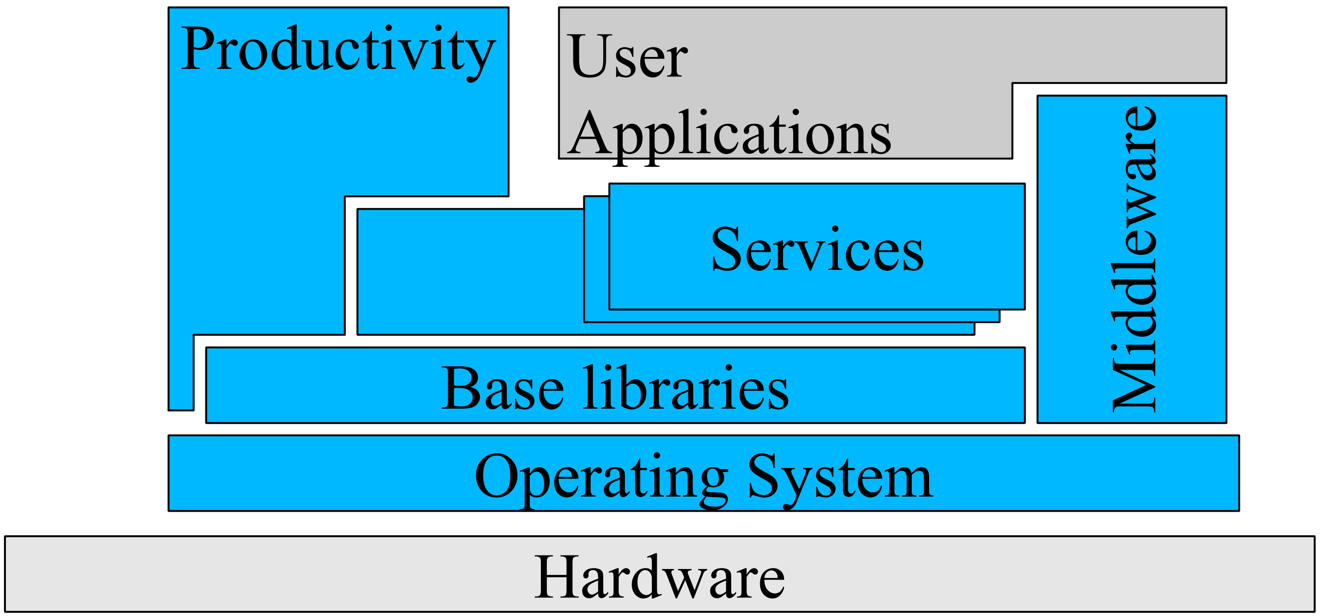
Commercial and Free & OpenSource Software

