

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

## The Yocto Project Eclipse plug-in: An Effective IDE Environment for Embedded Application and System Developers

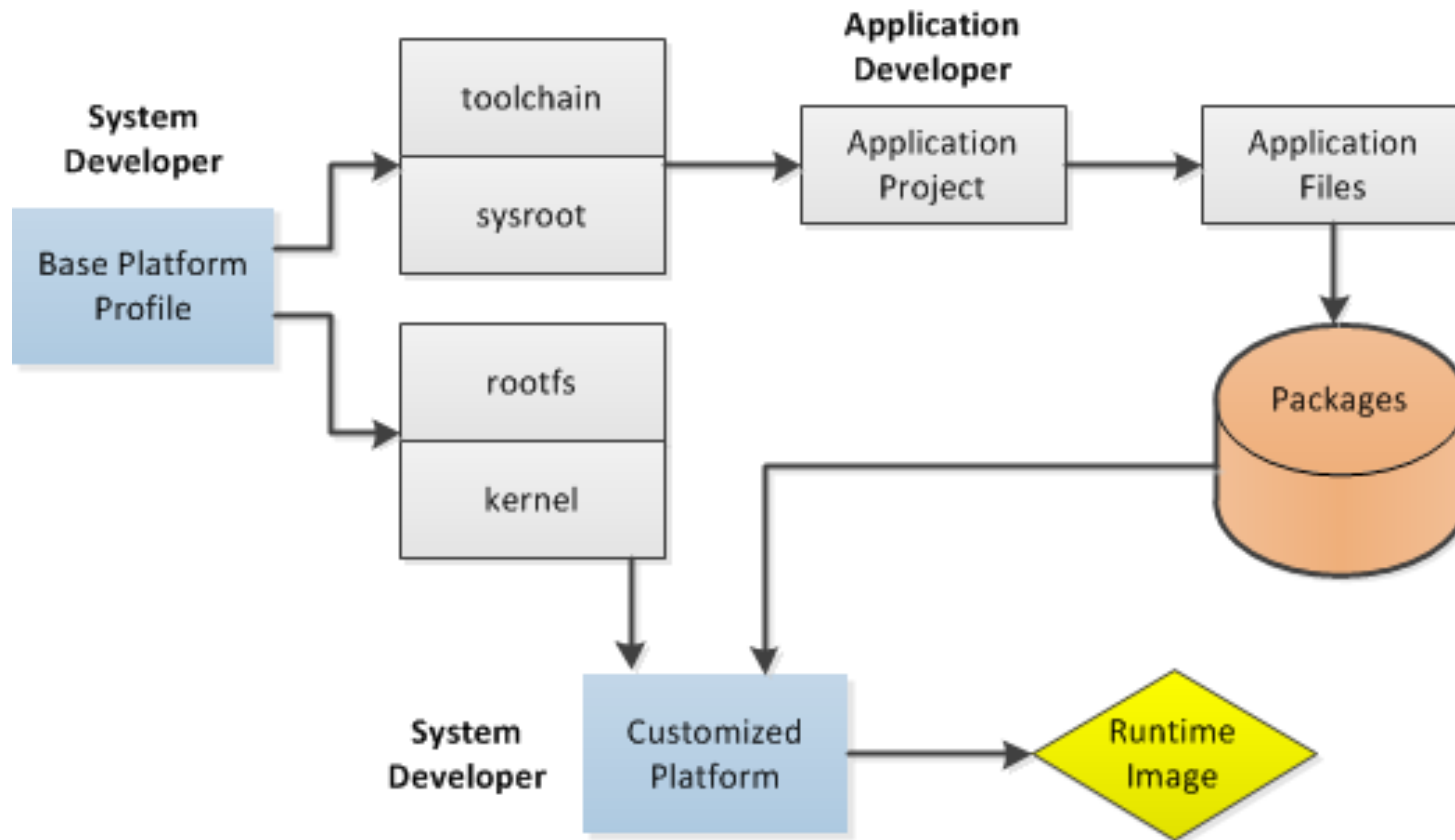
**Jessica Zhang**  
Intel Corporation  
Oct. 26, 2011



# Agenda

- **Embedded Linux Development**
- **What The Yocto Project Offers Embedded Linux Development**
- **The Yocto Project Eclipse Plug-in**
  - For System Developers
  - For Application Developers
- **What's Next?**

# Embedded Linux Development Flow

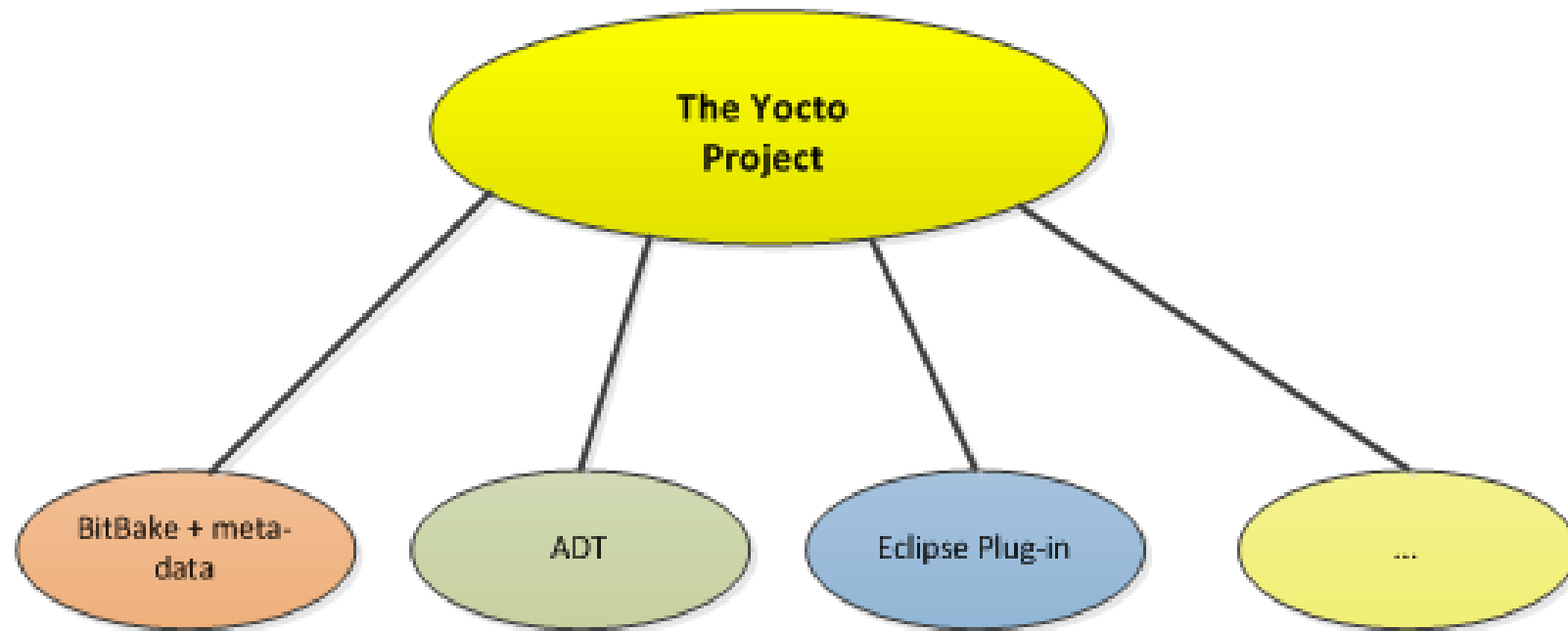


# Embedded Linux Development

- **System developer - develop the Linux systems for the targeted embedded devices:**
  - ✓ Build profile customization through package selection
  - ✓ Tune the image footprint
  - ✓ Create reproducible build with flexibility in customization (for example, target architecture, package format)
  - ✓ Build toolchain for application developers
- **Application developer - develop applications running on the targeted embedded devices:**
  - ✓ Use cross-toolchain
  - ✓ Take advantage of sysroot setup
  - ✓ Remotely debug application on target (real HW or emulator)
  - ✓ Tune performance using profiling/tracing tools

