

# Buildroot Eclipse Bundle : A powerful IDE for Embedded Linux developers



[www.flickr.com/photos/playdogi/3408511286/](http://www.flickr.com/photos/playdogi/3408511286/)

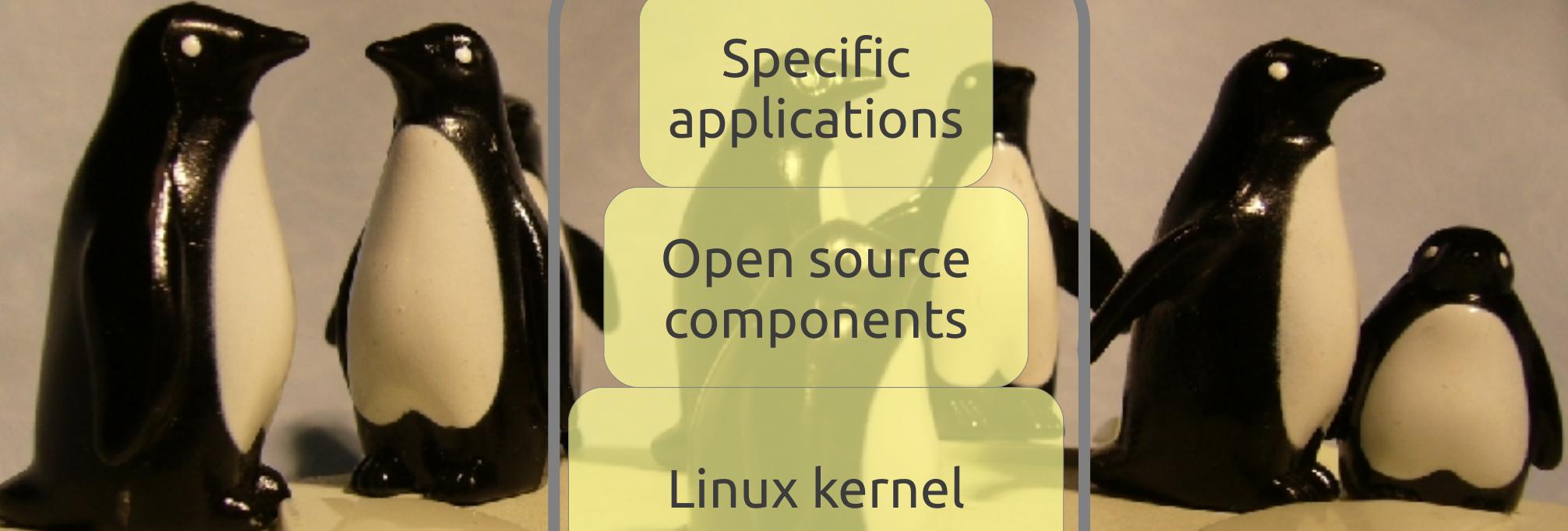
Mélanie Bats  
 ob eo  
OBSTACLES ET OPPORTUNITÉS

 **eclipseCON**  
**BOSTON 2013**

# Who Am I ?



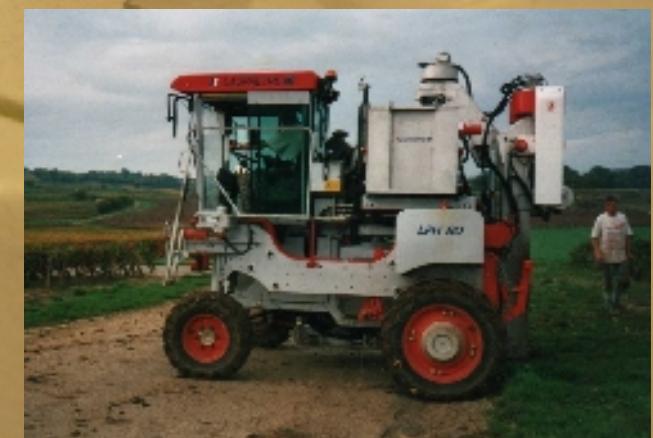
# What is embedded Linux ?



