PARALLEL_MAKE, based on the number of threads in the machine
    - Set MACHINE=“foo” for the CPU architecture
    - EXTRA_IMAGE_FEATURES adds features (groups of packages)
    - INCOMPATIBLE_LICENSE = “GPLv3” eliminates packages using this license (for example)

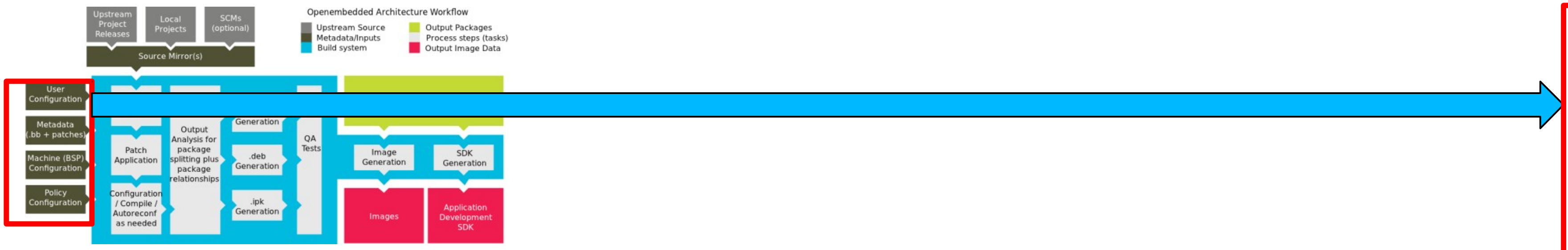


Recipes & Metadata



- **Metadata and patches:**
  - Recipes for building packages
  - Eg, `meta/recipes-core/coreutils/coreutils_6.9.bb` builds the core utilities (version 6.9) and installs them
  - `meta-recipes-core/coreutils/coreutils-6.9/` includes patches, also could include extra files to install

Recipes & Metadata



User Configuration

Metadata (.bb + patches)

Machine (BSP) Configuration

Policy Configuration

Developer-Specific Layer

Commercial Layer (from OSV)

UI-Specific Layer

Hardware-Specific BSP

Yocto-Specific Layer Metadata (`meta-yocto`)

OpenEmbedded Core Metadata (`oe-core`)

BSP Layers

- **Layers contain extensions and customizations to base system**
- **Can include image customizations, additional recipes, modifying recipes, adding extra configuration**
  - Really just another directory to look for recipes in
  - Added to the BBLAYERS variable in build/conf/bblayers.conf
- **BSPs are layers that add machine settings and recipes**
- **Machine settings are specified in a layer's conf/machine/xxx.conf file(s)**
- **Examples:**
  - Sandy Bridge + Cougar Point:
    - meta-intel/conf/meta-sugarbay/machine/sugarbay.conf
  - Routerstation Pro (MIPS)
    - yocto/meta/conf/machine/routerstationpro.conf