**A framework that streamlines the development flow is highly desirable**

# What The Yocto Project Offers Embedded Linux Development



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

# What The Yocto Project Offers Embedded Linux Development

## • The build system and meta-data:

- Using BitBake - a widely adopted build system by the embedded Linux developers
- Meta-data consists of recipe and configuration files
- Easy customization / extension of the core meta-data through layers
- HOB – A graphical user interface for BitBake

You don't need to be an expert of BitBake to be able to customize your build and image

## • The Application Development Kit (ADT):

- Cross-toolchain for the target device
  - ✓ Supports sysroot setup
  - ✓ Optimized for autotool-based projects
- Qemu emulator
  - ✓ Can be booted through unfs
  - ✓ Rootfs is extracted as sysroot
- Tools for target analysis, profiling and tracing

# The Yocto Project Eclipse Plug-in

- **An IDE environment to streamline the development flow:**
  - Wizard
  - Template
- **Based on open source solutions:**
  - Eclipse communities' CDT, RSE, TCF and LinuxTools projects
  - BitBake Commander Project
- **Within one IDE, users can fully benefit from Yocto Project offerings:**
  - BitBake (through Hob)
  - Meta-data
  - ADT
  - Qemu
  - Tracing and profiling tools

# The Yocto Project Eclipse Plug-in For the System Developers

**Without the Plug-in, you must do these steps all from the command-line.**

1. Clone the Yocto Project meta-data
2. Edit recipe files using your preferred editor: `emacs`, `vim`, ...
3. Source `oe-init-build-env` to setup your build directory
4. Edit the `conf/local.conf` file to configure the Yocto build and then use BitBake to kick off the build

