The 10 Top Mistakes Embedded Linux Users Make

Joel Isaacson

Ascender Technologies Ltd.

This Talk

- In this talk I will try to explain what are the top 10 mistakes made by Linux users as I see it.
- I'm aware that one person's mistake is another person's best practice. Thus this lecture is necessarily subjective.
- In order to remain within the time limits of this lecture some of the “mistakes” will be lightly treated.
- Apologies to David Letterman.

The Little Engine That Could

- I will use an embedded Linux device, the WRT54GS, a wireless router as an illustration of an embedded Linux device.
- An interesting article about this device can be found in:
  - http://www.pbs.org/cringely/pulpit/pulpit20040527.html

    “The Little Engine That Could”

    How Linux is Inadvertently Poised to Remake the Telephone and Internet Markets

    By Robert X. Cringely

Support Issues

10 – Pick a vendor.

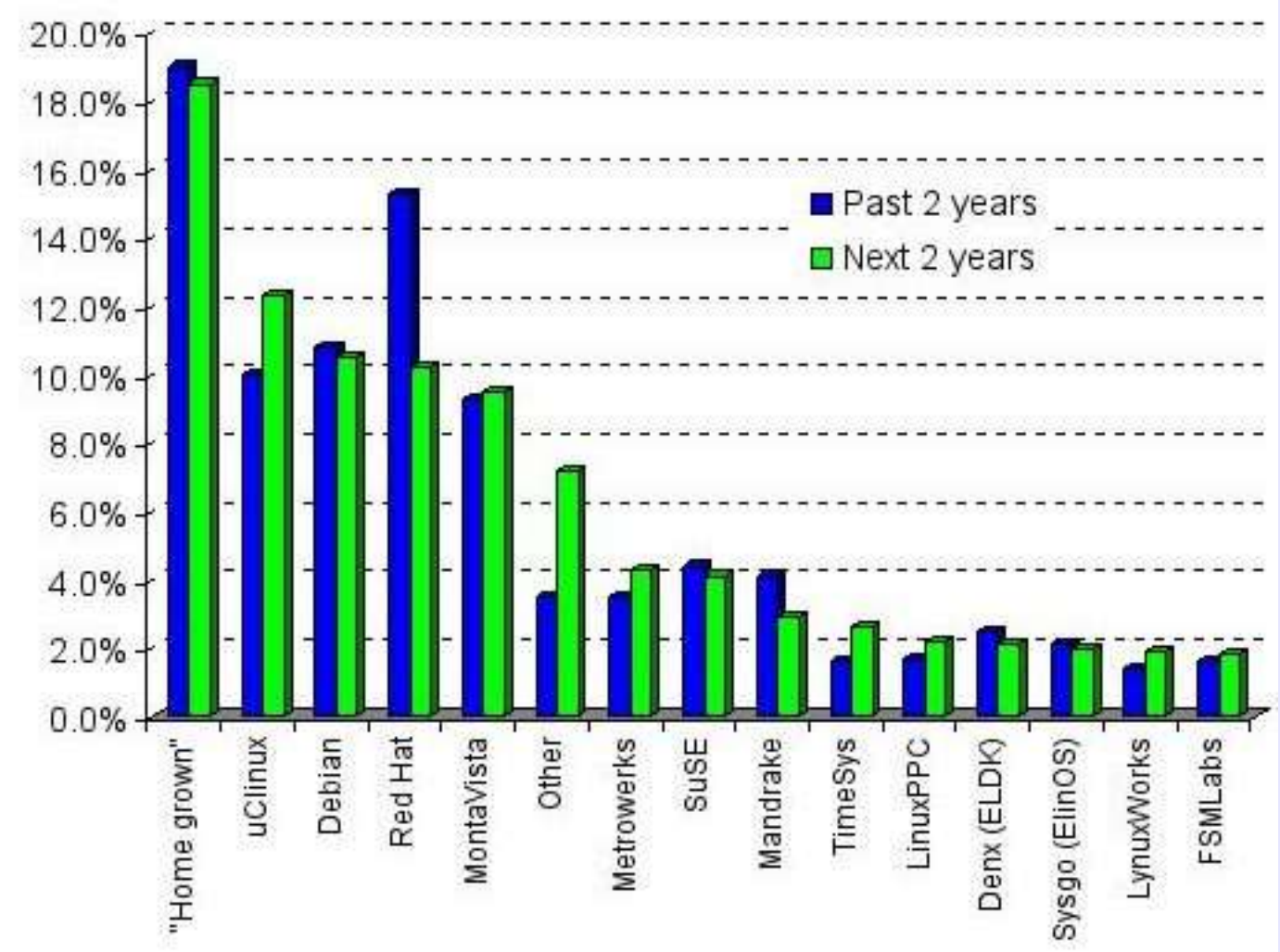
9 – Then pick a platform.

