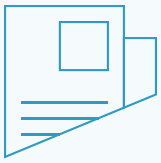

WHITE PAPER

Developer Ecosystems: Driving Innovation and Value with Third-Party Software

AppDirect

Developer Ecosystems: Driving Innovation and Value with Third-Party Software

WHITE PAPER

Introduction

As companies search for cost-efficient ways to increase the value of their software, an increasing number are exploring how to drive innovation by supporting and growing developer ecosystems.

Building an ecosystem of third-party developers that provide complementary services can deliver impressive results, including faster innovation and higher revenue. However, there are a number of different factors that companies need to consider before attempting to create one. This white paper explores the three general types of ecosystems and important operational aspects of each, such as integration, sales enablement, and more.

It also explores reseller enablement and management, as well as the innovation advantage that companies can get from building developer ecosystems. This includes, for example, how data generated by ecosystems can inform and improve technology build out or acquisition strategies.

Developer ecosystems are multi-sided and involve numerous participants, which can make them incredibly complex. However, a clear understanding of how these ecosystems work can make it far easier to create one for your own core products.

KEY TAKEAWAYS

- There are three types of developer ecosystems: add-on, sell-with, and commodity, each with their own unique benefits and challenges.
- Ease of participation in ecosystems is critical for driving growth.
- Making it easy for developers to sell their products in an ecosystem is essential to increasing revenue.
- To be successful, ecosystem builders need to consider a range of factors, including testing, pricing flexibility, submission control, and lifecycle management, among others.

The Three Types of Developer Ecosystem

Today, advanced technology is allowing companies to experiment with various ecosystem types, but most fall into three general categories:

Commodity: In this type, the services you are offering are basic building blocks that many developers will need to build out their own applications; for example, databases, caching services, or error reporting tools. These are "commodities" because you are offering functional tools, with defined shapes and sizes, which developers can take off the shelf and incorporate into their products.

Add-on: Here, you have existing products that could benefit from additional features and functions. Instead of incurring the cost and risk associated with building these capabilities yourself, you want to enable third-party developers to build these add-ons to expand your functional footprint. Generally these add-ons will have a smaller functional footprint than your core product and will provide enhancements.

Sell-with: In this type, you have an existing product that actually sits in an ecosystem of other products that complement your core offering. You want to be able to offer your customers these complementary products so you can offer comprehensive solutions to meet their needs. For example, your CRM system works well with a number of analytics and email marketing tools, and you want to package these as complete, easy-to-buy solutions.

A Detailed Look at Add-On Ecosystems

You have built a product platform and gained customer traction. Now, there are many features and functions you want to build in order to support and delight your customers. The challenge is that you only have a fraction of the resources you need to develop this long list of capabilities.

To fill the gap, an increasing number of companies are looking to enable third-party developers to create these features and functions. With this strategy, third-party developers take on the risk but also earn the monetary reward when the add-ons they develop provide customers with a more complete product.

An ecosystem of add-on solutions can help you increase existing customer usage by unlocking new use cases for your products. It can also drive retention, since customers are more likely to continue using your product as its feature set becomes more complete when supplemented with add-ons. In addition, this can hinder competitors who are trying to take advantage of weaknesses in your products and win customers away from you by closing the gaps in your product. Finally, you will be able to sell more product, because you can offer the market a more competitive feature set.

Success in this strategy usually hinges directly on your ability to attract developers to build add-ons for your product. Three key components will determine your success in attracting developers:

- Market opportunity: How big is total addressable market for your product?
- Ease of participation: How hard is it to build, submit, and begin selling add-ons?
- Ease of sales: How easy is it for developers to sell and promote their products on the platform?

There is a very simple logical expression that can define whether you will succeed at garnering developer attention: **P + S < M**. That is, if it's easier for a developer to build a product (that is, **P**articipate) and it's easier to start **S**elling and be promoted with your core service (the left side), and together, that effort is less than the capture of the total addressable **M**arket that a developer can reasonably expect (on the right side), you will be successful.

In other words, if you are a near monopoly or duopoly of a large market, then no matter how hard you make it on developers (not an ideal situation, obviously) they will participate because the monetary incentives for success are so high. However, assuming you aren't Google Play, Apple's App Store, or the Salesforce.com AppExchange—in other words, a well-established, popular marketplace—at least not yet, you will need to focus on reducing the left side of the equation while you grow the right.

Brokerage—technology that makes it easy to provision, consume, and manage cloud services—plays a big part in the left hand side of the equation by making it easy to participate in an ecosystem and making it easy to sell. The next sections explore these factors, ease of participation and ease of sales, as well as other tools you can use to reduce friction, that, while not related to brokerage, are still highly correlated to success.

