



Look Closer

You'll find GENBAND in nearly every network

A lawyer's strange love?

Or:

How

I Learned

To

Stop

Worrying

And

Love

Open Source

Software



DRIVING THE NETWORK EVOLUTION



produced for **ACCT Canada - Innovation 2011**

directed by **Christine Piché**

screenplay by **Thomas Prowse, IP Counsel**

located at **Montreal Hilton Bonaventure**

© November 22, 2011

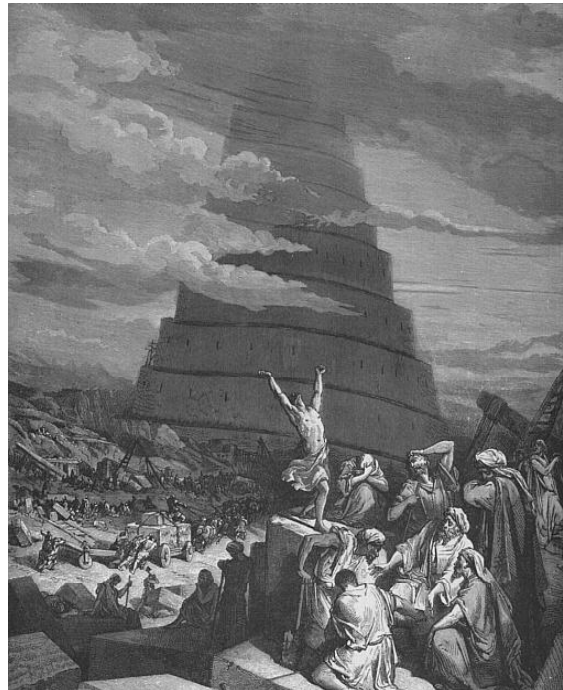




Disclaimer:

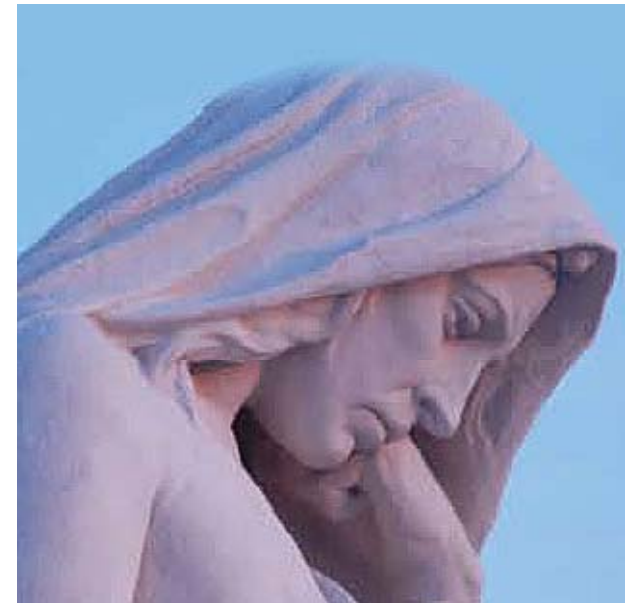
- This presentation will consist of general information and commentary and is not legal advice.
- I will be speaking as an individual and not as a representative of GENBAND.
- Any observations and opinions that I am putting forward are personal.
- *No lawyers were harmed in the making of this presentation!*

Legend tells us that in the beginning, we spoke one language...



and we all got along.

Then things changed....



and the Cold War brought us....

Distrust

MAD - Mutually Assured Destruction

Western World	Eastern Bloc
Free elections	No elections or fixed
Democratic	Autocratic / Dictatorship
Capitalist	Communist
'Survival of the fittest'	Everybody helps everybody
Richest world power	Poor economic base
Personal freedom	Society controlled by the powerful
Freedom of the media	Censorship

FEAR

In the beginning, all software was free....



and all computer programmers got along.

Then, some software was no longer free...



and some programmers really missed that freedom...

so they did something about it....



&



=



and this brought us the Open Source vs. Commercial Software “Cold War”....