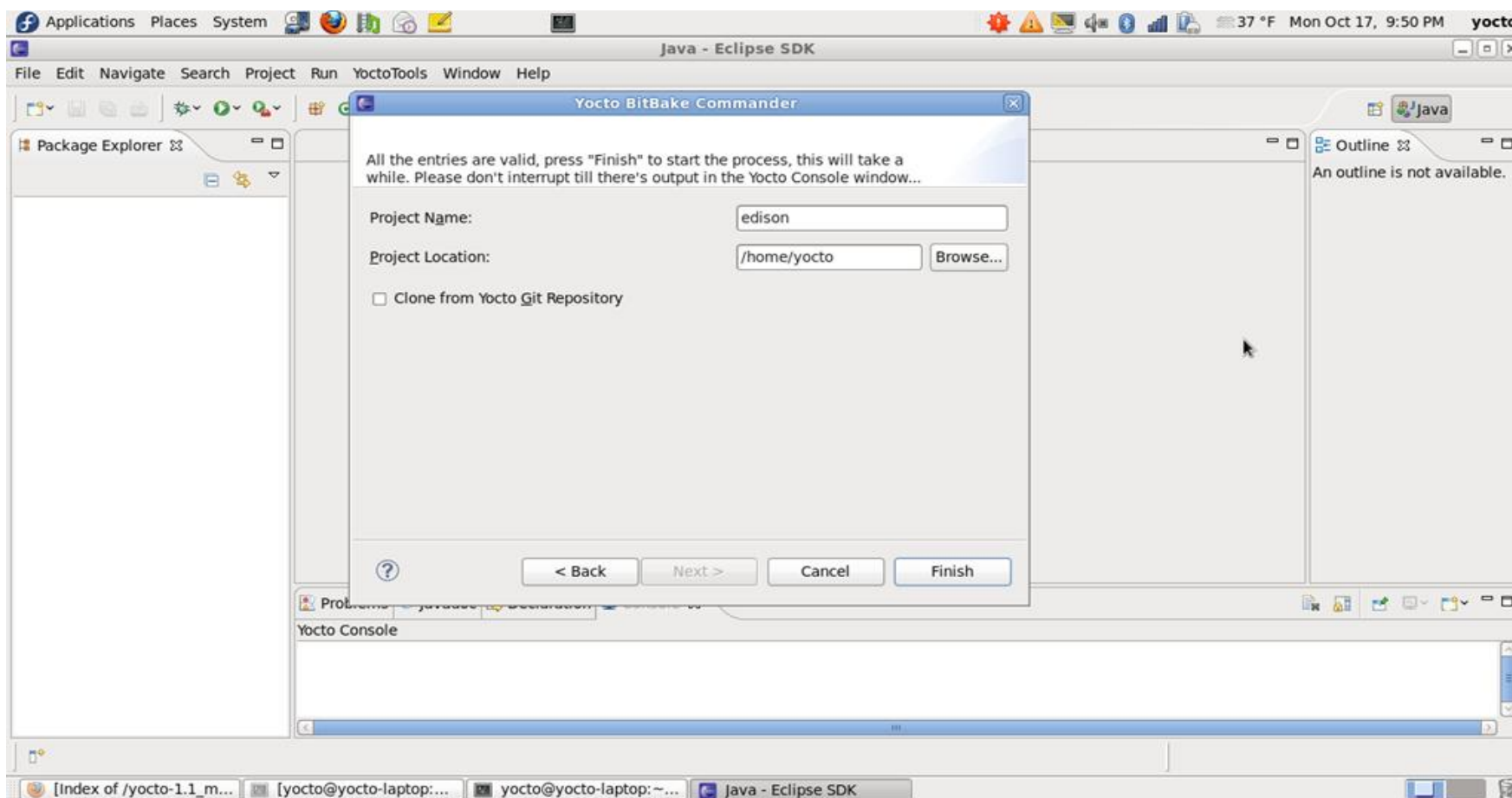
- OR -

4. Use Hob to facilitate further build customization, and then run the Yocto build from within Hob



# The Yocto Project Eclipse Plug-in For System Developers

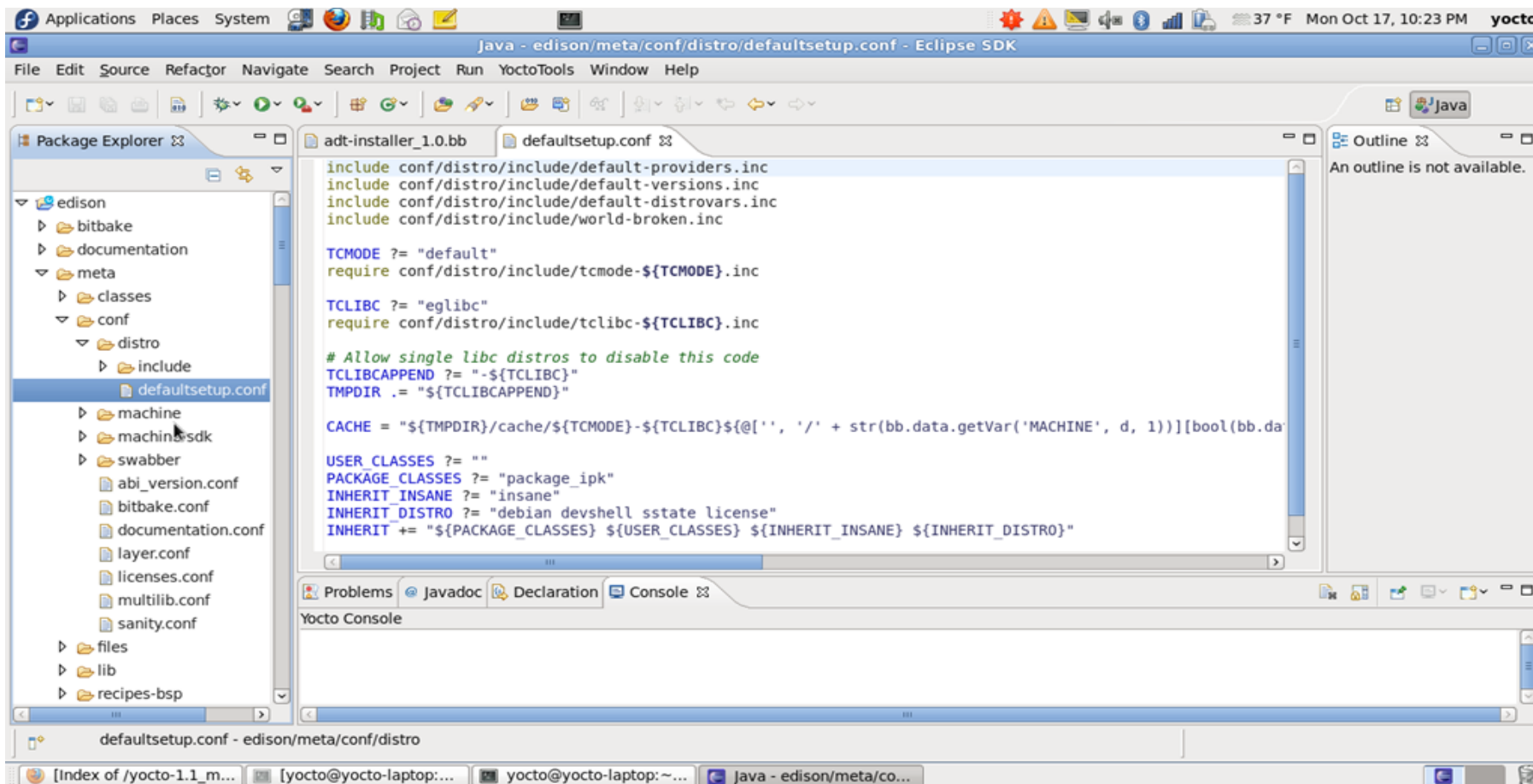
**Step 1: Create Yocto BitBake Commander Project for Yocto Project meta-data (Note: Collaborate with other open source plug-ins, e.g. egit)**



# The Yocto Project Eclipse Plug-in For System Developers

## Step 2: Customize meta-data recipe files

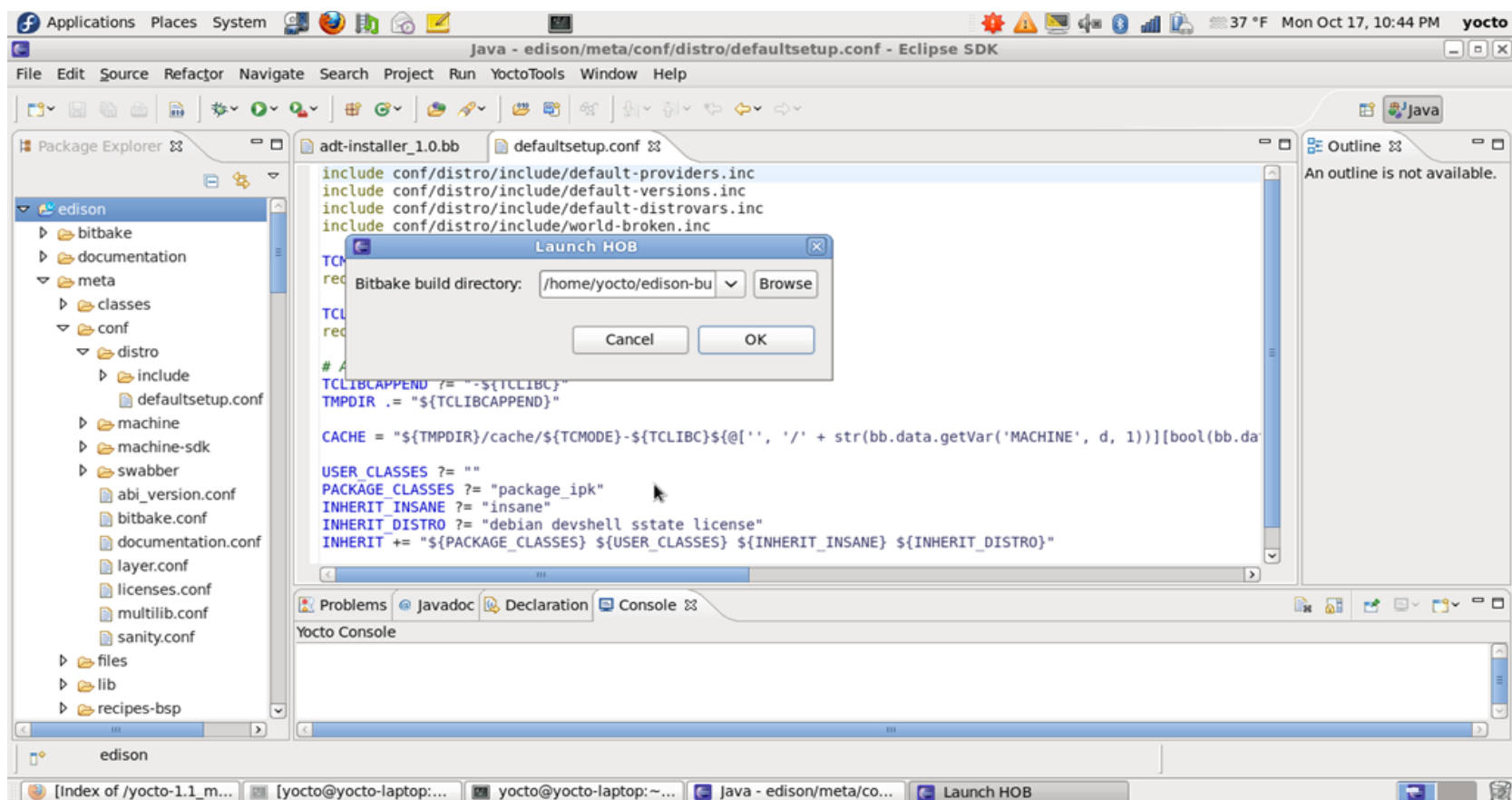
- Navigate the meta-data in the project tree view
- “Yocto BitBake Recipe Editor” with keywords highlighted
- New Recipe Wizard allows the user to quickly create new recipe files



