How Much Does it Cost to Develop Your Software?
Wayne Lobb and David Warburton
Published June 2011
This installment is Part 2 in a 4-part series.

Part one of this series introduced the concept of “Software Inventory,” which was defined as the number of lines of logic source code that are an essential ingredient in a company’s shipped products. The thousands or millions of lines of code in a company’s software inventory are a product of many work-years of effort by the development team, and may be a major contributor to the value of the organization’s products.

As is the case with many companies’ strategic assets, a firm’s software inventory has both ongoing maintenance costs and an acquisition cost. Both the maintenance costs and the acquisition cost of the inventory can be surprisingly large. This article describes some approaches to calculating these values, and to understanding a few of the underlying factors behind them.

What’s Been the Cost to Build and Own Your Software?
The yearly expense to maintain an existing software base and to develop new code primarily is driven by the fully burdened labor cost of the employees involved in software development. This is because there typically is very little material expense or capital depreciation in software development departments.

In calculating true labor costs, the following roles should be considered:

- Software architects, designers, programmers, testers and documenters;
- Staff who operate source control systems and build/test/package systems—but not IT staff, and not staff who operate issue tracking systems unless the systems are completely owned and managed by core research and development (R&D);
- Software R&D maintenance staff who fix bugs and do minor enhancements, and not staff who support customers in the field, i.e. not application engineers, even though those engineers might create large amounts of software in their own right; and
- Management of all the above groups, up to and including the highest software technical management positions within engineering or R&D.
The yearly labor cost of the staff in these roles is a good proxy for the ongoing costs of maintaining the existing software inventory and expanding the inventory by the creation of new products or product features.

Estimating the total inventory cost of the organization’s software can be a little more difficult, since the total inventory typically is the result of many years of work by teams that have varied in both size and composition. There are two ways to approach the problem:

- Either directly calculate the number of work-hours devoted to creating the software inventory, or indirectly estimate it by using calibrated metrics.

To use the direct method, simply add up the yearly labor costs of the software team for all the years it has devoted to creating and maintaining the active software inventory. For young companies or single product companies, this may be the most accurate way to cost the inventory, especially if there is a clear start date to the software development effort.

For many reasons, however, the direct method may be impractical. In that situation, the best approach may be to estimate the value of the inventory using the following metric: On average over time and across industries, a professional software R&D staff—including R&D management—generates about 700 lines per month per person of new or changed custom software logic. This number applies, again on average, across the entire useful lifetime of a body of software. At $150,000 per year on average for a fully burdened software staff person, a line of software logic thus costs about $18 in labor to build and own across its entire useful life.

Counts and costs do vary; regulated software development in a large medical or aerospace company with a complex product line and multiple development sites worldwide could cost $50 or more per line.

Using a per-line cost of $18 per line of code in the inventory, the estimated acquisition cost for the software quickly can be calculated, provided that the line count is in step with the methods described in Part One of this column series. (Did you miss Part One? See “How Big is Your Software Inventory?” in the March 2011 issue of Medical Product Outsourcing).

**Where Does the Money Go?**

The estimated dollar value of an organization’s software inventory often is surprisingly large. The breakdown of where all that time and effort has been spent also may be surprising.

On average, about three-quarters of the research and development labor costs for a body of software is incurred after the software has first been shipped to paying customers as a product or part of a product. Most of the labor and cost of software is in maintenance and support.
Software is almost never fully complete at the time that it first ships to a paying customer. The longer a body of software remains commercially viable, the higher the fraction of total effort that will go into enhancing it, fixing it and supporting it during its lifetime. Changes in third-party code may force changes within your software even if your software is stable in all other respects. New interfaces have to be created to meet evolving requirements. The more corner cutting there was during initial development—for example skimping on feasibility prototypes or on architectural analysis and design—the higher the costs will be during enhancements and maintenance, further increasing the fraction of post-shipment effort and costs.

In an example from a large medical diagnostic device company, a typical software application developed by this company takes approximately three years to develop. An application has about seven years of active life, and then another seven years of legally required maintenance and support.

For this company, a development team numbers about twenty people, so about 60 staff-years are required to create a new application. A team of about 15 people maintain and enhance the software during its seven-year active life; a total of 105 staff years. The final seven years of life require an additional 21 staff-years of effort. In this example 60 staff-years were required to develop the software, while 126 staff-years were required after its first release. This is a ratio of 33 percent to 66 percent. The slightly higher ratio of development to maintenance in this example is a reflection of the level of completion required before shipment in a regulated environment.

Most of the software effort and the majority of the total cost of software came after the first shipment. This is true for most complex product software. This should be a consideration when understanding software costs, especially when estimating costs and schedules of your next generation of software development. It is true across all kinds of applications and all industries, and true of all development methods including agile methods.

Also, on average across industries and products, for a product that has shipped to customers and is being actively enhanced and supported, each full-time development staff person accounts for about 50,000 lines of software source code (using our approach to counting lines of code from Part One of this column series, and accounting for development staff as described in this edition). That is, if your active product has 500,000 lines of source logic by the rules discussed in the previous article, then your company should have about 10 software R&D staff including management. Naturally, there is considerable variation in this number: for example, the number of lines per person tends to be significantly lower for safety-critical software or unusually complex software, and much higher for a product that either is under initial construction or nearing the end of its life.
There are many reasons why a company's software-inventory metrics over time might be very different from these numbers. Counting methods in your organization might differ from other companies. Other common reasons are poor or obsolete architecture, unusually complex or fast-evolving products or basic misunderstanding of the true costs of building and owning software. In the next two articles, we will look at costs and staffing across industries, and provide some metrics for assessing your internal development process.

**Editor's note:** This column is the second installment in a four-part software development series that will appear throughout the year, tackling the topic of what product software costs to build and own. The next two columns will help you evaluate how your company's costs compare to those of your competition.

Check out another installment in this series:
- Installment 1: [How Big is Your Software Inventory? (Part 1 in 4-part series)]
- Installment 3: [How Do Your Costs Compare to Your Competition? (Part 3 in 4-part series)]
- Installment 4: [How Do Your Costs Compare to Your Competition? (Part 4 in 4-part series)]