

WHITE PAPER

SOA directly impacts an ISV's bottom-line

by Jayaprakash Nair

When a software buyer (the customer) evaluates a software product to meet organizational requirements, there are several parameters that will come into play. One of the major factors that would ultimately affect the buying decision is cost. Now, there are four key cost components involved when buying a product from the customer's perspective...



SOA directly impacts an ISV's bottom-line

INTRODUCTION

When a software buyer (the customer) evaluates a software product to meet organizational requirements, there are several parameters that will come into play. One of the major factors that would ultimately affect the buying decision is cost. Now, there are four key cost components involved when buying a product from the customer's perspective (over the product's lifetime). They are:

- ❶ **Product cost:** The up-front cost of buying the product
- ❷ **Product Implementation costs:** The costs of customizing the product (the first time) to suit internal business processes.
- ❸ **Product Integration costs:** Integrating the product with existing technology investments (eg. a CRM product purchased might have to be integrated with the customer's existing ERP system resulting in integration costs for the customer and integration effort for the vendor) and
- ❹ **Maintenance costs:** Maintenance would typically involve either implementation (customization) or integration, and unlike 'b' and 'c' above, which are one-time cost investments, maintenance would be a recurring cost.

So, from the customer's perspective, the Total Cost of Ownership of the product would be the sum total of all the above four cost components.

Now, the higher the TCO for the product, the higher will be the sales barrier for you, as a software provider, as there is a good possibility that another vendor will undercut you on price for a similar product. Thus, a high TCO has a direct negative impact on your bottom-line.

Let's look at the different cost components of a product again: among the four mentioned above, the direct product cost is normally a function of the technical and domain features present in the product. As a vendor, you need to decide whether you want to reduce certain product features in order to reduce product cost. On the other hand, the other costs are directly proportional to

- ❶ The complexity of architecture and design of the product and
- ❷ The flux in the IT ecosystem of the customer's enterprise setup

In the above list, you will probably have very little control over 'II'. But if you can control 'I', then 'b', 'c' and 'd' cost components can be controlled. This in turn would give you more bargaining power for the actual cost of your product and ultimately increase your bottom-line growth.

ISV: So what do you suggest that we do, to reduce the cost of implementation and integration?

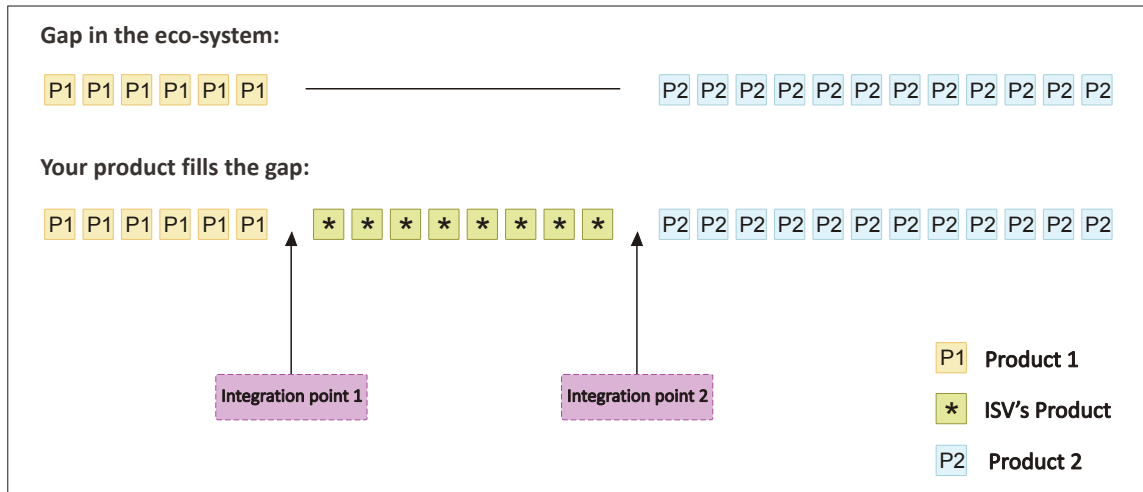
Before talking about the solution, let us delve a little deeper into what is observed in the industry today.

Observation 1: Your customer might have made investments in various 'best of breed' products for its Line-of-business (LOB) functionalities. These are typically monolithic systems tightly coupled together using traditional EAI approaches (and proprietary technologies to glue them together), leading to a heterogeneous and brittle IT eco-system. Any seasoned industry observer will identify this as an omnipresent scenario.



