Report of Luca Simoncini on session 2 - Software Executives

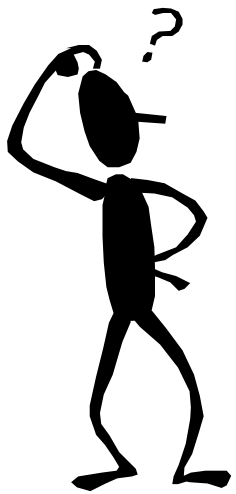
2 Presentations:

Karama Kanoun: Dependability Benchmarking of Off-the-Shelf OS Kernels

Ravishankar K. Iyer: Characterizing Linux Dependability on the Pentium and PowerPC Platforms

Both very interesting presentations

BUT



HOW are they related to our WS:

OPEN SOURCE and DEPENDABILITY ?

Both present approaches to benchmarking, and are complementary:

KK: closed source experimental environment (Microsoft)
measurement made at OS and application level
injection at level of critical system calls

RKI: semi-open source experimental environment (Linux)
injection at level of kernel code, data, stack and CPU registers

Why are they useful:

KK: to provide a set of benchmarks specs with reproducibility, portability and scalability properties
to aid in discovering singularities for guiding architects at system level to cope with them (wrappers, etc.)

RKI: to identify problem sources and provide both producers of platforms and OS with possible solutions

Lessons learned:

- Not everything can be measured through exp. verification
- Need to extend sensitivity analysis to different experiment profiles
- Need to evaluate level of portability in terms of effort for reproducing the benchmark on different gen. purp. OS
- There is a definite influence between the (basilar) architecture of the platform and the OS that runs on it
- Need to raise awareness in microchips producers on the dependability issues that influence their platforms

Benchmarking and Open Source

Open Source may help in:

- Incrementing the granularity of benchmarking so being able to identify subtle interaction problems at more detailed level between platforms
- Allowing a better level of abstraction to be used by architects to deal with problems
- Limiting blind patchworks activities (only usable in case of closed source) that may themselves influence overall dependability
- Allowing independent evaluations
- Limiting monopoly in SW, and help different communities

Open Source may NOT help since:

- It opens the way to very low cost systems of general and pervasive use (mainly used by non-trained users) with a strong push towards low quality applications and services that will definitely impair general dependability and security

The **basic idea behind open source** is very simple: When programmers can read, redistribute, and modify the source code for a piece of software, the software evolves. People improve it, people adapt it, people fix bugs. And this can happen at a speed that, if one is used to the slow pace of conventional software development, seems astonishing.

We in the open source community have learned that this rapid evolutionary process produces better software than the traditional closed model, in which only a very few programmers can see the source and everybody else must blindly use an opaque block of bits.

While this being relevant, should not be the case to start interacting with them to raise awareness that system dependability is more than building correct components ?