Specific  
applications

Open source  
components

Linux kernel



# Building an embedded Linux system

## Binary distributions



- Heavyweight systems
- Not all CPU architectures supported
- Not flexible to choose the configuration for each components

## Building tools



openembedded



OpenWRT      PTXdist

- Automate system reconstruction from source code
- Support many more CPU architectures
- High flexibility on components configuration
- Most used solution

# What is Buildroot ?

- Simple build system - <http://buildroot.org> :
  - automate the cross compilation process
  - generate kernel images, file system and bootloader
- Developped by :
  - an **open source** community
  - vendor neutral
  - under GPLv2
- Actively developped and used for many products
- Written in make language
- Choice of options in configuration interface

# Demo hardware platform

- Fairly typical ARM evaluation platform
- Many devices: Screen, Network, SD card, USB ...
- Used to develop prototypes
- Reference to design specific board



Atmel AT91SAM9G45 (ARMv5) processor

# Buildroot Demo



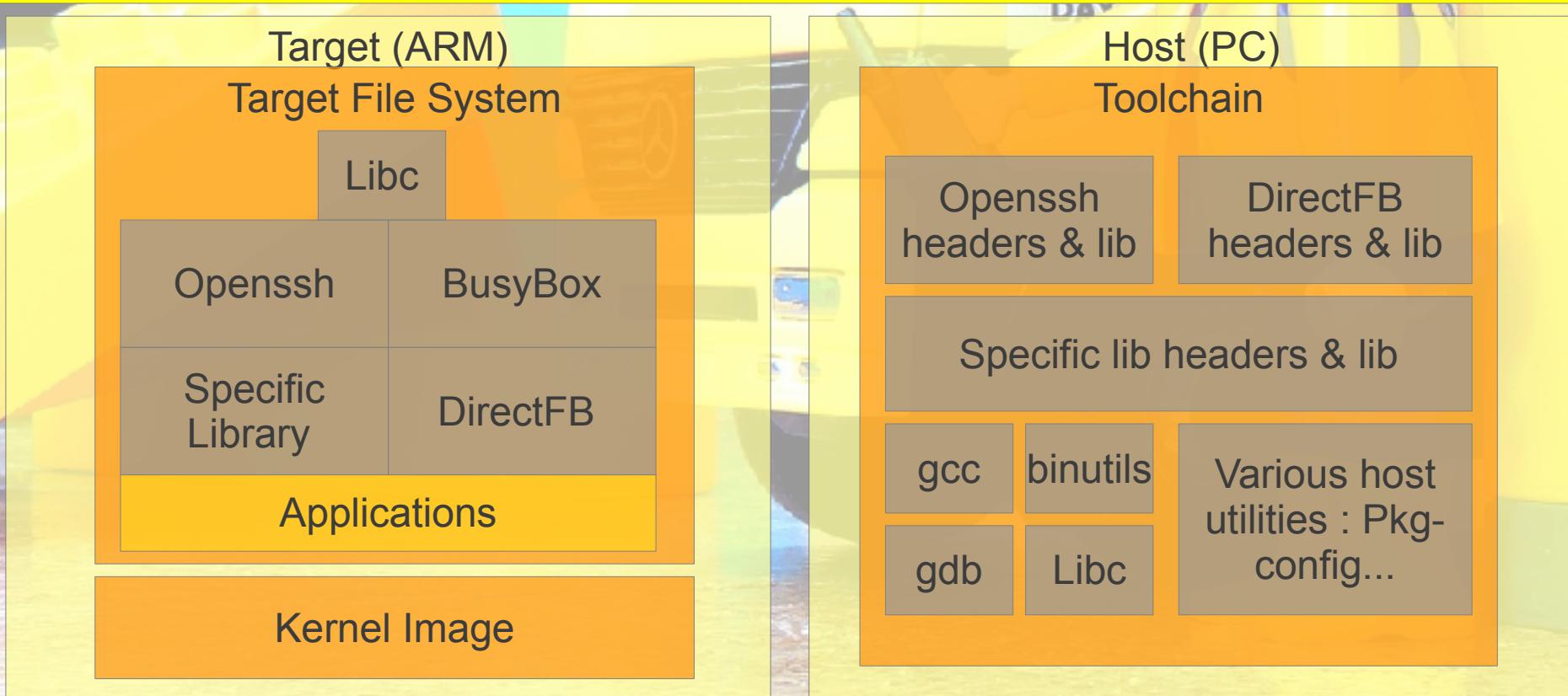
# Buildroot generates

Inputs



Buildroot

Outputs



# Why Integrating Buildroot in Eclipse ?

System developers



Application developers



# Why Integrating Buildroot in Eclipse ?

- Open source IDE for Application developers
  - Preconfigured with the cross compilation toolchain
  - Easy access to the libraries available on the target
  - Transfer / Execute remote application
  - Remote Debug



<http://www.flickr.com/photos/catcrispi/3095995888>

# Buildroot toolchain Eclipse plugin

- Integrate the toolchains to Eclipse CDT
  - Read the toolchain description file (/home/<user>/.buildroot-eclipse.toolchains)
  - Register dynamically the toolchains in CDT for :
    - Managed build projects : executable, static library, shared library
    - Autotools projects
    - Makefile projects

# Integration with CDT

- Inspired by the Eclipse Blackfin plugin and the GNU ARM Eclipse plugin
- Use CDT extension point: *org.eclipse.cdt.managedbuilder.core.buildDefinitions*
  - according to the Buildroot configuration file:
    - declare toolchains
    - register ProjectTypes

# Managed build Integration

- Add the Buildroot toolchains available for:
  - Executable project
  - Static library project
  - Shared library project
- Contribute to plugin.xml :

```
<projectType  
buildArtifactType="org.eclipse.cdt.build.core.buil  