8 – We are not in Kansas anymore.

10 - Pick a Vendor

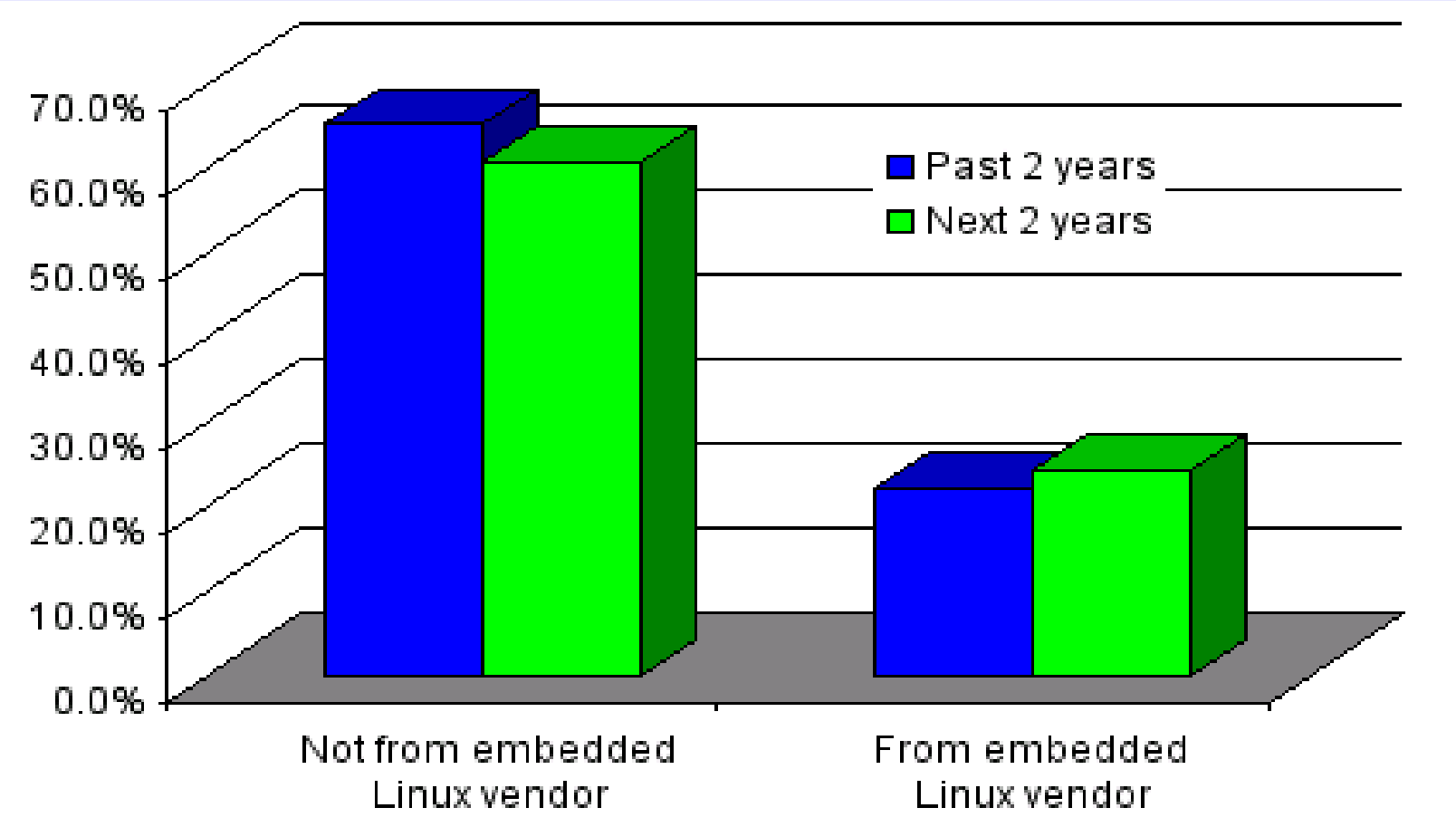
- In my experience picking a large foreign company for support is not the best way to go for various reasons.
- More about this later.

Which Linux?



From: “Snapshot of the Embedded Linux market -- March, 2004”
http://linuxdevices.com/articles/AT8693703925.html

Which Vendor?

