

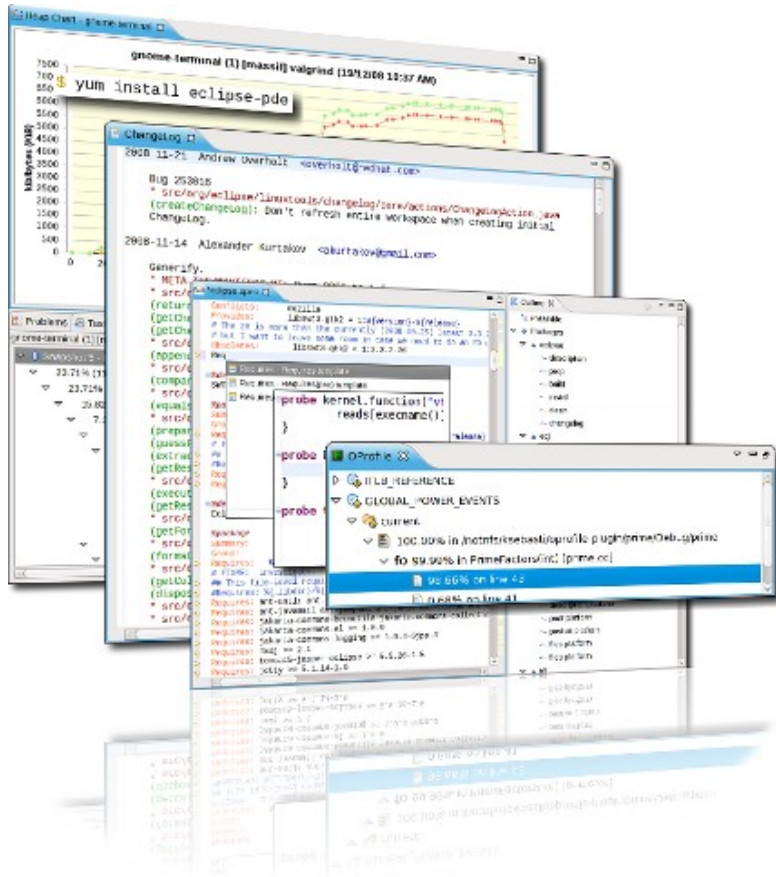


Linux Tools Project

EclipseCon 2009

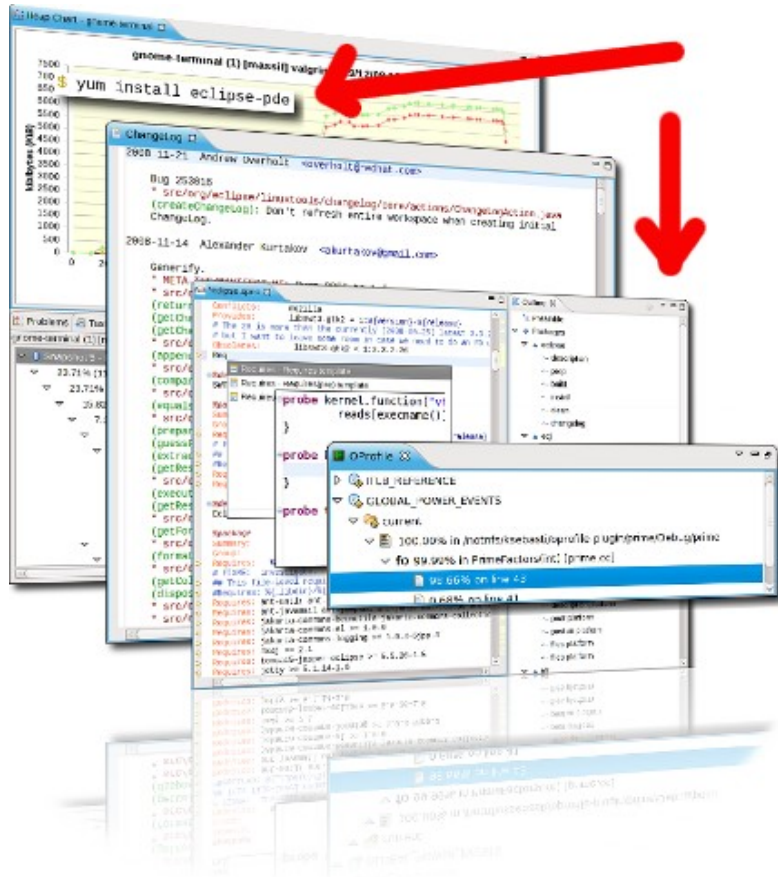
Andrew Overholt
Red Hat

Agenda



- Tools for Linux packagers
- Change of *main* focus
- Tools for C/C++ developers built on CDT
- Future plans and how to get involved

Agenda



- Tools for Linux packagers
- Change of *main* focus
- Tools for C/C++ developers built on CDT
- Future plans and how to get involved