dArtifactType.exe" ...>
```



# Buildroot Launch configurations

- Provide Remote configurations :
  - Launch configuration : execute the application on target
  - Remote Debug configuration : which point automatically on the correct cross-debugger



(cc) <http://www.flickr.com/photos/ejk/3255860779>

# Buildroot Eclipse plugin demonstration



# Pkg-config ?

- Applications need other libraries than just the standard C/C++ library
  - Graphical, network, crypto libraries ...
- Manually adding compiler and linker flags is annoying
- Pkg-config is a standard Linux tool to **query** the **compiler** and **linker flags** for a given library

```
$ pkg-config --cflags directfb  
-D_REENTRANT -I<...>/sysroot/usr/include/directfb  
  
$ pkg-config --libs directfb  
-ldirectfb -lfusion -ldirect -lpthread
```

# Managed build Integration : pkg-config plugin

- Contribute to existing pkg-config plugin :

*<http://code.google.com/p/pkg-config-support-for-eclipse-cdt/>*

- Fix bugs on PKG\_CONFIG\_PATH and pkg-config binary path
- Use the pkg-config environment variables directly on command
- Move the pkg-config configuration to project-level
- Improve the UI
- Allow a per-toolchain specified pkg-config binary

# Managed build Integration : pkg-config plugin

- Contribute to existing pkg-config plugin :

*<http://code.google.com/p/pkg-config-support-for-eclipse-cdt/>*

- Fix bugs on PKG\_CONFIG\_PATH and pkg-config binary path
- Use the pkg-config environment variables directly on command
- Move the pkg-config configuration to project-level
- Improve the UI
- Allow a per-toolchain specified pkg-config binary

# Managed build Integration : pkg-config plugin

- Contribute to existing pkg-config plugin :

*<http://code.google.com/p/pkg-config-support-for-eclipse-cdt/>*

- Fix bugs on PKG\_CONFIG\_PATH and pkg-config binary path
- Use the pkg-config environment variables directly on command
- Move the pkg-config configuration to project-level
- Improve the UI
- Allow a per-toolchain specified pkg-config binary

# Managed build Integration : pkg-config plugin

- Contribute to existing pkg-config plugin :

*<http://code.google.com/p/pkg-config-support-for-eclipse-cdt/>*

- Fix bugs on PKG\_CONFIG\_PATH and pkg-config binary path
- Use the pkg-config environment variables directly on command
- Move the pkg-config configuration to project-level
- Improve the UI
- Allow a per-toolchain specified pkg-config binary

