Four Elastic iPaaS Requirements That Must Not Be Ignored

A SNAPLOGIC WHITEPAPER

snapLogic

Table of Contents

Introduction

The concept of integration platform as a service (iPaaS), as defined by Gartner, Forrester, Ovum and other analyst firms, has gained significant awareness for enterprise IT organizations facing a new set of challenges posed by rapid cloud application, API and big data platform adoption. Once you recognize the need for a modern integration strategy to manage the social, mobile, analytics and cloud computing requirements of today's Elastic Enterprise, some of the iPaaS requirements to consider include:

- Metadata-driven integrations
- Drag-and-drop user experience
- Pre-built connectivity (no coding necessary)
- Management and monitoring that includes comprehensive error management
- Transactional support
- SOAP and REST API support
- Data transformation and other operations
- Hybrid deployment model

There are two themes driving the urgency of iPaaS in the Elastic Enterprise:

1. **"Cloudification"**: Cloud expansion has hit a tipping point and most IT organizations are either running to keep up or trying to get ahead of the pace of this transformation.
2. **Agility**: Business users' need for speed stemming from mobile, social and software as a service (SaaS) expectations.

As a result, four new requirements for Elastic iPaaS have arisen that must not be ignored:

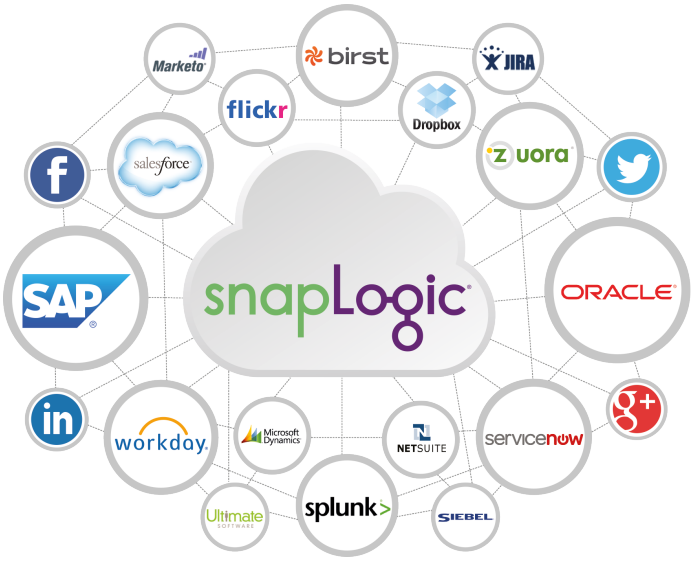
1. Resiliency
2. Hybrid deployment fluidity
3. Minimal lifecycle management of the platform
4. Future-proofing for the world of social, mobile, analytics, cloud and the Internet of Things (SMACT)

This whitepaper addresses these new requirements in detail while highlighting the importance of each.

Elastic iPaaS Requirement #1: Resiliency

Changing business requirements has always been the norm. But expectations of IT responding to these changes in real-time is a much more recent development. More often than not, these changes results in data changes that end up impacting the integration layer. The most common changes on the application, data and API side are additive, where a new column is added to the table or a field to an API to record or deliver additional information.

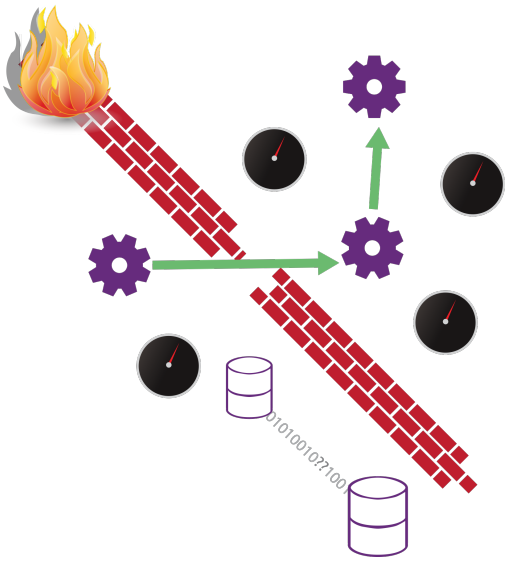
iPaaS technologies built in the last decade are strongly typed. This means the integration developer must define the exact data structures that will be passing through the integration flow while designing them. Any departure from this structure results in the integration layer breaking down. An example of this would be to run a simple file-to-file integration using a product built in the 2000s, then editing the source file by adding additional unexpected fields to it and run the flow again. The integration will fail because it cannot recognize the additional fields. This brittle integration can bring a business to its knees. An iPaaS should be resilient enough to handle such updates and variations in stride, and the right solution will include a data validation step specifically for that requirement.