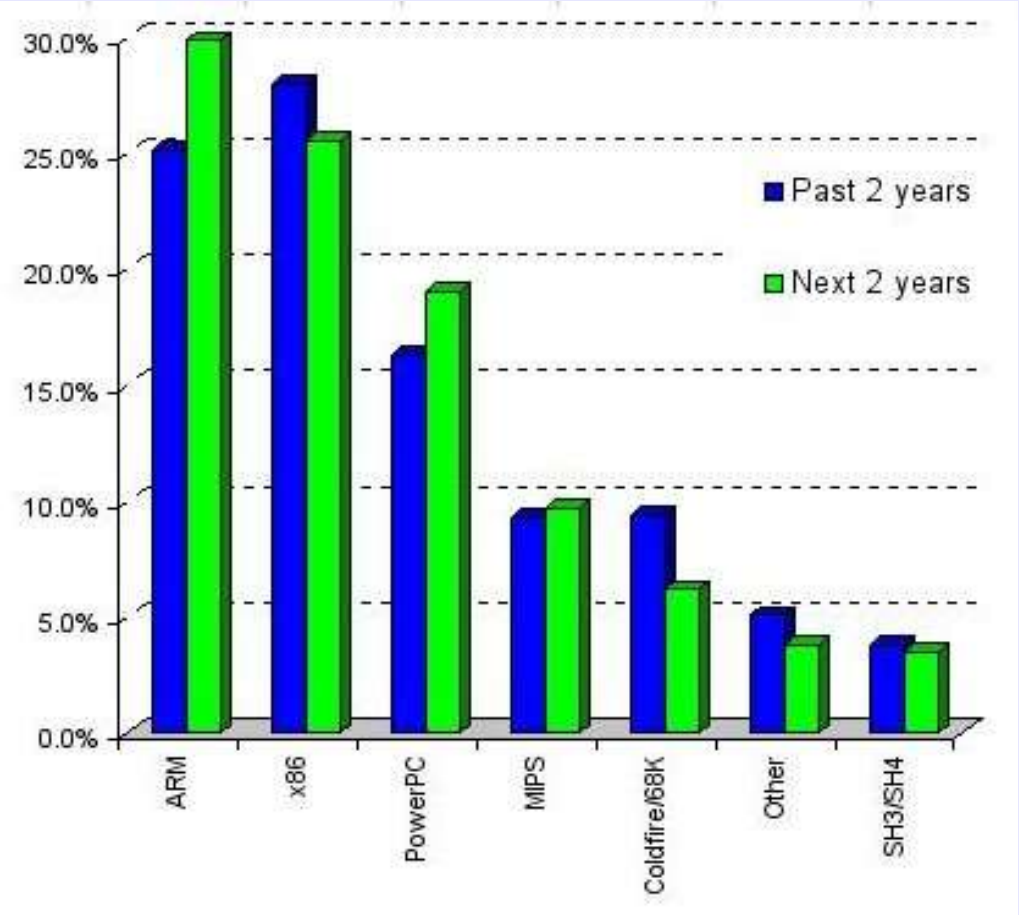


From: “Snapshot of the Embedded Linux market -- March, 2004”
http://linuxdevices.com/articles/AT8693703925.html

9 – Then Pick a Platform

- Most people immediately turn to Intel for a platform.
- If you are running high performance commodity systems this makes sense.
- For smaller embedded systems the Intel X86 architecture isn' t necessarily the best choice.

Which Processor?



From: “Snapshot of the Embedded Linux market -- March, 2004”
http://linuxdevices.com/articles/AT8693703925.html

8 – We are not in Kansas anymore.

- Linux is a disruptive technology. A once in a generation paradigm change.
- If you don't change your methods of dealing with software support, you will not benefit.
- Let's examine the issue of support in Open Source Systems.

Commercial vs. Open Source Knowledge Base

Commercial Software

Open Source Software

Who Do You Turn To?

- There are three viable approaches in dealing with support issues in Linux.
  - Get support from a large foreign software company.
  - Get support from a smaller local software company
  - Support yourself.

Support: Large Foreign Company

- There are a number of fairly large companies that support embedded Linux:
  - IBM
  - Montavista
  - RedHat
- You have to be careful of “vendor lock-in”
- Why go to Linux and then sell your soul to the devil?

Support: Small Local Company

- There are a number of local companies that can provide support for embedded Linux.
- The nice thing about this approach is that the local companies are not at a disadvantage since there is no proprietary or hidden software in the embedded Linux solution.
- Just look around you, there is plenty of talent in this room.
- No “vendor lockin”.

Support Yourself:

- Since everything is open you can provide your own support.
- This is definitely the most effective, but it needs the largest investment of time and talent.
- There is a lot of help available on the Internet and recently published books.

7 – I want it to run real fast.