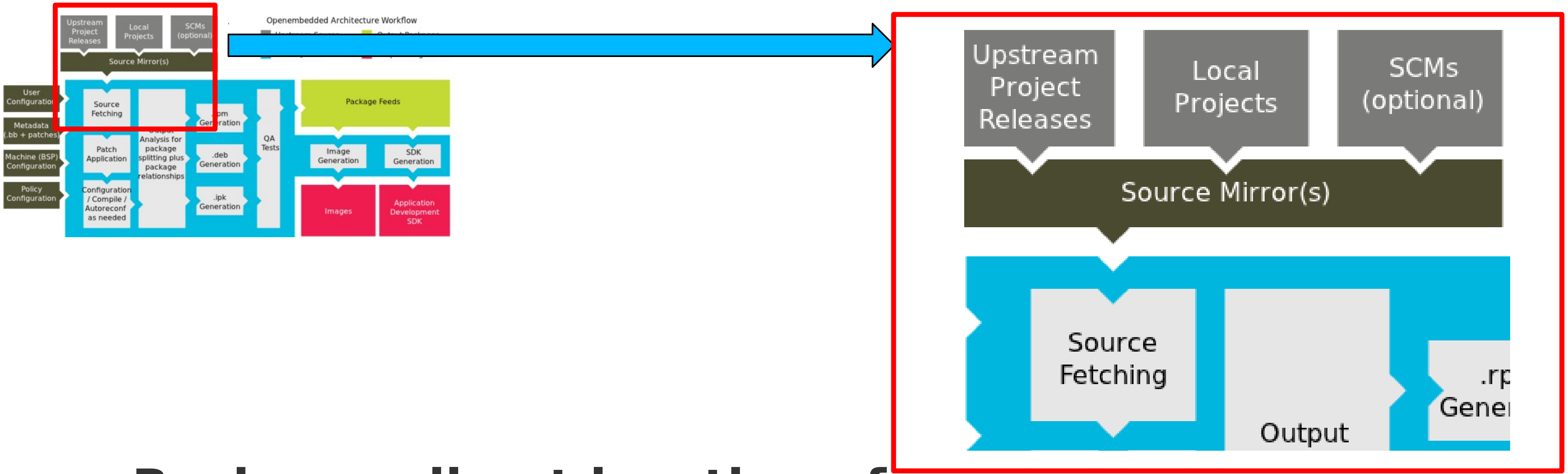
Kernel Development

- **We try to develop upstream wherever possible**
- **Two major advances in the Yocto Project:**
  - Branching tools: Per-BSP git branches contain machine-specific kernel sources. Tools collect up the relevant tree of branches
  - Kernel features: patches and configuration fragments managed as a functional block
- **Results:**
  - Can turn on a collection of features for a given BSP
  - Less code duplication
  - Easier to choose a config fragment and patches

Kernel Tools Details

- **Components**
  - Linux-Yocto recipe
    - meta/recipes-kernel/linux/linux-yocto*.bb
  - Linux-Yocto git repository
    - http://git.pokylinux.org/cgit/cgit.cgi/linux-yocto-2.6.37
    - http://git.pokylinux.org/cgit/cgit.cgi/linux-yocto-3.0
- **Kernel Versions**
  - Linux-yocto-stable: 3.0.2
  - Linux-yocto: 2.6.37 and 3.0.2
  - *linux-korg.bb: current HEAD of linux.git, base minimum of tools*
  - *Linux-yocto-rt : RealTime*