Distrust

MAD - Mutually Adversarial Development

Open Source	Commercial
Free	No elections or fixed
Democratic	Autocratic / Dictatorship
<i>“Communist”</i>	<i>Capitalist</i>
<i>Everybody helps everybody</i>	<i>‘Survival of the fittest’</i>
<i>Poor economic base</i>	<i>Richest software power</i>
Personal freedom	Society controlled by the powerful
Freedom of the media	Censorship

FEAR



But over the past several years, an Open Source and Commercial Software détente has emerged

- ***Détente:***
Means "peaceful co-existence". This refers to a period from the late 1960's to 1970's when the cold war tension between the United States and the USSR was reduced.

Formal definition of OSS :

1. Free Redistribution
2. Source Code
3. Derived Works
4. Integrity of The Author's Source Code
5. No Discrimination Against Persons or Groups
6. No Discrimination Against Fields of Endeavor
7. Distribution of License
8. License Must Not Be Specific to a Product
9. License Must Not Restrict Other Software
10. License Must Be Technology-Neutral

Source: Open Source Initiative
definition of "Open Source"

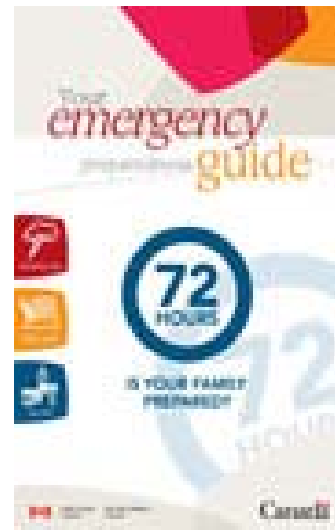
My informal OSS definition:

Open Source Software (e.g. Firefox)
is to
commercial software (e.g. Microsoft Internet Explorer)
what
Wikipedia
is to
Encyclopaedia Britannica

The emergence of “Commons Sourcing”

- While OSS is interesting, it is perhaps most interesting as the “tip of the iceberg” of a bigger societal shift in the way that value is created
- I coined the term “Commons Sourcing” some time ago to describe the pooling of shared information resources under OSS methodologies for use by companies for commercial purposes
- I believe that Commons Sourcing is the emerging third wave of commercial transformation after the earlier waves of in-sourcing and out-sourcing
- Commons Sourcing approaches are increasingly being found at the core of a broad range of new and promising commercial and public sector initiatives

While this world of détente is much safer than the Cold War world, we still need to focus on Emergency Preparedness



STEP 1. Know the risks



- Open Source Software (“OSS”) is a “Gordian Knot” of legal, business, and technical issues



STEP 1. Know the risks



- some threshold legal issues apply to all OSS (and almost all software!)
 - “pedigree” – are you getting what you think you are getting
 - “Bummer of a birthmark, Hal” risks – are you putting your hand up for a lawsuit
 - “license to???” – have you obtained and fully assessed the OSS software license



STEP 1. Know the risks



Pros and Cons of Open Source Software (“OSS”)

pros

- **Solid technology**
- **Familiar technology**
- **Very large ecosystem**
- **Peer review**
- **Community support**
- **Access to source code**
- **Faster time to productivity**
- **Reduced time-to-market**
- **Lower development costs**
- **“career portability”**

cons

- **License Contamination**
- **Product Constraints**
- **Unknown Origins**
 - **No Warranty**
 - **No Indemnification**
- **Possible lawsuits/injunctions**
- **Unpredictable evolution**
- **Customer Concerns**
- **No Formal Support**
- **Security Concerns**
- **ALL OBLIGATIONS must be met**



STEP 1. Know the risks



there are “2.3” OSS categories that you need to be aware of

- “permissive” – such as BSD & MIT
 - In most cases, the reproduction of the copyright notice / disclaimer is the only requirement
- “reciprocal” – such as GPL
 - “elastic” such as Eclipse Public License
 - “unified” such as GPLv3
 - “host-unified” such as Affero or aGPL

