



Open Source
in the Corporate World

Using Open Source

Brian McCallister



Open Source

Using Open Source Software is no different from using commercially licensed software.



Three Questions

- When should you use Open Source?
- How do you evaluate Open Source options?
- What do you need to consider if you do use Open Source?

Open Source
in the Corporate World

When Should You Use Open Source?





Buy vs Build

Open Source vs Buy

Open Source vs Build



Open Source vs Buy

- Examples
 - JBoss, Apache, Linux
- Why?
 - Onerous licensing terms
 - Often easier for developers to use
 - Meets evaluation criteria



Open Source vs Buy

- **Licensing Terms**
 - Per-CPU licensing and many CPU's
 - Low budget project
- **Reduces up-front cost**

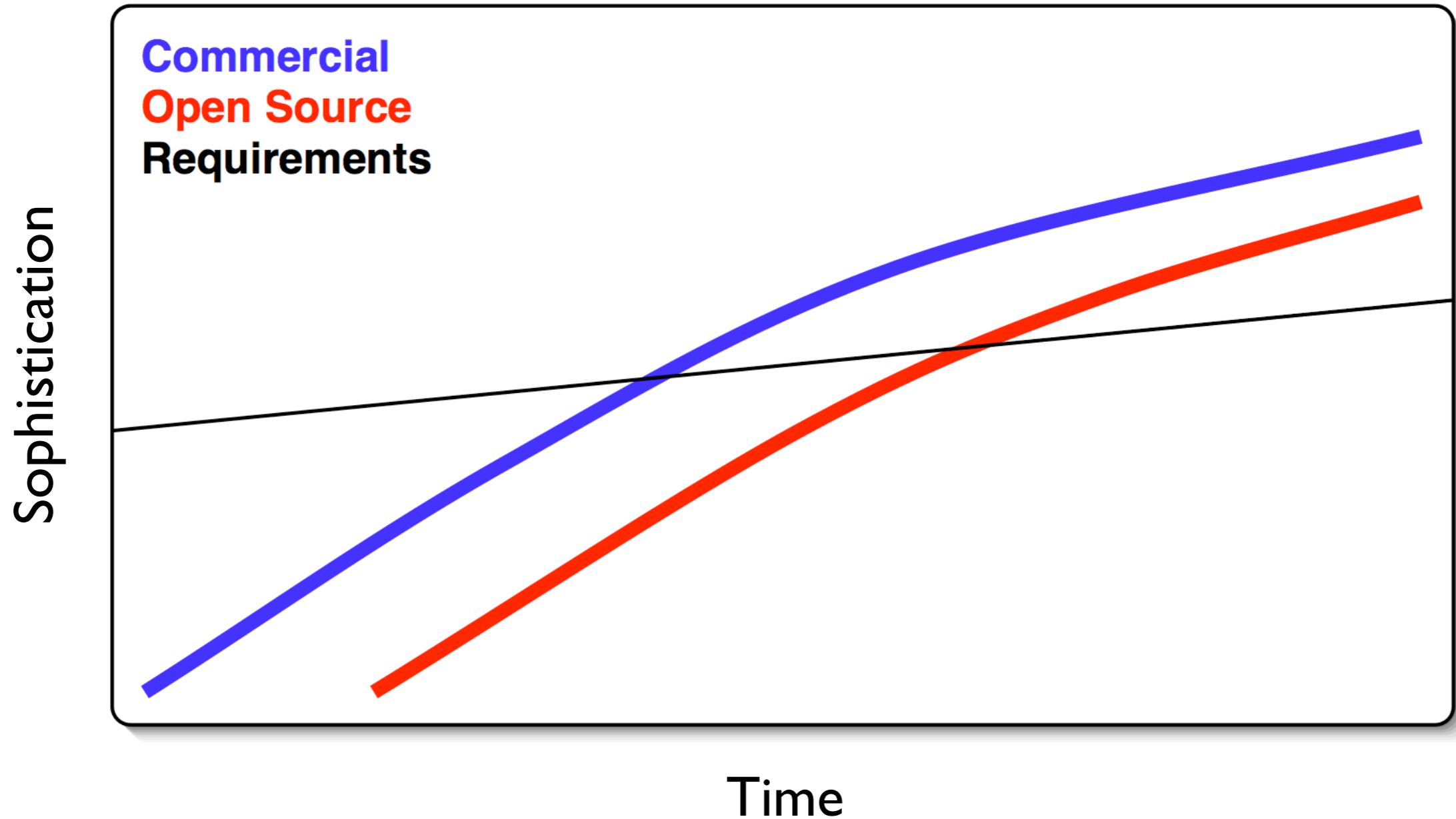


Open Source vs Buy

- Often easier for developers to use
 - Source code available
 - Usually optimized for developer productivity
 - Can fix bugs yourself and contribute back
- Sometimes this is a drawback
 - Formal separation of roles
 - Developer-oriented interfaces



Typical Feature Lineup





Open Source vs Build

- Building yourself you have
 - Architectural Design
 - Development
 - Training
 - Maintenance
- Using Open Source you have
 - Evaluation & Selection
 - Training



Quality

- Domain Specialists
 - Usually those expensive consultants
- Embodies Best Practices
 - Wide points of view, no barrier to entry
- Widely Tested
 - And widely reviewed
- Does one thing well, with clean interfaces
 - Open Source has strong incentives here



Maintainability

- Let other people maintain components
- Updated behind your back
- Clean separation of components
- 2/3 of cost is in maintenance
 - Try to only maintain **your** core code



Open Source vs Build

- **Licensing Issues**
 - Ensure license is compatible with your intended use
 - If you build it, you own it



Conclusion

- Open Source vs Buy
 - Establish criteria
 - Evaluate commercial and open source under same criteria
- Open Source vs Build
 - Think about what you are in for if you do not use a good package
 - Burden is choosing the right package

Open Source
in the Corporate World

Evaluating Open Source Options?