# The Yocto Project Eclipse Plug-in For System Developers

## Step 3: Launch Hob

- Set up a separate build area for the customized meta-data
- Using Hob, further customize and configure your build and image output
- Run the build from Hob



# The Yocto Project Eclipse Plug-in For System Developer

## Hob

Machine:  Base image:  Loaded

Packages Package Collections

Package	Description	License	Group	Included
a52dec	liba52 version 0.7.4-r3	GPLv2+	libs	<input type="checkbox"/>
a52dec-doc	liba52 version 0.7.4-r3	GPLv2+	libs	<input type="checkbox"/>
acl	acl version 2.2.51-r2	LGPLv2.1+ & GPLv2+	libs	<input checked="" type="checkbox"/>
acl-dbg	acl version 2.2.51-r2	LGPLv2.1+ & GPLv2+	libs	<input type="checkbox"/>
acl-dev	acl version 2.2.51-r2	LGPLv2.1+ & GPLv2+	libs	<input type="checkbox"/>
acl-doc	acl version 2.2.51-r2	LGPLv2.1+ & GPLv2+	libs	<input type="checkbox"/>
acl-locale	acl version 2.2.51-r2	LGPLv2.1+ & GPLv2+	libs	<input type="checkbox"/>
acl-staticdev	acl version 2.2.51-r2	LGPLv2.1+ & GPLv2+	libs	<input type="checkbox"/>

Search packages:

Estimated image contents (631 packages):

Package	Brought in by
acl	udev-cache, udev
alsa-conf-base	alsa-lib
alsa-lib	gst-plugins-base, alsa-utils-alsaucm, alsa-utils-midi, alsa-utils-iecset, alsa-utils-aplay, alsa-utils-alsactl, alsa-utils-speakertest, alsa-utils-
alsa-utils	alsa-utils-amiixer, alsa-utils-alsaucm, alsa-utils-midi, alsa-utils-iecset, alsa-utils-aplay, alsa-utils-alsactl, alsa-utils-speakertest, alsa-utils-
alsa-utils-aconnect	alsa-utils-aseqnet, alsa-utils-alsaloop, alsa-utils-aseqdump, alsa-utils-speakertest, alsa-utils-alsactl, alsa-utils-aplay, alsa-utils-iecset, al
alsa-utils-alsaconf	alsa-utils-aconnect, alsa-utils-aseqnet, alsa-utils-aseqdump, alsa-utils-speakertest, alsa-utils-alsactl, alsa-utils-aplay, alsa-utils-iecset, a
alsa-utils-alsactl	alsa-utils-aplay, alsa-utils-aseqdump, alsa-utils-aseqnet, alsa-utils-aconnect, alsa-utils-alsaconf, alsa-utils-alsaloop, alsa-utils-iecset, als
alsa-utils-alsaloop	alsa-utils-alsaconf, alsa-utils-aconnect, alsa-utils-aseqnet, alsa-utils-aseqdump, alsa-utils-speakertest, alsa-utils-alsactl, alsa-utils-aplay

Reset Bake

# The Yocto Project Eclipse Plug-in For System Developer

## Demo

# The Yocto Project Eclipse Plug-in For Application Developers

**Without the Plug-in, you must do these steps all from the command-line.**

1. Set up your cross toolchain and sysroot for cross development
2. Create your Makefile or autotool-based project
  - Best with autotool-based projects. Just pass host options to configure (e.g. `./configure host=i686-poky-linux --with-libtool-sysroot=/home/jzhang/x86`)
  - For other projects, ensure the cross-tools are used, (e.g. `CC=i686-poky-linux-gcc` and `LD=i686-poky-linux-ld` in makefile)
3. Compile your project
4. Optionally bring up the Qemu emulator using the command line
5. Deploy your application to the remote target: `rcp`, `scp`, `rsync`, etc.
6. Setup cross debugging against the desired target: Qemu or real hardware
  - Start `gdbserver` on target
  - Run `cross-gdb` on host side to connect to remote target
7. Perform target analysis tasks like tracing and profiling
  - Follow each tool's special setup for remote launch or interaction from the host

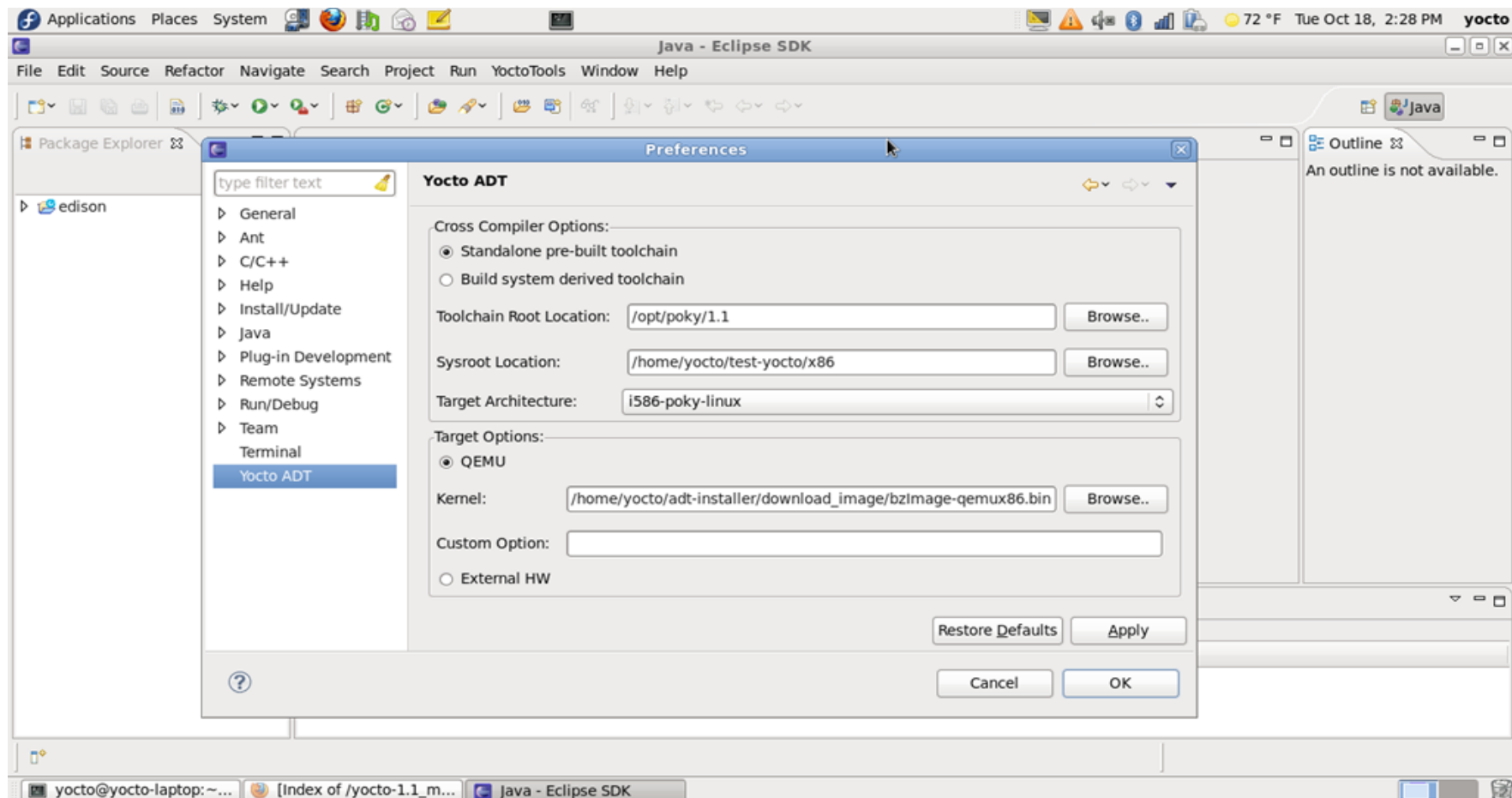
**This is a very complex task if doing everything on your own and can dramatically slow down your development cycle.**