STEP 1. Know the risks



- the manner in which OSS is used has significant implications
 - Most OSS can be used in a modified (or unmodified!) form either internally or for delivery of a hosted service (subject to Affero and any license restrictions e.g. non-commercial use only) without triggering an obligation to release the modified source code

STEP 1. Know the risks



- the manner in which OSS is used has significant implications (cont'd)
- you need to focus primarily on two main branches of analysis:
 - the modification of OSS, and
 - the combination of OSS with other code

STEP 1. Know the risks



- With respect to “modification (with distribution) of OSS”, you will *typically*,
 - for any permissive software, only have to reproduce the applicable copyright notice and disclaimer, and
 - for any reciprocal software, have a clear obligation to release the source code for the modified OSS when you distribute your software (subject to the terms of the OSS license)

STEP 1. Know the risks

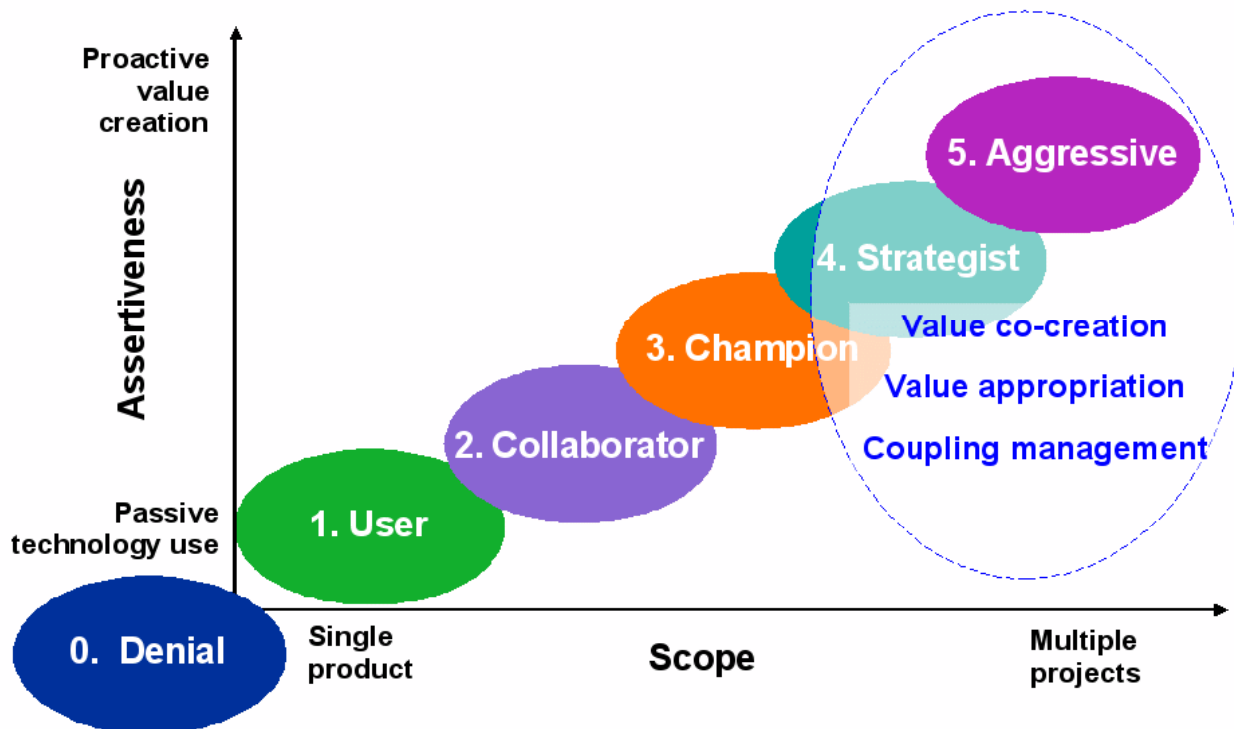


- The “combination of OSS with other code” branch includes (1) combination of OSS with other OSS, and (2) combination of OSS with other code that needs to be kept proprietary
 - While the permissive licenses allow the OSS code to co-exist with proprietary code, the reciprocal licenses require detailed analysis of the software architecture. The “relationship” between the OSS code and the proprietary code determines whether the code can remain proprietary, not merely whether or not the OSS code has been modified (a common misconception)
 - *This sub-branch is one of the most complex areas of OSS practice!*

