Purchasing Systems

Alex Sverdlov
alex@theparticle.com

1 Introduction

Purchasing software/system is often a viable alternative to building it. Software development is long and expensive—so when an opportunity to share the cost arrises (such as when a vendor builds one system and distributes the cost across multiple customers), the outcome is generally cheaper for everyone involved—including a profit for the vendor.

Besides cost benefits to such an arrangement, the vendor supplied software is often already there—ready to be used immediately. Building software often takes a long time with a lot of uncertainties. In addition to already being there, vendor supplied systems may be more mature and support industry best practices, as compared to custom build software within corporations.

This obviously applies to software that is common enough to have a vendor. If the function is common enough—often there are vendors ready to supply that market.

2 Make Vs Buy

The make-vs-buy decision involves a surprising amount of work.

First, the detailed business requirements need to be gathered—can't purchase something without knowing what the business needs are. This is similar to the requirements gathering during software development—similarly documented, and just as detailed.

Once the needs are documented, business and technology need to collaborate in reaching out to potential vendors to determine if any available products meet the required needs (or can be customized to meet the needs).

The purchasing cost is often a small portion of the process—from the total cost perspective, the savings is on the 'coding' and potentially the testing phase. But even purchased software needs to be tested, integrated, configured, and learned—training costs will exist with purchased as well as built software.

The vendor needs to be evaluated as well—is the vendor a stable company that will exist when you really need them? Is there a way to ensure that the system is maintained or maintainable past the life of the vendor?

3 The process

The process of purchasing systems or software:

1. Document business requirements
2. Determine whether it is something that is worth purchasing.
3. Establish evaluation criteria
4. Reach out to potential vendors, perhaps by initiating a request-for-proposal (RFP: a document that is distributed to potential vendors inviting them to submit a proposal describing their software package).
5. Short list the vendors/systems
6. Evaluate the available software systems: This may involve presentations by vendors.
7. Evaluate the vendors for reliability
8. Choose package based on evaluation criteria: evaluation criteria often measures the overlap of business needs with what the software package provides.
9. Negotiate contract with the vendor, including items such as:
    - Price of the system.
    - Number of licenses.
    - Time tables (when will the system be ready).
    - Protection of trade secrets (in case logs need to be shared, or the specification for the system contains trade secrets).
    - Maintenance and cost of maintenance.
    - Liabilities due to failure (are there financial penalties if software does not perform as documented).
    - Options and conditions for termination of the agreement.
    - Rights and obligations for future updates and support. This may include a clause to have 3rd party keep the source-code, in case the vendor goes out of business or refuses to support the product.
    - Payment structure: upfront costs, fixed costs, monthly payments, etc.
10. Customizing/Modifying/Integrating the software.
11. Implementing: deploying it to users.

12. Documentation: The vendor will obviously have documentation, but during implementation most companies create their own version of key documentation—to support their own users in their own environment.

13. Retraining: Often users will need training (or re-training) to use the new system.

14. Modifications: the modifications may be vendor dependent, or may be done by the company.

15. Maintenance: who ensures updates and bug fixes get implemented.

16. Operations: who ensures the software is running properly.

4 Open Source

Open Source software is awesome, if you ignore all costs except price of the software. As you can imagine, the actual price of the software is often a tiny portion of the whole cost of ownership.

There are benefits that come with open source: the projects often outlive a single vendor. In fact, the software can often be supported by multiple vendors. The software tends to be "standard" in a sense that finding personnel to work on the system is relatively easy and cheap.

5 Software-as-a-Service: SaaS

Many vendors noticed that the primary business need is often to lower total cost of ownership, and started offering hosted systems. These are configured and managed by the vendor, where the company just connects to use them. Often, payment structure is by user or feature, or time usage, not lump sum.

In such situations, it is critical to have service-level agreement as part of the contract: minimum uptime, speed, etc.

Because of tremendous vendor lock-in with such products, it is also critical to be able to integrate and off-load data from such services.