# The Yocto Project Eclipse Plug-in For Application Developer

**Step 1: Set up your cross-toolchain and sysroot for cross-development**

**Step 2: Set up your Eclipse IDE with the Yocto Project Plug-in installed**

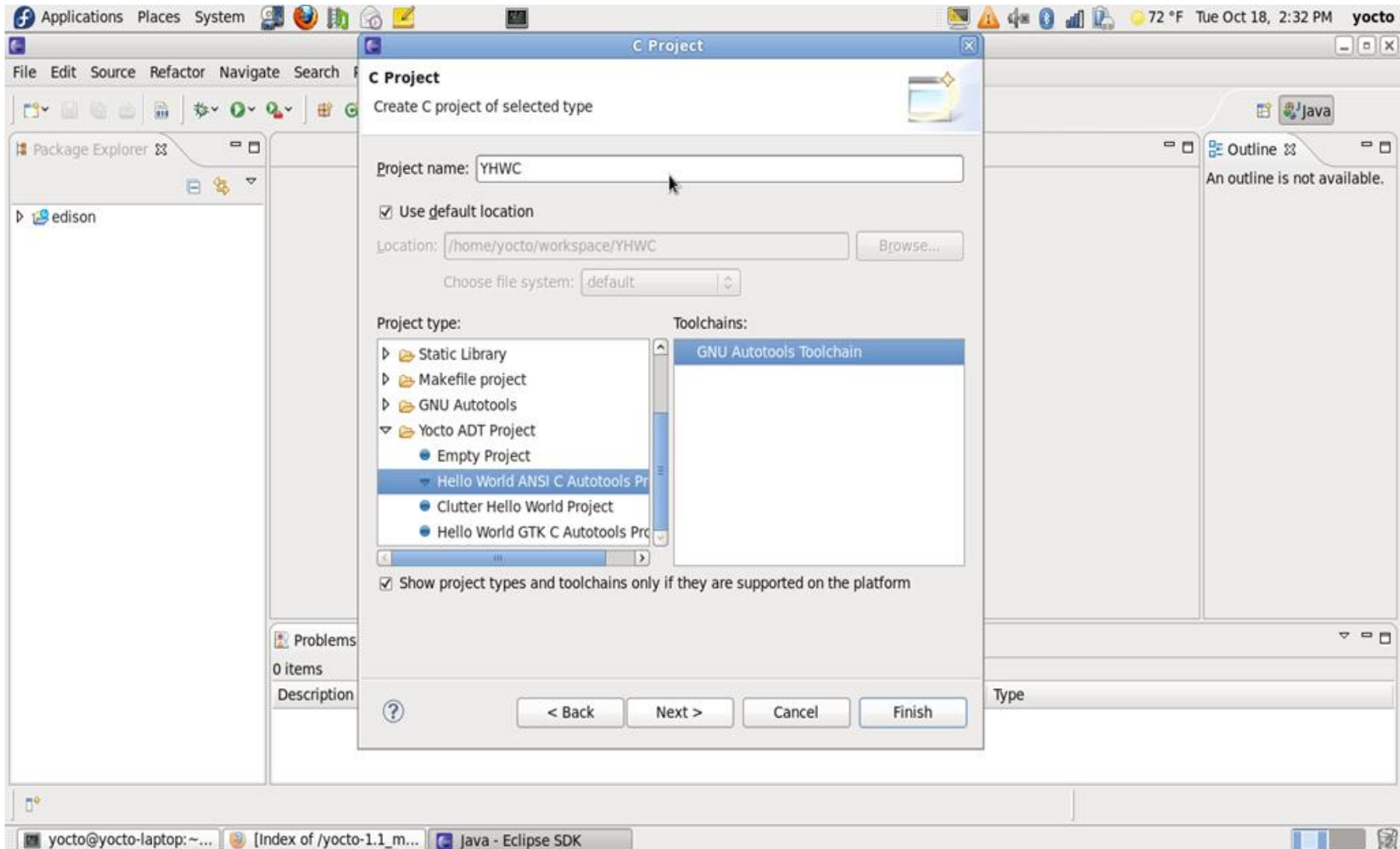
**Step 3: Configure Yocto Project ADT plug-in for IDE**





