



Open Source Software 2.0
*Software, licensing models, and
building a sustainable business*

Maria Woods, General Counsel
StillSecure®

August, 2008

- **Introduction**
- **The evolution of open source software (“OSS”)**
- **Licensing models**
 - True open source
 - Community source
 - Shared source
 - Dual use license
- **What model meets your strategy?**
 - What factors to consider in choosing a license?
 - Cobia – a case study
- **Where is OSS headed?**
- **Q&A**



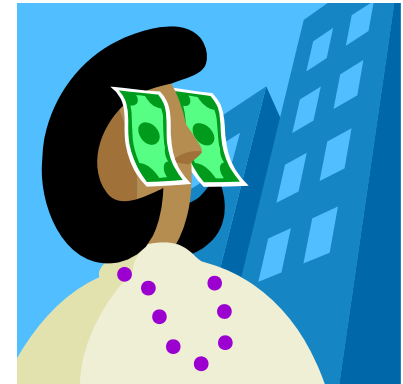
- **Open source 1.0 was primarily for open source “zealots”**

- Open Source began in the 1980s
- Linux appeared on the scene in 1991
- Became much more well known in the late 1990s
- Once companies became aware of developers using OSS/some companies issued moratoriums against use
 - Not “Enterprise ready”
 - Risks were seen as outweighing benefits
 - No support
 - Enforceability of licenses was unclear



- **Open source 2.0:**

- Use of OSS has exploded
- No longer only in developer communities
- Being quickly adopted for enterprise applications (started with Linux but increasingly being used for CRM, content management, security, etc.)
- Companies have demonstrated that you can make money with open source (MySQL, RedHat, etc)



- **Vendors are increasingly leveraging OSS**

- Decreased costs
 - No initial license fee
 - Valuable engineering resources not used to recreate existing software
- Increased innovation – can customize software for needs
- Faster time to market
- Stable code



- **Consolidation in vendor markets leaves room for open source vendors to flourish**

- As proprietary players consolidate, the door opened for open source alternatives

- **Open Source is also becoming big business**

- Open source companies are being purchased by commercial entities at unprecedented rates
- Sun purchased MySQL for \$1 billion, January 2008
- Yahoo purchased Zimbra for \$350 million and Citrix bought XenSource for \$500 million in 2007

In 2007, 30+ open source companies were bought for more than \$1 billion (2x that of 2005)

- the 451 Group

- **According to Gartner, vendors use the open source software model via four strategies:**
 - Support and integrate selected products and services with open source solutions
 - Many vendors have products certified on Linux
 - Sponsor open source projects and solutions as an active member of a community
 - Leverage open source software embedded within other products and services
 - Certain OSS licenses are compatible with proprietary products
 - Provide direct commercialized products and services of open source solutions
 - Aggregations
 - Service and support
 - Subscriptions
- **What changed from open source 1.0 to 2.0?**
 - OSS became mainstream; Gartner states that in 2007 OSS could be found in mission critical applications and that conservative IT organizations began using OSS
 - Have more experience with the license terms and risks haven't materialize
 - Open source vendors emerged offering to support OSS, minimizing risk

By 2012, 80% or more of all commercial software will include elements of open-source technology

- Gartner, Inc.

- **“True open source”**
 - Open Source Initiative (OSI) certifies licenses such as GPL, BSD, Apache, Mozilla license
 - There 72 numbers of OSI certified licenses
- **Community license**
 - Shared development among users with similar challenges/needs
 - Shares the risk across the peer organizations
- **Shared source license**
 - Source is provided but certain other restrictions apply
 - Microsoft licenses fit this category
- **Dual use license**
 - Increasingly being used by commercial entities
 - Typically a company couples an open source license with an alternative commercial license
 - There are some open source packages that offer a choice of open source licenses
 - Becoming very popular