RPM .spec editor

The screenshot displays the RPM .spec editor interface. The main window shows the content of the `eclipse-oprofile.spec` file. The file content is as follows:

```
Name:          eclipse-oprofile
Version:       0.1.0
Release:       3%{?dist}
Summary:       Eclipse plugin for OProfile integration

Group:         Development/Tools
License:       EPL
URL:           http://www.eclipse.org/linuxtools/projectPages/oprofile/
## sh %{name}-fetch-src.sh
Source0:       %{name}-fetched-src-%{src_repo_tag}.tar.bz2
Source1:       %{name}-fetch-src.sh
Patch0:        %{name}-includefixes.patch
BuildRoot:     %{_tmppath}/%{name}-%{version}-%{release}-root-%(%{__id_u} -n

ExcludeArch:  ppc ppc64

BuildRequires: eclipse-pde >= 1:3.4.0
BuildRequires: eclipse-cdt >= 5.0.1
BuildRequires: eclipse-linuxprofilingframework >= 0.1.0
BuildRequires: oprofile >= 0.9.3
BuildRequires: oprofile-devel >= 0.9.3
BuildRequires: binutils-devel >= 2.18.50.0.6
Requires:     eclipse-platform >= 3.4.0
Requires:     eclipse-cdt >= 5.0.1
Requires:     eclipse-linuxprofilingframework >= 0.1.0
Requires:     oprofile >= 0.9.3
Requires:     usermode >= 1.98

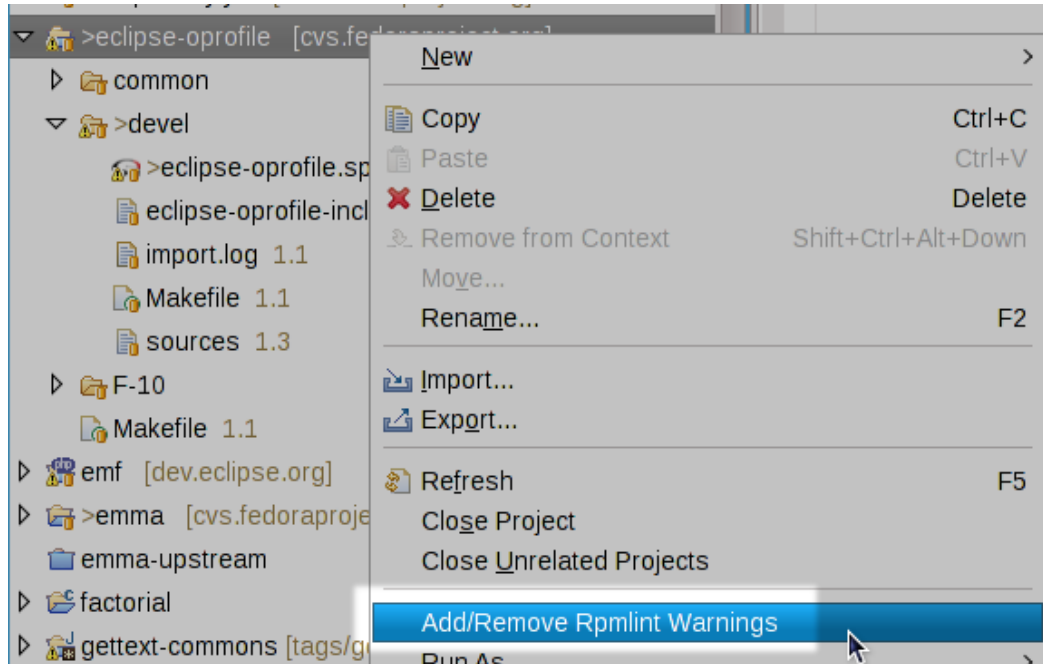
%description
Eclipse plugins to integrate OProfile's profiling capabilities with the CDT.

%prep
%setup -q -c
#remove binaries
rm -f org.eclipse.linuxtools.oprofile.core.linux.*os/linux/*/opxml
```

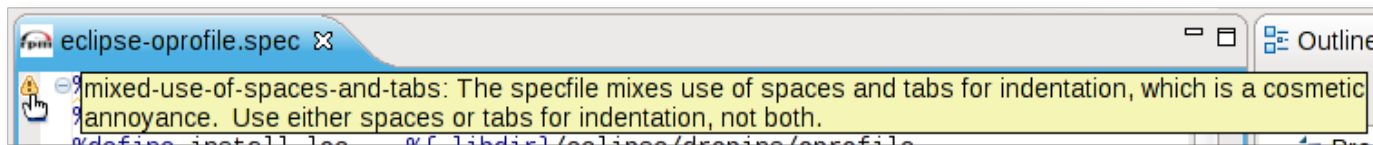
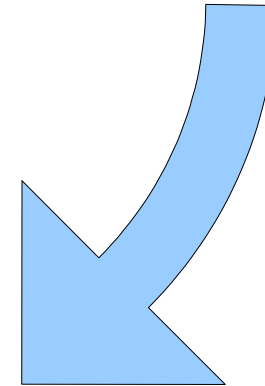
The right-hand side of the editor shows an Outline view with the following structure:

- Preamble
- Packages
 - eclipse-oprofile
 - description
 - files
 - prep
 - build
 - install
 - clean
 - changelog

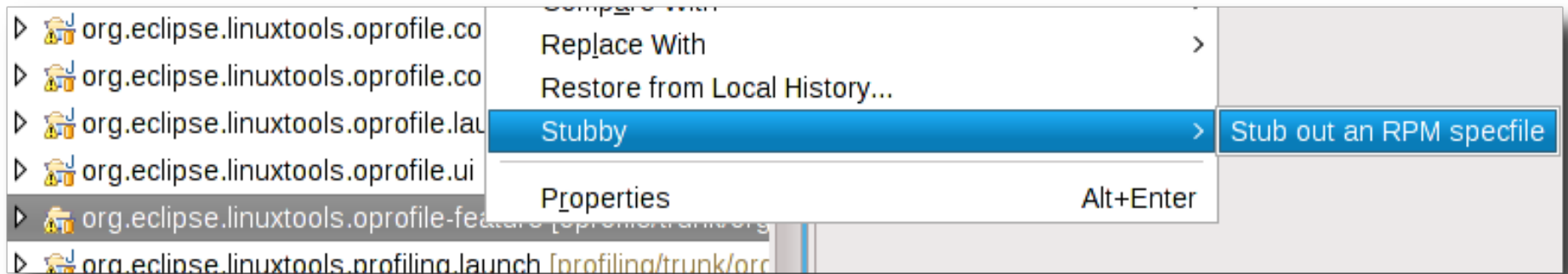
rpmlint warnings



Add/Remove rpmlint Warnings



“RPM Stubby”



“Stub” out an RPM specfile



RPM .spec editor

The screenshot shows a window titled "eclipse-oprofile.spec" with a text editor containing the following content:

```
Name:          eclipse-oprofile
Version:       0.1.0
Release:       3%{?dist}
Summary:       Eclipse plugin for OProfile integration

Group:         Development/Tools
License:       EPL
URL:           http://www.eclipse.org/linuxtools/projectPages/oprofile/
## sh %{name}-fetch-src.sh
Source0:       %{name}-fetched-src-%{src_repo_tag}.tar.bz2
Source1:       %{name}-fetch-src.sh
Patch0:        %{name}-includefixes.patch
BuildRoot:     %{_tmppath}/%{name}-%{version}-%{release}-root-%(%{__id_u} -n

ExcludeArch:  ppc ppc64

BuildRequires: eclipse-pde >= 1:3.4.0
BuildRequires: eclipse-cdt >= 5.0.1
BuildRequires: eclipse-linuxprofilingframework >= 0.1.0
BuildRequires: oprofile >= 0.9.3
BuildRequires: oprofile-devel >= 0.9.3
BuildRequires: binutils-devel >= 2.18.50.0.6
Requires:     eclipse-platform >= 3.4.0
Requires:     eclipse-cdt >= 5.0.1
Requires:     eclipse-linuxprofilingframework >= 0.1.0
Requires:     oprofile >= 0.9.3
Requires:     usermode >= 1.98

%description
Eclipse plugins to integrate OProfile's profiling capabilities with the CDT.

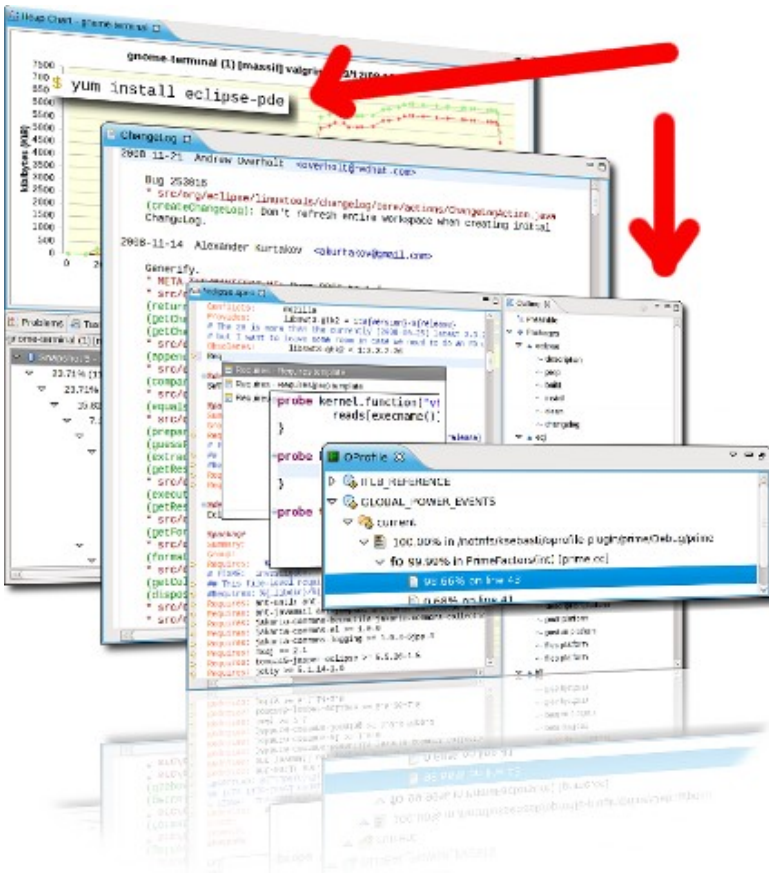
%prep
%setup -q -c
#remove binaries
rm -f org.eclipse.linuxtools.oprofile.core.linux.*os/linux/*/opxml
```

