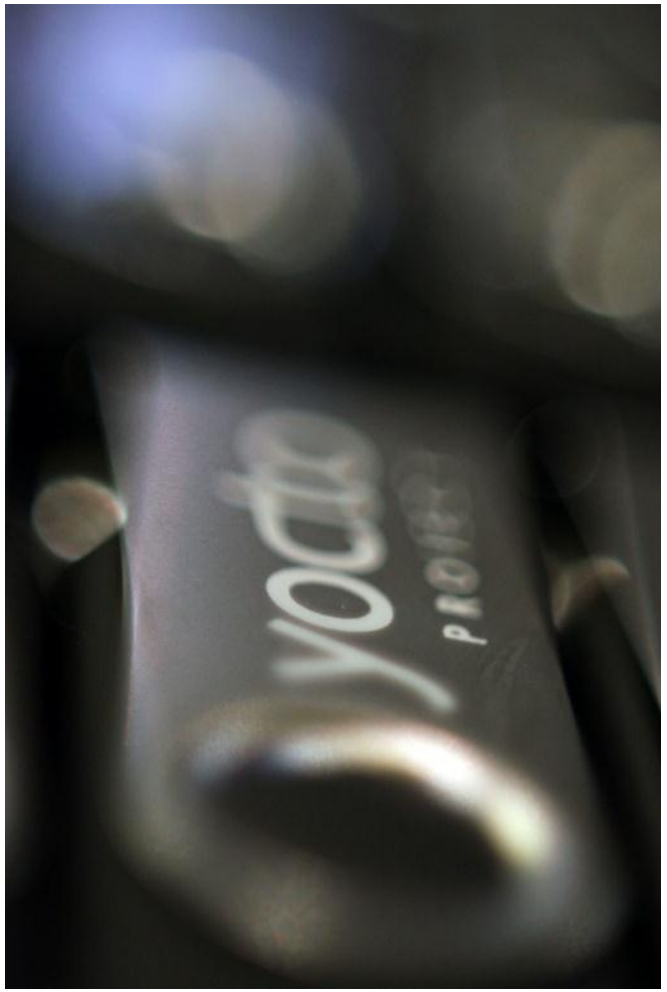
yocto PROJECT

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

Developing Embedded Linux Devices Using the Yocto Project™



David Stewart
Intel Corporation
October, 2011

yocto PROJECT · THE LINUX FOUNDATION

Agenda

- What is the Yocto Project (YP)? … and what's new
- How does it work?
- How to get started with building OS, apps, and debugging
- What's Next?
- Q&A

What is the Yocto Project? The Story

- Linux is becoming increasingly popular for Embedded
- Non-commercial and commercial embedded Linux has many distros
- Result is:
  - 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)
  - Linux grows more in embedded

What is the Yocto Project?

- Distribution build environment and tools for embedded
- Supports ARM, PPC, MIPS, x86 (32 & 64 bit)
- Open source project with a strong community
- Content
  - Complete Linux OS with package metadata
  - Releases every 6 months with latest (but stable) kernel, toolchain, and package versions
  - Place for Industry to publish BSPs
  - App Dev Tools which allow development against the stack, including Eclipse plug-ins and emulators
  - Full documentation representative of a consistent system

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

Why Should a Developer Care?

- Build a complete Linux system in about an hour from sources (about 90 minutes with X).
- Start with a validated collection of packages (toolchain, kernel, user space).
- Access to a great collection of app developer tools (performance, debug, power analysis, Eclipse). We distinguish app developers system developers and we support both.
- Manage patches with included kernel development tools.
- Supports all major embedded architectures (x86, x86-64, ARM, PPC, MIPS), just change a line in a config file and rebuild.
- Easy path to a commercial embedded Linux (Mentor Graphics, Montavista, Timesys, Wind River).

What's new in Yocto v1.1

- Hob – graphical interface for selecting options and packages and doing a build
- Multilib –
  - 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 for more
- 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