- “IP” overview
- Criteria
- Examples from CERN
- Forecast

Scope

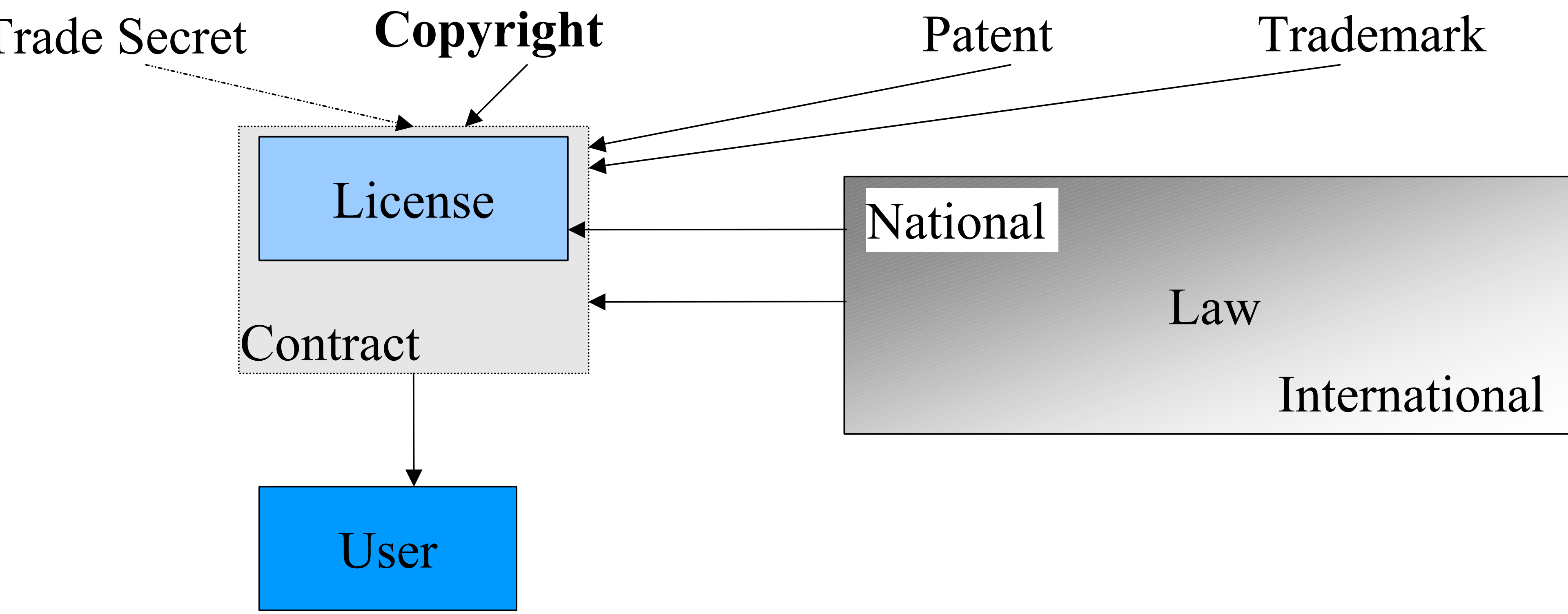
- Physics computing
- Service "building blocks"
- Not: Physics Application software == written by the users



#include <std_disclaimer.h> /* IANAL */

Bias: Linux support

"Intellectual Property"



- Idea: temporary **monopoly** to encourage innovation
- **Copyright**: automatic, "all rights reserved"
- **Patent**: "optimistic" grants

Software Licenses: Key areas

(Definitions)

Execution: run the software

Modification: changes, bugfixes, rights to the changes

Re-Distribution (yes / no)

Warranty (rather: exclusion of ...)

Restrictions (examples):

- keep original license or credits, identify changes
- No reverse engineering
- No use for certain applications (nuclear, medical, military...)
- Patents & Trademarks usage
- No reporting of performance results
- Need to consent to remote checks / automatic updates
- ....

[....] THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAV
EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIE
PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPL
INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNES
PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGR
WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE CO
OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY
COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROG
AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL,
INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE
PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCUR
LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WIT
OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBI
SUCH DAMAGES.