Well boy you need **real time**.

Real Time Systems

- A large amount of confusion exists about the uses of commercial RTOS's
- This confusion is largely propagated by companies that sell RTOS's.
- The use of RTOS's in embedded systems is mostly a historical anomaly.

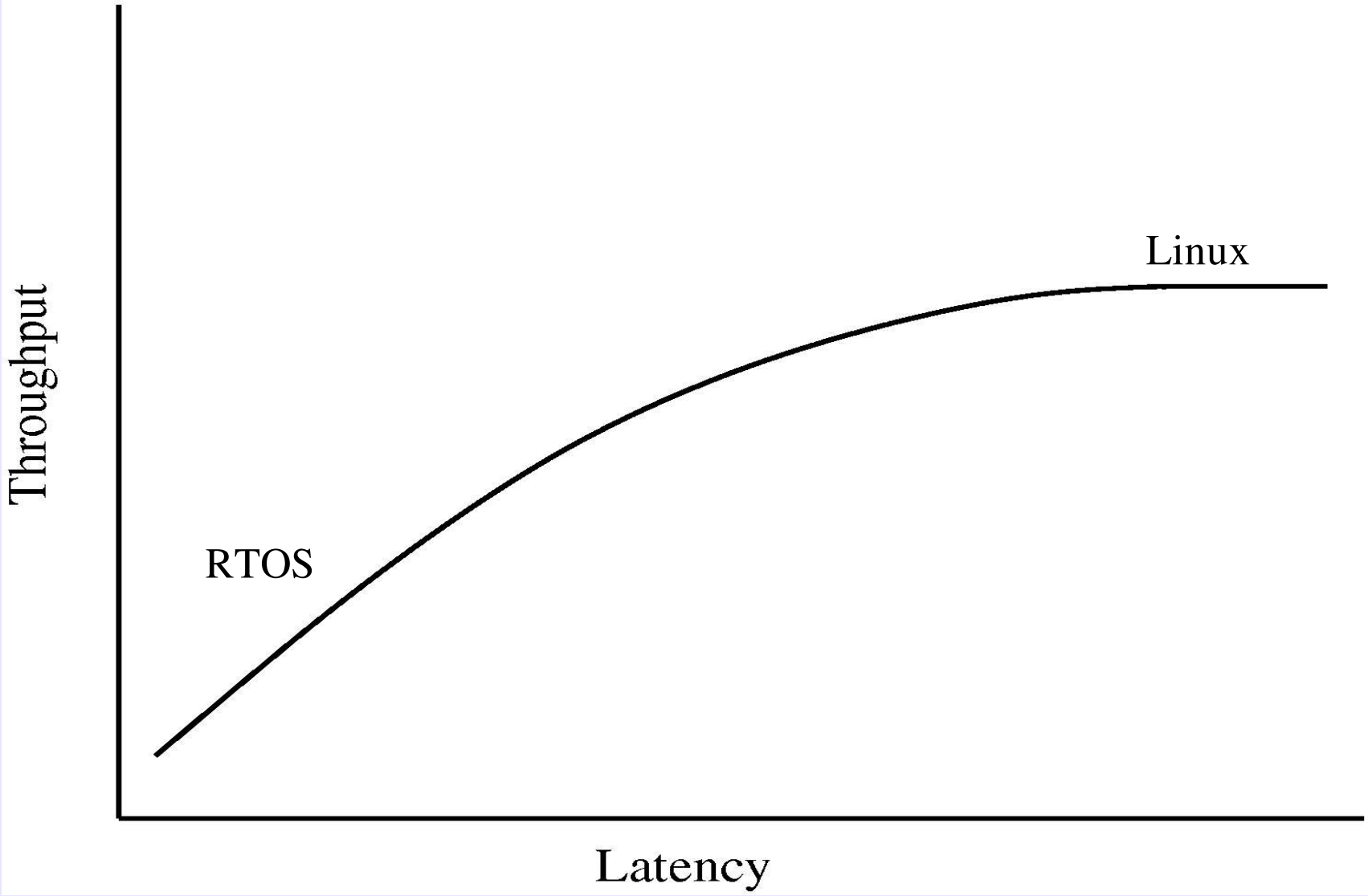
Real Time Systems

- Real time systems are optimized to minimize worse case latency (the response time).
- Interrupt latency is usually the criterion that defines how "Real Time" the operating system is.
- RTOS are usually needed to control hardware that has strict time constraints.

Embedded Systems

- Embedded systems are systems with limited human interaction.
- These systems are sometime very small but not necessarily.
- The embedded computer market is huge The shipment volume of embedded systems is much larger than the PC computer market.