# Buildroot pkg-config plugin demonstration



# Autotools ?

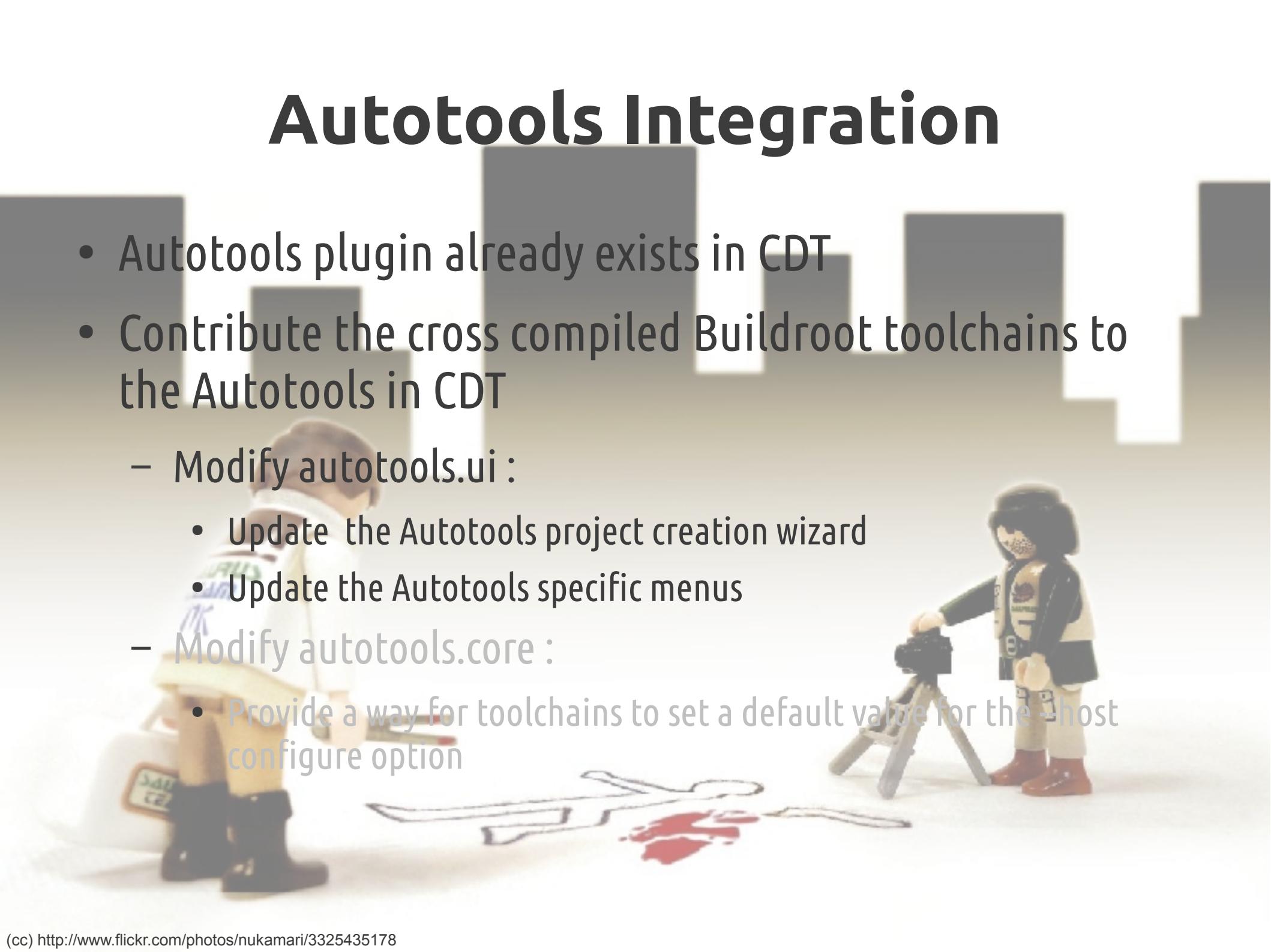
- The most **common configuration and build system** for Linux software components
- Composed of autoconf, automake and libtool
- Developers write :
  - Configure.ac
  - Makefile.am
- Final **makefiles generated by those tools**