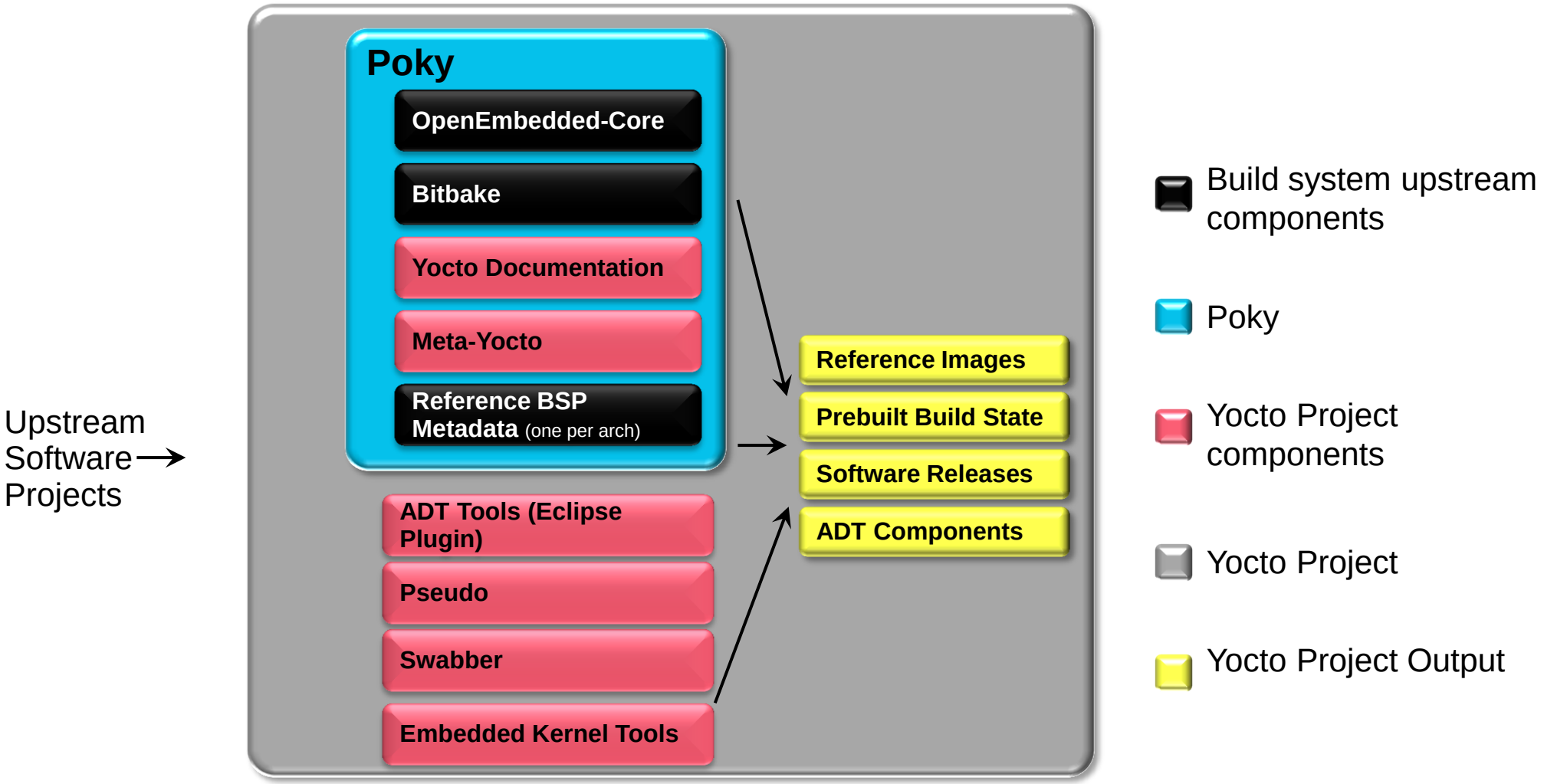
How Does It Work? – Quick Start

1. Go to http://yoctoproject.org, click “documentation” and consult the Quick Start guide
2. Set up your Linux system with the right packages (and firewall access, if needed)
3. Click “Download” and download the latest stable release (or check out “bernard” from the git repo)
4. Edit `conf/local.conf` and set MACHINE, BB_NUMBER_THREADS and PARALLEL_MAKE
5. Source `oe-init-build-env` script
6. Run `$ bitbake -k core-image-sato`
7. Run `$ runqemu qemux86` (if MACHINE=qemux86)

Note: File or command names in this presentation are subject to change, several are different now in master.