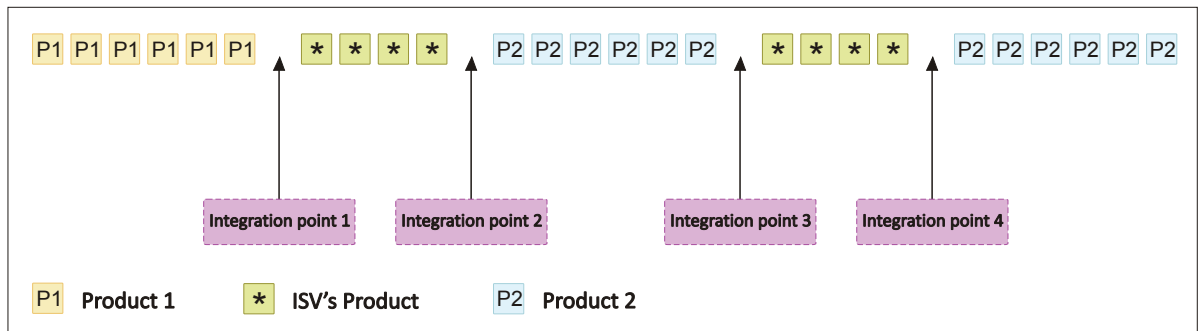
SOA directly impacts an ISV's bottom-line

Observation 2: The major software product companies had hitherto left gaps in their offerings, which were filled by smaller ISVs with their niche offerings. Let us look at this scenario from your point of view. Today, you have a product that caters to a business/technical gap in the IT eco-system.



Having made this observation, the following problems are likely to occur.

Problem 1: In due course of time, the customer may want to extend your product to replace part of the functionality of P2, which belongs to another vendor. Now the functionality that needs to be changed could be anywhere amidst the existing workflow provided by P2. So, you'll need to build more interfaces between your product and P2, to ensure that the flow is seamless.



All this leads to an increasingly fragmented IT ecosystem. You can consider this akin to the continuous fragmentation that happens on the hard-disk because of deletion of some files/folders and addition of new files/folders which do not fit the size of an existing gap. This increased fragmentation leads to an increase in the cross-references between the different storage blocks on the hard disk, which is exactly what happens in a typical enterprise IT setup, leading to a spaghetti type of integration.

Problem 2: The customer may decide to replace product P1 with another product (say P3), by another vendor. There's a good likelihood that you would need to customize your product, or at least repair your integration point with the product, in order to maintain the workflow/dataflow within the enterprise.

Observation 3: The customer might introduce some changes in its business processes/rules, and would expect you to make the necessary changes in your product.



SOA directly impacts an ISV's bottom-line

As you would be well aware, the above observations are, by no means, hypothetical, and such occurrences are expected to increase, going forward. This inference, in turn, means that you will have 2 options:

- ❶ Either continue with the existing monolithic product and keep on increasing the ever-so-brittle integration interfaces with the other products, thus increasing the complexity of the integration/implementation of your product. This would ultimately lead to a negative impact on your bottom line OR:
- ❷ Convert your monolithic product into an SOP (Service Oriented Product)

Out of the above options, the first one is not really a 'solution' and just amounts to maintaining the status quo (or maybe worsening the situation). Whereas the second option is a solution that you can offer to your customer - for that you need to adopt a Service Oriented Architecture for your product. Service-orientation describes an architecture that uses loosely coupled services to support the requirements of business processes and users.

ISV: Hmm... now that you've explained that, can we get to the actual means of reducing the cost of implementation and integration using SOA?

Sure. Let's look at some ways in which SOA can be useful in reducing costs.

- ❶ **Product Integration:** A successfully implemented SOA will reduce the coupling between the different components in the system. **This is possible because of the standards based (typically XML) interfaces that SOA adopts (this manner of using services for integrating the disparate systems, is called Service Oriented Integration, or SOI).** This approach is totally different from the proprietary interfaces used by traditional EAI.

What happens in an SOA scenario is that the different components/products use the same language to communicate with each other, and it becomes easy to continuously realign or 're-integrate' them. This in turn leads to a very 'amoebic' enterprise architecture, i.e. an architecture which can continuously change shape, without any harm to its fundamental constitution.

- ❷ **Product Implementation:** A typical software product solves one or more business problems, and business problems are manifested in business processes. In other words, a product can be effective only if it has a direct relevance to the business processes followed by the company, and most of them are likely to change very frequently. There are 2 problems identified in traditional product implementations:
 - ❶ **Atomicity:** The atomic level of a traditional product is normally a module, which is nothing but a set of closely knitted classes (here we make an assumption that the product has been developed in a disciplined object-oriented manner with the right composition of modules; otherwise, the condition is even worse). Modules are typically significant in size, highly cohesive, and put into production after different levels of rigorous testing. Now if there's a change in any of the business processes contained within a module, it would mean that the module needs to be broken open, possibly gutted and reworked on. It would require multiple analyze-code-test loops for the module to be put back in production. And then, there's always the huge risk of regression errors (and subsequent negative ripple effects) creeping into the module.
 - ❷ **Non-segregation of variable and fixed business processes:** Every business has a set of core business processes which are relatively fixed in nature i.e. they hardly change over a period of



SOA directly impacts an ISV's bottom-line

time. Then there are business processes which change very frequently. Now in a typical traditional product, modules are structured around features, and as such, contain a mix of business processes which are fixed, and the ones which are not. Thus, every time a change is required in a business process, a module needs to be dismantled and reworked on.