STEP 2. Make a plan - OSS Maturity Model



Company Behaviour Appears to Follow a Maturity Model



STEP 2. Make a plan



- **OSS Policy & Procedure**
 - Your organization should have an OSS Policy & Procedure (“OSS Process”) in place
 - This OSS Process needs to be developed on a collaborative basis among the business, technical, and legal teams
 - Needs to reflect (or in some cases lead) the overall objectives of your organization

STEP 2. Make a plan



- **OSS Policy & Procedure (cont'd)**

- Your OSS Process should ensure that the pedigree, risk, and license assessment steps are consistently followed and that the license information and the OSS code are properly “archived”
- For the reasons set out above, the OSS Process needs to focus primarily on the modification (with distribution) of OSS and the combination of OSS with other code
- The OSS Process should ensure that obligations are fully understood and fully complied with
- The OSS Process should also address employee contributions to OSS projects

STEP 2. Make a plan



- **OSS Best Practices**
 - In many instances, good up-front and well informed architectural decisions will adequately address the OSS license compliance situation
 - In other cases, more sophisticated approaches (such as the creation of a middleware layer) will need to be adopted

STEP 2. Make a plan



- **OSS Best Practices (cont'd)**
 - Contracts need focus on both the procurement and customer licensing fronts
 - On the procurement front, your contracts, including commercial software license agreements and development agreements, need to set out an OSS identification process together with appropriate representation, warranties, and remedies

STEP 2. Make a plan



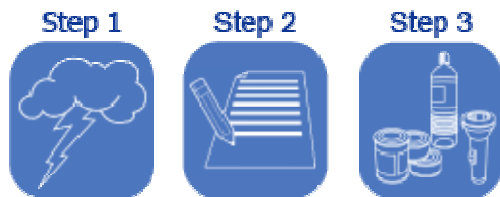
- **OSS Best Practices (cont'd)**
 - Contracts need focus on both the procurement and customer licensing fronts (cont'd)
 - On the customer licensing front, your contracts need to clearly make reference to, contain license text from, and grant to its customers any rights they may have under the applicable OSS licenses for software included in its offering

STEP 3. Get an emergency kit



- Given the complexity surrounding both the various forms of OSS licenses and the use of OSS software itself, you will need to ensure that your organization has the expertise necessary to make these critical assessments
- There are organizations such as fosslc.org that are providing leadership for educational initiatives in the OSS area
- There are companies, such as Protecode, that can assist your organization by providing services such as code analysis
- There are true OSS lawyers who would be pleased to assist!