GPL version 2

Software Licenses

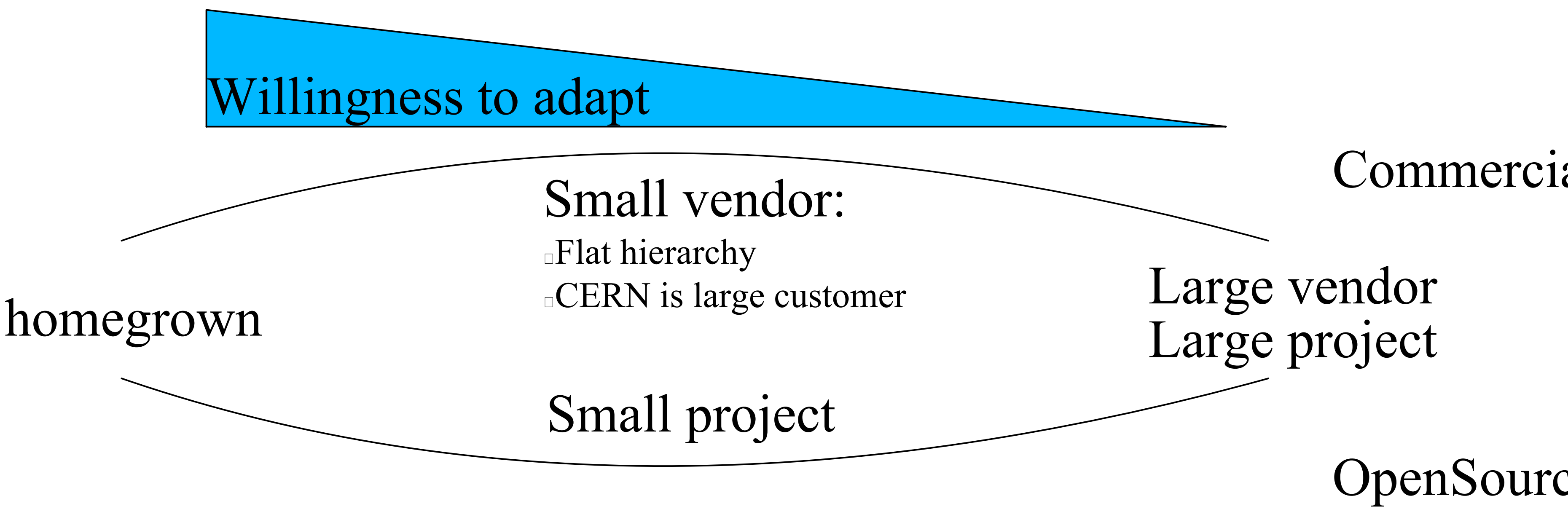
- Ownership: author has all rights anyway
- Public Domain: author grants all/most rights to public
- **F**ree: GNU Public License and compatible
  - Grants rights to use, modify and redistribute, but only under same license. Requires to redistribute source code. http://gnu.org. Emphasis on Freedom and social contract
- free:
  - BSD: right to use,modify and redistribute. May need to contain "prominent notices".
  - OpenSource licenses: http://opensource.org/docs/definition.php
- Everything else: single-user, binary only, "do not distribute" EULA.
  - Not always enforceable in local jurisdiction (click-through, shrinkwrap, restrictions on reverse engineering...)

Criteria for "software service"

- Ideal: Provide zero-cost service with all required features forever and without interruption.
  - Criteria (== cost. In the end.)
    - Suitability
    - Stability
    - Lifetime management
  - Costs
    - Short-term (Migration) / long-term (support, service, lock-in)
    - Manpower / money
    - Direct (to service provider) / Indirect (to users)
    - Immediate / Potential (risks)

Customization issues

- Off-the-shelf does not fit (most of the time).
  - Somebody will need to adapt it: Author / Vendor or CERN
  - Or CERN will have to change procedures



Implications on support:

- Large vendors try to raise the hurdle (whitelist of supported configs)
- Projects tend to ignore you unless you are "sexy" or hit widely-interesting problems

Stability & Lifetime issues

- Stability
  - Instability == wasted resources (human & technical) == cost.
  - Vendor warranty can offset (real) costs, but is rarely offered and too expensive
  - Instability == direct damage to reputation
  - Vendor reputation can decrease risk, both service-related and personnel ("...nobody ever go fired for buying XXX")

- Lifetime management
  - Software is constantly evolving (new features, old bugs)
  - Software market is changing
  - Upgrades (same software) and migrations (new software) will happen
    - Lock-In can be expensive (persistent data, proprietary APIs, support conditions)
    - Collateral requirements can disrupt service change management (upgrade cycles)


Examples: CASTOR (CERN Advanced STORage mananger, http://cern.ch/castor)

- CERN homebuild, in-house development and service, few external users
- License: CERN proprietary, being clarified at the moment (commercial opportunities vs. community involvement)
- Unique storage needs of CERN. IBM HPSS was evaluated 98/99 but
    - Customization was expensive and continuous effort (vendor not taking up mods)
    - No perceived advantages in terms of support cost
    - Not suitable for random access patterns (Disk pool via UNIX file system)
    - Heavy requirements (DCE on all clients)
- CASTOR prototype performed well during ALICE mock DC 2001 → decision to adop in 2002
- Alternatives:
    - dCACHE + HPSS, IBM StorageTank ?
    - Any candidate would need in-depth evaluation (years)
    - No strong contender in sight
- Migration will be difficult (2 PB of data)
    - Current development effort is to modularize and move to standard (internal) interfaces