Project Evaluation

- Level of Investment
- Licensing
- Documentation
- Community
- Support
- Maturity



Level of Investment

- Architectural Investment
- Productivity Libraries
- Development Tools
- Infrastructure



Architectural Investment

The core design of the application or system relies upon the specific framework or library.



Productivity Libraries

Libraries, which could be relatively easily replaced later, used to speed development of an application or system.



Development Tools

Tools used in the build or configuration management process upon which the deployed application does not rely.



Infrastructure

Modular systems used by the application which conform to open standards and are relatively interchangeable.



Cannot Escape Licenses

- Commercial Licenses
- BSD Licenses
- LGPL License
- GPL License
- Custom (Open Source)



Commercial

- Licensing Fee
- Terms for evaluation
- May or may not receive source
- Redistribution usually subject to fees



BSD Style

- No license fee
- May change code and redistribute
- No reciprocity
- Apache, FreeBSD, PostgreSQL



LGPL

- No License Fee
- May redistribute
- Changes to library's source must be freely available
 - Limited Reciprocity
- JBoss, OpenOffice.org



GPL

- No License Fee
- May Redistribute (with restrictions)
- All sources using GPL'd code must be freely available
 - Full reciprocity
- Linux, MySQL



Other Open Source

- Many others meet Open Source Definition
- Not in common use
- Details vary, **really** need to read the license
- OpenSolaris, XFree86



Licensing Conclusions

- If in doubt, ask your lawyer
- Consider whether you are using a product or extending a product
- Consider whether you will redistribute or sell
- Establish guidelines for each license type or license



Documentation

- **Primary Documentation**
 - Part of the project
- **Secondary Documentation**
 - Books, articles
- **Tertiary Documentation**
 - Mailing lists, forums, blogs



Developer Community

What is more important, the developer community or state of the code?



Developer Community

- Long Term Architecture
 - Long-term viability of a project is a function of the developer community, not the existing codebase
 - Diversity and Inclusiveness
 - Reactions to new contributors



Developer Community

- Short Term Need
 - Capabilities and maturity of the codebase matters more for non-architectural resources.
 - If it doesn't already solve the problem, you will have to change it until it does.



User Community

- How active?
- How diverse?
- How do they communicate?
 - Mailing lists
 - Forums
 - Conferences
- Case Studies and Success Stories



Support

- **Primary (free)**
 - Remember that user community?
 - Developer involvement in user lists
- **Secondary (commercial)**
 - Individual support from developers
 - Support and services companies



Maturity

- Versioning Guidelines
- Release Cycle
- Developer Community
- Support and Documentation



How to choose

- Create internal guidelines for selecting projects
 - Licensing
 - Documentation
 - Community
 - Support
 - Maturity
- Apply consistently to all open source
- How formal depends on situation

Open Source
in the Corporate World

When You Do Use Open Source





Guidelines for Using OSS

- Pay attention to community direction
- How to interact with the community
- When it makes sense to become a part of the community
- Building internal practices
- Re-evaluate at end of each project
- Be aware of releases and upgrades



Community Direction

- Where is the project going?
- Who is taking it there?
- What about major, incompatible, releases?



Community Interaction

- Understand Expectations
 - Where to ask for help
 - Users Mailing Lists
 - Forums
 - How to ask for help
 - With sufficient detail
 - Without logins and passwords
 - Patiently



Joining a Community

- When do you join a community?
 - Architectural dependency
 - Need new features
- How do you join a community?
 - As individuals
 - Contribute
 - Documentation
 - Bug Fixes
 - Modular Components
 - Core Development



Building Practices

- Doesn't end when you decide to use it
- Set standards and common understandings
- Know where to go for documentation
 - Internal or External -- but use same assumptions
- Common reusable code for projects



Reevaluation

- Reevaluate Projects
 - Did it meet goals?
 - What were the sore points?
 - What did we learn?
- Feed it into the common practices



Releases and Upgrades

- Do you even want to upgrade?
 - Security Fixes
 - Bug Fixes
 - Feature Releases
 - Compatibility
- What testing should you do?
- Should all apps be upgraded together?



Using Open Source

- **Community Focus**
- **Building Knowledge**



Conclusions

- The process for using Open Source Software is the same as the process for evaluating, selecting, and maintaining any other software.
- There are some elements unique to open source, but the majority is that it is more the same than different.



Open Source
in the Corporate World

Thank You!

<http://www.chariotsolutions.com/>

CHARIOT
SOLUTIONS