Power management and the Linux desktop

Matthew Garrett

March 15, 2005

Abstract

Mobile computing is a part of everyday life. Unfortunately, so is running out of battery. When every single operation reduces the amount of time the system can run for, it's clear that applications need to be aware of whether they're running on AC or battery, and cope accordingly. This BOF aims to work out what needs to be done in order to make this information available to applications, and how the applications should respond. Covered material will include what information modern hardware provides to userspace, the right layering model for ensuring that this is effectively passed on and ways that applications can make life easier for themselves and the user when complete system failure may only be minutes away.

Introduction

Modern hardware has introduced increasingly sophisticated power management functionality, and there has been an increasing tendancy for this to be managed by the running operating system. As a result, it is now possible for the operating system to be notified about a wide range of power management related issues. Currently, Gnome does very little with this information.

Hardware implementations

- APM

  APM was the first standardised power management system on PC hardware. It is mostly implemented at the BIOS level. Certain hardware events (such as lid switches or sleep buttons) generate a signal that is sent to the operating system. The OS has the opportunity to decline the event [1], and if it does not then the hardware puts itself to sleep. The only other information exposed by apm is the battery charge and whether the machine is on ac or battery power.

  Since APM was so widespread throughout the x86 world, many userspace applications have been written to use it. As a result, many other hardware power management implementations are exposed as APM under Linux.

- PMU

  Portable Apple hardware contains a power management unit (PMU) that is supported by Linux. This provides more information to the OS than APM, including whether the lid is opened or closed, direct notification of hotkey events, control of the backlight and a greater amount of battery information.

- ACPI

  The Advanced Control and Power Interface deals with a wide range of issues, only some of which are relevant here. ACPI machines pass the job of managing system power management to the OS, exposing the functionality to do so. ACPI will generally provide support for hotkey events, a great deal of battery information, control over the processor speed, temperature statistics and fan control.

- Cpufreq

  cpufreq is the name of the Linux kernel code responsible for management of voltage and frequency scaling on a wide range of hardware. In contrast to simple frequency scaling (which reduces power consumption linearly, and so has

[1] However, Linux does not support declining of hardware-generated suspend events, only user-generated ones

no significant impact on productivity time before running out of battery), voltage scaling allows significant power savings as the speed is dropped.

Existing software

- apmd

  apmd listens for apm events such as ac plug/unplug and suspend notifications. In response, it can run commands. If a user presses a suspend button, apmd is not able to prevent the suspend from occuring.

- pmud

  pmud fulfils a similar role to apmd, but has the authority to refuse to respond to certain events.

- acpid

  acpid lists for acpi events and then executes configurable scripts.

- battfink

  battfink was originally intended as a general power management control applet for Gnome – sadly, development has stalled.

- gnome-battery-applet

  gnome-battery-applet displays battery information and can be configured to run commands on various events. However, as it is running as a user, it is difficult for it to trigger many hardware-affecting actions.

- HAL

  HAL exposes much of the functionality of power management interfaces in a generic way.

Usage cases

In general, the user wants their machine to run for as long as possible. How can we achieve this?

- CPU throttling

  This is a semi-solved problem. The CPU is throttled down on battery, and may be sped up when performance is required. However, there is currently no way for applications to give advance warning of performance requirements. Certain applications may also be upset about speed changes, and may wish to request that the speed be kept constant.

- Algorithmic choice

  Applications may wish to drop output quality in return for consuming less CPU time when running on battery.

- Suspend on idle time

  It is currently difficult to produce a generic interface to trigger a suspend on user idleness, as the privilege requirements vary between low-level implementations.

- Delay suspend

  The user may wish disable suspend temporarily – alternatively, it may be undesirable to suspend if there is high processor or network activity even if there is no user input.

- Delay non-urgent activities

  Applications may wish to avoid unnecessary disk IO on battery. Database indexing is something that is unlikely to be urgent while away from an AC supply.

- Respond to suspend requests

  Applications may wish to be able to block suspend requests. Alternatively, they may wish to take the opportunity to save important state to disk.

Project Utopia

The Project Utopia stack provides a mechanism for achieving many of these goals. HAL can already generalise away the different power management implementations. dbus allows us to notify applications of changes in power state or user requests, allowing them to behave as necessary. Management of this functionality will be performed by PowerManager, a session-lifetime daemon. PowerManager will allow configuration of system-wide power state, such as the events that should trigger suspend, the length of time that the screen should remain switched on and (depending on hardware provision) screen brightness.

Conclusion

Much of the required infrastructure for providing a high-quality laptop experience already exists. Now we simply[2] face the problem of putting it all together and ensuring that we provide that experience.

[2] ha ha ha