Both the above problems can be solved if the product adapts to an SOA. Here's how...**SOA requires that the code implementation of stable business processes be segregated from those which change very frequently. Also, it requires that the atomic unit of cohesiveness be the 'service' and not the module.** This makes the whole product easily customizable, thus reducing product implementation costs.

To sum it up, as the complexity of the IT ecosystem in a company increases (primarily because of heterogeneity) the revenue savings obtained from SOA also increases - provided SOA is implemented in a disciplined manner, with the proper governance in place. And it has been observed that the IT ecosystem is indeed getting more and more complex.

ISV: Alright, I've understood the issues and your explanation that I should seriously consider SOA, but we already have a significantly sized product, which has already been implemented at multiple client locations. Wouldn't the shift to SOA consume too much money and time? Can you suggest a quick-fix that we could start with immediately, for the integration blues that we're facing?

There are 3 approaches that companies typically consider, for SOA adoption

- ❶ **Top-down approach:** Here, the company builds an SOA framework suitable for their requirements (in terms of governance), creates services which cater to specific business processes, and then plugs these new services into the framework. The cons with this approach are the same as what has been seen with the waterfall model of software development.
 - ❶ By the time the framework is built, some of the assumptions/policies of the company's business processes would have changed, and the framework would need to be reworked
 - ❶ It takes a lot of time to actually see any tangible result in the 'SOAfication' of the enterprise.
- ❷ **Bottom-up approach:** In this approach, the company starts at the leaf level, i.e. builds services, and then tries to stitch them together and arrive at a framework. With this approach, the risk is that, being focused on the trees, it is easy to lose sight of the forest. So, instead of an SOA, one could very well end up with JBOWS (Just a Bunch Of Web Services), without a uniform governance framework which is so critical for any SOA implementation to be successful.
- ❸ **Start in the middle and scale out' approach:** Here, SOA is implemented in a cross-section of the enterprise, i.e. in a complete end-to-end LOB chain. A good cross-section of business work-flow is identified as a candidate, the required governance framework is built only for that section, and the corresponding services are designed, built, tested and deployed. This approach overcomes most of the cons of the above 2 approaches.

For you, the third option is the best...In short, start small, identify your main bottlenecks, hit at those places with well governed services, see the results, and then extrapolate.

ISV : OK, I've understood all that you've told me. I am also aware that my professional services team is under a lot of direct pressure from some of my existing customers to get service-oriented. Some of my prospective



SOA directly impacts an ISV's bottom-line

leads are asking me how SOA-compatible my product is. As a result, I would like to start moving my product to SOA. So where do I go from here, what should be my next step?

As a first step, you will need to talk to SOA consultants/vendors and identify those with the required capability.

Here are some of the things that you could keep in mind for narrowing down your list of prospective vendors:

- Check if the vendor has any standing in the ISV space, since that will be crucial for your vendor to understand your stated as well as unstated challenges, and work out solutions to alleviate those.
- Does the vendor have consulting capabilities? Can they identify the cross-section within your product where SOA can be implemented end-to-end with maximum impact? Note that at this stage, your vendor should be focusing mainly on business process, and not too much on the technology.
- Check if that vendor has a solid implementation team, because once the candidate cross-section for (converting to) SOA has been identified, and the architecture is in place, the next step would be to actually implement the solution using services. Here, you'll need to ensure that the vendor has a proven track record in the requisite technologies.
- Does the vendor have proven capabilities in the appropriate software development methodologies (Engineering as well as Project Management) which are apt for SOA implementation? For instance, using a waterfall model for this would be akin to fitting a round peg in a square hole. SOA and waterfall normally don't work well together. What you need is a vendor who has sufficient expertise in Agile (or at least some other Iterative) methodologies.
- Any code that is developed needs to be tested. Now 'SOA Testing' is a different beast. So check if your vendor has that expertise.
- Check if your vendor has a Quality Management System in place, to ensure consistently good quality.
- Needless to say, check if your vendor will do all these at a cost that will not pinch you

One option is to go for an amalgamation of different vendors to meet the above goals, but you then need to be prepared to shell out the extra cost involved in project management, communication management and risk management. Also, which vendor will you hold responsible if there are any issues?