(cc) <http://www.flickr.com/photos/rotia/8082650255>

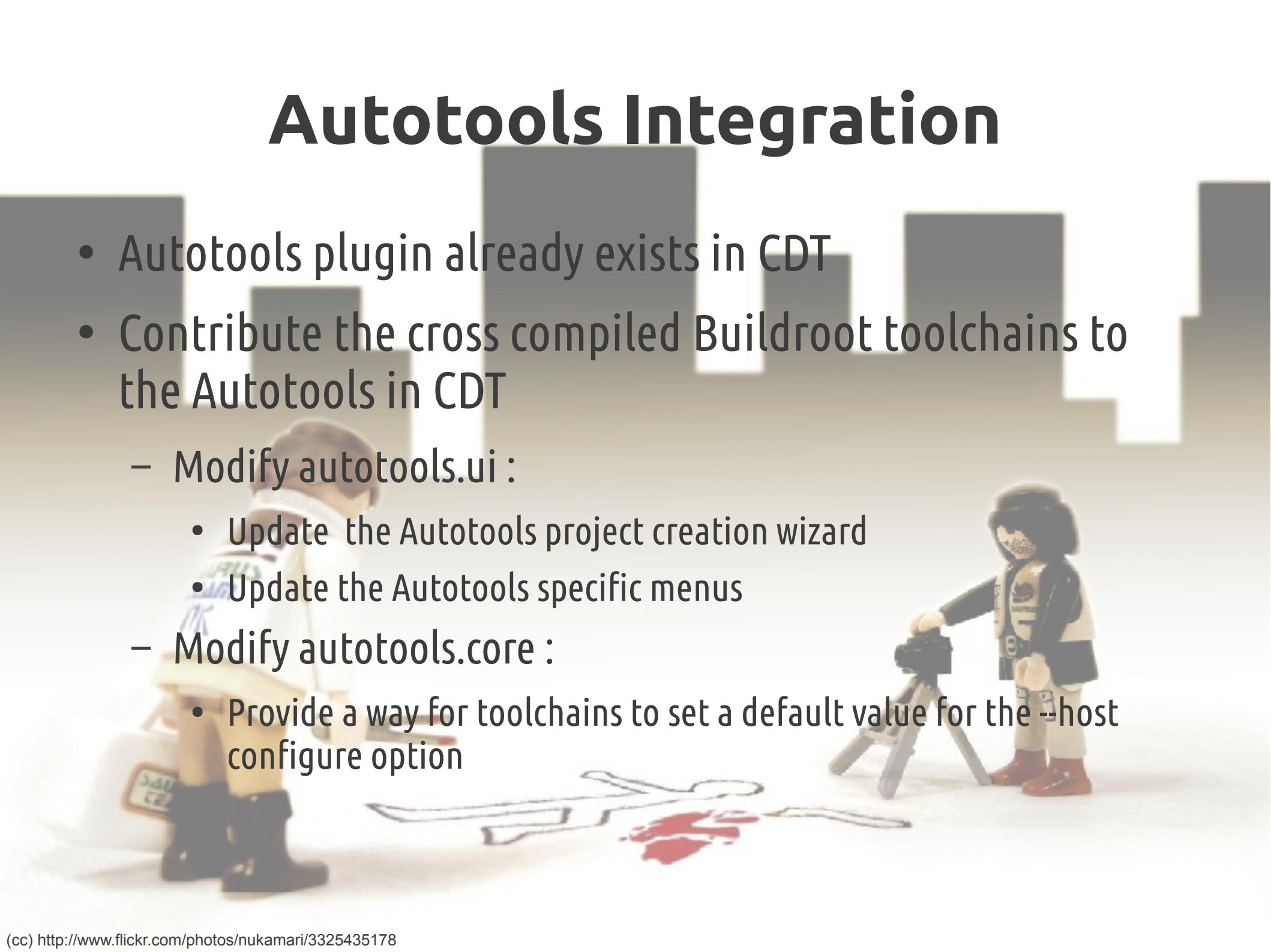
# Autotools Integration

- Autotools plugin already exists in CDT
- Contribute the cross compiled Buildroot toolchains to the Autotools in CDT
  - Modify autotools.ui :
    - Update the Autotools project creation wizard
    - Update the Autotools specific menus
  - Modify autotools.core :
    - Provide a way for toolchains to set a default value for the `--host` configure option