On the right side, there is an "Outline" panel showing a tree view of the file structure:

- Preamble
- Packages
 - eclipse-oprofile
 - description
 - files
 - prep
 - build
 - install
 - clean
 - changelog

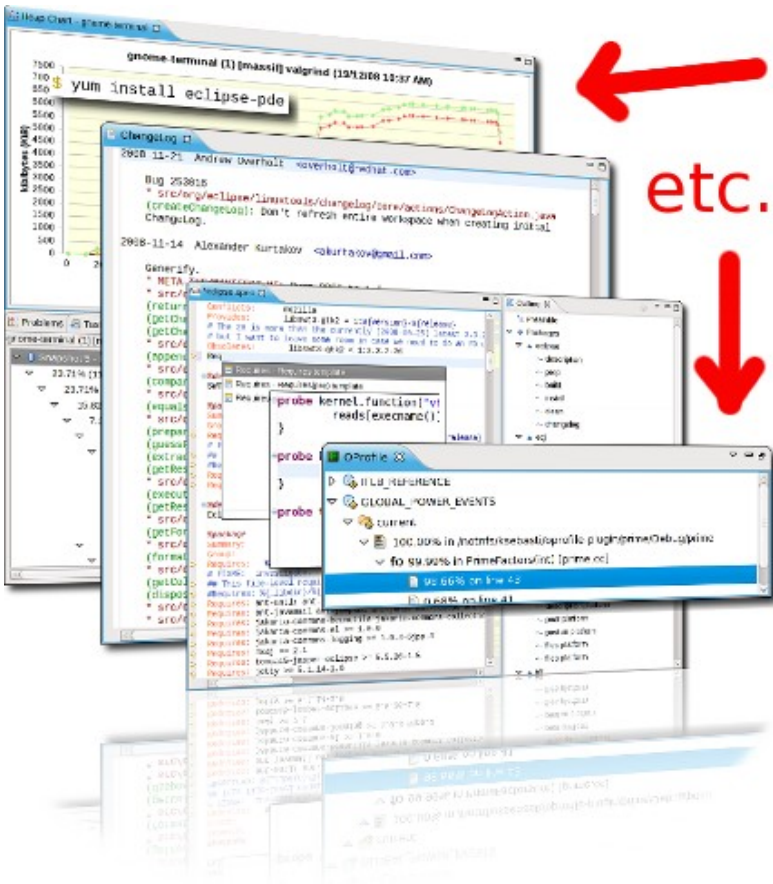
Agenda

- Tools for Linux packagers
- Change of *main* focus
- Tools for C/C++ developers built on CDT
- Future plans and how to get involved



Agenda

- Tools for Linux packagers
- Change of *main* focus
- Tools for C/C++ developers built on CDT
- Future plans and how to get involved





Focus

“Linux Distros” -> “Linux Tools”