yocto PROJECT · THE LINUX FOUNDATION

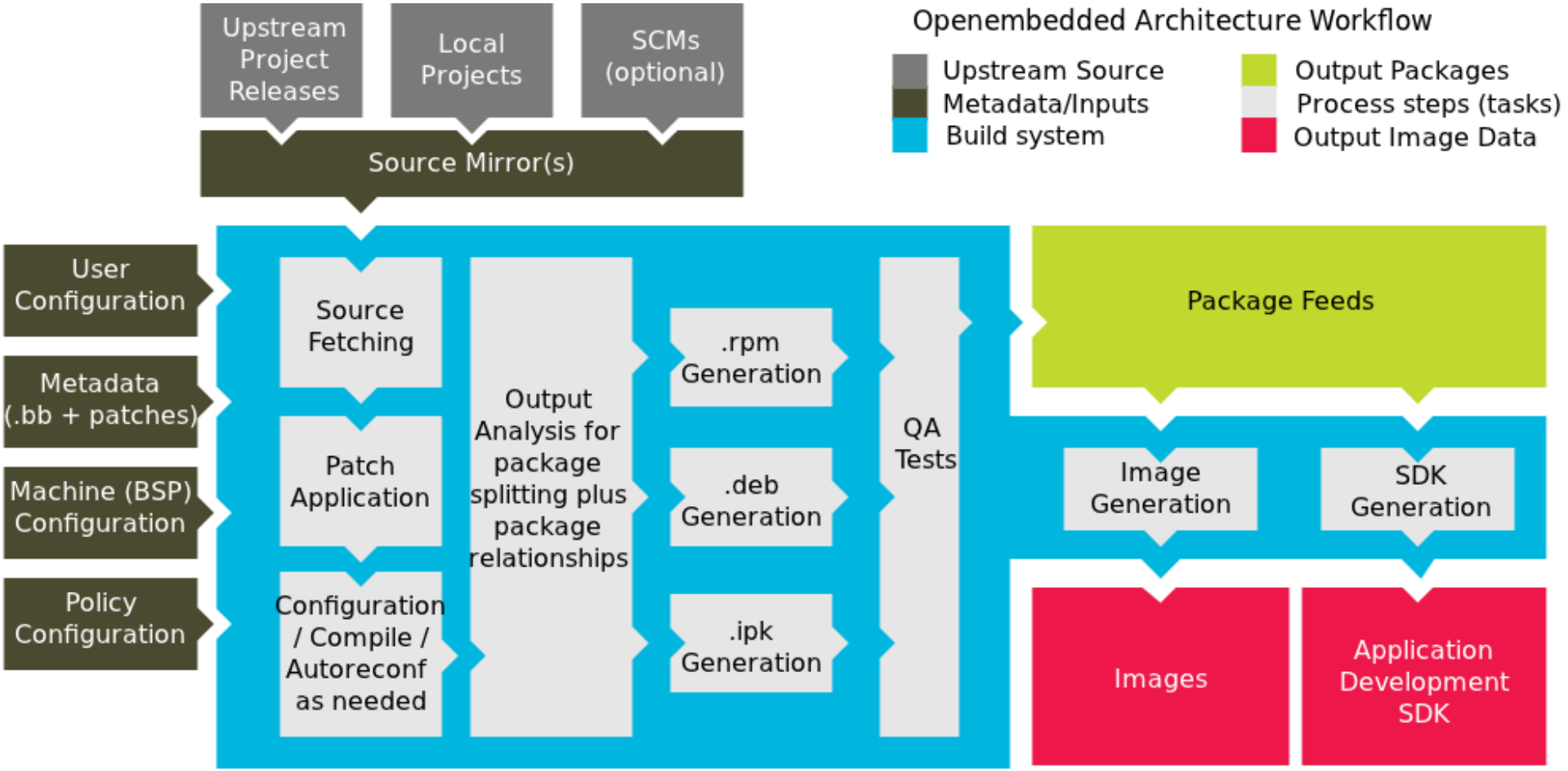
YP = Poky + Upstreams + Tools



YP provides best of upstream for a stable base

yocto PROJECT · THE LINUX FOUNDATION

How Does It Work? More Depth

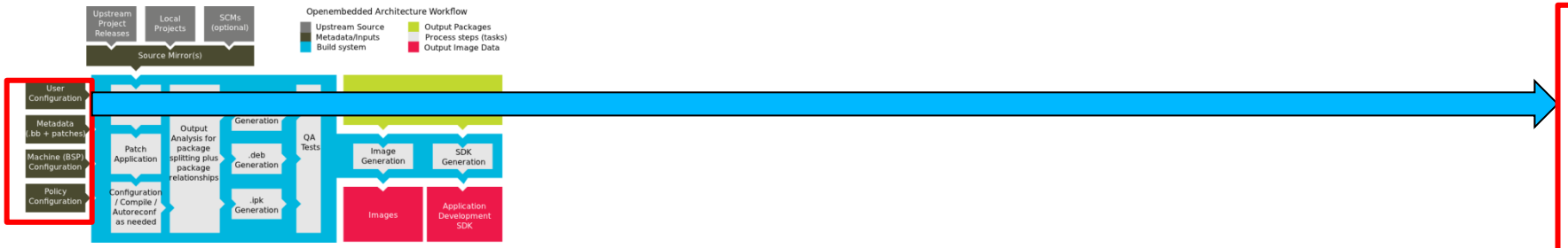


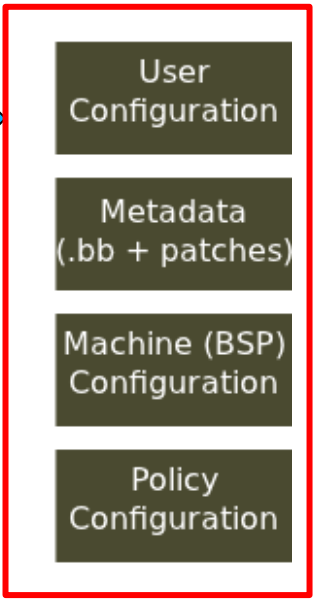
Look here for links to slides and video tutorials!

More info: http://bit.ly/it9rkB

yocto PROJECT · THE LINUX FOUNDATION

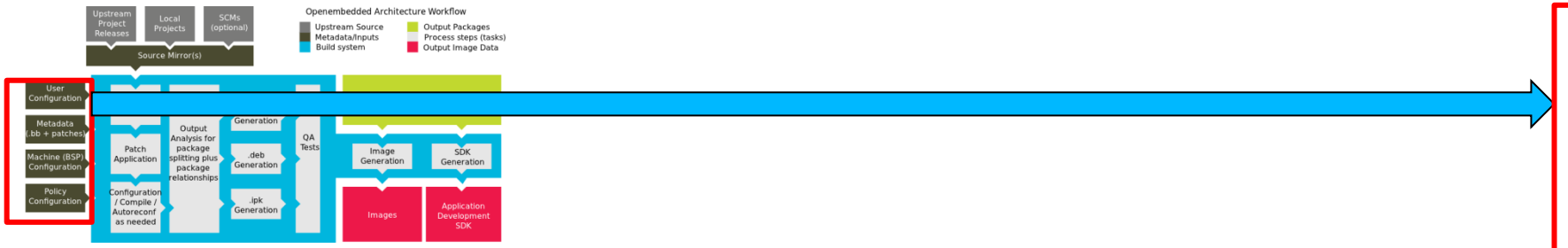
How Does it Work? Configuration





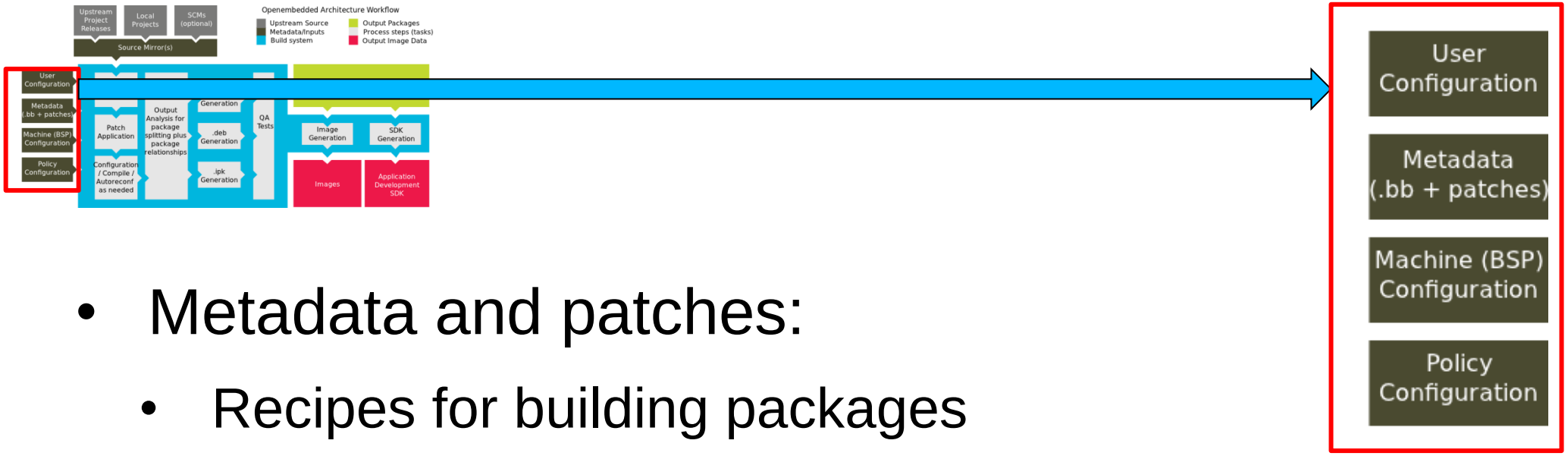
- Configuration (*.conf) – global definition of variables
  - build/conf/local.conf (local user-defined variables)
  - distro/poky.conf (Yocto policy config variables)
  - machine/routerstationpro.conf (machine-specific variables)