# The Yocto Project Eclipse Plug-in For Application Developer

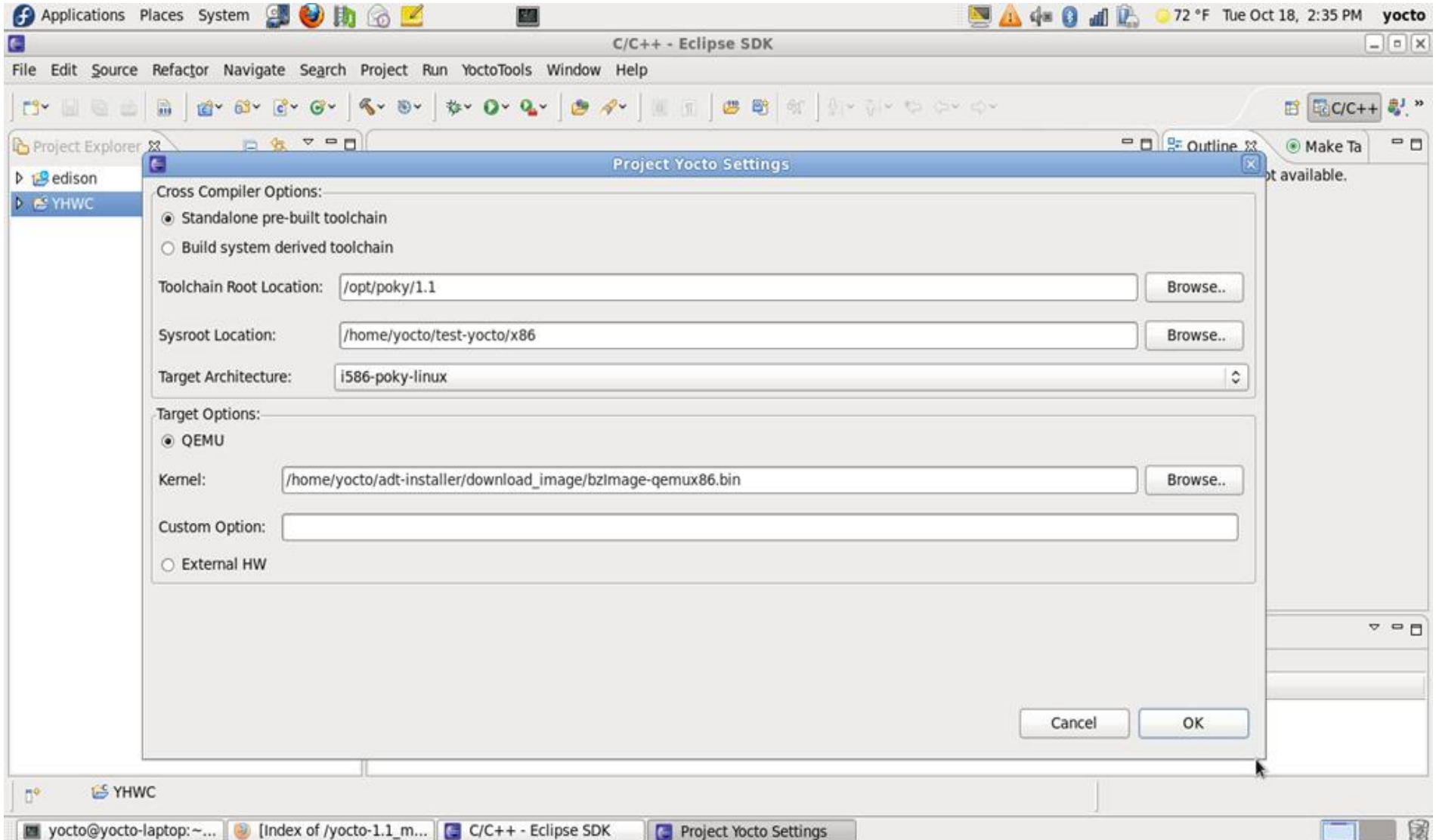
## Step 4: Pick one of the ADT autotool-based project templates





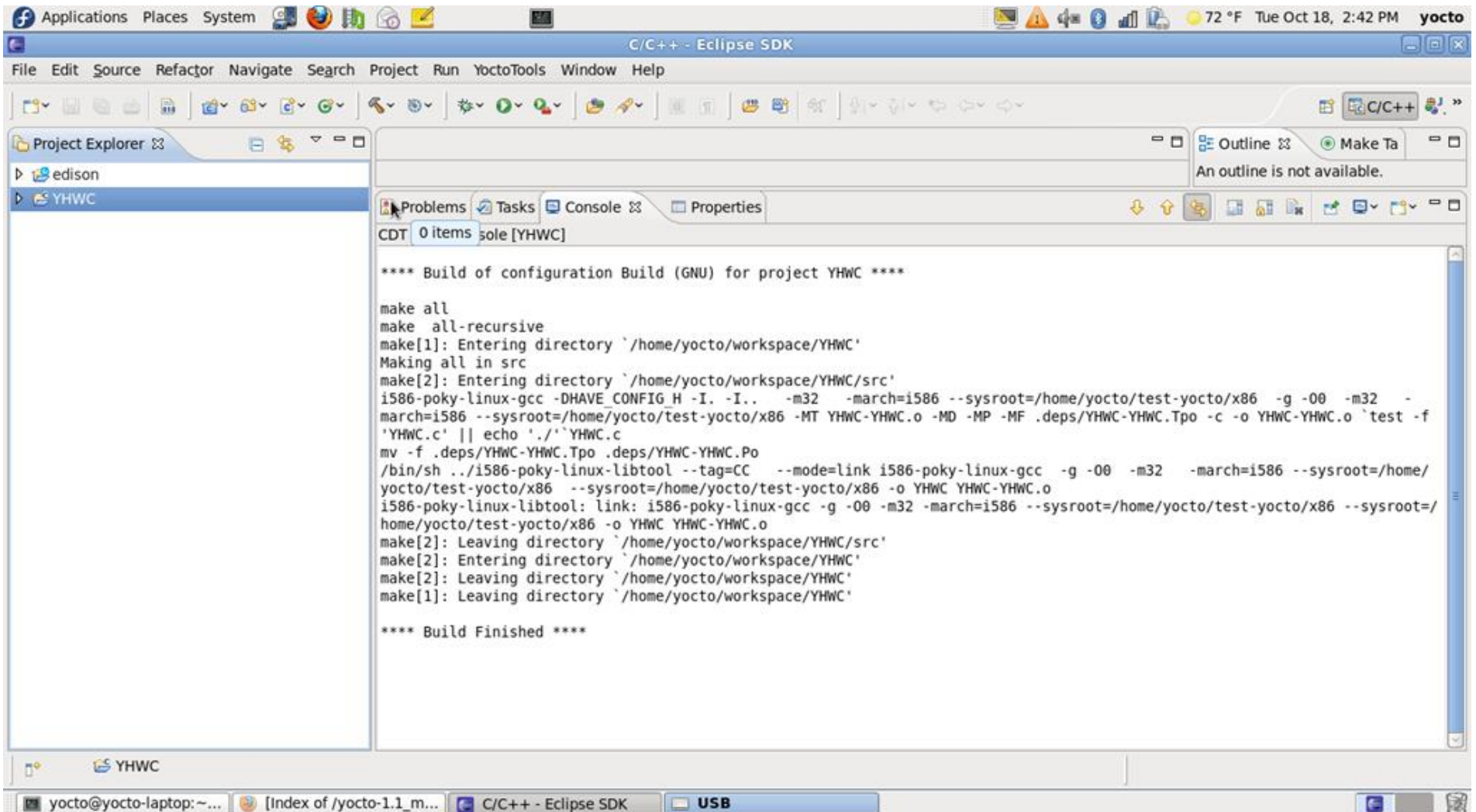
# The Yocto Project Eclipse Plug-in For Application Developer