In short, you need to find a vendor who has the ability to provide you with a **cost-effective** and **pragmatic turn-key** solution without disrupting any of your existing business activities. Do not get confused or overwhelmed by the plethora of technology acronyms and other balderdash.

ISV: Thanks for all the interesting inputs. I have a hypothetical question here. Assume that we're looking at being acquired by another company, would such a big investment make sense at this juncture?

That's an interesting point that you bring up. If the due-diligence is already completed, then fine. But if not, then one stands a much greater chance of getting the right valuation by becoming Service Oriented, because of the increasing awareness of the importance of SOA. Most of the companies around the world have realized that SOA is here to stay for quite a while, and it is better to start moving in that direction as early as possible. In fact, we've got reports where the trading partners of companies which have B2B integration with them, are forcing companies in their business eco-system to adopt SOA (at least at the interface points) or else they'll look for other options.

ISV: Now that I'm in the realm of hypothesis, another one...what if my product is in the last stages of its life-cycle?

Then there's a good likelihood that going for SOA might not be the right choice for you, after all.



SOA directly impacts an ISV's bottom-line

The factors that you would need to consider for the decision making would be:

- ❶ When do you intend to stop making new releases?
- ❷ When do you intend to stop going for new customers?
- ❸ What is your SLA/support agreement with your existing customers?
- ❹ A complete overhauling of your architecture may not be feasible. But will the evergreen Pareto Rule be applicable to your product? That is, will converting 20% of your product into SOA reduce 80% of your integration pains and if yes, will the ROI justify the 20% investment?

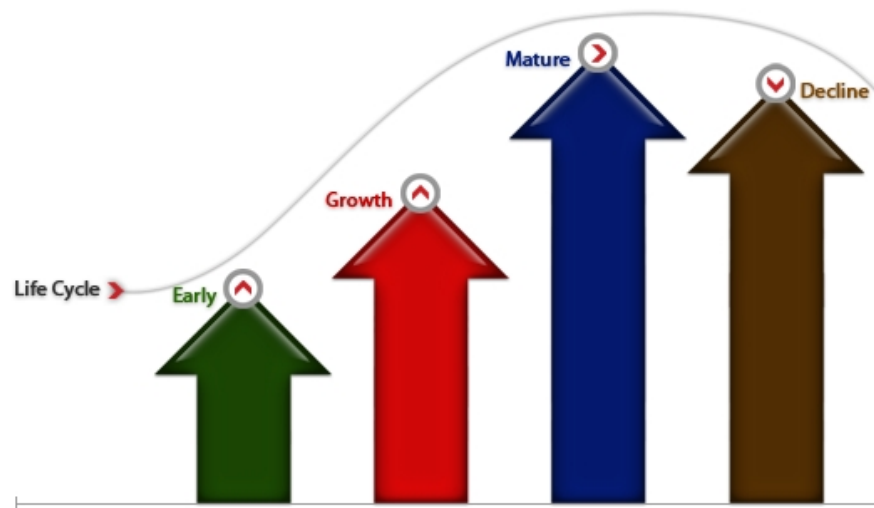
Based on these factors, if your product is really going to be phased out within a short period of time, the applicability of adopting a SOA for your product may not make sense.

However, at this stage, it's a good opportunity for you to invest and redesign the product. A new product that encompasses the existing functionality and more can incorporate SOA right from the start and that could prove to be a huge advantage to your product and bottom-line.

To sum it up, a typical product goes through the following stages:

product that encompasses the existing functionality and more can incorporate SOA right from the start and that could prove to be a huge advantage to your product and bottom-line.

To sum it up, a typical product goes through the following stages:



Out of the above, the probability of SOA being applicable and beneficial is very high in all the stages except for the last one, where the applicability needs to be contextually ascertained.



SOA directly impacts an ISV's bottom-line

ABOUT ASPIRE SYSTEMS

Aspire Systems is an Outsourced Product Development firm committed to helping our customers build software products better and faster. We work with some of the world's most innovative Independent Software Vendors and software-enabled businesses, ranging from start-ups to established industry leaders, transforming the way software is built.

Aspire provides complete product lifecycle services, ranging from new product development and product advancement to product migration, re-engineering, sustenance and support. Our product development teams are spread between our Global Innovation Center in Chennai, India and offices in the United States.



Aspire Systems India Private Limited
Plot No 1/D-1, SIPCOT IT PARK, Siruseri, Tamil Nadu - 603 103
Tel : +91-44-67404000. Fax: +91-44-67404234
E-mail : info@aspiresys.com
Web: www.aspiresys.com