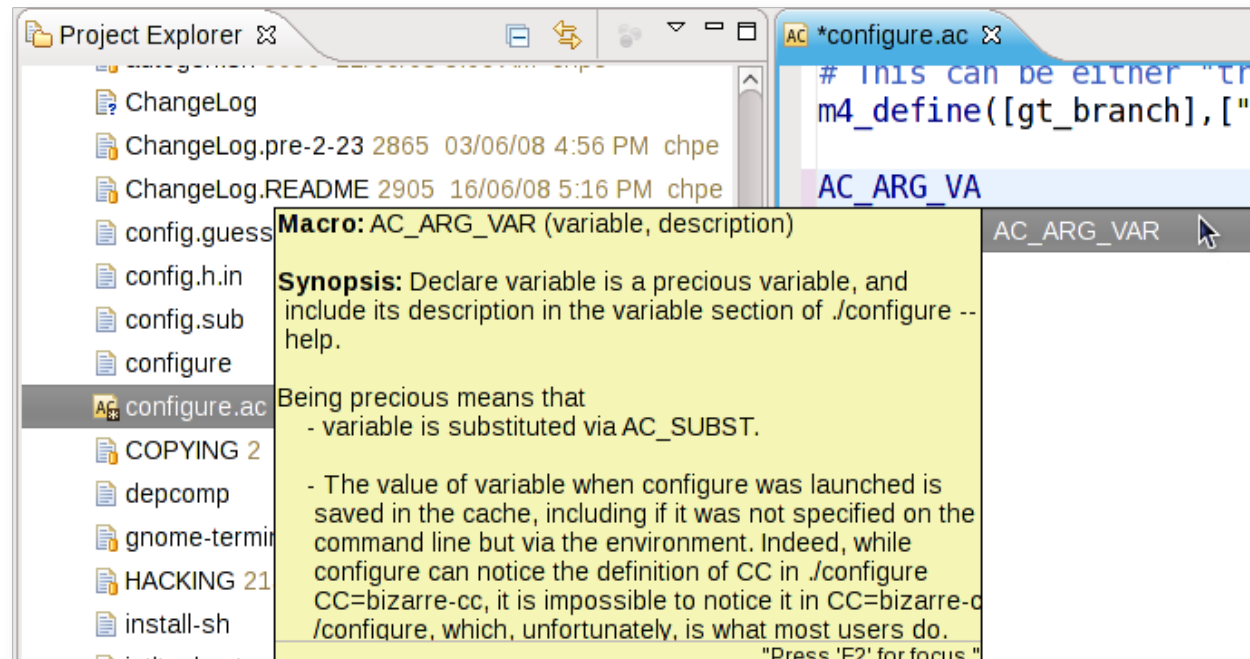
Haven't forgotten our roots



C/C++ Tools

- Developer-focused
- Sane defaults
- Integrate with CDT functionality

GNU Autotools



The screenshot shows an IDE window with a Project Explorer on the left and a code editor on the right. The Project Explorer lists files including ChangeLog, ChangeLog.pre-2-23, ChangeLog.README, config.guess, config.h.in, config.sub, configure, **configure.ac**, COPYING 2, depcomp, gnome-termin, HACKING 21, and install-sh. The code editor shows the following code:

```
# This can be either "true" or "false"
m4_define([gt_branch], ["b"])

AC_ARG_VAR
```

A tooltip is displayed over the `AC_ARG_VAR` macro, providing the following information:

Macro: AC_ARG_VAR (variable, description)

Synopsis: Declare variable is a precious variable, and include its description in the variable section of `./configure --help`.

Being precious means that

- variable is substituted via `AC_SUBST`.
- The value of variable when `configure` was launched is saved in the cache, including if it was not specified on the command line but via the environment. Indeed, while `configure` can notice the definition of `CC` in `./configure CC=bizarre-cc`, it is impossible to notice it in `CC=bizarre-cc ./configure`, which, unfortunately, is what most users do.

"Press 'E?' for focus."

Libhover

```
void *blah = mall;  
  
if (display_name  
    display = gdk_d  
else  
    {  
        GSList *displ  
        const char *p  
  
        period = strr  
        if (period)  
            {  
                gulong n;
```

- mallinfo (void) struct mallinfo
- mallinfo (void) struct mallinfo
- malloc (size_t nbytes) void *
- malloc (size_t size) void *
- malloc_stats (void) void
- malloc_usable_size (void *aptr) size_t
- mallopt (int param, int value) int
- mallopt (int parameter, value) int

Press 'Ctrl+Space' to show Template Proposals

glib.h
terminal-intl.h

This function returns a pointer to a newly allocated block size bytes long, or a null pointer if the block could not be allocated.