## Step 5: Change the project's cross-developmen settings if needed



# The Yocto Project Eclipse Plug-in For Application Developer

## Step 6: Work on your project, configure and compile using cross-development settings



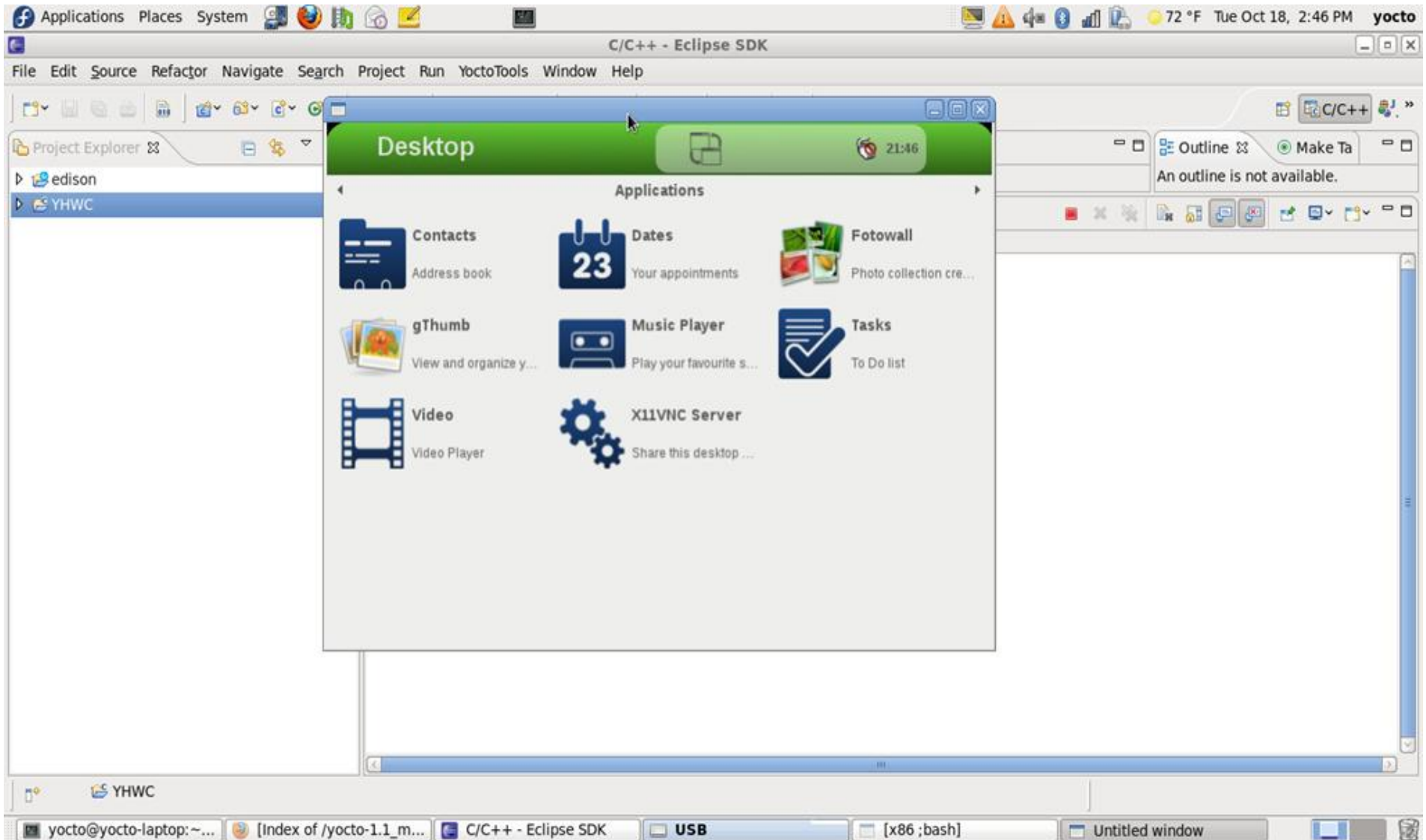
```
**** Build of configuration Build (GNU) for project YHWC ****

make all
make all-recursive
make[1]: Entering directory `/home/yocto/workspace/YHWC'
Making all in src
make[2]: Entering directory `/home/yocto/workspace/YHWC/src'
i586-poky-linux-gcc -DHAVE_CONFIG_H -I. -I.. -m32 -march=i586 --sysroot=/home/yocto/test-yocto/x86 -g -O0 -m32 -march=i586 --sysroot=/home/yocto/test-yocto/x86 -MT YHWC-YHWC.o -MD -MP -MF .deps/YHWC-YHWC.Tpo -c -o YHWC-YHWC.o `test -f 'YHWC.c' || echo './'`YHWC.c
mv -f .deps/YHWC-YHWC.Tpo .deps/YHWC-YHWC.Po
/bin/sh ../i586-poky-linux-libtool --tag=CC --mode=link i586-poky-linux-gcc -g -O0 -m32 -march=i586 --sysroot=/home/yocto/test-yocto/x86 --sysroot=/home/yocto/test-yocto/x86 -o YHWC YHWC-YHWC.o
i586-poky-linux-libtool: link: i586-poky-linux-gcc -g -O0 -m32 -march=i586 --sysroot=/home/yocto/test-yocto/x86 --sysroot=/home/yocto/test-yocto/x86 -o YHWC YHWC-YHWC.o
make[2]: Leaving directory `/home/yocto/workspace/YHWC/src'
make[2]: Entering directory `/home/yocto/workspace/YHWC'
make[2]: Leaving directory `/home/yocto/workspace/YHWC'
make[1]: Leaving directory `/home/yocto/workspace/YHWC'

**** Build Finished ****
```

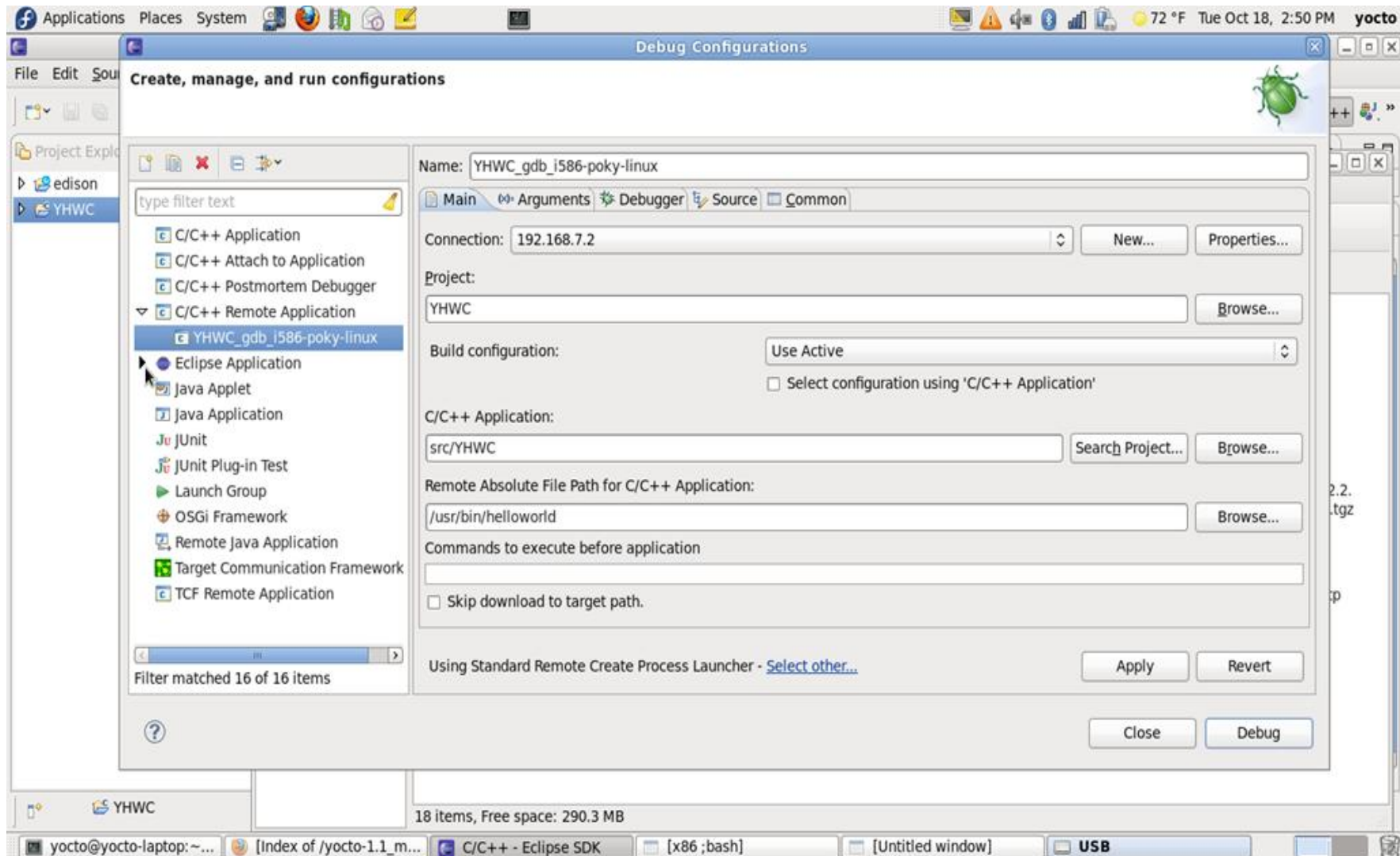
# The Yocto Project Eclipse Plug-in For Application Developer