Latency vs Throughput



Real Time – Says Who?

- The majority of real-time systems aren't.
- Embedded systems are often misclassified as real-time systems. However, most systems simply do not require real-time capabilities, in fact these capabilities are detrimental.
- Real-time requirements are often simply designed out through the use of a deeper hardware FIFO, scatter/gather DMA engines and custom hardware.

So You Still Want Real Time!

- There are a number of approaches that can be used to provide Real Time Response:
  - Soft Real time: There are various low latency patches to the standard Linux kernel:
    - Montavista's
    - Redhat's
  - Hard Real time: The are a number of hard real time kernel patches:
    - Rtai
    - RtLinux

6 – Posix Real-Time Extensions

Posix.4 Real-Time Extensions to Linux

- Posix.4 adds real-time facilities to Posix.
- This standard add the facilities typically used in RTOS's.
- In my opinion using these facilities are a recipe for trouble.
- There are no standard Linux programs that use these facilities, just look at your favorite Linux distribution.

Use Linux's Strong Simple Abstractions

- Linux supports some very powerful abstractions that should be preferred over many weaker techniques.
- The major strong abstractions of Linux are:
  - Files
  - Processes
  - Memory spaces
  - IPC

5 – Java

Java

- While this is difficult to classify as a mistake, it is worth noting that virtually no standard Linux programs are written in Java.
- Sun itself uses Gnome-Gtk for its desktop, which is written in C. If Java is so good why doesn't Sun use it.
- Sun releases a SUSE version of Linux, without any Java programs, and dubs it the “Java Desktop System”.

4 – Scaling

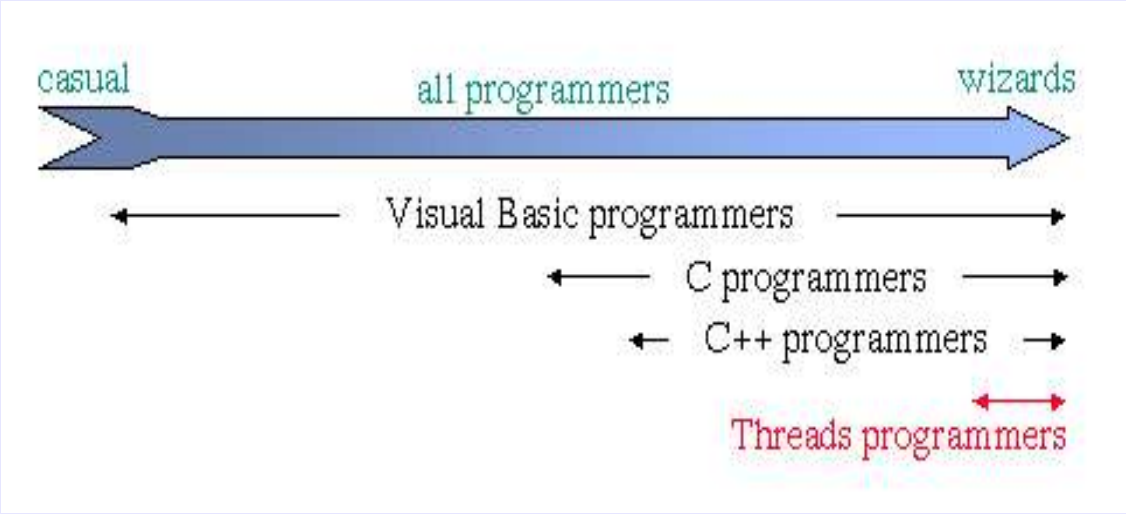
Scaling

- Embedded environments many times have restrictive resources and the software must be properly scaled to run on the platform.
- Things that are appropriate for a large enterprise server, such as Apache, PHP, graphical toolkits that are familiar to many Linux users are just too big for restricted embedded hardware.
- Trying to squeeze these large programs into small flash memory is just no fun.

3 - Threads

3 - Threads

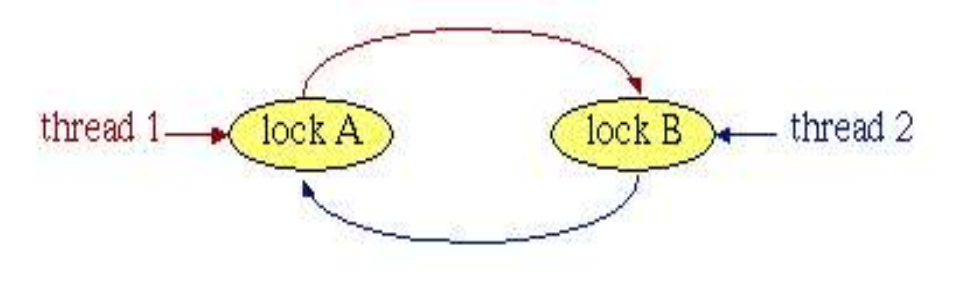
- The main problem with threads is that they are hard to use correctly. Even for experts, development is painful.