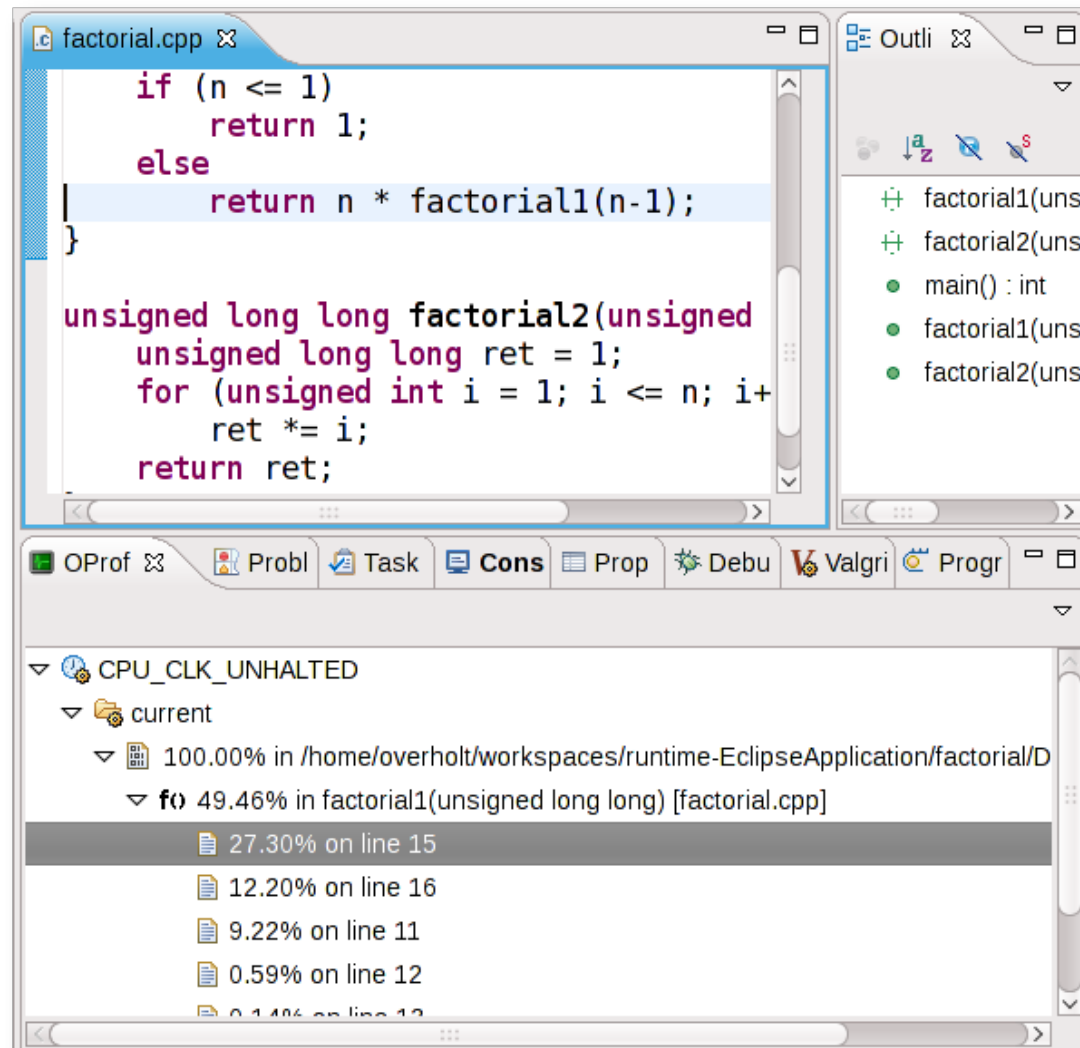
SystemTap

```
*test.stp ✕
eprobe kernel.function("vfs_read").return {
    reads[execname()] += $return
}

eprobe kernel.function("vfs_write").return {
    writes[execname()] += $return
}

eprobe timer.s(1) {
    foreach (p in reads)
        total_io[p] += reads[p]
    foreach (p in writes)
        total_io[p] += writes[p]
    foreach (p in total_io- limit 10)
        printf("%15s r: %8d KiB w: %8d KiB\n",
            p, reads[p]/1024,
            writes[p]/1024)
        printf("\n")
    # Note we don't zero out reads, writes and total_io,
```

OProfile



The screenshot displays the Eclipse IDE interface. The top editor shows the source code for `factorial.cpp`. The code includes a recursive `factorial1` function and an iterative `factorial2` function. The `factorial1` function is highlighted in blue.

```
if (n <= 1)
    return 1;
else
    return n * factorial1(n-1);
}

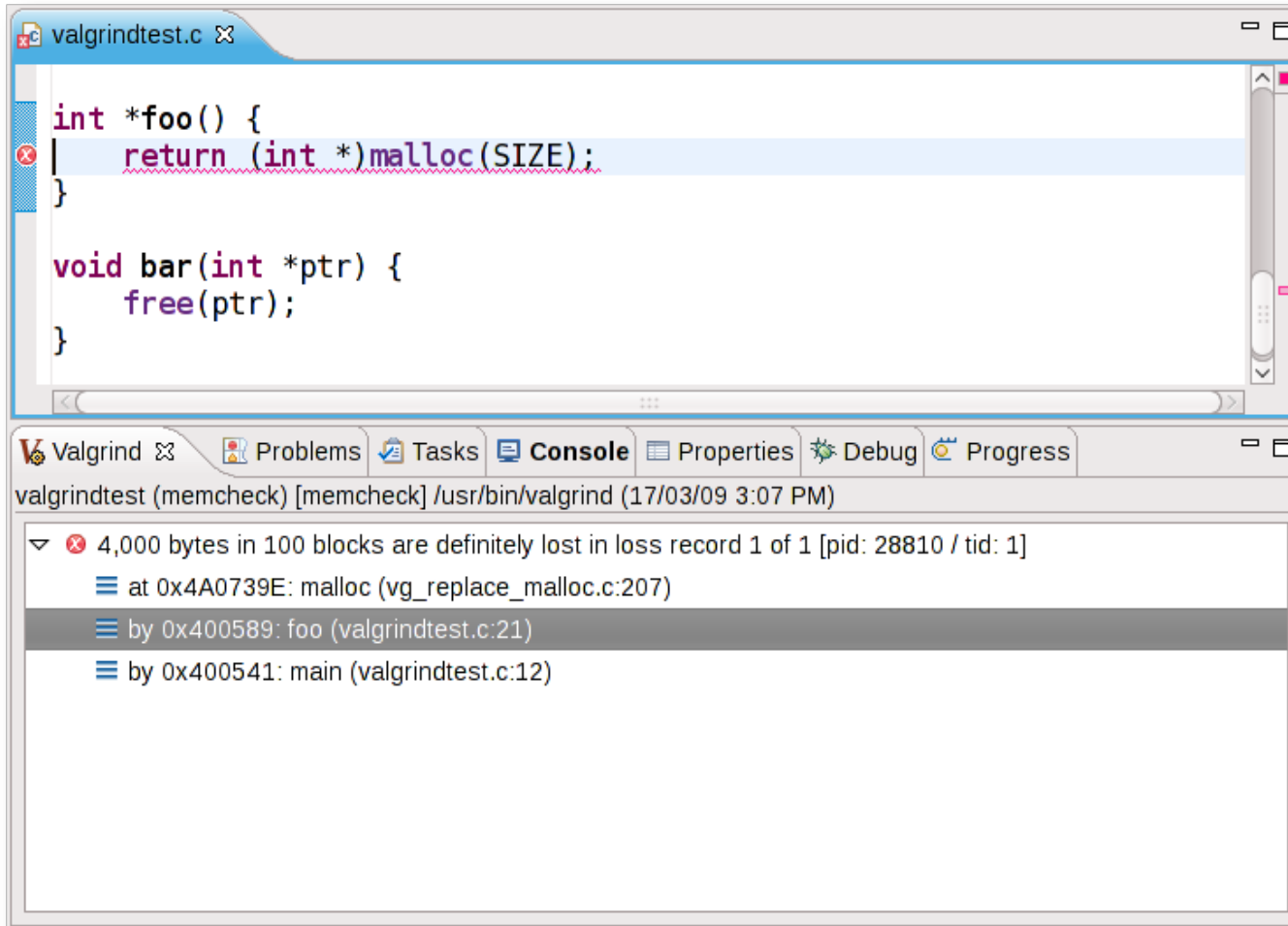
unsigned long long factorial2(unsigned
unsigned long long ret = 1;
for (unsigned int i = 1; i <= n; i+
    ret *= i;
return ret;
```

