All I want to do is make money writing Free Software!

BUSINESS PLANS IN FREE SOFTWARE

When you write free software, it's a good idea to consider your goals.

You might even want to draw up a business plan.

BY JON "MADDOG" HALL

Someone recently approached me at a conference and told me that they had a free software product as a business and the business had failed. They had conceived of a product, hired a set of developers, invested a lot of money in the project, and when it came time for people to buy the software, no one purchased it. He lost all of his investment money. I asked him what his business plan had been before he started hiring the programmers. He told me that he had never gotten to the point of writing a business plan. This was part, if not all, of his problem.

Before you write free software, you need to determine what your goals are and how you plan to achieve those goals. Some people write free software "just for fun," or to learn more about writing software for a particular need. They have no incentive to make money with the software. Even if they never make a single dollar from the software, they will keep writing it.

Other people have an interest in a field such as photography, astronomy, or general systems administration, and they need the software to help them in that field. Often they create a project to get others who are also interested in that field to help them make the software better.

Some people write free software for the political or sociological reasons of "Freeing the Internet" or "Digital Inclusion." They believe (I feel correctly) that society in general would be better with the ability to change the software to meet their needs, and they dedicate their time and skills to making this happen.

Finally, some people see in free software a chance for people to make or save money, either directly through the sales of services or through the application of new software techniques that add value to the user's business.

This last group of people are usually not writing the software for their own use, so more of the compensation for their time and effort is monetary. Therefore, you have to know from where and at what point in the project the compensation will arrive before you start.

Is your plan for making money based on service? If so, what type of services will you offer? Integration? Training? Tailoring the software? Will your service delivery be able to scale? Will people perceive your service as necessary and a "value-add" to the software?

A friend of mine had a business offering training on a certain package of free software. Because of a switch in the direction of the communities, fewer and fewer people were using the free software that he had built his business around. He then found a closed-source, proprietary piece of software from a company that was floundering and he purchased the company.

Recognizing that 60 percent of the customers that did buy the software bought a support contract and training, he decided to make the software free to download, and he put the source code under an open source license. Immediately the number of companies using the software increased by more than 100 times, and (after a brief usage of the software) 60 percent of the companies that downloaded it bought service contracts and training. However, my friend had the staff, resources, and experience to reach and service this large influx of people, so he could take advantage of the service revenue. He made much more money than when the company was charging per license.

Another friend had a huge amount of software that his engineering staff had developed and which he sold and serviced. As he received more competition in the marketplace, he was forced to lay off engineers and quality assurance people. He decided to make the bulk of the software free software and allowed the community to download, maintain, and improve it. Certain portions of the software he kept proprietary and he refocused his smaller group of engineers on that software, which only the largest customers would find useful. Second, he cut his cost of sales by selling only to the large customers that downloaded the free software. By reducing his expenses and refocusing his engineering staff on new functionality that these large customers found useful, he maintained his business.

Some people will argue that the last example of software was not completely "free" for the large customers. However, the actions taken by my friend allowed small companies to use and maintain the free software to their advantage, rather than have the software disappear completely. A lot of really great technical projects are out there. People who hope to someday support themselves by writing free software.will want to start with a good business plan. ■