- http://www.cc.gatech.edu/ccg/people/rob/software/threads/ousterhout_threads.html

Why Threads are Hard

- Synchronization:
  - Must coordinate access to shared data with locks.
  - Forget a lock. Corrupted data.
- Deadlock:
  - Circular dependencies among locks.
  - Each process waits for some other process.



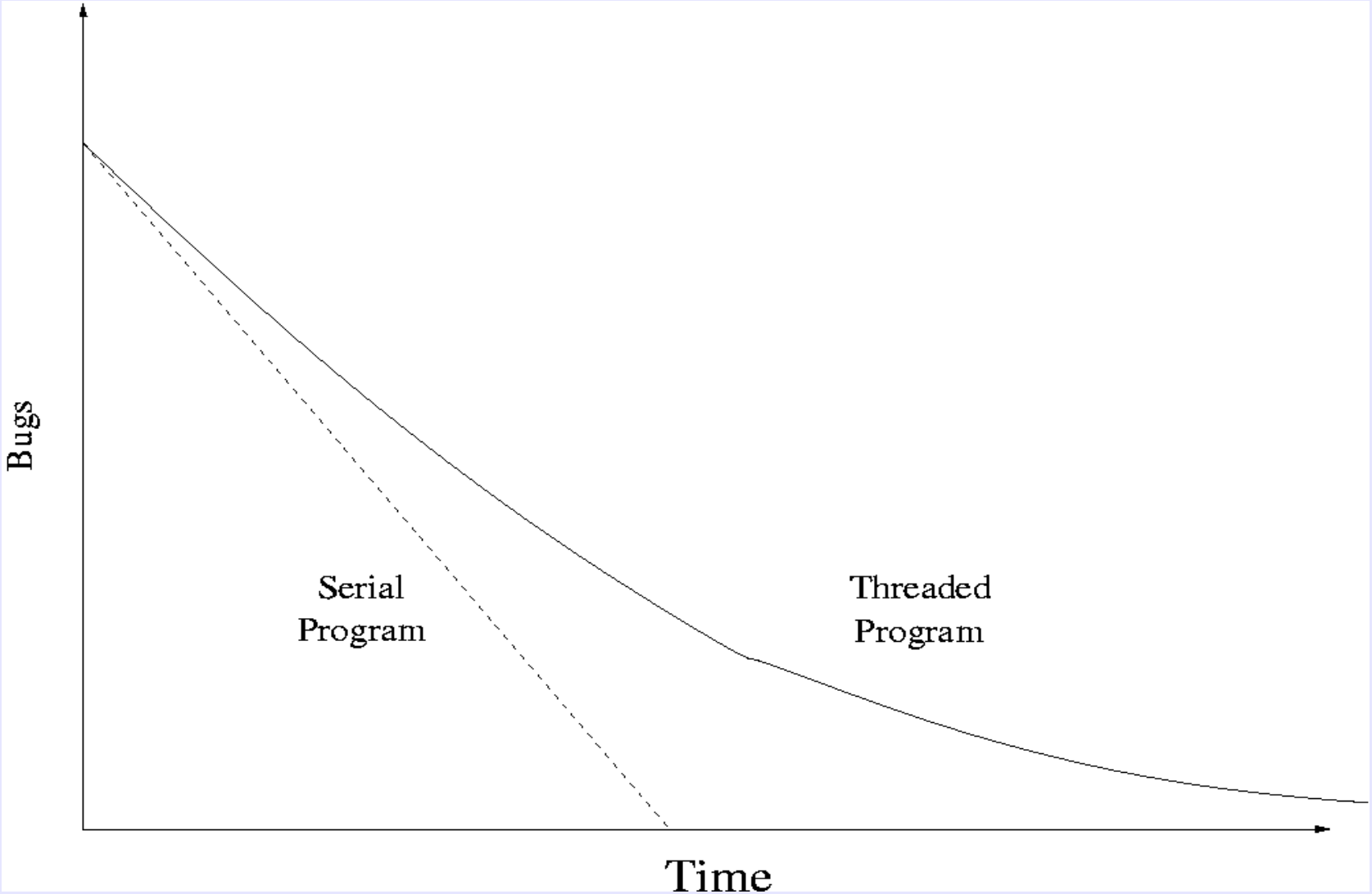
Why Threads are Hard, cont'd

- Achieving good performance is hard:
  - Simple locking (e.g. monitors) yields low concurrency.
  - Fine-grain locking increases complexity, reduces performance in normal case.
  - OSes limit performance (scheduling, context switches).