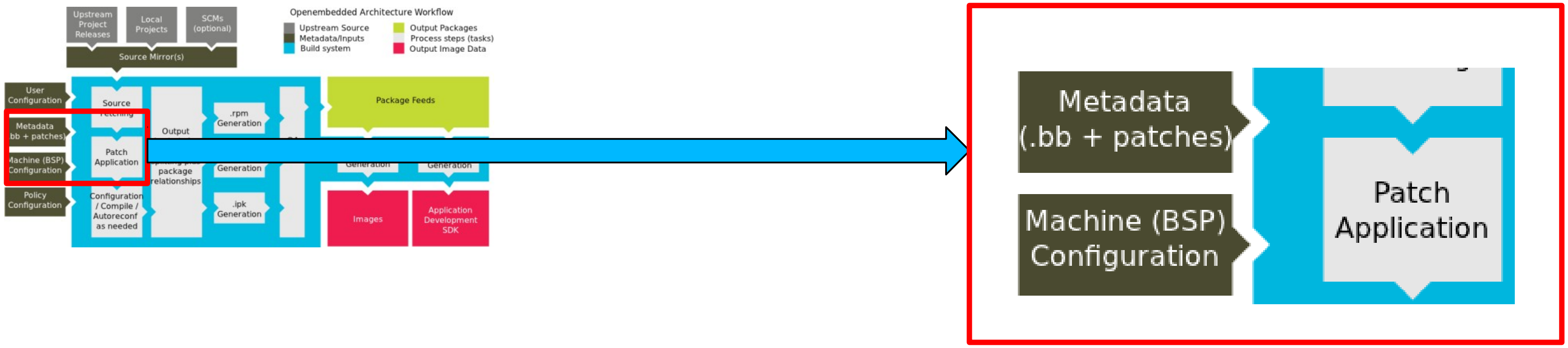
Source Fetching



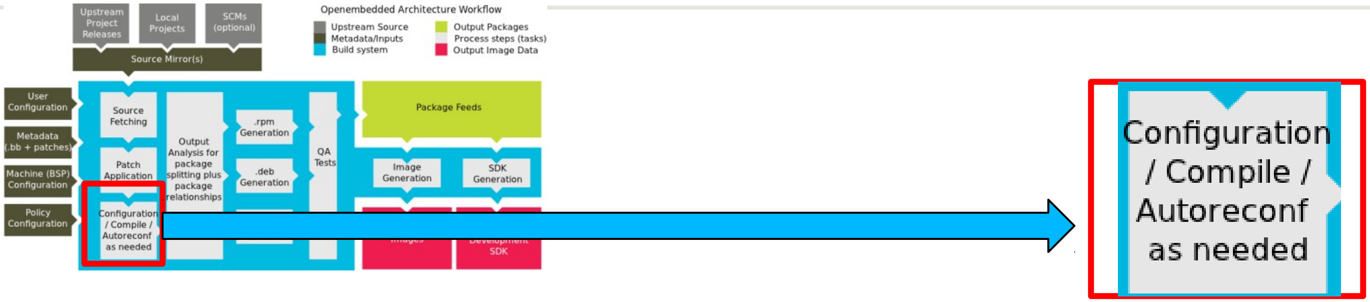
- **Recipes call out location of sources**
  - SRC_URI can be local or in the ether
- **Bitbake can fetch from various types**
  - git, svn, bzr, from tarballs, and many, many more*
- **Versions of packages can be fixed or updated automatically**
- **Yocto Project sources mirror available as a fallback, if the sources move on the internet**

Patching

- **Once sources are obtained, the patches are applied**
- **This is a good place place to patch the software yourself**
- **However, we encourage you to contribute development upstream whenever possible (we try to)**



Configure/Compile



- **Autoconf can be triggered automatically to ensure latest libtool is used**

```
DESCRIPTION = "GNU Helloworld application‟
SECTION = "examples"
LICENSE = "GPLv2+"
LIC_FILES_CHKSUM = "file://COPYING;md5=751419260aa954499f7abaabaa882bbe"
PR = "r0"

SRC_URI = "${GNU_MIRROR}/hello/hello-${PV}.tar.gz"

inherit autotools gettext
```

- **CFLAGS can be set**

```
CFLAGS_prepend = "-I ${S}/include "
```

- **Install task to set modes, permissions, target directories, done by “pseudo”**

```
do_install () {
        oe_runmake install DESTDIR=${D} SBINDIR=${sbindir} MANDIR=${mandir}
```

Packaging

- **Once configure/compile/install is completed, packaging commences**
- **The most popular package formats are supported: RPM, Debian, and ipk**
  - Set `PACKAGE_CLASSES` in `conf/local.conf`
- **You can split into multiple packages using PACKAGES and FILES in a *.bb file:**

```
PACKAGES =+ "sxpm cxpm"
FILES_cxpm = "${bindir}/cxpm"
FILES_sxpm = "${bindir}/sxpm"
```