How Does It Work? Configuration



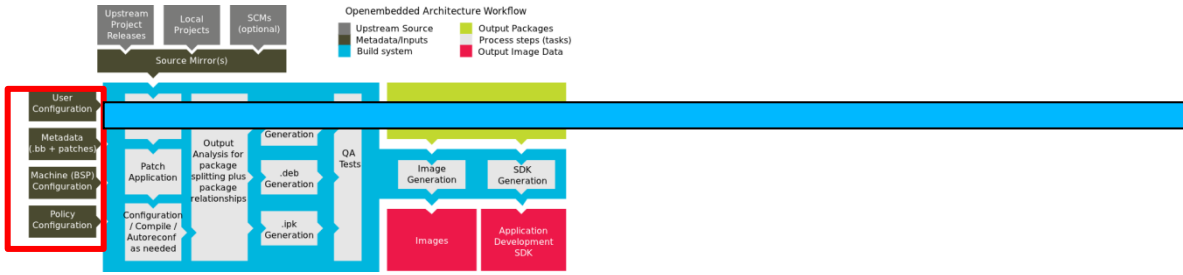
- User configuration:
  - conf/local.conf – some things to set:
    - Set BB_NUMBER_THREADS and PARALLEL_MAKE, based on the number of threads in the machine
    - Set MACHINE="foo" for the CPU architecture
    - EXTRA_IMAGE_FEATURES adds features (groups of packages)
    - INCOMPATIBLE_LICENSE = "GPLv3"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

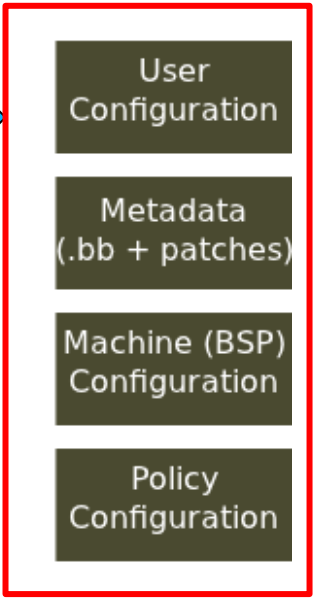
How Does It Work? Metadata

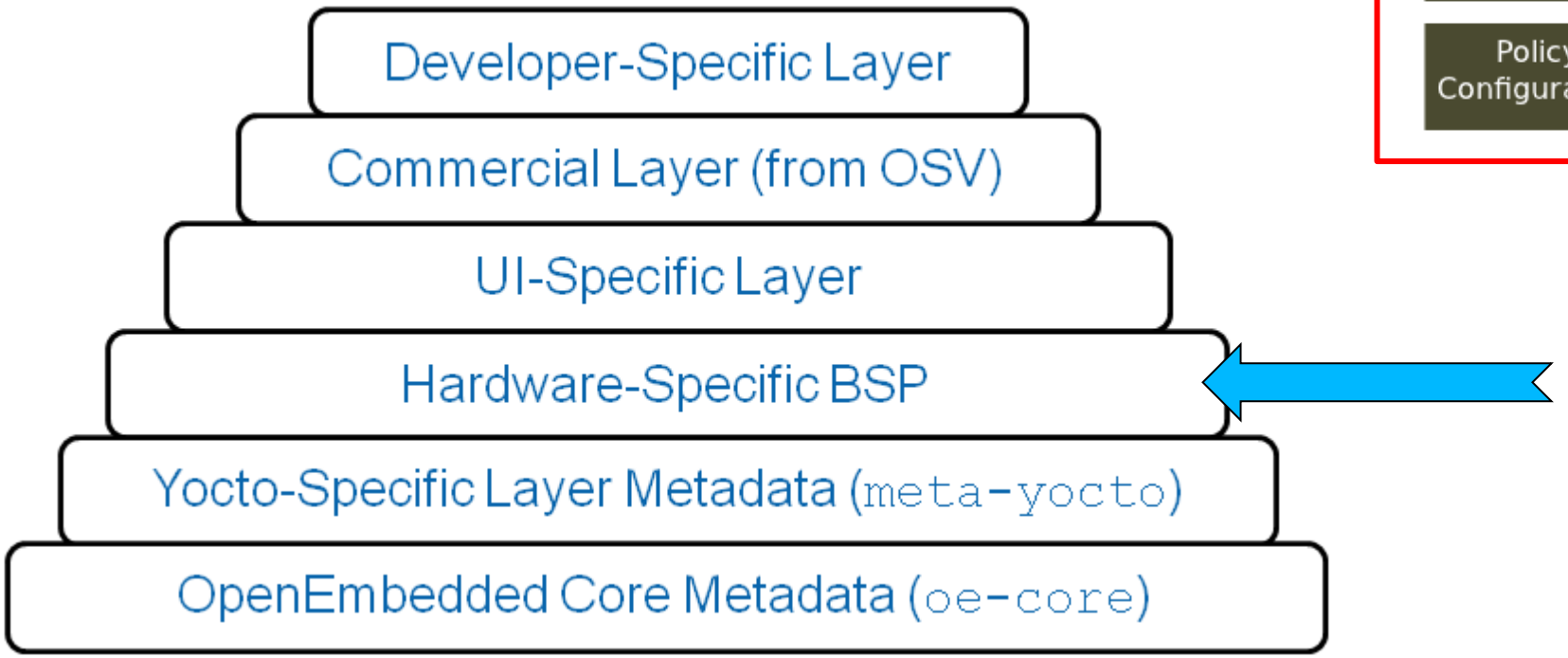


- Metadata and patches:
  - Recipes for building packages
  - Eg, `meta/recipes-core/coreutils/coreutils_6.9.bb` builds the core utilities (version 6.9) and installs them
  - `meta-recipes-core/coreutils/coreutils-6.9/` includes patches, also could include extra files to install