Why Threads are Hard, cont'd

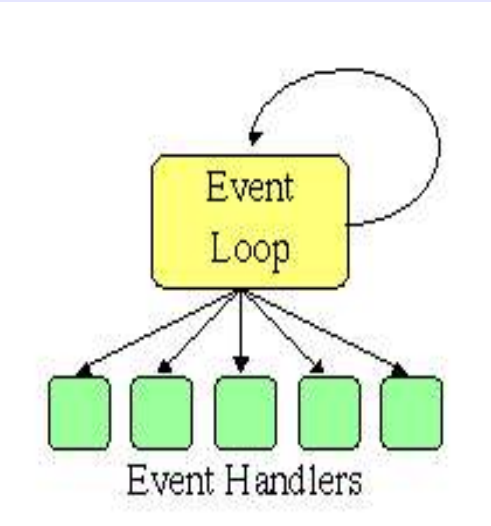
- Threads not well supported:
  - Hard to port threaded code (PCs? Macs?).
  - Standard libraries not thread-safe.
  - Kernel calls, window systems not multi-threaded.
  - Few debugging tools (LockLint, debuggers?).

Debugging Threaded Programs



If Not Threads Then: Event-Driven Programming

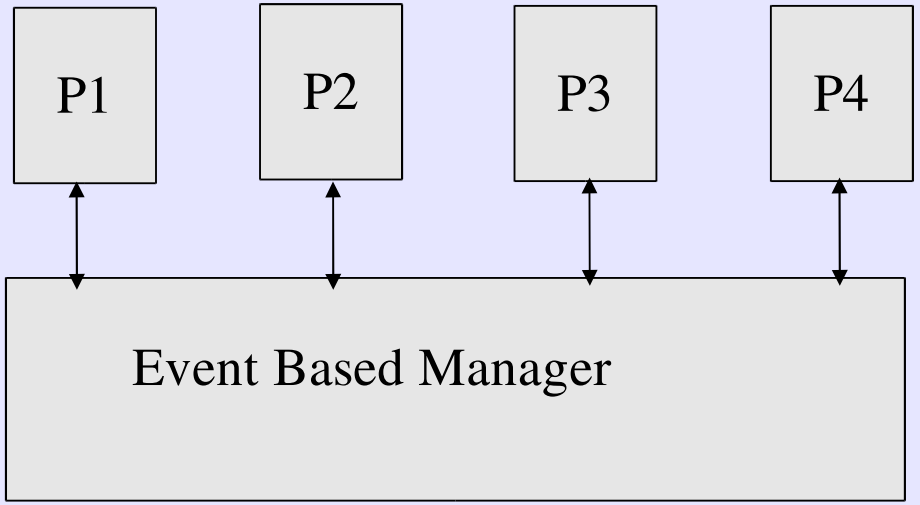
- One execution stream: no CPU concurrency.
- Register interest in events (callbacks).
- Event loop waits for events, invokes handlers.
- No preemption of event handlers.
- Handlers generally short-lived.
- Main loop architecture.



Process Based Concurrency Another Alternative

- Use processes for concurrency rather than threads.
- Synchronize processes with event based IPC.
- Advantages:
  - Simpler and surprisingly more efficient synchronization than threads.
    - send/rcv is self synchronizing and buffered.
  - No race conditions. Much simpler to debug. Trivial to distribute.