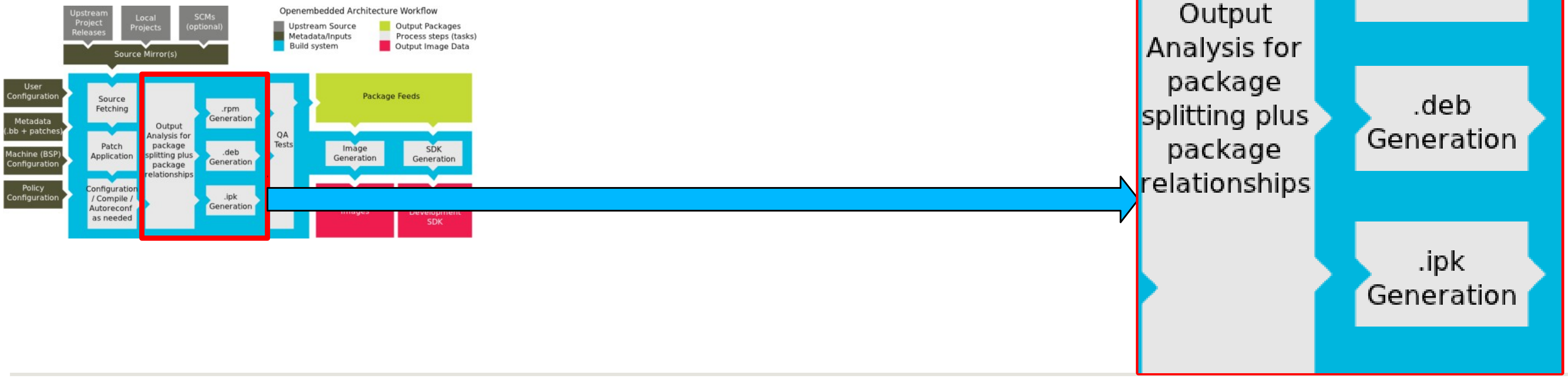
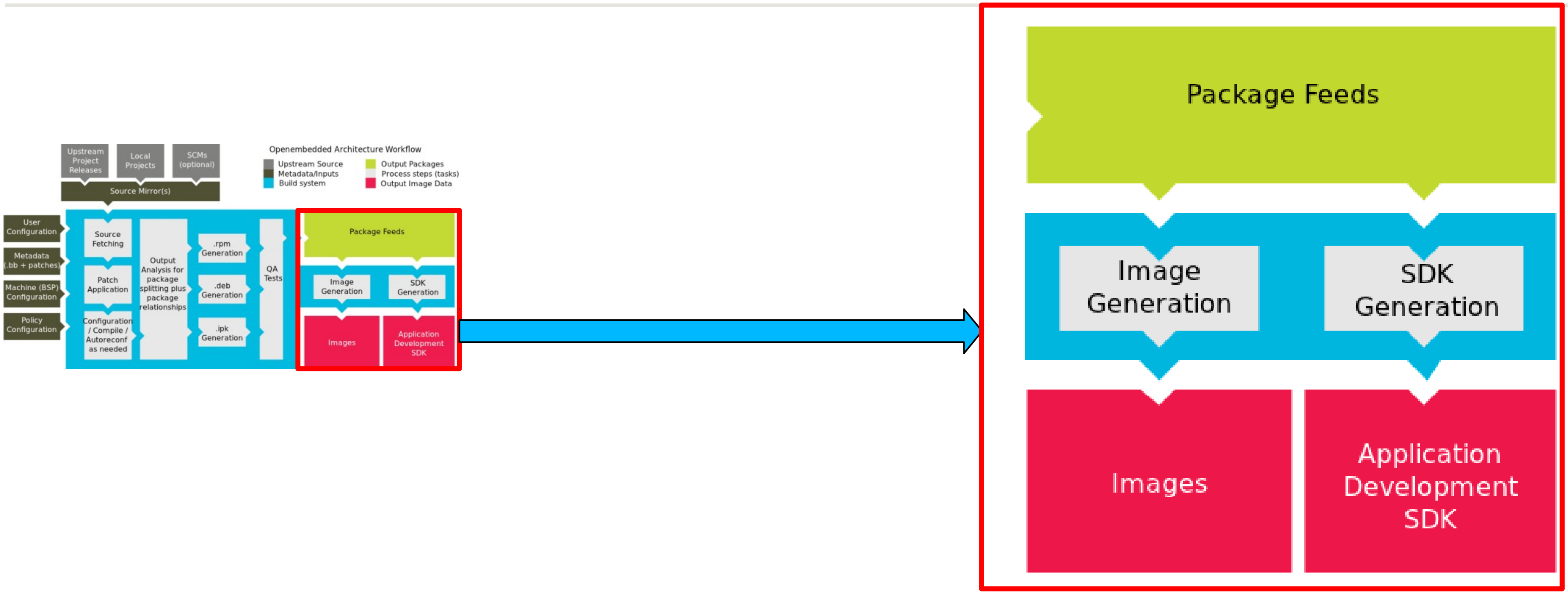


Image Generation



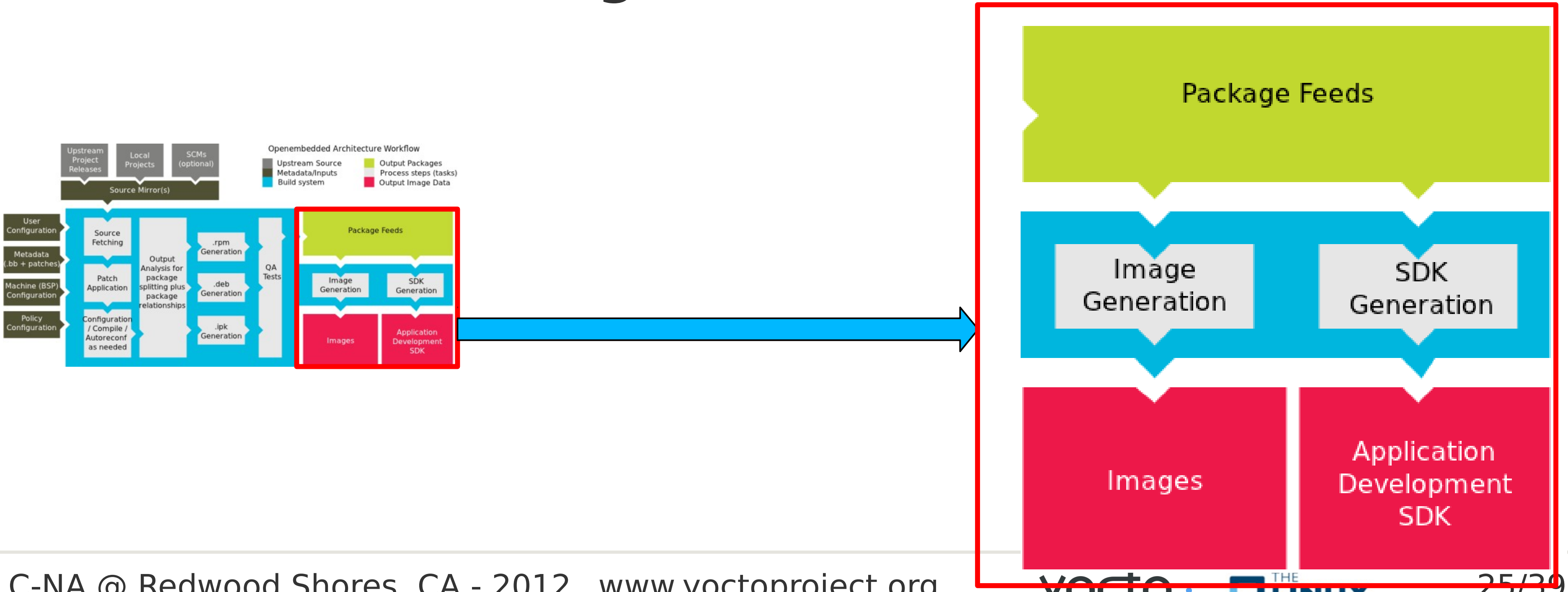
- **Images are constructed using the packages built earlier in the process**
- **Uses for these images:**
  - Live Image to boot a device
  - Root filesystem for QEMU emulator
  - Sysroot for Application development

