S-Docs
Simple. Smart. Secure.

TECHNICAL WHITEPAPER

A Guide to Evaluating Salesforce AppExchange Apps

There are thousands of apps available today on the Salesforce AppExchange. This paper will help you assess security, speed and scalability differences between different app architectures. It will help you identify which apps share your Salesforce data as they interoperate with Salesforce, and provide you with some key questions to ask vendors during your evaluation process.

Overview

Since 2005, the AppExchange has provided a store for Salesforce customers to find, install, use and share experiences on thousands of apps that can add real value for your business. While the benefits of an app are touted, the risks associated with implementation are often omitted or left unclear. Most listings on the AppExchange give little technical detail on the App construct or how it interoperates with the Salesforce platform and your customer data. Thes e details are critical to understanding performance, security, adoption, compatibility and total cost.

"app listings use phrases like "native integration" or "without ever leaving Salesforce". This can mislead you on how these apps use and share your Salesforce data.

Vendors are not making your research efforts any easier. They are constantly updating their listings by adding terms like "Native", "on-platform" or "Intelligent" to convey a sense of security, ease of use, or leveraging the latest technologies. The vagueness of some technical terms lends them to being exploited. For example, some AppExchange listings use phrases like "native integration" or "without ever leaving Salesforce". This can mislead you on how these apps use and share your Salesforce data. Most probably, the latter phrase is referring to the user experience of "not leaving" the Salesforce user interface, but this does not apply to your data.

This whitepaper examines how most apps on the AppExchange are constructed, the tradeoffs of differing architectures and key questions you should ask vendors during your evaluation. It is intended as a guide to help you understand the benefits and risks of various app architectures.

First mover disadvantage

Today, we universally assume speed to market is a benefit, where laggards continue to be at a disadvantage. We will explain why this is not the case with Salesforce apps.

In 2005, Salesforce introduced the AppExchange that allowed installation of third party apps to help expand Salesforce capabilities and interconnectivity. At that time, Salesforce did not have the broad feature set they boast today. The force.com platform would not be available for another four years. This meant that apps originally listed on the AppExchange during the early years were built on other clouds and used Salesforce APIs to communicate with Salesforce. Some continue to use antiquated protocols like SOAP more than a decade later.

In 2009, force.com was introduced as Salesforce's Platform as a Service (PaaS) and this dramatically changed the ecosystem. Developers could now build a Native App, meaning built and run entirely on the Saleforce cloud. For existing vendors who already had a product built with existing customers and many year's worth data on their existing stack, the cost to convert to force.com was not advisable.

Most of these early vendors continued on their current stack. But many of their customers soon found they were using yesterday's interface and technology as Salesforce continued to rapidly evolve. Despite this, some of these first movers continued to thrive based on market share and brand, not technology.

"While being native is not a panacea, investigating native solutions is strongly advised before you seek out non-native apps."

Summary of Native Benefits and Disadvantage

For solutions entering the market after 2009, Native meant a new way to deliver functionality within Salesforce by leveraging the customers' existing investment in Salesforce. Here are some very important benefits:

- Data Security: Your customer data doesn't leave Salesforce
- Data Residency: Meets EU and other regulatory requirements
- IT Compliance: Builds on 'already approved' infrastructure
- Speed: No latency due to network traffic between clouds
- Scalability: Platform elasticity matches Salesforce
- Ease of use: Uses Salesforce UX/UI and existing admin tools
- Cost: Infrastructure already paid for with Salesforce licenses
- Uptime/Reliability: Availability matches Salesforce
- Data Residency: Meets stringent citizen data laws
- Integration: Does not count against your API usage, simplifies authentication and reporting integration
- Future Proof: New Salesforce features must work with existing stack

While building on platform certainly has enormous advantages, it does not work for all cases. There are use cases when off-platform solutions are often more suitable.