How Does It Work? Layers







yocto PROJECT · THE LINUX FOUNDATION

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

More info: bit.ly/lWaszt & bit.ly/m8nirP

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

More info: bit.ly/iZUkvk & bit.ly/jRSfwO

Kernel Tools Details

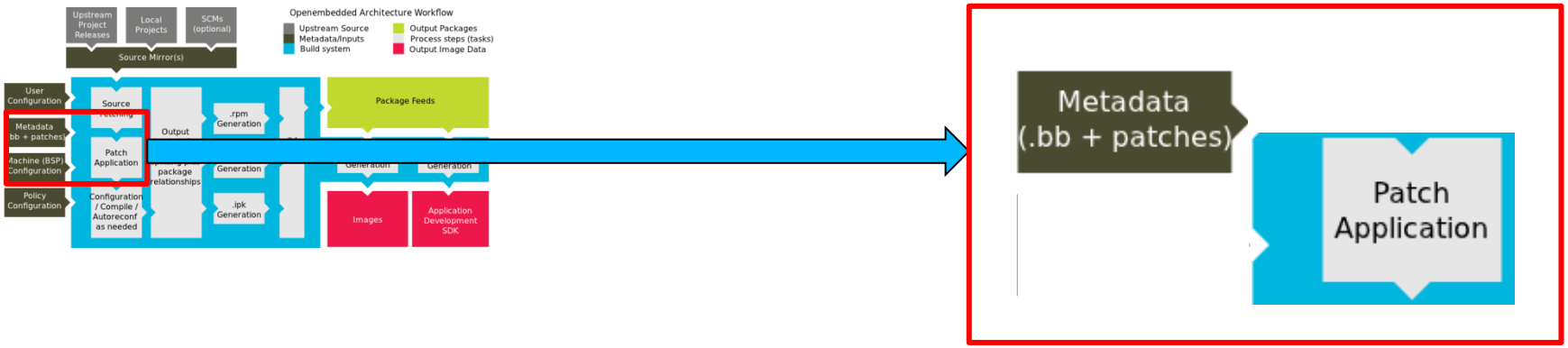
- Components
  - Kernel class
    - meta/classes/kernel.bbclass
  - Linux-Yocto recipe
    - meta/recipes-kernel/linux/linux-yocto*bb
  - Linux-Yocto git repository
    - http://git.pokylinux.org/cgit/cgit.cgi/linux-yocto-2.6.37
- Kernel Versions
  - linux-yocto-stable: 2.6.34
  - linux-yocto: 2.6.37
  - *linux-yocto-dev: 2.6.39 (meta-kernel-dev)* ***(soon 3.0)***
  - *linux-2.6: current mainline git (meta-kernel-dev)*

Source Fetching

- Recipes call out location of all sources, whether on the internet or local (Look for SRC_URI in *.bb files)
- Bitbake can get sources from git, svn, bzr, from tarballs, and many, many more*
- Versions of packages can be fixed or updated automatically (Add SRCREV_pn- PN = "${AUTOREV}" to local.conf)
- Yocto Project sources mirror available as a fallback, if the sources move on the internet

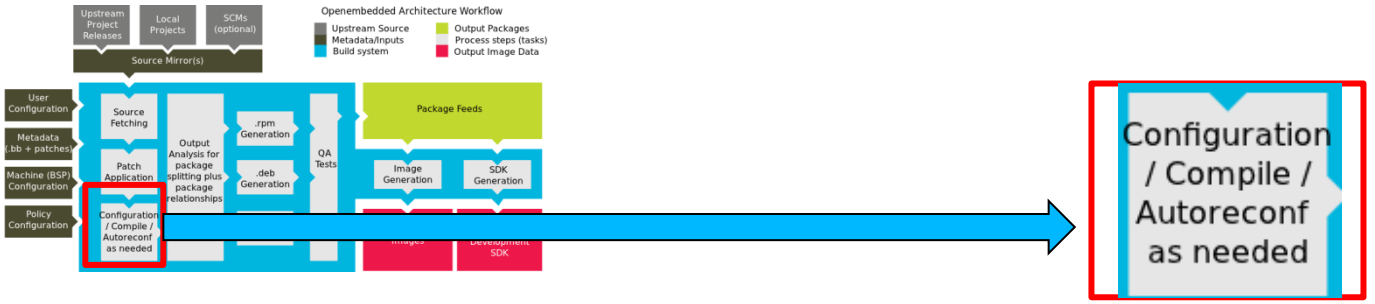
* Complete list includes: http, ftp, https, git, svn, perforce, mercurial, bzr, cvs, osc, repo, ssh, and svk and the unpacker can cope with tarballs, zip, rar, xz, gz, bz2, and so on.

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
```

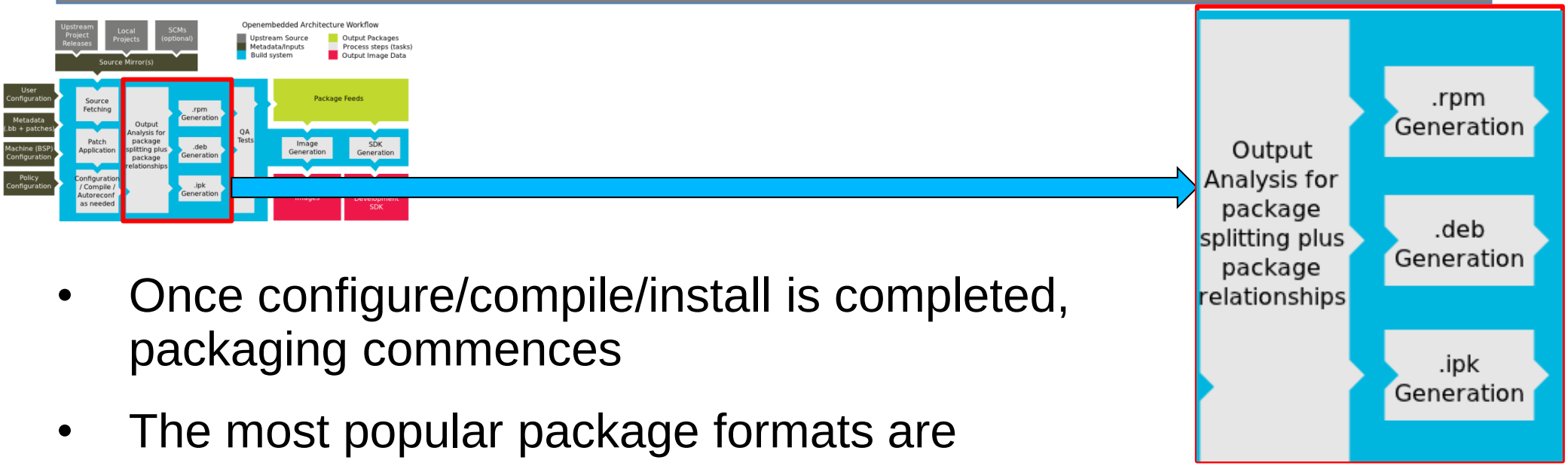
- CFLAGS can be set

