

DataFlow - The SnapLogic Architecture and Ecosystem

White Paper

March 2010



SnapLogic, Inc.
28 East 3rd Avenue, Suite 300
San Mateo, California 94401

(650) 525-3540

Data Volumes are Exploding

The amount of data a business has to access and manage today is exploding not only in volume, but in complexity and heterogeneity of formats and sources as well. The proliferation of mission critical software applications and services, combined with the requirement for better, almost universal access makes the difficult problem of data integration bigger and more complex than ever before.

Technology

The data deluge

Businesses, governments and society are only starting to tap its vast potential

Feb 25th 2010 | From *The Economist* print edition



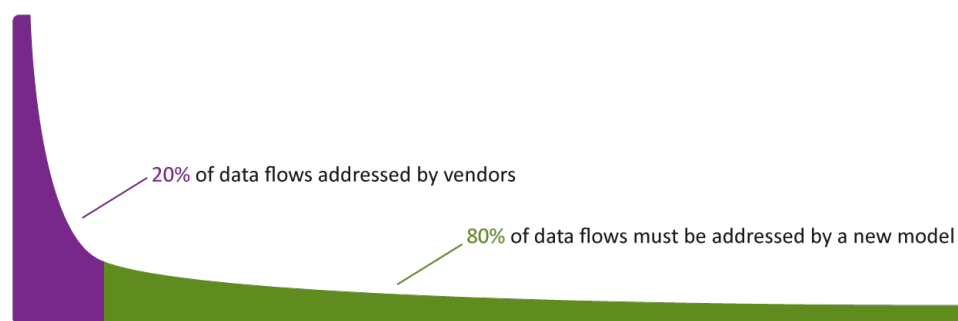
“...According to estimates, mankind created 150 exabytes (billion gigabytes) of data in 2005. This year, it will create 1,200 exabytes. Merely keeping up with this flood, and storing the bits that might be useful, is difficult enough. Analysing it, to spot patterns and extract useful information, is harder still. Even so, the data deluge is already starting to transform business, government, science and everyday life...”

Not only is the quantity of data produced projected for exponential growth, but the major sources of that data are projected to shift from predominantly enterprise sources to cloud and hosted SaaS applications and the web itself.

A key problem with this shift is that unlike the historical growth pattern of more and more data from a relatively consolidated, and therefore limited, number of leading enterprise applications, the projected growth in SaaS and web data sources comes from a much larger *number* of rapidly proliferating SaaS, cloud and web applications as well as data streams such as those from rapidly proliferating social networks and syndicated feeds such as RSS and Atom. This rapid proliferation of many and varied data sources is creating a “Long Tail” of data sources that need to be integrated into everything from enterprise applications and databases to rich internet applications (RIAs), enterprise mashups, and both business and market analytical tools.

This “Long Tail” creates a second, even more critical issue for those desiring to integrate these important, widely diverse data sources. Gartner estimates that today, even with all the “traditional” data integration solutions available from vendors for implementing EAI, EII, ETL and the newer approaches of ESB’s, SOA’s and other MOM solutions, 80% of the \$10B integration market still consists of hand coded, point-to-point solutions.

As the rapid proliferation of new data sources floods organizations with needs for even more data connections, this need for a better solution than current integration products or hand coded patchwork solutions will only grow. If vendors’ current business models cannot keep up with anything more than 20% of the integration demand, it’s clear that a new model is required to address what is an ever increasing long tail.



Obviously, this ever increasing demand for connectivity to an increasingly diverse universe of data sources can't be addressed by conventional architectures and business models. A new approach is required.

Alphabet Soup Doesn't Solve the Problem

The alphabet soup of ETL, EAI, EII, approaches to data integration architectures evolved to solve a specific information management challenge, and just as quickly became rigidly defined based on the need to solve a specific type of data integration as a monolithic, proprietary and purpose-built integration solution.

However, the kinds of integrations that developers spend most of their time on today generally don't fit within the confines of a conventional ETL, EAI, or EII solution.¹ This is because they require integrating data with a wide variety of endpoints including files, spreadsheets, reports, public websites, cloud-based SaaS applications and services as well as web services, social media and the like.

The availability of easy-to-use dynamic languages like Perl, Python, PHP, and Ruby has exacerbated this bespoke approach. Unfortunately, this approach also results in code that is fragile, hard to maintain, not reusable, and not easily extensible as APIs and data requirements change. This is the classic iceberg scenario where the cost of development is just the visible tip of the iceberg -- the cost of maintenance is the menacing amount of ice below the water and is often overlooked.



¹ ETL: Extract, Transform, and Load. EAI: Enterprise Application Integration. EII: Enterprise Information Integration.

As a rapidly increasing portion of important business data either originates or migrates outside an organization, a greater portion of developers' time is spent on these integrations. And, as the number of data feeds and the volume of web based application data grows exponentially, the challenge of integrating, filtering, modifying and analyzing this data multiplies exponentially as well.

Unfortunately, hand-coded integration solutions have high hidden costs due to their single-use focus and fragility as well as their lack of documentation and metadata support. The irony of the easy-to-use programming methodologies is that they – like the ice below the waterline -- mask the growth of the underlying problem of integrating an exponentially growing and diverse collection of data sources.