- Working with files greater than 30MB or containing video
- App functionality requires file system, or OS-level access
- Apps require streaming data
- Heavy use of external (non-Salesforce) data
- Computation intensive applications
- Portability: Interfacing App with other CRM systems

While being native is not a panacea, investigating native solutions is strongly advised before you seek out non-native apps.

Native Salesforce App Defined

Today, even the term “on Salesforce” can be confusing because it no longer refers to a unified platform but encompasses recently acquired technologies that are now operating under the Salesforce corporate umbrella. Products like Sales Cloud, Service Cloud, force.com and Communities all share the same infrastructure or stack. This is often referred to as the “Core Stack”. That core stack contains a complete set of components needed to run an instance of Salesforce including the database, application servers, load balancers, cache servers, and search servers. This is what we should consider “on platform”. However, Salesforce acquired technologies like ExactTarget and Heroku that are not on the core stack. Nonetheless, these are often marketed as “on Salesforce”, which can be misleading. Some technologies like Salesforce CPQ are marketed as Native, which they mostly are, but they still use some off-platform components for services like processing calculations or document generation.

“A Native Salesforce app is one that resides entirely within the confines of the existing Salesforce Core Stack.”

For our purposes, a Native Salesforce app is one that resides entirely within the confines of the existing Salesforce Core Stack. It is built in same Salesforce development environment and uses the same commercially available development tools that are available to all Salesforce clients to customize their own instance of Salesforce. It does not rely on any external systems, third-party web services or browser plug-ins - even if those services are owned by Salesforce.com. All web pages must served from the force.com domain and all code is compiled and executed on Salesforce core stack for an app to be truly Native.

A Native app consists of a code base called a “managed package”, which is made available on the Salesforce AppExchange. Packages are installs into an existing Salesforce instance. Once installed, a Native app should be considered part of Salesforce in the same way your own customizations become part of Salesforce. This should be akin to how Chatter is considered part of Salesforce.

S-Docs is 100% native to Salesforce.

Non-Native Salesforce App Defined

A non-native Salesforce app does not live 100% within the Salesforce platform. Elements of the application may be within Salesforce and could be considered a “hybrid” app, however, the app itself may be hosted elsewhere (such as Amazon) and data may leave and enter the platform and be stored on an outside, third party server. Be mindful that some vendors state that “your data is never stored externally”. This claim is made because your data is processed in-memory and then flush from that memory cache. This does not change the fact that your customer data has changed possession.

Details of Native App Benefits

EASE OF USE

- Since Native app are exclusively written for Salesforce on Salesforce, they better leverage existing Salesforce user experience. For example, S-Docs uses the same styles, picklists, grids, icons, labels and taxonomy as your existing Salesforce UI. This makes S-Docs more intuitive to Salesforce users rather requiring them to use a generic interface that was designed for a variety of CRM vendors.

SECURITY

- All client data stays within Salesforce (unless you download, export or email it). It is not sent nor stored on external servers nor does data enter from external servers, which may be the case with a non-native application.
- A native Salesforce application conforms with the same security settings and sharing rules that you have already created within Salesforce. With a non-native app, these rules and settings may not be applied, as a result, data could be vulnerable. Hybrid apps may well be secure, however, they may not necessarily be using Salesforce security and sharing settings.
- Integrations with a Native App are more secure. For example, API interactions with any Native App services require authentication with Salesforce. This ensures the highest security, and minimizes the need for specially built integrations. For example, a single-sign-on (SSO) integration requires no additional consideration to work with a native app.
- All apps on the Salesforce AppExchange must pass the Force.com ISV Security Review prior to being published on the AppExchange. https://developer.salesforce.com/page/Security_Review

DATA RESIDENCY

- Since all data continues to be exclusively stored on existing Salesforce servers, Native apps inherently meet the strictest government data residency requirements – just like Salesforce.