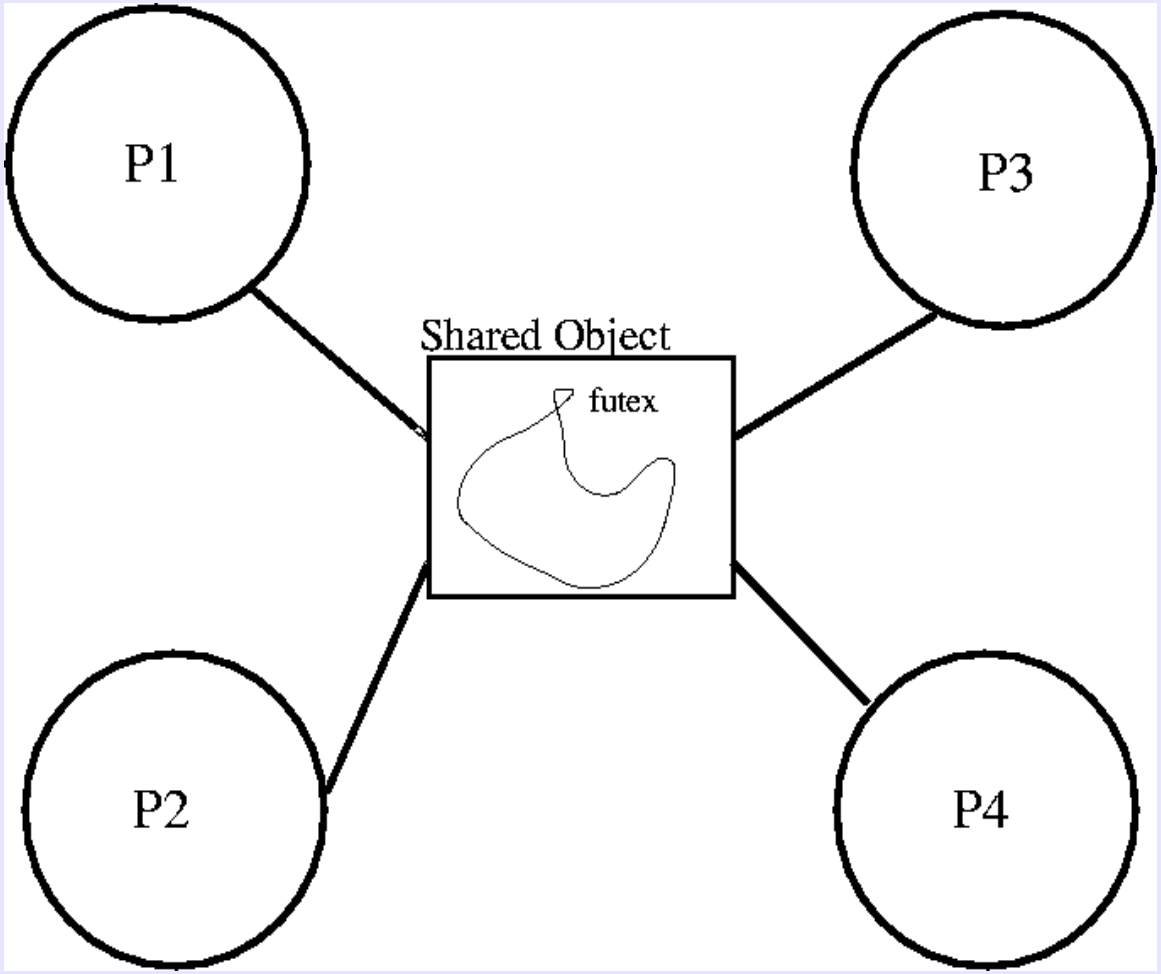
Process Based Concurrency



Process Based Threading Another Alternative

- Instead of sharing all memory, create processes with a shared memory region.
  - This allows you to minimize the interaction of the processes to a well defined subset of the total memory space of the application.
  - Thread safe libraries are not needed.
  - Use POSIX 1003.1b semaphores to synchronize shared data.
  - No performance hit.

Process Based Threading Another Alternative



2 – Use the Source Luke

2. The Source

- The source is your friend.
- The GPL creates a unique situation that makes many embedded devices transparent.
- The WRT54G wireless router is a case in point.
- Linksys (a Cisco company) shipped this box without any indication that the software was GPL'ed
- Someone noticed that this was a Linux box and sent an email:

The Letter

From Andrew Miklas <>
Subject Linksys WRT54G and the GPL
Date Sat, 7 Jun 2003 22:41:23 -0400

Hi,

Awhile ago, I mentioned that the Linksys WRT54G wireless access point used several GPL projects in its firmware, but did not seem to have any of the source available, or acknowledge the use of the GPLed software. Four weeks ago, I spoke with an employee at Linksys who confirmed that the system did use Linux, and also mentioned that he would work with his management to ensure that the source was released. Unfortunately, my e-mails to this individual over the past three weeks have gone unanswered. Of course, I also tried contacting Linksys through their common public e-mail accounts (pr@linksys.com, mailroom@linksys.com) to no avail.

Linksys Releases The Source

- Linksys eventually released the sources.
- You can just download it from their web site.
- This launched “The Little Engine That Could”.
- They have done very well with this product.
- If you want to design an embedded Linux product just look at the completely transparent design of the WRT54G for a guide on how to design an embedded system.

1 -GPL Violations

1. GPL Violations

- Israeli companies tend to ignore the finer details of legalities.
- Violations of the GPL are a serious matter.
- Recently the GPL has been upheld in its first court test in Germany.
- The authors of netfilter, Linux's firewall, has been granted an injunction against Sitecom Germany GmbH for GPL violations.