LSF: Load Sharing Facility, http://platform.com/products/LSF

- platform.com: small company (140 employees), dynamic, academic env.
- CERN is major customer (largest installation in #CPUs)
- Excellent relationship (requirements discussion, developer on site, direct access to developers)
- Responsive to feature requests
  - AFS, Kerberos4 integration (now add-on products)
  - access to source (NDA)
  - CERN requests/ideas taken into mainline products (1 year delay)
- low CERN cost (1FTE) for integration and support
- License cost « DIY cost (features & support)
- Replaced (opensource) NQS at CERN in 1997
- Competing against opensource/freely available MAUI, PBS, SUN GridEngine...
  - Today: LSF is more flexible and sophisticated, CERN uses (some) advanced features
  - Platform is fast-moving (free GLOBUS integration with per-pay support)

Example: ORACLE http://oracle.com

- Large company (40.000 employees), CERN is small but priviledged customer
  - Access to developers
  - Beta tests
  - Proposed modifications
- Strict on requirements (certified = supported on Red Hat & SuSE “Enterprise”)
- Already used at CERN (HR, administrative):

  → Synergies
  - Proven record for stability and support
  - 'Good deal':
    - HEP-wide license
    - CERN actively uses ORACLE support for operational problems, cheaper than DIY
- Any migration would be expensive
- Future:
  - Commercial contenders (SQLserver, DB2) offer no strategic advantages
  - FOSS: MySQL, PostgreSQL lack advanced features (code instrumentation), scalability and support structure, would need “confidence building” period

Example: Linux kernel http://kernel.org

- Large OpenSource project, CERN is 'small' user
- Multiple levels of involvement:
  - (CERN users): custom hardware
  - CERN-IT: overall support, customization
  - Distributors (Red Hat, SuSE)
  - Developers
  - "tree dictator" (Linus Torvalds for 2.6)
- CERN needs to maintain own customizations
  - occasionally feeding back to community
- Future:
  - Commercial (Solaris x86, MS Windows, LynxOS, ...):
    - Restrict co-development and deployment (licensing)
    - Off-the-shelf does not fit, slow turnaround, no advantages if modified
    - Not driven by user community: Linux was introduced at CERN via “grass-root” mvmt
  - FOSS alternatives (xxBSD, GNU Hurd): less momentum, less users, narrow hardware

Example: CERN Linux distribution http://cern.ch/linux

- CERN has customized distribution (software packages and procedures), used by smalle HEP institutes
- Based on Red Hat 7.3 (http://redhat.com/), using freely available software and update services
- In 2003, Red Hat splits into
  - Instable "home use" Desktop release (now Fedora Project) - for free.
  - Stable "production" version ("Red Hat Enterprise") with commercial support and certified 3rd party software.
- CERN now has do decide what to use next
  - stay with Red Hat or change (Debian, SuSE = Novell)
  - Spend manpower and do "self-support", perhaps community-wide (HEP / universities), or
  - Spend money, buy licenses.
- Currently negotiating, together with other HEP institues (SLAC, Fermi, ...)

Forecasts

- Projects tend to go OpenSource, not vice versa:
  - Netscape → Mozilla, Transarc/IBM AFS → OpenAFS, StarOffice → OpenOffice, SAP-DE Kylix,..
  - Typically for strategic reasons (get out of maintenance, annoy competitors)
  - Typically for products on the decline (market share, technology evolution)
- New OpenSource projects either address niche markets or copycat commercial product
- OpenSource tends to catch up and commoditize

  → OpenSource eats existing market share
- OpenSource most fundamental advantage: choice between

  MONEY and MANPOWER
- To compete, commercial products either
  - Need to evolve constantly ("run away")
  - Need to offer substantial support and / or warranties to offset license cost
  - Need to be cheaper than DIY, over time

Resources

- F.Fluckiger: DMM presentation on CERN software licensing
- Windows XP EULA: %systemroot%\system32\eula.txt
- GNU General Public License
- OpenSource definition and list of licenses: http://www.opensource.org/