PERFORMANCE / SCALABILITY

- In general, Native apps will exhibit better execution performance because all processing occur on the same platform. Performance is augmented by the elimination of having multiple calls to send and receive data from external services.
- All Native apps must adhere to Salesforce's execution and governor limits. https://developer.salesforce.com/docs/atlas.en-us.apexcode.meta/apexcode/apex_gov_limits.htm These limits apply to code created on the Salesforce platform and help enforce rules that prevent runaway processes, use of unsupported methods or executions that monopolize resources. These governors ensure that S-Docs code is efficient and follows best practices.
- Native Apps do not count against your Salesforce orgs daily API limits. By contrast, non-native apps may require multiple API calls per invocation.

These limits need to be considered especially when other apps are also sharing in those limits.

- Native apps exhibit the same benefits of scalability as the Salesforce platform. So, adding thousands of users to your system, or generating thousands of documents is all handled by the existing force.com infrastructure.

RELIBILITY

- Since native apps reside 100% within Salesforce they will always be up and running when Salesforce is running. Non-native and hybrid apps depend on outside servers and network connectivity which may or may not have the same Enterprise-class infrastructure as Salesforce. Downtime may not only cause loss to a business, but often requires time to reconcile failed requests.
- As a native app, S-Docs contains all test classes and test code coverage as required by Salesforce. Comprehensive testing ensures that the app runs as designed and continues to run uninterrupted. https://developer.salesforce.com/blogs/developer-relations/2012/11/how-code-coverage-works.html

ADMINISTRATION

- Since a native app is part of Salesforce, it is much easier to implement, and maintain. S-Docs exclusively uses existing Salesforce admin setup menus for administration.
- User provisioning is handled through standard Salesforce setup screens. Native apps do not require a separate procedure, login or service to manage user accounts. Native apps do not require setup of account tokens.
- Access within the Native app is also managed through standard Salesforce permissions and record sharing rules.

CUSTOMIZATION

- Native apps can be customized to your needs by adding new fields, workflow rules, validation rules, triggers etc. Your customizations are not affected by subsequent upgrades of Salesforce or S-Docs.

INTEGRATION

- All Salesforce objects inherit APIs including REST APIs. This allows for an easy integration with external systems.
- Native apps are even easier to integrate with your existing Apex code. Calls to global services are available that make custom integration straightforward.

COST

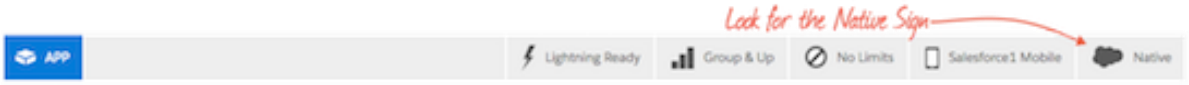
- A Native app leverages the world-class Salesforce platform, which is already paid for with your existing Salesforce licenses. Therefore, you do not pay for twice for infrastructure.
- Non-native vendors usually charge an additional fees for API/Batch Processing capabilities or each workflow event because it taxes their infrastructure. Since S-Docs is part of the Salesforce infrastructure, paid for by your existing user licenses, there is no added cost for these capabilities with S-Docs.

FUTURE COMPATIBILITY

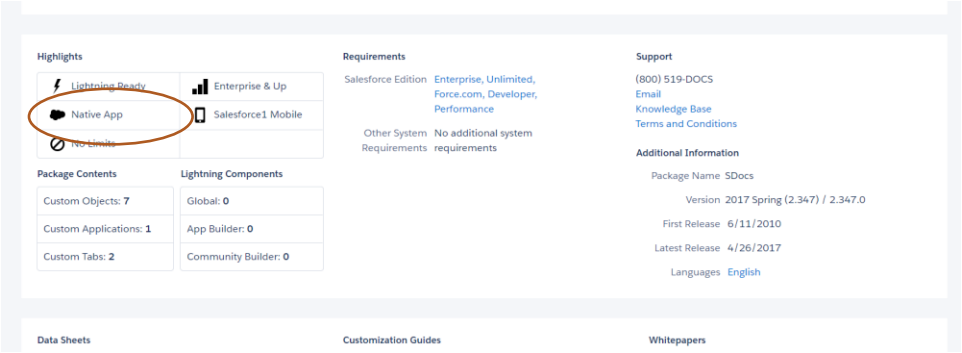
- As Salesforce continues to innovate and release new features and functionality. It is critically important to Salesforce that customizations previously made not only continue to work, but can easily take advantage of new capabilities. Native apps are uniquely positioned alongside your own Salesforce customizations because they are done using the same technologies. Non-native apps need to constantly retrofit their product with each Salesforce release to prevent brakeage whenever features are added, changed or decommissioned.