The right-hand side of the IDE shows the Outline view, listing the functions: `factorial1(uns`, `factorial2(uns`, `main() : int`, `factorial1(uns`, and `factorial2(uns`.

The bottom panel shows the OProfile performance data. The CPU is in the `CPU_CLK_UNHALTED` state. The current process is `/home/overholt/workspaces/runtime-EclipseApplication/factorial/D`. The function `factorial1(unsigned long long) [factorial.cpp]` is the most time-consuming, accounting for 49.46% of the total time. The breakdown of time spent in `factorial1` is as follows:

Line	Percentage
15	27.30%
16	12.20%
11	9.22%
12	0.59%
13	0.14%

Valgrind



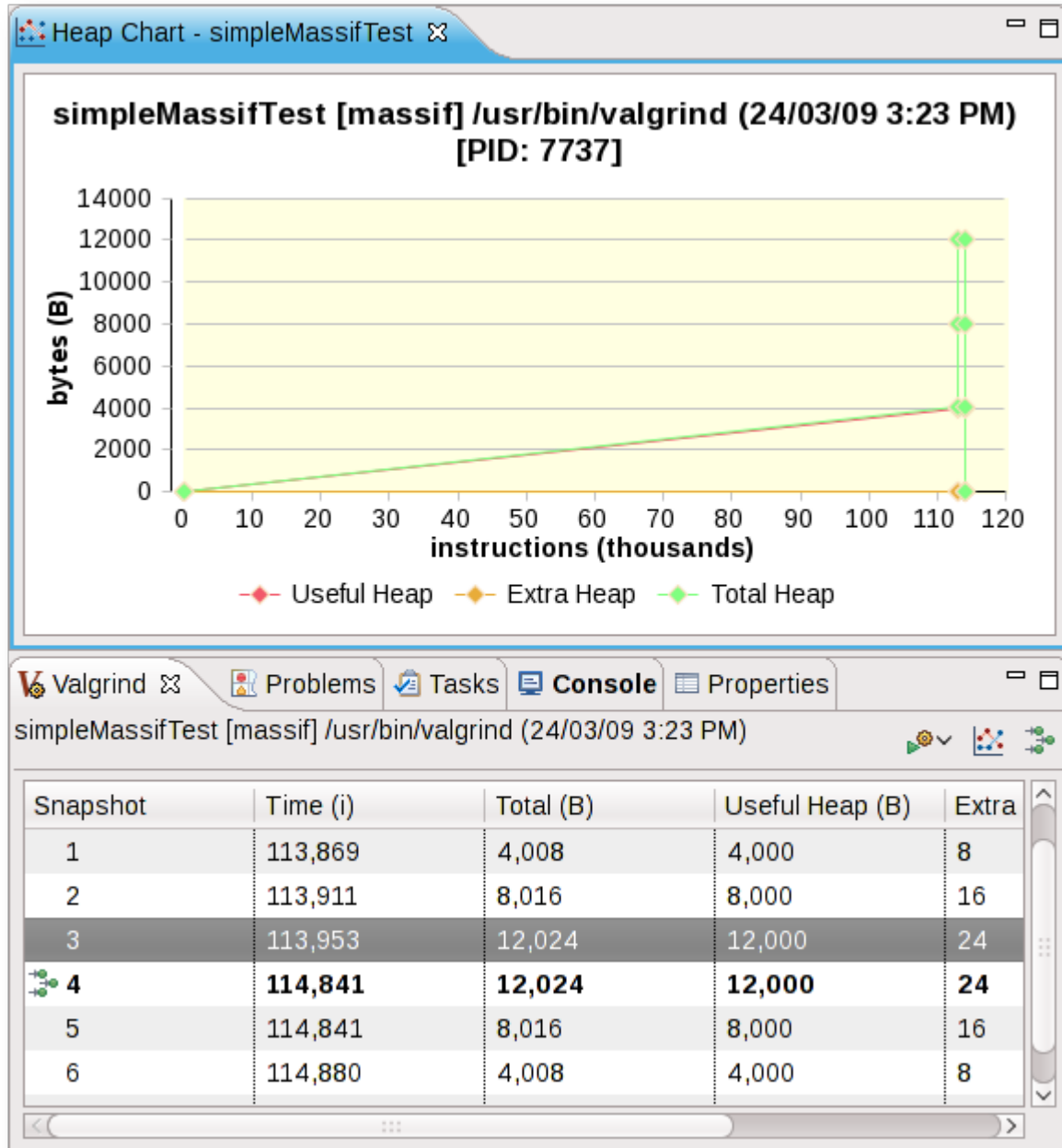
The screenshot shows a code editor window titled 'valgrindtest.c' with the following C code:

```
int *foo() {  
    return (int *)malloc(SIZE);  
}  
  
void bar(int *ptr) {  
    free(ptr);  
}
```

