Readings reviewed: “IBM’s Pragmatic Embrace of Open Source” by Pamela Samuelson and “Open Source Baselines: Compared to What?” by Lawrence Lessig

The Samuelson piece discussed three possible reasons why IBM embraced the concept of open source software: an anti-Microsoft approach, open innovation and a better business model story. Of the three, the combination of business model and open innovation seem the most realistic. It is completely logical for a company like IBM to want to share the burden of software development with other investors; like IBM does with Nokia, Intel and Hitachi in the development of Linux. If new software fails upon entering the market or simply cannot compete with established firms like Microsoft, then multiple companies share the financial loss. Losing $100 million will hurt a lot less than losing $1 billion.

In the United States, where business promotes its personal interests over that of the general public, it may seem odd that a company (or group of companies) has found any success in developing open source software. The idea of developing software with lessened intellectual property restrictions is in fact the opposite of what capitalism represents – why share innovations and improve service for the general public when we can keep all the profit for ourselves? As Lessig mentions in his article, countries like Germany, France and China are currently pushing for mandatory (or at least highly recommended) open source software development. This has of course caused many in the United States to challenge the role of government in determining whether or not software must be open source. (Perhaps because we use the threat of big government anytime profit is challenged)
The Lessig reading also contained several key points about open source vs. proprietary software. The difference between the two still confused me before reading this article. I understood that proprietary software allowed users (or lesers) to run the programs while not fully understanding the code behind it but how proprietary software “hid” its code was still confusing. Lessig explained that proprietary software contained only object code (readable only to computers) while free source software contained both the object and source material (readable to humans). His Kentucky Fried Chicken Metaphor was quite possibly the most useful thing I’ve read about open source vs. proprietary.

Lessig also cleared up a few terms that went completely over my head in lecture: copylefted and noncopylefted. Copylefted refers to the open source development of a system like Linux, which is protected under the general public license meaning all innovations must be done in accordance with the GPL. Noncopylefted on the other hand refers to a system like Apache that requires no licensing for any advancement.