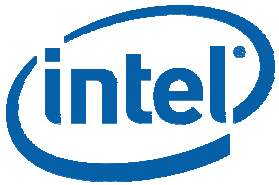


Direction check on proposal for Open Source Driver Enabling Framework

Max Alt, Dario Rapisardi
Channel Platforms Solutions Group
Apr 2006

Agenda

- Introduction
- Problem statement and its scale
- Examples of problem complexity
- What do we have now ?
- What do we need to go forward ?
- Issues and Risks
- Need PDT’s feedback, brainstorm issues/opportunities.

Linux Enabling Problem

There is no central repository of information that provides the "intelligence" about the drivers and their configuration that is needed to fully enable Linux on the myriad of Intel hardware combinations that are available on the market today.

The channel frequently asks Intel enabling engineers if a certain software stack will run on a particular system.

Conversely, OSVs will frequently ask what drivers are needed to make their distro run on a particular hardware configuration.

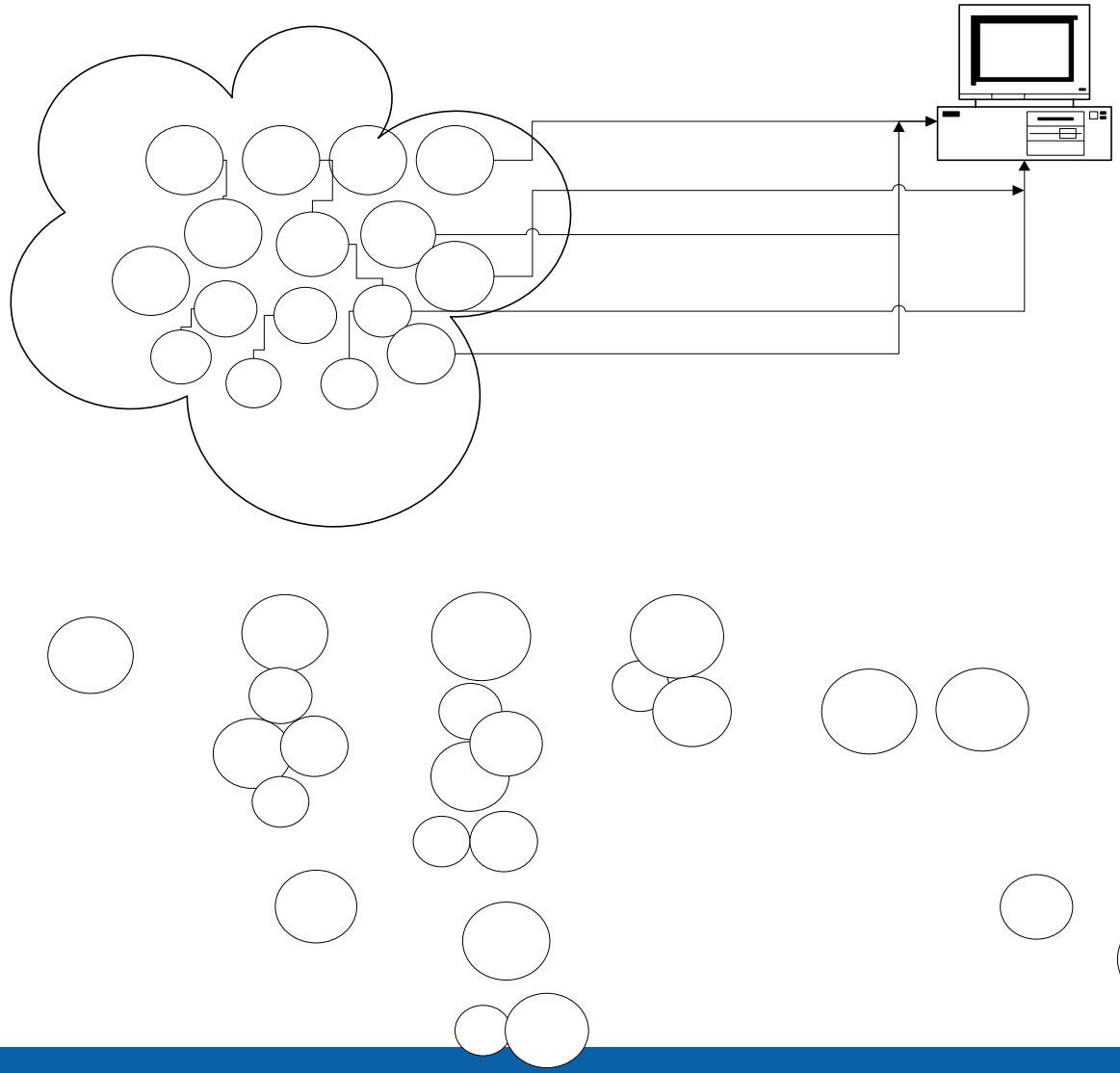
Proposed solution: The Open Source knowledge database harnessing:

- Output of all driver tests used with various kernels and OSs on Intel hardware
- Known S/W and H/W dependencies rules representing system requirements

Problem's scale

- Not unique for CPSG, but problem is an industry wide
- Not unique for clients or servers
- Not unique to non-commercial Linux distributions
- Not OS distribution dependent (both RPM and DEB worlds)
- Not architecture or platform dependent : the framework's essence is in gathered data (which does depend on platform)
- Public statistic data on broken support
- Generates OS communities around issues & solutions
- Creates credible drivers/platform readiness rating
- Microsoft somewhat resolved similar issue for Windows

Problem: Support Complexity in the Channel



Oper

Kerne
l
2.6.12

Als

Typical questions from our Fellow Travellers:

Channel (*when the software stack is specified*):

- Q: Is the specified hardware fully supported on the stack/OS?
- Q: What hardware *could be* fully supported on the stack/OS?
- Q: If not, what do I need to do to ensure the support?
  - For skilled users (instructions on how to make/rebuild)
  - For non-skilled users (instructions on where to get a fix)

OSVs (*when the hardware is specified*):

- Q: What software do I need to have in my distribution in order to support a given set of hardware?

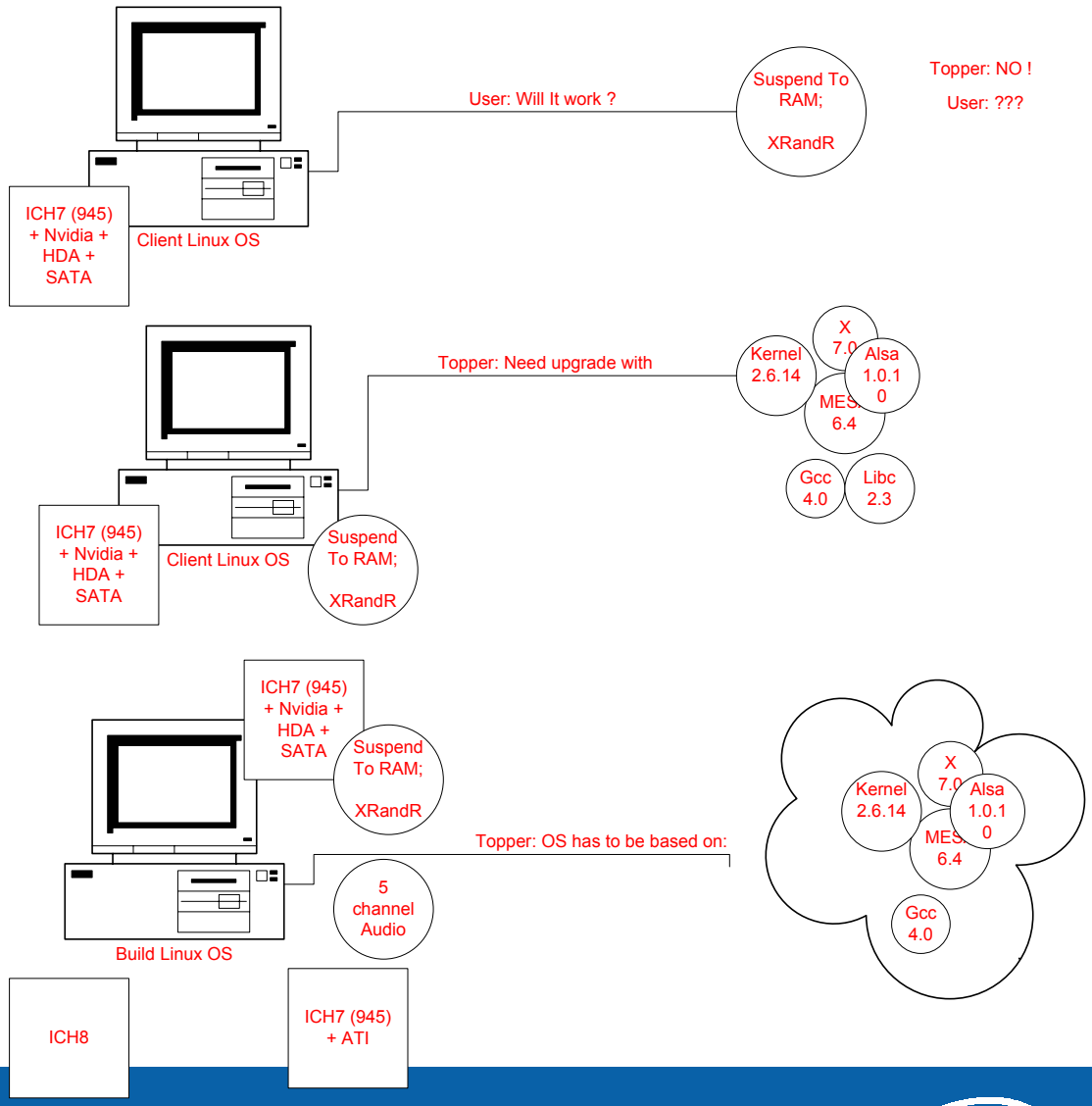
Three Frequent Scenarios

Local channel partner wants to use:

- **A local, non-commercial Linux derivative which contains specific packages:**
  - NVIDIA or ATI video card, or a specific device on their IA system
  - HDA audio
  - 2.6.12 kernel

What should they do to ensure proper functionality of the entire system ?

Will the system "suspend to RAM" ok?



Solution Goals & Benefits

Channel & ODMs: For non-Linux savvy partners, how do we ensure that their Linux/IA-based solution is functional?)

- Coordination amongst pool of OSS SW packages to ensure solution acceptance
- Trusted resource of information on solution's functionality

Scalable enabling strategy:

- Create a consistent way to retain device's test case generation
- Capture the knowledge & best practices into an easy-to-navigate expert system of H/W, S/W dependency analysis

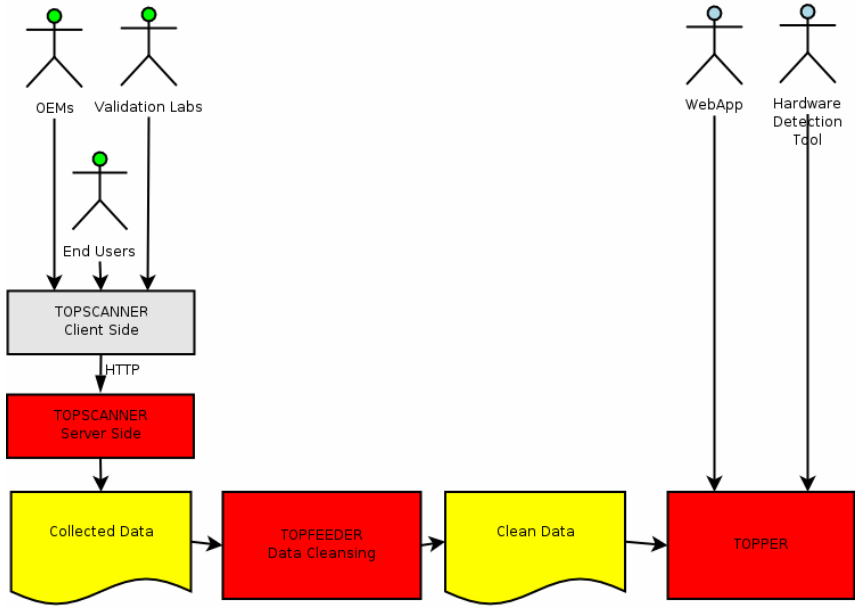
OSV & OSS:

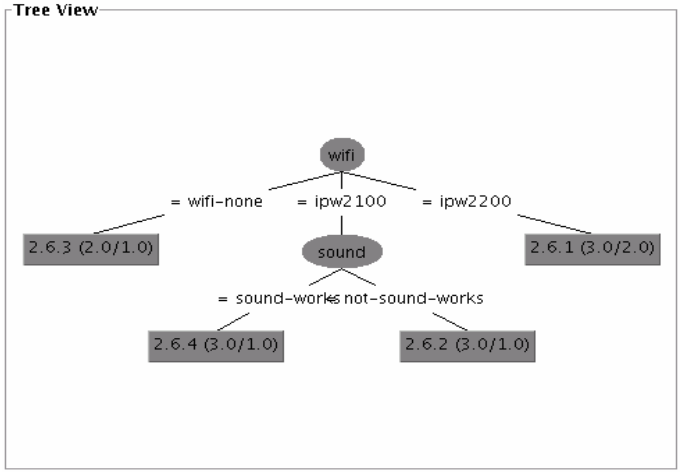
- Use the knowledge on H/W and S/W dependencies to create more intelligent ways to build distributions; promotes community-powered issues resolution
- Trusted source of information to propagate into the Linux core
- Helps direct OS development to tackle most critical H/W support issues

What Do We Have Now

- A prototype, called “Topper”, is ready & can be demoed now!
- Topper workflows and usage models are being scoped
- Topper data structures created – see backup
- Topper interface: User and Validation framework
- Interest from channel to try it out
- Ubuntu, Mandriva, Linspire are interested in participating in this project:

  All OSVs, OSS, channel would contribute





What Do We Need to Go Forward?

Finish architecting the solution

- Jointly with contracted Linux-savvy & OSS trusted partner
- Detailed scoping
- Good for both packaging worlds: DEB and RPM
- Publicly accessible knowledgebase
- Populating with references to existing OSS solutions (driver download page)
- Start conversations with OSDL for Topper's after-POC phases to complete their driver initiatives and provide an umbrella for further maintenance & development

Begin data gathering

- Ensure ICH7, ICH8 information is in the database (from CPSG engineering collateral)
- Have major IPPs and one testing agency to populate tested data
- Ubuntu had given Intel 200,000 data entries of validated SKUs

Issues and Risks

- Main challenge and value is not by the framework itself, but obtaining the data !
- Trusting the validity of entered data. Who is the authority ?
- Scope - What does it mean “device works”? The breakdown of functionality increases tool complexity
- Generic interface to validation processes will be required
- What is the product package: web-site? CD? Collateral? Knowledge management solution? *All of the above?*
- Concept’s success depends on mass adoption
- Needs to be mirrored for both the RPM and DEB worlds. Distro independent architecture
- The Linux kernel just does not like the separation of the Core from the Driver code ! Dependencies are going to continue to represent complex system of constraints to resolve.

Feedback ? Questions ? Comments ?

Backup

Gaps and Initiatives

"Topper" Prototype is ready:

- Written in Prolog, to populate and access dependency data for SW packages with a device and its features. Some 945 chipsets data was already populated from published materials.