```
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}
```

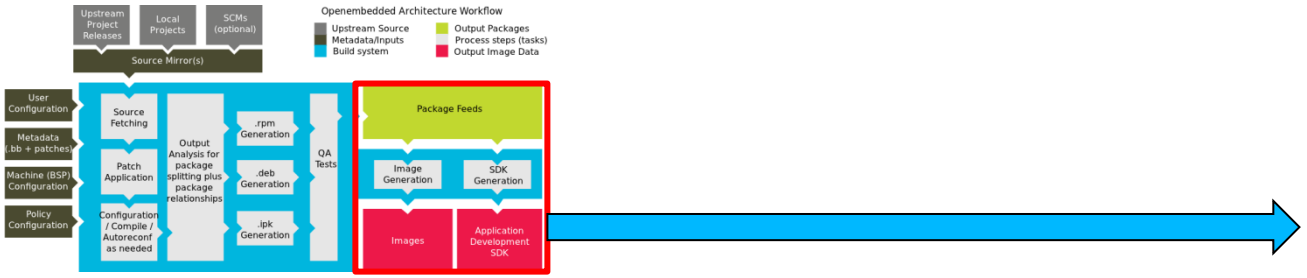
Packaging

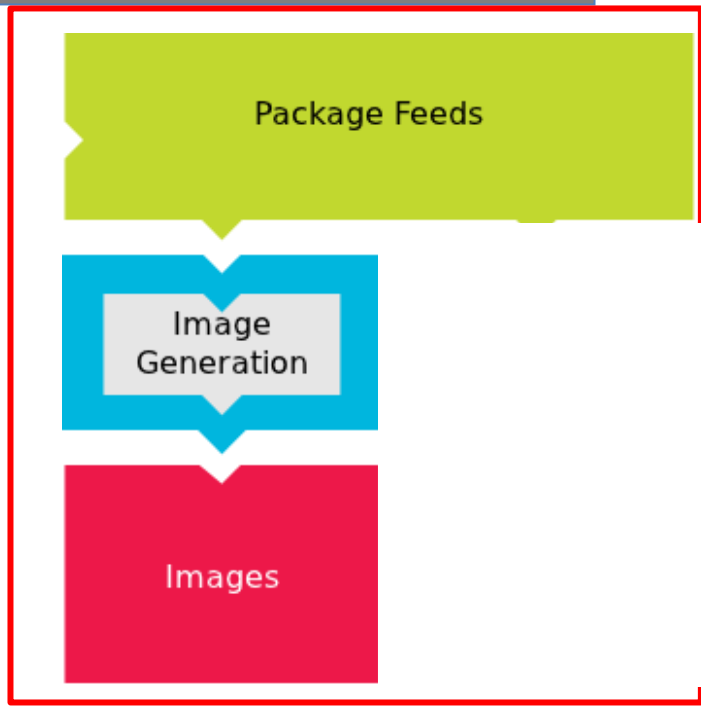


- Once configure/compile/install is completed, packaging commences
- The most popular package formats are supported: RPM, Debian, and ipk
  - Set `PACKAGE_CLASSES` in `conf/local.conf`
- You can split into multiple packages using PACKAGES and FILES in a *.bb file:

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

Image Generation

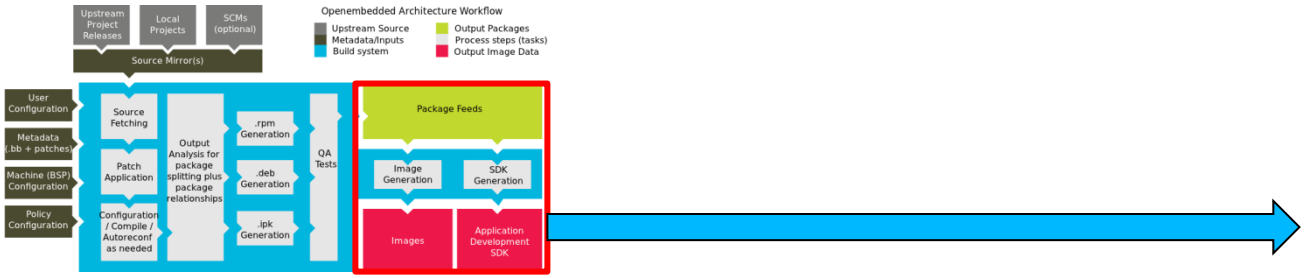


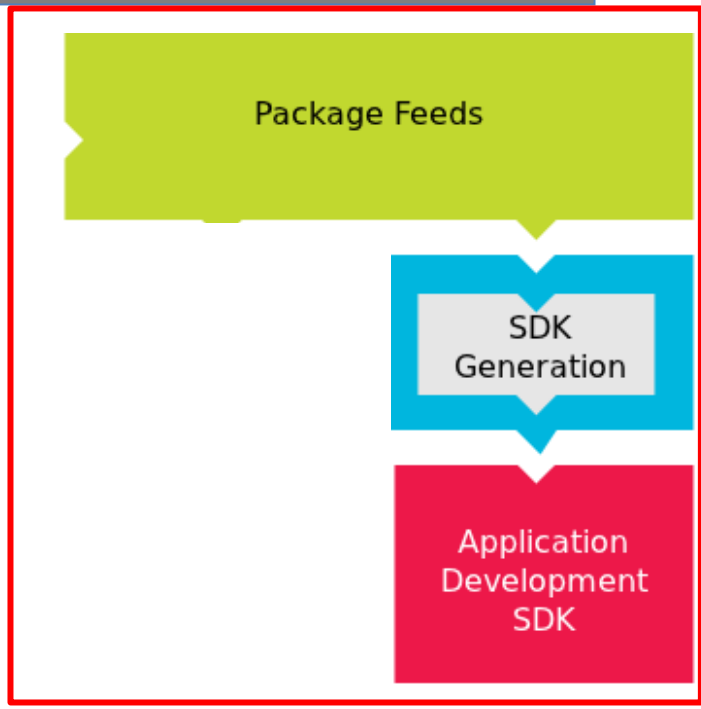


- Images are constructed using the packages built earlier in the process
- Uses for these images:
  - Live Image to boot a device
  - Root filesystem for QEMU emulator
  - Sysroot for App development

YP lets you customize your embedded Linux OS

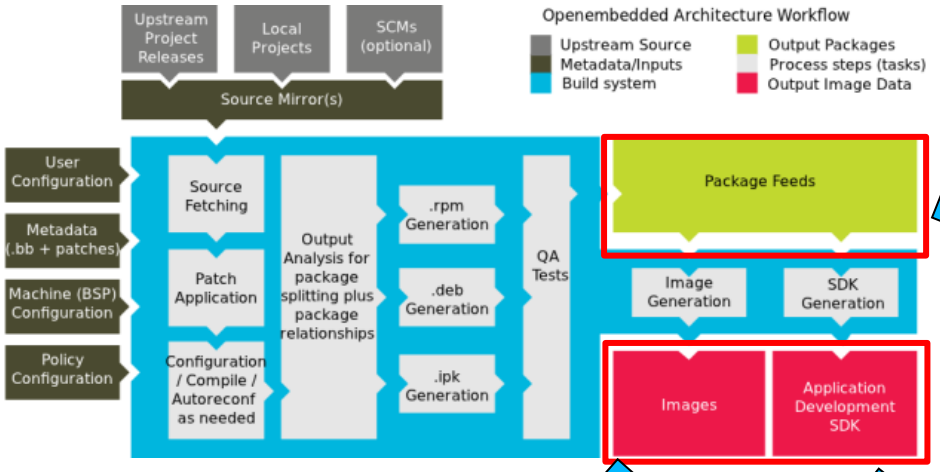
ADT Generation





- 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