A FORMULA FOR SUCCESS

P + S < M

If ease of Participation and Selling within an ecosystem is less than the capture of the total addressable Market that a developer can reasonably expect on their own, that ecosystem will be successful.

Ease of Participation in the Ecosystem

In order to be successful, you need to design and support an ecosystem that makes it easy for developers to participate. When sign up, on-boarding, marketing, pricing, and other essential steps are simple and straightforward, developers will flock to your ecosystem and drive more revenue, both for themselves and for you.

From the start, a developer program should be designed to reduce friction wherever possible. This means making it as easy as possible for developers to get test accounts and API access, as well as crystal clear documentation with samples. Generally speaking, you want to make these critical elements as self-service

as possible; developers are busy, and your ability to grab and hold their attention will be limited. If you make it hard to participate in your ecosystem, you will try developers' patience and lose them.

From the start, a developer program should be designed to reduce friction wherever possible... If you make it hard to participate in your ecosystem, you will try developers' patience and lose them.

You must ensure that your process for developer on-boarding is as good as the sign-up experience for the program as a whole. Some friction most likely won't drive developers away after they've built an add-on product, but the quality of the experience will dictate whether they build the next add-on, especially if the first product doesn't sell as well as they hope.

Overall, the on-boarding experience should be self-service and well-documented, just like the development experience. However, there are some additional key considerations:

On-Boarding: What steps do developers need to take to on-board their products? Is integration work required for web services, and if so, how are integrations tested? For downloadable products, what does the upload mechanism look like?

Testing: A full sandbox for testing is critical for both you and the developer. You want the developer to test their software as much as possible before they submit it to increase the quality of the add-ons you see. If you can make testing a mandatory step before submission, it is a huge, time-saving bonus.

Marketing: How does a developer create a marketing profile? How easy is it to edit and customize the profile to make it attractive and meet the needs of the developer?

Pricing: Can you enable the pricing and flexibility developers need to offer plans that make sense for their products?

Submission Control: You're going to need back-end tools to manage all of the add-ons submitted by developers. You should have the ability to see which developers have requested to publish, as well as a sandbox for you to review and test each submission.

Lifecycle Management: You need to provide dashboards to enable developers to see how they are selling and who their customers are. You also need to provide tools for developers to manage their product listings, integrations, and uploads on an on-going basis. This will allow them to continuously update their product as they improve it.

Ease of Sales in the Ecosystem

Once your developer program is up and running, what does a developer do next? This is where cloud service brokerage comes into play. Brokerage offers your ecosystem the simplest route to market, giving your end users a single place to find the add-on products they need to make your product the perfect solution for them.

A critical part of the brokerage process is enabling customers to find and purchase products. A marketplace that provides reviews, product profiles, and feature information is a great way to do this. To be most effective, the site should be categorized by relevant function, user persona, and other major categories, but searching for software is the only beginning. Once customers find solutions they want to buy, how do they transact?

Enabling customers to transact on the marketplace is essential. However, to make this process as seamless as possible, you need to be able to accept payment on behalf of the developer as well.

Enabling customers to transact on the marketplace is essential. However, to make this process as seamless as possible, you need to be able to accept payment on behalf of the developer as well. That way, you'll not only provide a better customer experience, but also get an opportunity to earn margin on the sale. If you don't allow customers to transact, developers will have to set up their own payment acceptance methods—and all of the infrastructure that goes along with it—which creates unacceptable levels of friction.

Even better, you should offer your customers a way to sign in with credentials so they can access payment methods on file (e.g., a credit card) and easily purchase additional items. This not only dramatically reduces friction for customers and developers alike, but it also makes your core service stickier by making it an anchor for a constellation of services that deliver more value to your customers.

Another way to drive additional marketplace sales is to enable cross-product bundling. This capability will allow you to create and merchandise "starter" or promotional packages based on user persona or function. You can use these bundles to increase sales, and also lower the amount of effort customers have to put in to find and consume products that will benefit them.

In addition, your sales team can play a key role in marketplace transactions. Enabling sales to provision services for your customers allows you to treat your developer ecosystem as a catalog that significantly extends your product footprint. Your sales team will figure out which tools help them close more deals, and your account managers can use these tools to help engage your existing customer base to save at-risk customers.

Finally, you need to consider how you track these sales so you can compensate your sales team, and most importantly, pay the developers. To do this, you need

to keep close tabs on what was sold to whom, as well as the revenue shares owed to each party. This can be particularly challenging if you do not have a uniform revenue share rate across your ecosystem.