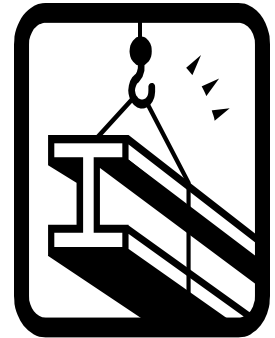
- **OSI has outlined the requirements for licenses meet the Open Source Definition**
 - Includes standard principles (e.g. access to source code, right to modify, freedom to distribute, no discrimination)
 - Not all true open source licenses are “viral”
- **GNU General Public License –most common used license**
 - Drafted by the Free Software Foundation
 - v2, v3, and the Affero license
- **Pure open source licenses: Apache, BSD, LGPL, MIT**
- **Pure open source projects: Linux, JBoss, Eclipse, Apache**
- **Benefits:**
 - Accepted and embraced by the open source community
 - Many “true” open source licenses to choose from that meet various licensing strategies
 - Industry understanding of the terms and conditions of the licenses (although still some ambiguity)



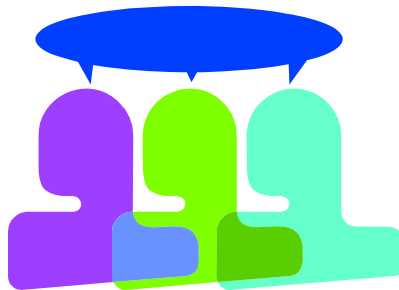
- **Community source license is a development and distribution model that brings different users together that share the same purpose and a set of common requirements.**
- **It is user driven and not generally vendor driven.**
 - Example IT groups from different companies pool internal resources to create a mutually beneficial application
- **Examples include ADULLACT in France; programverket in Sweden**
- **Cobia**
 - Distributed under our own community license that allows for use, modifications and free redistribution when not for profit
 - Foster community, the users have access and use application as each sees fit (except commercial distribution)
 - More on this topic later
- **Benefits:**
 - Control over applications and future features (e.g. no reliance upon vendors)
 - Sharing risk and cost with peers
 - No custom applications required for each organization/user



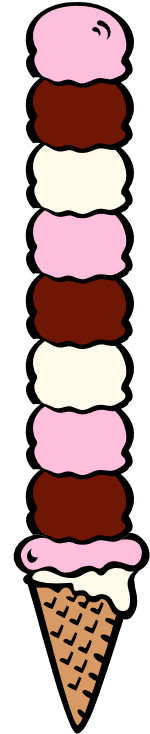
- **After initial “incubation phase” - release runnable and testable code to interested users and developers**
 - Look to friends and family to help with this and provide critical feedback
- **Early projects often run by “benevolent dictators”**
 - Need to have one person in charge of the program deciding what code comes in and what code is kept out
- **Engage with other open source projects and developers to expand awareness**
 - Cross market with other projects
- **Support is critical - feedback needs to be early and often**
- **Encourage all forms of participation - users, testers, developers, documentation**
 - Statistics show only about 7% of community will actually review and possible contribute to code
- **Encourage “out-of-the-box” thinking - the more ways the code is used, the more ways the community can participate**
- **Guerilla marketing - people can’t participate if they don’t know about the project so get creative!**
 - Build the buzz



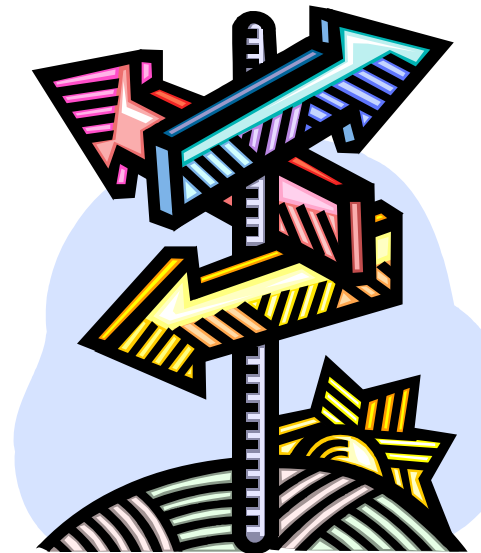
- **This license is not as well defined**
 - Wikipedia says it as “software where the source is available for viewing, but which may not legally be modified or redistributed “
 - The term shared source or “source available” was coined by Microsoft in opposition to true open source
- **No predominant license form,**
 - Companies draft to meet its specific strategy
- **Benefits**
 - No forking permitted
 - Providing source allows for interoperability and complementary products to be built