# Autotools Integration

- Autotools plugin already exists in CDT
- Contribute the cross compiled Buildroot toolchains to the Autotools in CDT
  - Modify autotools.ui :
    - Update the Autotools project creation wizard
    - Update the Autotools specific menus
  - Modify autotools.core :
    - Provide a way for toolchains to set a default value for the --host configure option



# Buildroot Autotools integration demonstration



# Integration with Makefile

- Add Buildroot toolchain to Makefile projects
  - Add some environment variables:
    - PATH
    - CC
    - CXX

Those are typically used in standard makefiles



(cc) <http://www.flickr.com/photos/22425840@N00/2648507349>

# Buildroot Makefile integration demonstration



# Integration with CDT : Issues

- V1.0 uses *dynamicElementProvider*
  - Method returning a hierarchy of objects implementing IManagedConfigElement
  - Contributing environment variables through configurationEnvironmentSupplier requires a DefaultManagedConfigElement
- Consequently for V2.0 build dynamically the plugin.xml
  - Use startup extension point ;(
  - Missing dynamic extension points in CDT ?
    - scanner configuration discovery profile
    - launch configuration

# Integration with CDT

The screenshot displays three code editors within the Eclipse IDE interface, illustrating the integration of the Buildroot project with the CDT (C/C++ Development Tools) environment.