Sell-With Ecosystems: Biggest Risk, Biggest Reward

In the sell-with ecosystem model, you have an existing product which actually sits in an ecosystem of other products that complement your core offering. By offering these complementary products, both yours and those developed by third parties, you can provide complete solutions to meet your customers' business needs. Here's a concrete example: You offer a CRM system that works well with a number of analytics and email marketing tools, and you want to package these as complete solutions for your customers.

Tackling a sell-with ecosystem should be considered expert level partnering with developers in the ecosystem and the platform provider you may be working with.

This scenario presents many of the same merchandising and sales challenges as the add-on ecosystem model does, only bigger. That is, cloud service commerce aspects like billing, provisioning, and identity become even more complex in this environment. In fact, tackling a sell-with ecosystem should be considered expert level partnering with developers in the ecosystem and the platform provider you may be working with.

Overall, integration will also be significantly more complex, so you should carefully consider how you will provision products for your customers, and then how you will facilitate the integrations between your product and partner products. If the products in the ecosystem are not integrated, the combined offerings will not provide enough of a value-add to make them a good buy for your customers.

Billing is, of course, another critical element and billing complexity in a sell-with ecosystem can be daunting. Your partnership arrangements may differ between partners—one may be a referral-only partner while another may get a revenue share—and you will need to track and reconcile every transaction, no matter what type, across all of your partners.

Going further, if customers can purchase products with their existing identities, do you give them access to the partner product with their existing credentials (via single sign-on)? Does this happen from within your existing user portal, such as Google Apps, or does the user have a separate entry point? Enabling your sales channels with the proper configuration will also be vital, but how can you accomplish this across the ecosystem?

Clearly, there are many questions to consider. However, when done correctly, a sell-with ecosystem can drive significant value and revenue for your core product.

The Unique Challenges of Ecosystems and Commodity Services

Commodity services present a special case. Like the "commodity" name implies, it can be very difficult to build these types of ecosystems because the products are just that, commodities, and they don't have the sales pull-through effects of other ecosystems. For example, if you are a provider like SendGrid, one of the best email delivery services on the market, you have a large developer community that uses your service.

However, email delivery can be considered a commodity service; in other words, it's a well-defined common building block that is widely used across applications. One of the only things those customers will have in common is that they use SendGrid, but unless the products directly enhance or improve SendGrid, the diversity of the customer base means they are just as likely to find the right solution via a Google search.

Instead of trying to build an ecosystem around a commodity service, it's a better idea to place this type of software into an ecosystem with complimentary services that act as a collection point for like-minded target customers. For example, SendGrid does this by participating in the marketplaces of IBM, Cloud Foundry, and Red Hat, among many others. These markets attach to service offerings where developers go to build software, making them excellent locations to capture future customers. For SendGrid and other companies like it, commodity plays can still provide tremendous uplift for their products, if placed where the right customer base can find them.

Conclusion

Clearly, a common thread that runs through this white paper is that each type of developer ecosystem can be incredibly complex to build and requires advanced capabilities, including billing, provisioning, identity and access management, and more. The costs, measured in both time and money, of building and integrating all of these functionalities into one platform can add up fast.

While building and supporting a developer ecosystem can deliver a range of benefits, from increased revenue, to higher customer stickiness and satisfaction, to more overall product innovation, the decision to pursue this strategy depends on each company's unique needs and goals. With careful planning and strong execution, any company with a cloud-based software offering can benefit from this rapidly emerging trend in the global cloud service commerce industry.

AppDirect: The Industry-Leader in Enabling Developer Ecosystems

AppDirect has deep expertise and experience helping companies launch developer ecosystems quickly and cost effectively. AppDirect can provide all of the necessary features and functionalities—including billing, identity, provisioning, and more—with a single platform. Learn more by visiting our website at www.appdirect.com or emailing info@appdirect.com.

ABOUT APPDIRECT

AppDirect is the leader in cloud service commerce making software accessible globally. The AppDirect Cloud Service Commerce Platform unites providers, developers and consumers of cloud services into a single ecosystem. This makes it easy for businesses to find, buy, and manage cloud services from a central location and delivers new opportunities to distribute, sell, and market cloud services.

AppDirect-powered marketplaces, billing and distribution, and reselling services help providers—including Telstra, ADP, Samsung, Deutsche Telekom, Cloud Foundry, Rackspace, and others—connect millions of businesses to solutions from Google, Box, DocuSign, Intel Security, and more.

For more information contact
info@appdirect.com
or visit **www.appdirect.com**.

650 California Street, 25th Floor
San Francisco, CA 94108
(877) 404-2777