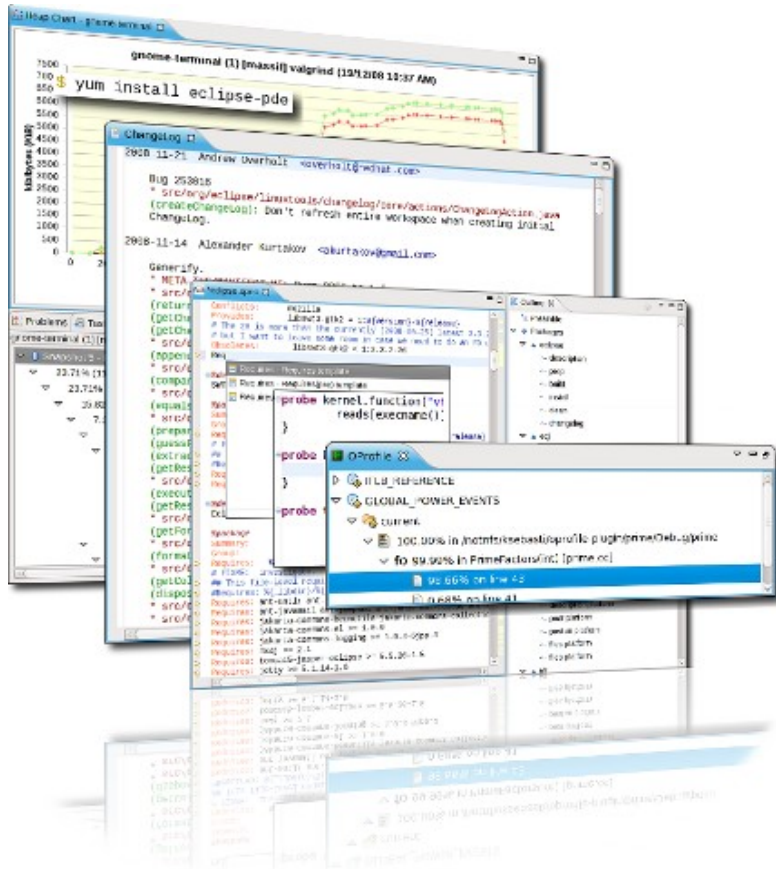
Below the code editor is a Valgrind console window. The title bar includes 'Valgrind', 'Problems', 'Tasks', 'Console', 'Properties', 'Debug', and 'Progress'. The console output shows a memory leak error:

```
valgrindtest (memcheck) [memcheck] /usr/bin/valgrind (17/03/09 3:07 PM)  
4,000 bytes in 100 blocks are definitely lost in loss record 1 of 1 [pid: 28810 / tid: 1]  
    at 0x4A0739E: malloc (vg_replace_malloc.c:207)  
    by 0x400589: foo (valgrindtest.c:21)  
    by 0x400541: main (valgrindtest.c:12)
```

Valgrind



Agenda



- Tools for Linux packagers
- Change of *main* focus
- Tools for C/C++ developers built on CDT
- Demonstrations
- **Future plans and how to get involved**

Plans

- SystemTapGui contribution
- Tracing tools (LTTng, SystemTap)
- Unit testing and coverage (CPPUnit, gcov)
- GCC static analysis
- Virtual machine integration



Join us

- We welcome contributors of all forms
 - Plug-in testers
 - Plug-in developers
 - Web designers
 - Documentation authors
 - Graphic designers
 - Commercial adopters