Package Repository (networked or local)

Sysroot (Bootable Linux filesystem tree with development headers)

Yocto plug-ins

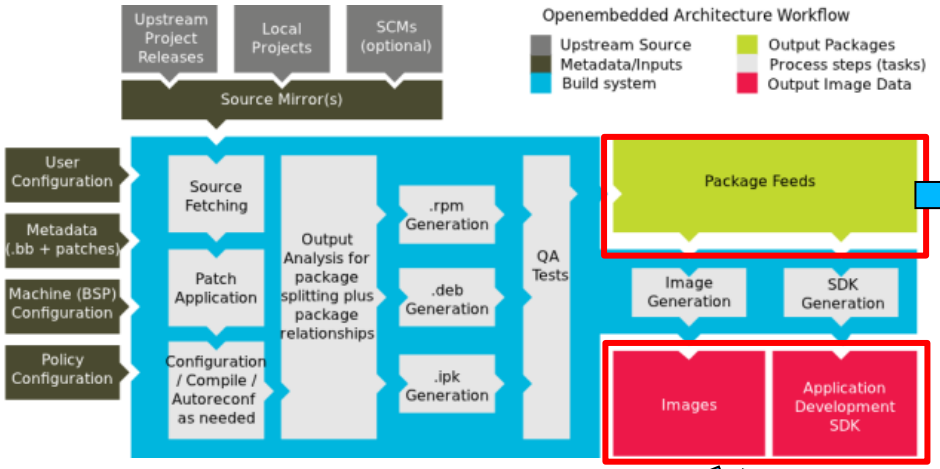
Cross toolchain installation (such as: /opt/poky)

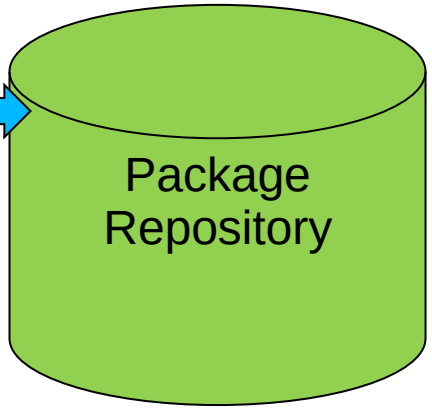
YP helps set up the embedded app developer

More info: bit.ly/mz6uRv & bit.ly/j55IQ3

yocto PROJECT · THE LINUX FOUNDATION

Use NFS/Local Disk, Pkg Manager





System Developer

App Developer

QEMU Device emulator

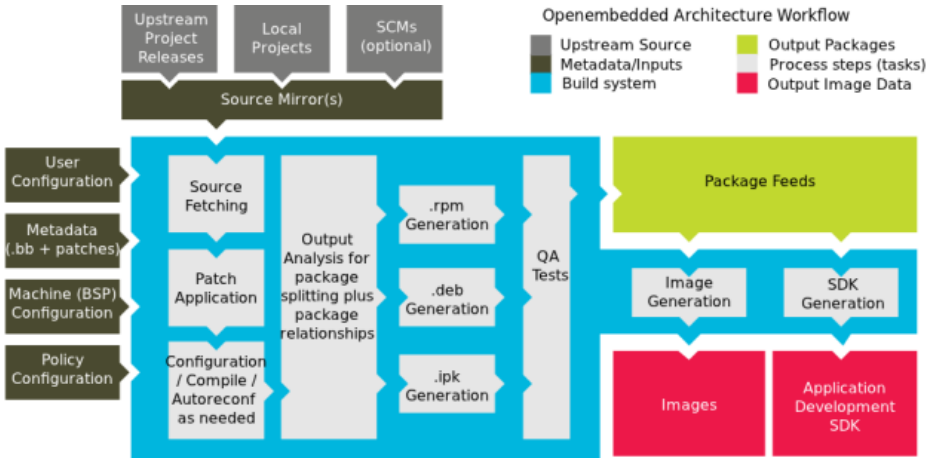
Sysroot

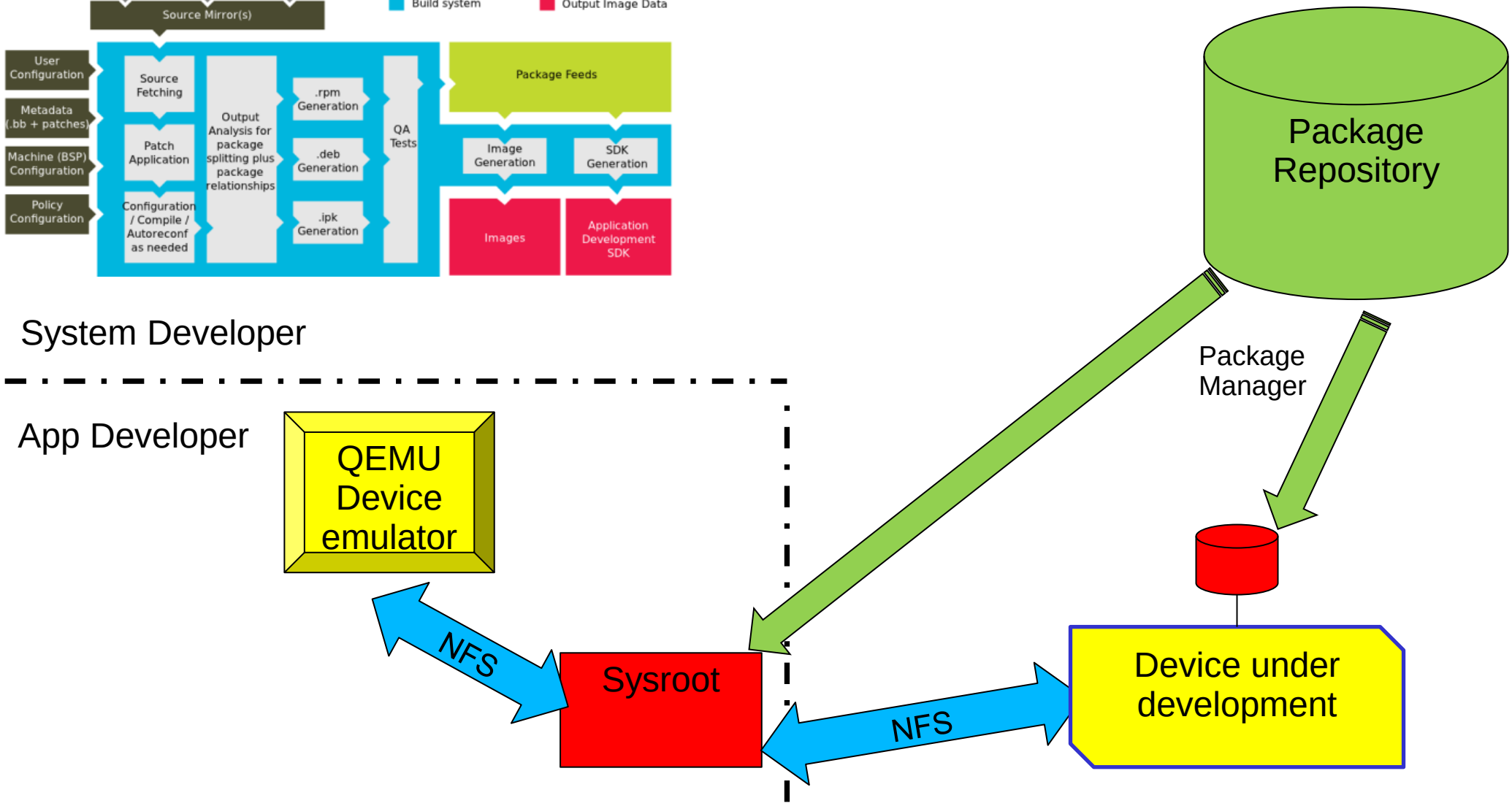
Device under development

yocto PROJECT

THE LINUX FOUNDATION

Use NFS/Local Disk, Pkg Manager





Both Device and App Development Models Supported

What's Next?

- Constantly improve the developer's experience
  - Identify areas which are confusing and constantly improve them
  - Improvements on the Hob
  - Isolate all Linux development system uncertainties
- Updated kernel, toolchain, user land packages