How to identify a Native Salesforce App

All Native applications on the Salesforce AppExchange are listed with a graphic on the page that denotes their Native status.

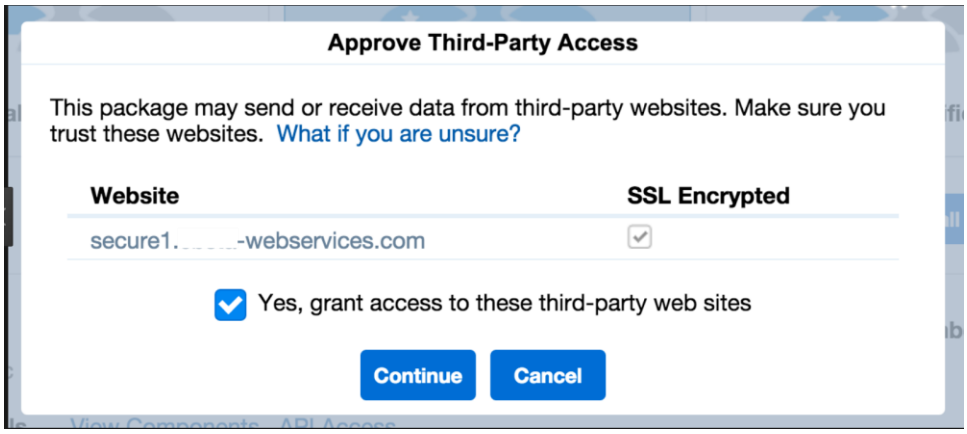


Likewise, in the AppExchange listing only Native Apps *should* be listed as such in the Highlights table. We have found that this does not always hold true because this attribute is self-reported by vendors.



The true test happens during the installation of your App. If a message similar to the one below appears, it indicates that the App you are installing is NOT

native. This message is letting you know that this App is specifically requesting that you allow Salesforce to communicate with an external service.



Key Questions to Ask during your evaluation of a AppExchange Application:

1) Does your app require that I whitelist any IP addresses in Salesforce to use any feature of your app?
2) Are there ANY features of your App that requires a connection from a Salesforce domain you a web service or URI?
3) Where are your service physically hosted? Is it regional or distributed and how does that affect my data residency requirements?
4) How are your encrypting my data between your service and salesforce?
5) Is my salesforce data visible in your debug logs? Is it plain text? How is debugging accomplished when needed? Who has access to those logs?
6) How are you separating your corporate network from your services?
7) What are your SLAs for service and performance regarding uptime, bandwidth and latency?
8) What is the communication process for downtime and breaches? Can you provide a history of those for the previous 24 months?
9) When was the last time you implemented your emergency management procedures?
10) What audits and certifications do you have?

S-DOCS COMPANY OVERVIEW

Founded in 2010, the S-Docs team is led by a former Director and Technical Architect from Salesforce and is comprised of experts in Salesforce and document solutions. S-Docs is used by thousands of global subscribers from all industries with a multitude of use cases, but they all share one key theme: The need for a powerful yet easy-to-use native solution. We are proud that they have made us the #1 native document generator and put us in the top 1% of all apps on the Salesforce AppExchange.