A schema-less iPaaS is the best approach to building dynamic integrations. With no rigid dependency on a pre-defined schema, the integrations can flexibly continue to run even when endpoint data definitions evolve and change over time. This approach also maximizes pipeline reuse across uses cases that share common endpoints. Additionally, there are tremendous cost-saving and business risk containment benefits that arise from avoiding downtimes

resulting from changing API and data structures. A resilient, elastic iPaaS will keep a business running, even when unexpected changes occur. A high degree of integration resilience leads to greater business agility.

Elastic iPaaS Requirement #2: Hybrid Deployment Fluidity



Similar to data structure changes being a common occurrence and resulting in the requirement for resiliency, the introduction or retirement of applications is a common occurrence for IT organizations. Software as a service (SaaS) applications continue to fuel worldwide software growth, while infrastructure as a service (IaaS) and platform as a service (PaaS) providers such as Amazon Web Services offer customers the flexibility to build up systems like relational data services (RDS) or Redshift and tear them down in short cycles. Change is the only constant in the Elastic Enterprise and the impact of these changes on the integration layer is that it is expected to seamlessly transition from connecting on-premises systems to cloud systems or vice versa while still ensuring a high degree of business continuity.

Integration technologies built in the last decade were not built for the fluidity needed in today's Elastic Enterprise. Even though many legacy integration solution providers offer a dual deployment model – one on-premises and one in the cloud – they are typically not peers when it comes to management, monitoring and configuration (not to mention development

functionality). Here are some common issues that customers will face with these so-called "hybrid" integration technologies:

- Deployment in the cloud is not the same as deploying the on-premises engine. The provider may need a different set of preparatory and configuration requirements between the two environments. For example, setting up configurations that were typically local like connection pooling, or even providing driver locations with their on-premises products, still must be done locally and manually for each of the runtime installations. This can be a very manual, cumbersome and error-prone approach.
- Dual management and monitoring dashboards – one for the on-premises execution engine and the other for the cloud engine – means an administrator needs to manually stay on top of two environments. This is both time-consuming and risky.
- The on-premises engine was built for connecting on-premises systems and often requires many network ports to be open for communication in order to receive data processing instructions or send back monitoring information to the server. If the monitoring and metadata servers are running in the cloud, customers often will be requested to punch holes in their network firewall in order for all of the iPaaS functionality to work.

A truly modern, elastic iPaaS solution must deliver fluidity in a hybrid deployment. Specifically, two things must be considered while making an iPaaS purchasing decision:

1. **When looking at a cloud integration solution from a legacy vendor, look under the hood to ensure the vendor isn't simply *"cloud-washing"* its on-prem product by hosting it in the cloud.** Even if the same code base is hosted in the cloud and inside the firewall and can be deployed to with a drop-down option, there will be management, monitoring and configuration issues. The management and monitoring of these two runtimes will need separate consoles as they lack the ability to communicate to the monitoring server across the firewall. Additionally, on-premises runtimes were designed to have configuration files that are not centrally managed but rather local to the installation. Managing and configuring such hybrid environments becomes a

recurring cost with every software upgrade and can add up significantly.