- More partner's products

How to Get Started

- 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
- Join the community to get help
  - #yocto on freenode and yocto@yoctoproject.org (http://lists.yoctoproject.org/listinfo/yocto)

Getting started with the Yocto Project is easy

Get Involved

- The Yocto Project is a collaboration of individuals, non-profits, and corporations under the Linux Foundation
- We urge you or your organization to join
- yoctoproject.org/documentation/getting-started has a number of ways to learn and contribute
  - Contribute code, documentation, fix bugs, provide BSPs
  - Use YP for your embedded projects
  - Work with the community to make YP better

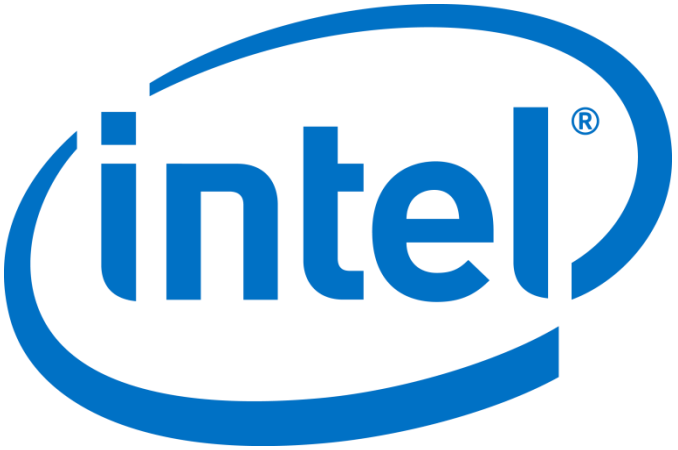
Make an impact – collaboration in its purest sense

It’s Time to Take Action

- 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
- Make an impact – collaboration in its purest sense

Q & A

Thanks!


intel