- **Dual licenses come in a many flavors:**
 - Same product/two different licenses
 - Different products/different licenses
 - Separate open source product can complement commercial products or are “lite” versions of a commercial product
- **Combination of licenses:**
 - Community or Open Source license
 - For individual use, use in completely open sourced project, or to distribute for free as a stand-alone product
 - Commercial license
 - For individuals/organizations that want to distribute the code and not be required to comply with the open source license
- **Note dual licensing works if a company is the copyright holder or has the appropriate rights to dual license**
- **Benefits:**
 - Allows the vendor to support open source community and profit from proprietary sales
 - Open source version can drive revenue of commercial version
 - Can be seen as a “try to buy” strategy.
 - Can have very unique license terms (e.g. Dansguardian)



- **It depends on facts and circumstances**
- **Consider the following factors:**
 - What is the goal for providing an open source license?
 - To drive adoption of the product
 - Create interoperability with other products
 - Upsell to the more feature rich version
 - Where do you expect your revenue to come from?
 - OEMs
 - End users
 - Sale of support
 - Drag along hardware or other software
 - Do you want to allow code to be used in non-open source programs?
 - What amount of control does you want to maintain over the code and future versions?
 - To control forking?
 - Allow widespread, unfettered use to drive adoption
 - Do you want modifications contributed back under the same license?
 - What is the company's risk tolerance for non-compliance and enforcement plan?
- **Review the benefits and drawbacks of the various licenses as compared to your strategy**



- **StillSecure developed Cobia, a unified network platform, and released it under a dual license strategy; an “open source” license and commercial distribution license**
 - We chose a dual license from the get go
- **Business model**
 - License back end to OEMs with customized GUI and proprietary modules
 - Support revenue
 - Provide free platform and charge end users for value added modules
 - Build community at the grassroots level
- **The licensing selection process**
 - Wanted license transparency:
 - Wanted to avoid ambiguous terms
 - Wanted to require sharing of commercial profit; we only make money if you make money
 - Want contributions to come back to the source tree
 - Wanted to control forking, to the extent possible
 - Dual licensing – allow free, unfettered use in the individual and internal business environments and require a commercial license for revenue generating distributions



- **Open source software is here to stay!!! But will continue to evolve and develop**
- **What is “open source?”**
 - The “pure” open source supporters and commercial users will continue to battle over what is open source
 - As community source, shared source and other variations of “open source” become more prevalent, the term will speak to the development model as oppose to license certification
 - The key is the source being available
- **How will the GPL v3 affect the world of open source?**
 - GPL v2 still widely used for software projects,
 - GPL v3 adoption taking longer than expected
 - Linux’s refusal to change is a factor
 - As industry standards for the GPLv3 develop and are understood, adoption may increase

Will OSI remain the “standard” for certifying open source projects?

- To date, OSI has focused on the “true open source” model
- As “open source” includes various licensing strategies, OSI has the opportunity to expand to help direct the development of these models and educate the market
- If OSI doesn’t evolve, their impact will diminish because they will no longer relate to all models of open source





Thanks and support materials:

- **Open Source Alternative: Understanding Risks and Leveraging Opportunities, Heather Meeker © 2008**
- **Gartner Research: Community Source: When Users Can and Want to Retain Control © October 1, 2008**
- **Gartner Research: Open Source in Vendor Business Strategies © March 31, 2008**
- **Open Source Mergers and Acquisitions, Socialized Software © January 18, 2008 (Quoting 451 Group brief on Open Source Mergers and Acquisitions)**
- **How to Pick an Open Source License (part 1 and 2); Ed Burnette's Dev Connection © June 14 and 20, 2006**
- **Gartner Research: Hype Cycle for Open Source Software © July 7, 2008**