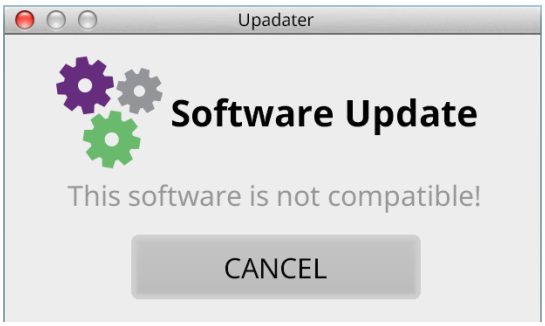
2. **Don't fall for the map trap of "map once and run everywhere."** There isn't sufficient value gained when mappings built once can run everywhere, because mappings are typically very specific to the sources and targets that are being integrated. Most times, they are not transferrable from on-premise to the cloud as on-premises endpoints - including Oracle ERP, SAP and Teradata – are typically very different from cloud endpoints, such as Salesforce, Workday and Redshift. This renders the "run everywhere" story quite ineffective. Other issues with this approach include:
    - It is really masking the reality that the "anywhere" implies a variety of different products being made to appear similar.
    - This set of distinct products implies management and monitoring headaches.
    - Mappings are a one-time cost; therefore, reusability doesn't provide much and the management and monitoring costs end up being more significant.

Because of the above challenges, a software-defined elastic iPaaS has become a key enabler of *"enterprise IT cloudification."* A high degree of hybrid integration fluidity results in a higher degree of business continuity.

Elastic iPaaS Requirement #3: Minimal Lifecycle Management

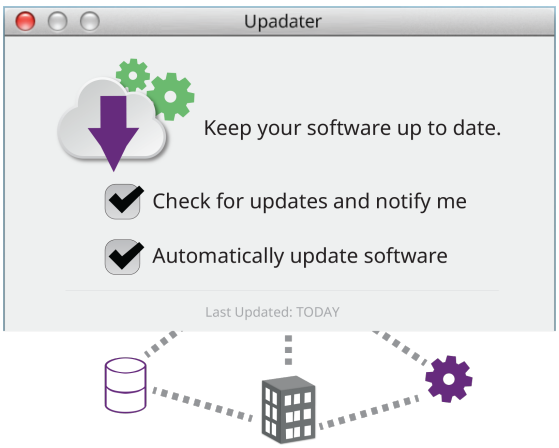
With increasingly hybrid integration deployments, lifecycle management can very quickly become a nightmare for users of legacy enterprise service bus (ESB) and extraction, transformation and loading (ETL) technologies. Upgrading on-premises integration software, such as the core product libraries, typically means binary updates for every installation across hybrid environments. While each vendor is different, there are still a surprising amount of cloud integration installations that are simply hosted on-premises software and not truly multitenant SaaS. Nevertheless, the more challenging upgrades are the on-premises installations that are customer-managed; and there are many enterprise customers still running old, unsupported versions of integration software because of the fear of upgrades and the mindset of *"if it ain't broke, don't fix it."*

Cumbersome manual upgrades of on-premises integration installations are error-prone and result in significant testing cycles and downtime. The bigger the implementation, the bigger the upgrade challenge – and connector libraries can be equally painful. Local configuration changes and the need to rebuild mappings also demand thorough testing cycles.



SaaS customers are accustomed to interacting with complex business processes, such as opportunity-to-order management in a CRM application, through a simple web interface. Therefore, the bar set for the modern integration platform is higher where these customers expect the vendor to shield them from as much complexity as possible. There is a similar expectation with the management of the lifecycle of the iPaaS. Here are a few examples of new lifecycle management requirements:

1. **Customers want zero desktop installations.** Customers want to move away from integrated development environments (IDEs) that are highly developer-centric and require their own upgrades. Instead customers want browser-based designers for building integrations with automatic access to the latest and great functionality.
2. **Customers expect the installation of the runtime engine to be self-upgrading.** This is particularly important for the on-premises installations to avoid cumbersome, error-prone tasks and endless testing cycles. Today's iPaaS runtime engines should be smart enough to push binary upgrades to every runtime engine, regardless of its location - either on-premises or in the cloud. This is particularly efficient with a software-defined integration architecture because each of the runtime engines are stateless containers awaiting execution instructions from the control plane.
3. **Customers expect the execution instructions to also include connector libraries and configuration information.** This means that a customer no longer needs to worry about manual configuration changes at every installation location.