Summary

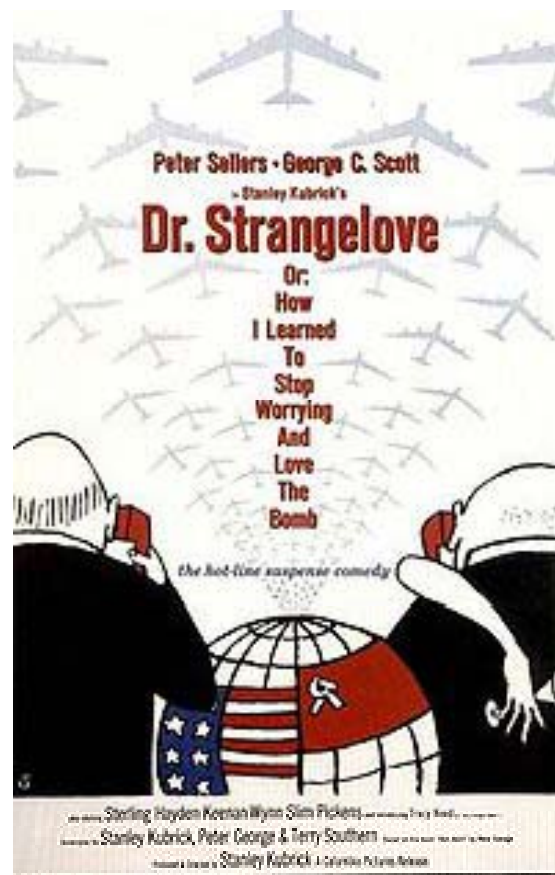



- A policy based approach to OSS will enable your organization to balance the tensions among competing legal, business, and technical considerations
- The development and implementation of a comprehensive OSS process will require both a clear articulation of your organizational objectives and access to certain specialized expertise and resources
- This OSS process will help your organization manage itself through the complexities of OSS on its journey to success

Questions?



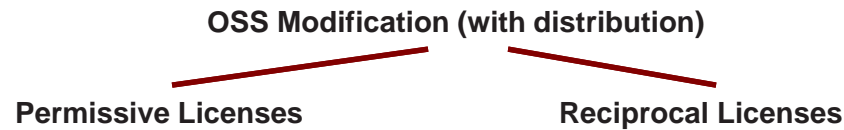
Thank you!





*Back-up slides on the
modification of OSS, and
the combination of OSS
with other code*

STEP 1. Know the risks



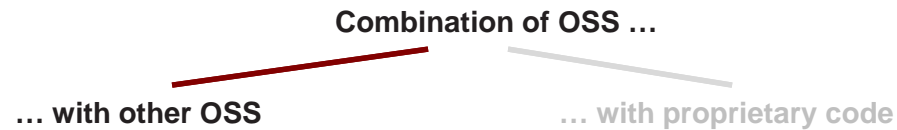
- With respect to “modification (with distribution) of OSS”, you will *typically*,
 - for any permissive software, only have to reproduce the applicable copyright notice and disclaimer, and
 - for any reciprocal software, have a clear obligation to release the source code for the modified OSS when you distribute your software (subject to the terms of the OSS license)

STEP 1. Know the risks



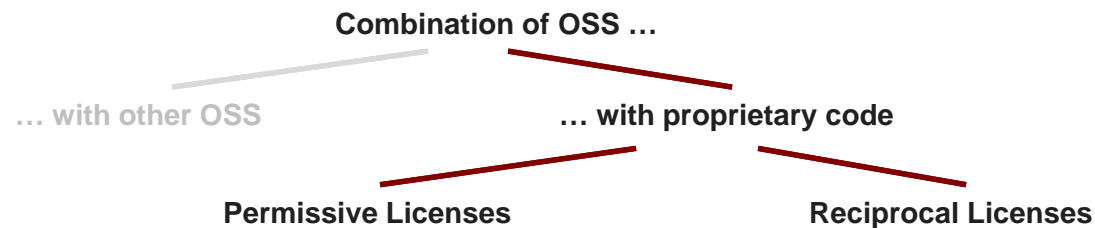
- The “combination of OSS with other code” branch has two main sub-branches:
 - combination of OSS with other OSS, and
 - combination of OSS with other code that needs to be kept proprietary

STEP 1. Know the risks



- For “combination of OSS with other OSS”:
 - If you intend to combine the OSS software with other OSS software, you will need to ensure that the terms of the licenses are compatible
 - Given the sheer number and complexity of the OSS licenses, you will need to ensure that you secure the resources necessary to make these assessments

STEP 1. Know the risks



- For the “combination of OSS with other code that you wish or need to keep proprietary”:
 - “Permissive” OSS licenses allow the OSS code to co-exist with proprietary code
 - “Reciprocal” licenses require detailed analysis of the software architecture in order to determine whether the proprietary source code must be shared as a condition of distribution. It is the nature of the “relationship” between the OSS code and the proprietary code that will determine whether the code can remain proprietary, not merely whether or not the OSS code has been modified (a common misconception)
 - *This sub-branch is one of the most complex areas of OSS practice!*