ADT Generation

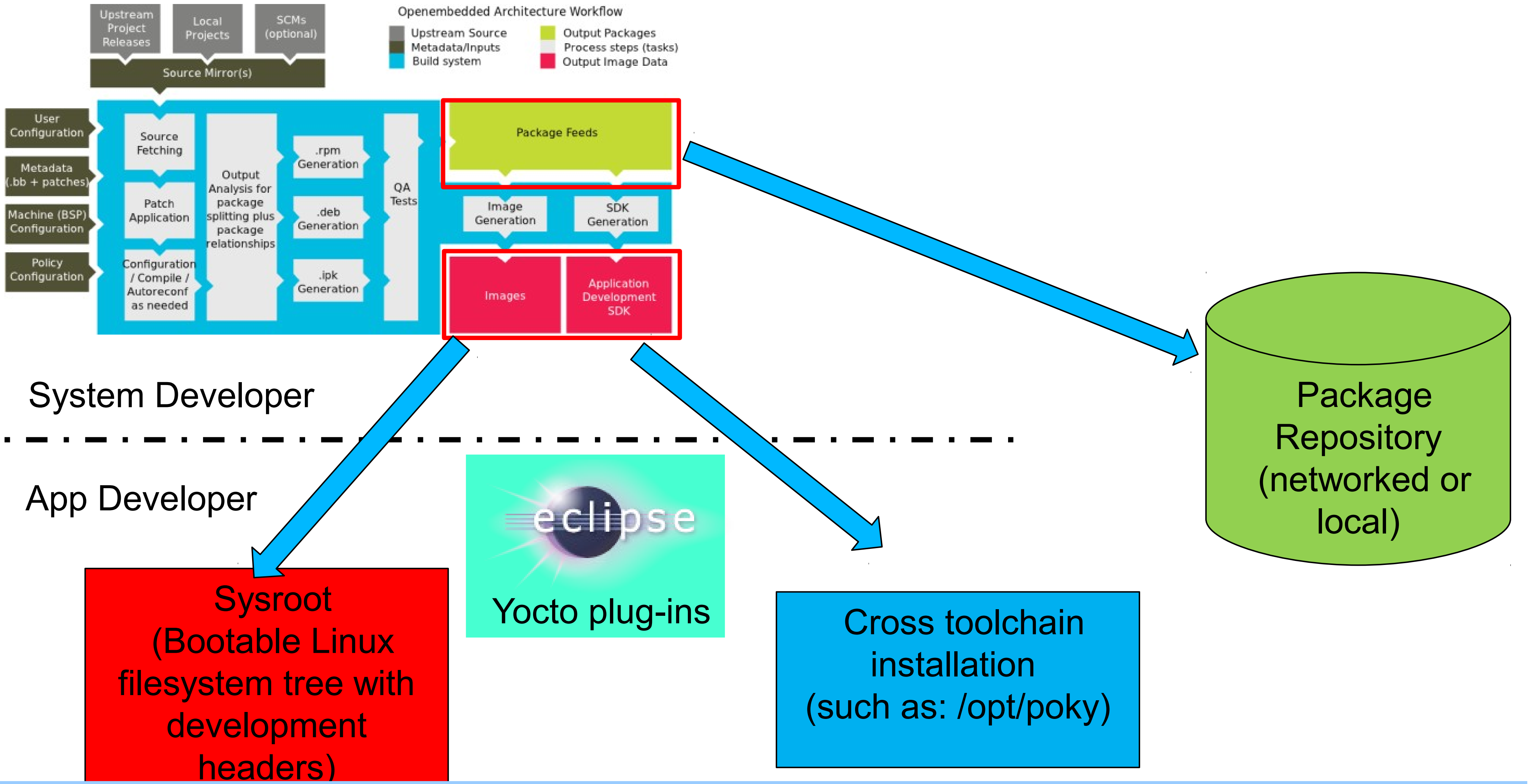
- Cross toolchain and installation script generated.
- This can be used to set up an application developer’s cross development environment to create apps

```
$MACHINE=qemuarm bitbake poky-image-sato-sdk
 meta-toolchain package-index
```