A truly modern, elastic iPaaS will deliver on all of the above and deliver an integration solution that eliminates much of the pain of traditional lifecycle management. The cost and risk of not having self-upgrading software is an order of magnitude in today's age of agile software

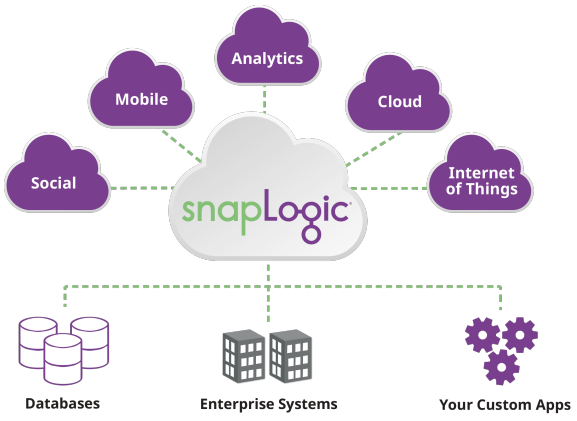
snapLogic

delivery. Benefits to this approach include the assurance that customers are always on the latest platform and automatically keep up with the innovation that vendors deliver. Customers no longer have to plan long and costly upgrade cycles that are typically associated with infrastructure downtime and hinder business continuity. Lastly, the biggest benefit is that the integration platform is built to run at cloud speed.

Elastic iPaaS Requirement #4: Future-Proofing for Social, Mobile, Analytics, Cloud and Internet of Things (SMACT)

Even if an organization has been traditionally conservative and is currently only at the assessment stage around the value of a cloud-centric IT infrastructure, it is likely that business stakeholders will soon expect the IT organization to handle social data and enable mobile channels. And for those companies that sell products with hardware components, the "Internet of Things" (IoT) will be a pressing need, if it isn't already.

It can be daunting to consider the volumes, velocity and variety – as well as the need to harvest, integrate and analyze data - in the world of Social, Mobile, Analytics, Cloud and the Internet of Things (SMACT). Today, many IT organizations are using massively parallel or big data technologies such as Hadoop and Amazon Redshift to build enterprise data hubs or data lakes. These technologies are ideal for writing sophisticated algorithms to analyze data either for problem detection or historical and predictive trends. Most times, this data is aggregated from a variety of sources – including business transaction data sources such as Oracle databases or Salesforce.com, and activity data such as website click stream data. However, enthusiasm for transitioning to these big data technologies can sometimes result in a lack of attention to the key prerequisite of getting that data into these systems in the first place, causing the "Integrator's Dilemma" that is crippling many enterprise IT organizations.



In order to be able to handle the new SMCAT data and API requirements seamlessly, an elastic iPaaS needs to deliver scale without slowing data initiatives down. It needs to deliver elastic scale that expands and contracts its compute capacity to handle variable workloads while

pumping data into analytics infrastructure. A cloud integration platform also needs to move data in a lightweight format and add minimal overhead; JSON is regarded as that compact format of choice when compared with last generation formats such as XML. Lastly, an elastic iPaaS should be able to handle REST-based streaming APIs to continuously feed into a data analytics platform. Without these new approaches to iPaaS, organizations are setting themselves up for falling short on big data and SMACT initiatives.

With a modern, elastic iPaaS, the benefits for these initiatives are:

1. Only one platform is needed. Instead of one enterprise application integration (EAI) and one ETL platform, an elastic iPaaS allows for all connectivity and orchestration styles and patterns. As a result, there are fewer software and hardware costs, and fewer skilled resources needed for maintenance, and developers and administrators can double up to build and manage ETL as well as EAI workloads.
2. With elastic iPaaS scale, there is no need to plan and provision resources around black swan events of traffic spikes.
3. A more compact REST + JSON-based software-defined architecture will optimize hardware resources and seamlessly deliver on the growing mobile requirements of a business.

Summary

In addition to the classic data and application integration requirements, these four new requirements are driving the demand for elastic iPaaS in the enterprise. To summarize, legacy integration tools and technologies typically have the following characteristics: metadata-driven, drag and drop, pre-built connectors, management and monitoring, transaction support, SOAP and REST APIs, transformation and cloud deployment. In addition to these functionalities however, the modern, elastic iPaaS also requires resiliency, hybrid architecture fluidity, lifecycle management and future-proofing social, mobile, analytics, cloud and IoT initiatives.

To learn more about the SnapLogic Elastic Integration Platform visit www.SnapLogic.com.