## Step 7: Use the auto-created Qemu launcher for the target to launch Qemu



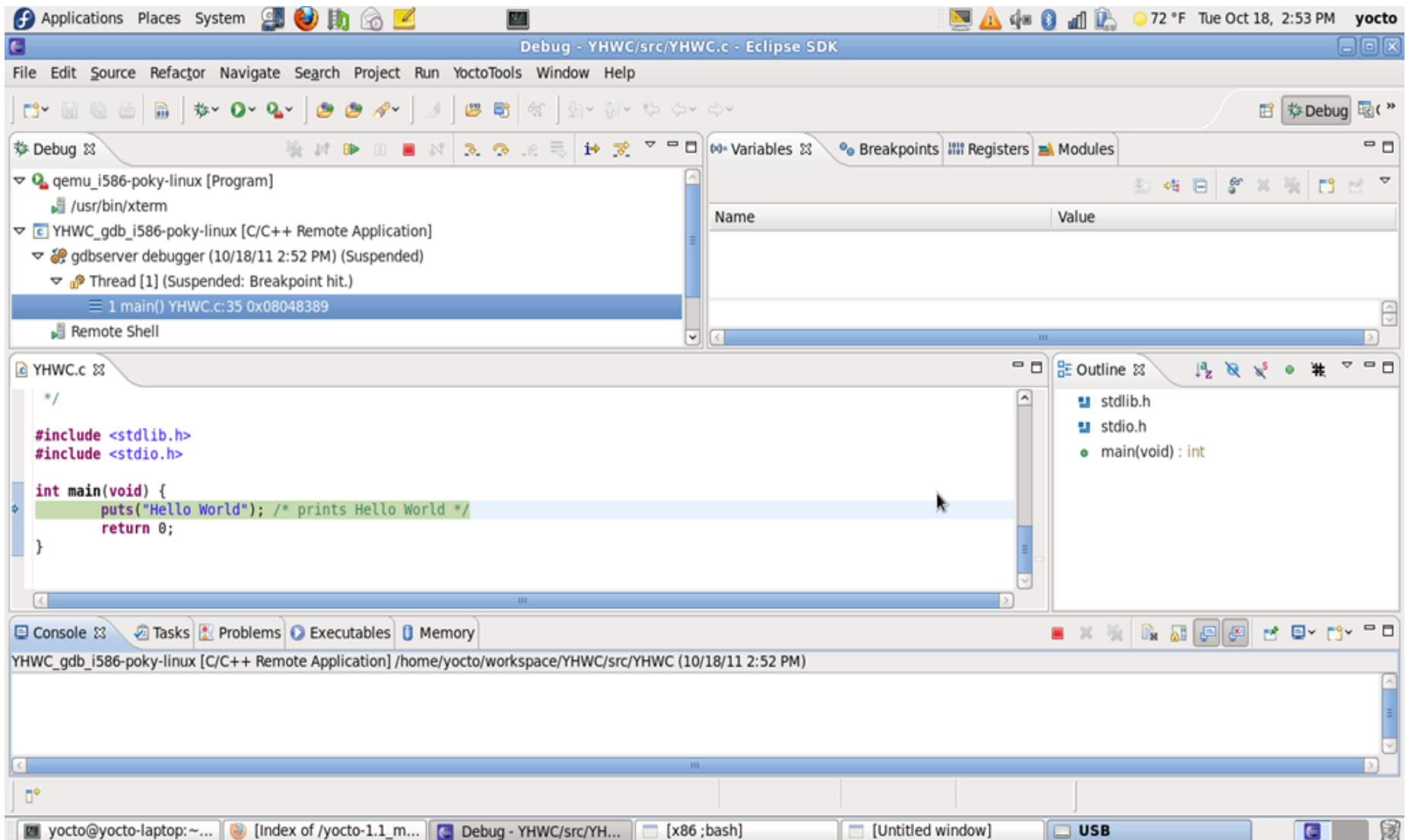
# The Yocto Project Eclipse Plug-in For Application Developer

## Step 8: Finish the auto-created remote debug configuration template for the project



# The Yocto Project Eclipse Plug-in For Application Developer

## Step 9: Launch the remote debug session





# The Yocto Project Eclipse Plug-in For Application Developer

**Step 10: Use the tools under the “YoctoTools” menu. The tools suite contains the following essential tools that provide target analytical capabilities:**

- PowerTop
- LatencyTop
- Oprofile
- Perf
- Lttng-ust
- SystemTap

# The Yocto Project Eclipse Plug-in For Application Developer

## Demo

# What's Next?

- **Continue to improve the Yocto Project's overall user experience is the main theme of the next release**
- **Add new tools:**
  - BSP/Kernel configuration tools
- **Improve existing tools:**
  - BitBake Commander
    - ✓ Create recipe wizard extensions
    - ✓ Add more features to make it easier for the user to create recipes
  - Hob:
    - ✓ Working on near- and longer-term plans for creating a better infrastructure to support a back-end BitBake server and front-end user interface model
    - ✓ Deliver key missing functionality, e.g. packages deselection, precise package information
  - Tracing and Profiling Tools:
    - ✓ Make tools easier to setup
    - ✓ Improve tool functionality in the long term



# Other Sources

## **Yocto Project:**

<http://www.yoctoproject.org/>

## **ADT manual:**

<http://www.yoctoproject.org/docs/current/adt-manual/adt-manual.html>

## **Yocto Project Eclipse Plug-in Video:**

<http://vimeo.com/30557368>

## **Hob Video:**

<http://www.youtube.com/embed/W3IXTdajqH4>

## **ELC 2011 ADT Video:**

“The Yocto project and its application development toolkit (ADT) -  
The answer to effective embedded application development”:

<http://free-electrons.com/blog/elc-2011-videos/>

# Legal Information

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.

Copyright © 2009, Intel Corporation. All rights are protected.