- QEMU built for target architecture emulation



Setting up the App Developer



System Developer

App Developer

Sysroot (Bootable Linux filesystem tree with development headers)

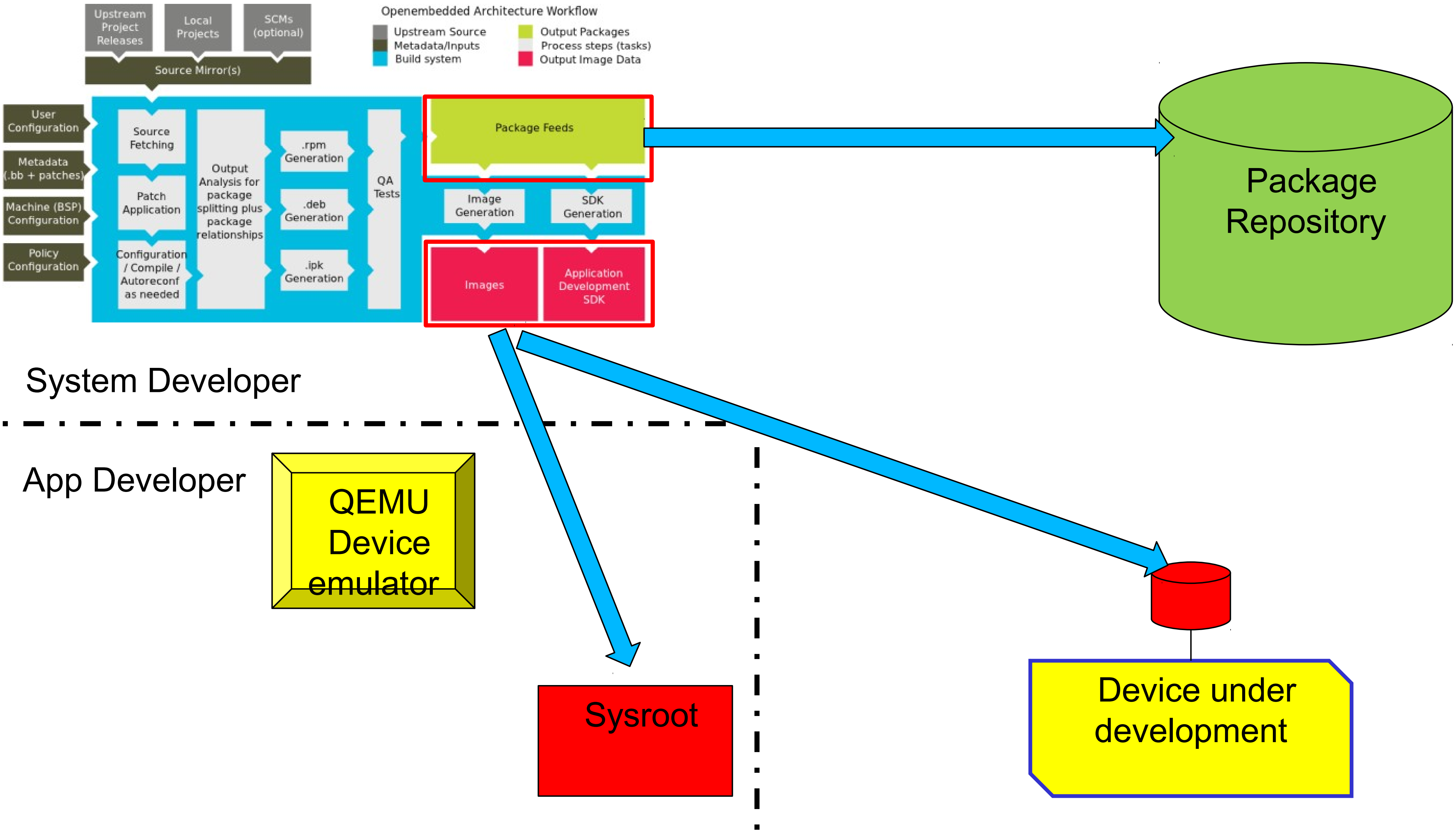
Yocto plug-ins

Cross toolchain installation (such as: /opt/poky)

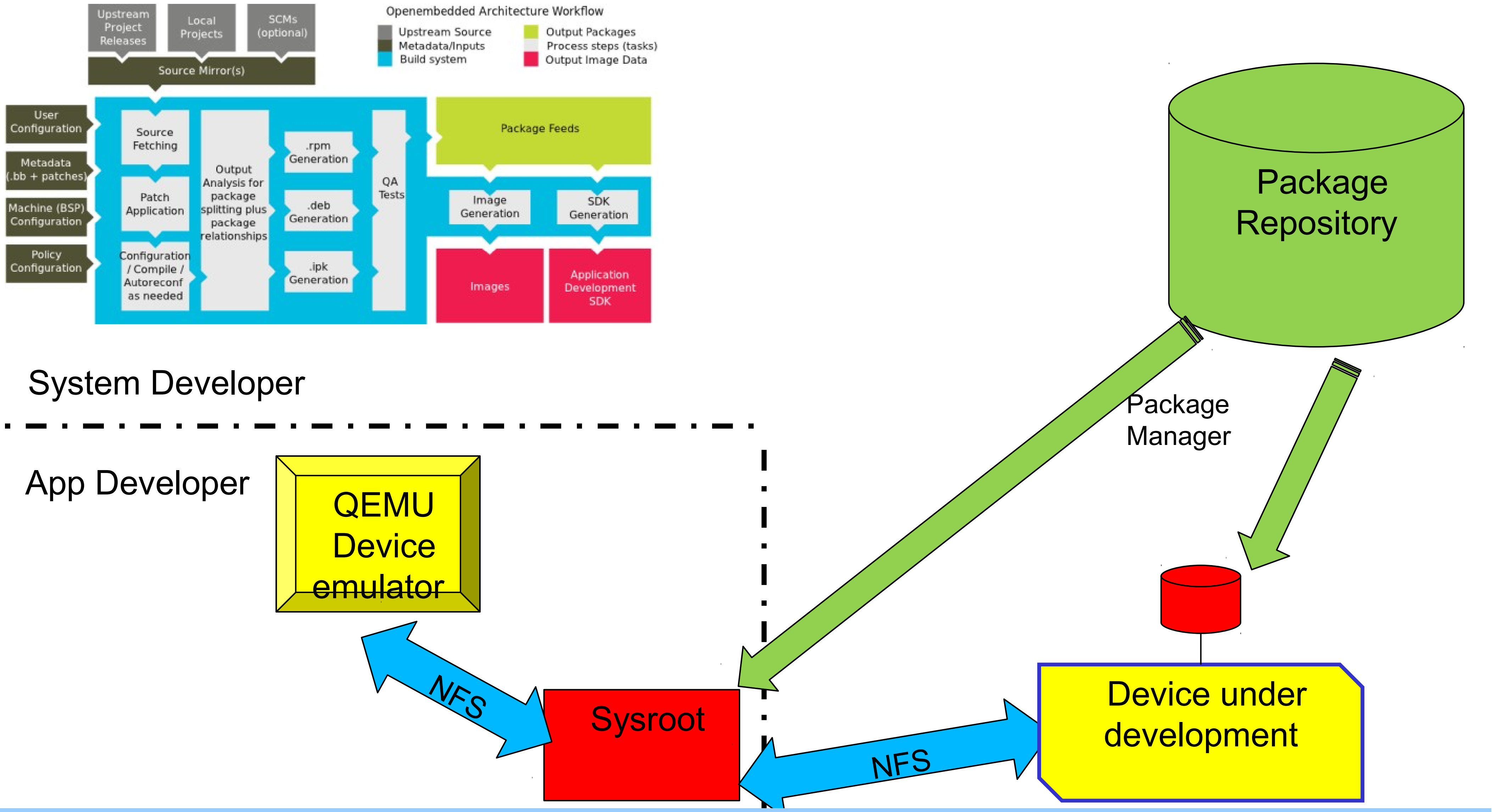
Package Repository (networked or local)