- BuildrootInputType.java:** This file contains Java code for generating XML configuration files. It includes imports for `org.eclipse.core.internal.registry.RegistryFactory`, `org.eclipse.core.runtime.IExtensionRegistry`, and `org.eclipse.core.runtime.IObjectDescriptor`. The class defines methods for creating scanner configuration discovery profiles and extension points, utilizing internal classes like `ScannerInfoCollector` and `ScannerInfoConsoleParser`.
- BuildrootLaunchConfiguration.java:** This file contains Java code for creating launch configurations. It includes imports for `org.eclipse.debug.core.ILaunchConfigurationType`, `org.eclipse.debug.core.ILaunchConfiguration`, and `org.eclipse.debug.core.ILaunchConfigurationTabGroup`. The class implements methods for creating launch configuration types and tab groups, referencing internal classes like `BuildrootUtil` and `BuildrootActivator`.
- BuildrootUtils.java:** This file contains Java code for utility functions. It includes imports for `java.util.List`, `java.util.Map`, and `java.util.Properties`. The class provides static methods for registering extension points and managing contributor factories.

```
private StringBuffer createScannerConfigurationDiscoveryProfile() {
    StringBuffer buffer = new StringBuffer();
    "<?xml version='1.0' encoding='UTF-8'?>";
    buffer.append("<?eclipse version='3.4.1'?>");
    buffer.append("<plugin>");
    buffer.append("<extension>");
    buffer.append(" id=''" + getScannerConfigurationId() + "'");
    buffer.append(" name='Buildroot ManagedWakeForProjectProfileCV'");
    buffer.append(" point='org.eclipse.cdt.make.core.ScannerConfigurationDiscoveryF");
    buffer.append(" <scannerInfoCollector>");
    buffer.append("   <class='org.buildroot.cdt.toolchain.defaultScannerInfoCollector'>");
    buffer.append("     <scanner>'project'</scanner>");
    buffer.append("   </scannerInfoCollector>");
    buffer.append(" <buildOutputProvider>");
    buffer.append("   <provider>'open'</provider>");
    buffer.append("   <scannerInfoConsoleParser>");
    buffer.append("     <class='org.buildroot.cdt.toolchain.ManagedGCCScannerInt";
    buffer.append("       <scannerInfoConsoleParser>'>");
    buffer.append("     <id>'BuildrootProfile'</id>");
    buffer.append("     <scannerInfoProvider>");
    buffer.append("       <providerId>'$specsFile'</providerId>");
    buffer.append("       <run>'");
    buffer.append("         arguments='-$I -f -O0 -m $plugin.state ToolChain'>");
    "        <getSpecFileName()>'</run>'");
    buffer.append("       <class='org.eclipse.cdt.make.internal.core.scannerconfig";
    buffer.append("         command=''" + command + "'>'>");
    buffer.append("     <name>'");
    buffer.append("       <scannerInfoConsoleParser>'>");
    buffer.append("     <class='org.eclipse.cdt.make.internal.core.scannerconfig";
    buffer.append("       </scannerInfoConsoleParser>'>");
    buffer.append("     </scannerInfoProvider>'>");
    buffer.append("   </extension>");
```

```
public void createLaunchConfiguration() {
    StringBuffer buffer = new StringBuffer();
    "<?xml version='1.0' encoding='UTF-8'?>";
    buffer.append("<?eclipse version='3.4.1'?>");
    buffer.append("<plugin>");
    buffer.append("<extension>");
    buffer.append(" point='org.eclipse.debug.core.launchConfigurationTypes'>");
    buffer.append(" <launchConfigurationType>");
    buffer.append("   <delegate>'org.eclipse.debug.core.launching.RunnableLaunch";
    buffer.append("     id=''" + getLaunchConfigTypeId() + "'>'>");
    buffer.append("   <noexec>'run,debug'</noexec>");
    buffer.append("   <name>'";
    "     <BuildrootUtil>.getToolName(architecture, path, null) + '</name>'");
    buffer.append("   <public>'true'</public>'>");
    buffer.append("   <sourceLocatorId>'org.eclipse.cdt.debug.core.sourceLocator'</sourceLocatorId>");
    buffer.append("   <sourcePathComputerId>'org.eclipse.cdt.debug.core.sourcePathComputer'</sourcePathComputerId>");
    buffer.append("   <launchConfigurationType>'>");
    buffer.append("   </extension>");
```

```
public static void registerExtensionPoint(StringBuffer buffer) {
    byte[] inputString = new byte[buffer.length()];
    buffer.getBytes();
    IExtensionRegistry registry = RegistryFactory.getRegistry();
    registry.addExtensionPoint("restriction");
    Object key = IExtensionRegistry.getTemporaryToken();
    Bundle bundle = BuildrootActivator.getDefault().getBundle();
    IContributor contributor = ContributorFactoryOSGi
        .createContributor(bundle);
    if (registry.addContributing(key, contributor, false, null, null, key)) {
        BuildrootActivator.getBuilder().warning(
            "Contribution is not registered : " + buffer.toString(),
            null);
    }
}
```

# Buildroot Eclipse Update-site

- Update-site integrating embedded Linux tools :
  - Buildroot Toolchain Eclipse Plugin
  - Pkg-config
  - CDT
  - Linux tools
  - Autotools ...
- Make a **Ready-to-Use** development **platform** for embedded Linux
- Use Tycho to create update-site

# Availability

- Buildroot 2013.02 :  
<http://buildroot.org>
- Buildroot Eclipse plugin sources (EPL licence):  
<https://github.com/mbats/eclipse-buildroot-toolchain-plugin>
- Buildroot Eclipse bundle sources :  
<https://github.com/mbats/eclipse-buildroot-bundle>
- Pkg-config Eclipse plugin :  
<http://code.google.com/p/pkg-config-support-for-eclipse-cdt/>
- Buildroot Eclipse bundle update-site (based on Eclipse 4.2)



# Getting started

- On Github :  
<https://github.com/mbats/eclipse-buildroot-bundle/wiki/>
  - Installation details
  - Tutorials
  - Bug tracker
- Give us feedback !



# Future work

- Contribute Autotools updates to CDT
- Integrate :
  - RX-TX
  - LTTNG
  - CMake
  - Other relevant Eclipse plugins for embedded Linux development
- Bug fixes
- What do you need ?

# Questions ?



Contact :  
[melanie.bats@obeo.fr](mailto:melanie.bats@obeo.fr)