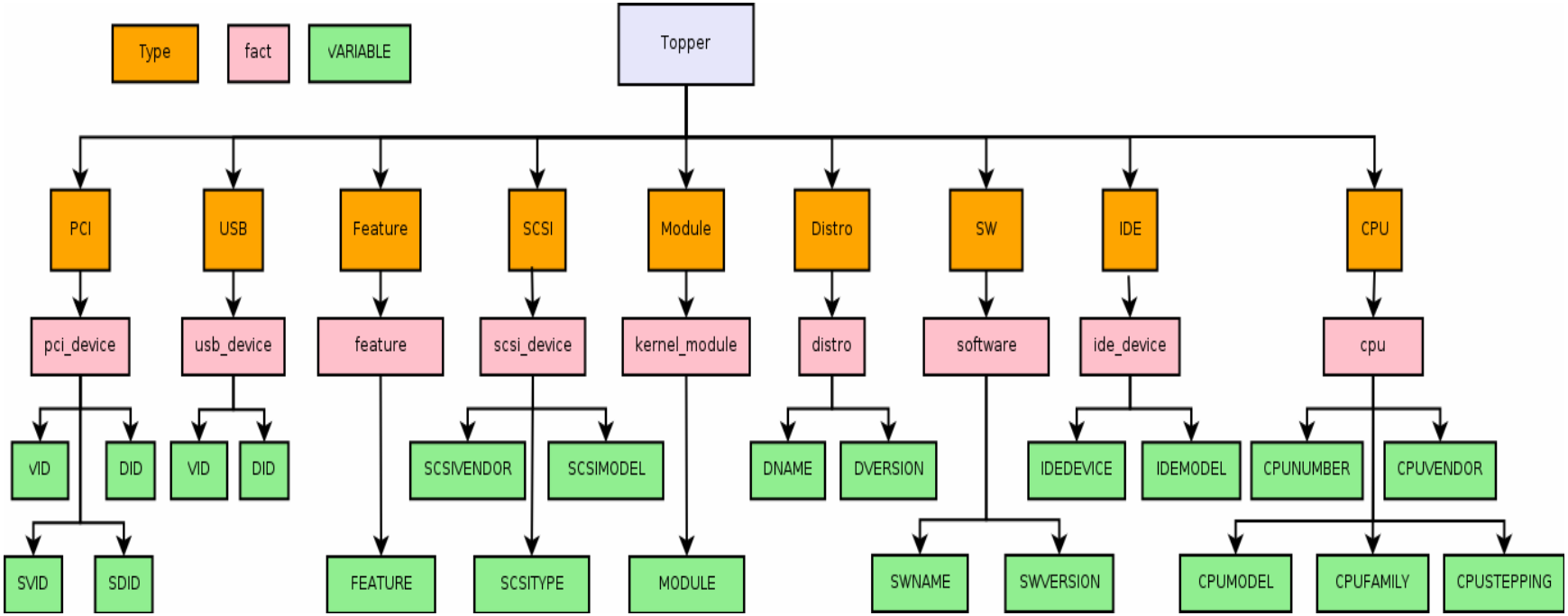
Critical path:

- Need a Linux savvy and OSS trusted partner to team up on covering DEB and RPM worlds to jointly architect suitable solution due to the issue of inseparable kernel core and drivers
- Immediate data mining through existing testing channels: OSVs, LOEMs, Sis, OSS
- Immediate scoping of the project: "What does it do" and "what doesn't it do"

Independent steps

- Pilot to use Topper during red.es workshop and on site at one of Spanish LOEMs
- Pilot to use Topper as a guidance on device's test case engineering
- Ensure web-base public access to repository
- Introduce relation to locations with remotely executable/accessible solutions, i.e. driver downloads, build scripts, code, papers, etc.

Topper Data structures



Analogy– Creating a Food Dish

Assume: Raw ingredients are 1c each (almost free)

Red Hat, Novell, etc: Are restaurants, will still be in business
- Food can go bad, hard to carry
- Pay, but its good and tasty, no hustle, no wasted time on cooking

Debian repository: Is a supermarket of raw ingredients
- Cultivates home cooking
- Great if someone wants to cook at home (e.g. Spain)

Ubuntu, DCC: Represents pre-cooked food covering as many tastes as possible

IA: "meat-lovers" or "spicy-lovers" (for vegetarians)

If you are a meat-lover, how do you cook for yourself at home?
- Recipe book of proven tasty dishes for meat helps (validated, dependencies, precise measurements, order, ingredients)
- People are motivated to use pre-cooked food if its 1c too
- Kitchen supplies (tools)