cto Project helps setup the embedded application develo

Use NFS/Local Disk w/ Pkg Manager



Use NFS/Local Disk & Pkg Manager



oth Device and Application Development Models Support

Hob 2 Demo

Agenda

- **What is the Yocto Project**
  - Overview
  - Community
- **Overview of Poky – the build system**
  - Bitbake
  - Application Developer Kit
  - Hob2 Show & Tell
- **The Yocto Project Update**
  - 1.1 / 1.2 Features
  - Beyond 1.2

Yocto Project / Poky 1.1

- **Hob**
  - graphical interface for selecting options, packages and doing a build
- **Multilib support**
  - mix and match 32 and 64 bit binaries on the target
  - Pick the architecture on a per package basis
  - https://wiki.yoctoproject.org/wiki/Multilib
- **Initial x32 support**
  - X86-64 systems running 64 bit registers and 32 bit data types – see meta-x32 repository
- **System builder tasks now in Eclipse**
- **Layer tooling -**
- **Build Statistics – Disk IO**
- **LSB**

Yocto Project / Poky 1.2

- **Hob2**
  - graphical interface for selecting options and packages and doing a build
  - Additional tool to assist with Deployment
- **Build Appliance**
  - Boots to a mini-x session and Hob2 using vmware or qemu
- **Build History**
  - Tools for comparing build results
- **Lots of usability and stability work**
  - Sstate, License manifests

Beyond 1.2

- **In planning now!**
  - Get involved give your enhancement suggestions by email or bugzilla
- **Always working to improve the experience**
  - Error Handling and Output
  - Performance improvements (of build system)
- **Improvements to Hob and Build Appliance**
  - Proxy settings via preferences
  - Improved Deployment tools
- **Continual update of kernel, userspace**
  - Clean meta-data (License info, Descriptions, Packaging)
  - Improve documentation output and reduce warnings

Other Talks / Activites this week

- **Embedded License Compliance Patterns and Antipatterns**
  - Beth Flanagan, Thursday at 3:00
- **Yocto Project Community BOF**
  - Jeff Osier-mixon, 5:15 on Thrusday
- **Producing the Beaglebone and Supporting It**
  - Koen Koon, Friday at 11:30
- **OpenEmbedded - A Layered Approach**
  - Khem Raj, Friday at 2:00

Take Action Now

- **It’s not an embedded Linux distribution – it creates a custom one for you**
  - YP lets you customize your embedded Linux OS
  - YP helps set up the embedded app developer
  - Both device and app development models supported
- **Getting started is easy**
  - Download the software today
  - Be sure you read the Quick Start to set up your system to use the Yocto Project
  - Build, test on QEMU or real hardware, develop apps

Make an impact – collaboration in its purest sense

Join the community

- **#yocto on freenode.irc.net**
- **http://www.yoctoproject.org**
- **http://wiki.yoctoproject.org**
- **Development through public mailing lists:**
  - **yocto@yoctoproject.org**, **poky@yoctoproject.org** **http://lists.yoctoproject.org**
  - **openembedded-core@lists.openembedded.org**
- **Git Code repositories**
  - **git://git.yoctoproject.org**
  - **git://git.openembedded.org**
- **Bug reporting and features requests via**
  - **http://bugzilla.yoctoproject.org**

Title:yocto-project-transp.eps
Creator:Adobe Illustrator(R) 14.0
CreationDate:09/10/2010
LanguageLevel:2

Legal

INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL® PRODUCTS. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER, AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY RELATING TO SALE AND/OR USE OF INTEL PRODUCTS, INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT, OR OTHER INTELLECTUAL PROPERTY RIGHT.

Intel may make changes to specifications, product descriptions, and plans at any time, without notice.

All dates provided are subject to change without notice.

Intel is a trademark of Intel Corporation in the U.S. and other countries.

*Other names and brands may be claimed as the property of others.

Slide Title

- **Bitstream Vera Sans**
- **Font color “Yocto Dark Gray”**
- **Topic Level 1**
    - Detail Level 2

    1 Numbered Detail Level 2

    2 Plain Detail Level 2

```
$ code
#privileged code
```

        - Detail Level 3

        1 Numbered Detail Level 3

        Plain Detail Level 3
- **http://www.yoctoproject.org**