Clearly, this situation is unsustainable: The problem grows exponentially as requirements accelerate, while the solutions scale linearly, and IT staff and budget continue to shrink. To solve the integration problem in an era of modern IT architectures, a fundamentally new approach is needed.

That approach must address several key issues:

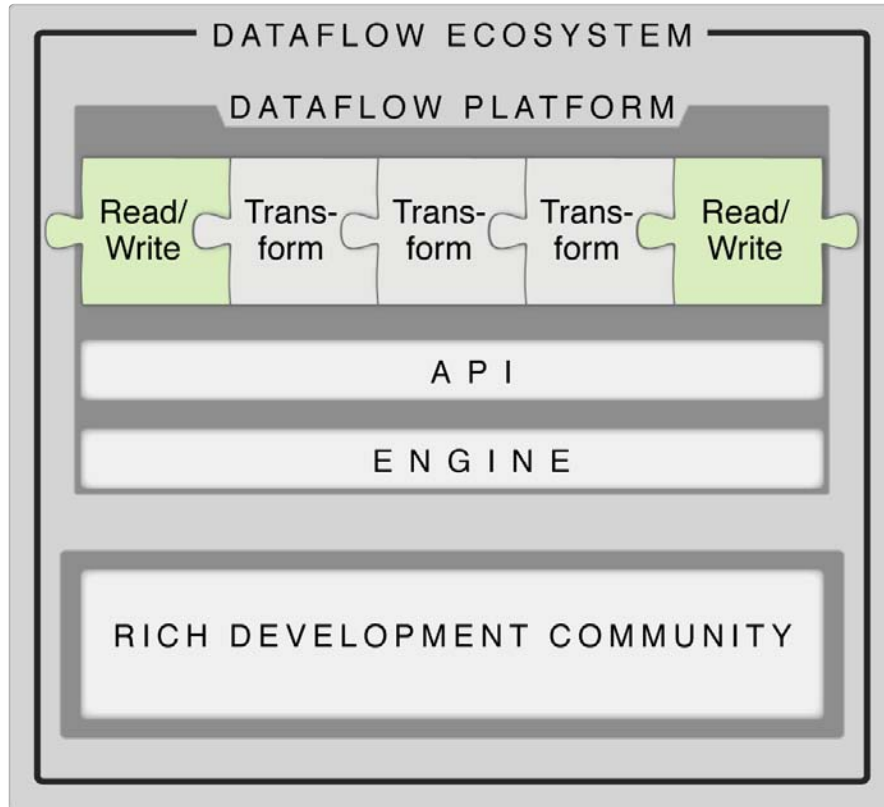
- Simple, consistent and reusable integration modules that can be written once and reused in infinite combinations to satisfy the exponential growth of the combinations of data sources that they address.
- A simple, powerful and infinitely extensible server framework that can be distributed across multiple instances and yet supports, manages and configures the needed combinations of integration modules to allow for both infinite connectivity and infinite combinations of data operations between endpoints while not imposing the overhead of any unneeded functionality.
- A simple internal architecture for both server and modules that maximizes ease of use, ease of interconnection, robustness and security.
- An open architecture that allows developers to create, share and sell reusable components, and a store and ecosystem that enables development and reuse of connectors to meet the exploding demand.

DataFlow is the Answer

The data deluge and mushrooming data sources that organizations now need to handle requires an entirely new approach to data integration:

- A new Architecture that allows data to flow from any number of sources to any number of destinations via a highly adaptable and extensible platform that provides:
 - The flexibility to go beyond the rigid, proprietary solutions that have focused on point-to-point data integrations and the core concepts of ETL (sic.) to enable not only data transfers and transformations between data stores and applications, but true data flows between applications and data streams originating from not only other applications, but the rich web, RSS, ATOM and social media data feeds as well.
 - A “less is more” framework that allows developers to include only the connections, transformations and data functions needed to enable the data flows required without imposing any additional overhead.
 - The ability to distribute multiple integration components to precisely where they are needed - be it behind the firewall, in the cloud or across multiple environments for performance and virtualization.
- A new business model that lowers integration costs and enables re-use and propagation of solutions
 - A subscription-based pricing model with multiple tiers that makes it possible to pay for only the level of product and support that you need.
 - A developer ecosystem that enables the development, sharing and re-use of integration components to optimize efficiency and make pre-built solutions readily and easily available through a “clean well-lighted” store where they can be shared and monetized.

The SnapLogic DataFlow architecture and the developer ecosystem created by the SnapStore address all of these needs and solves the problem of the data explosion by changing the paradigm for data integration.



Value of SnapLogic Dataflow Architecture - Simplicity

According to a recent Netcraft survey, as of January 2010, the Web has grown to over 270,000,000 active sites and nearly 160,000,000 hostnames.² Your browser knows nothing about any of the sites you might visit, and you don't need any prior knowledge about what the site contains or how it is laid out or structured. Yet you have no problem seeing it all. How does it do that?

The answer is: **Simplicity**

² See http://news.netcraft.com/archives/2010/01/07/january_2010_web_server_survey.html

The Web is a collection of loosely coupled servers that rely on simple, stateless interactions between client and server. Standard access protocols (TCP/HTTP), stateless access methods (GET/PUT/POST/DELETE) and simple data formats (HTML) make it possible for any client (e.g., a browser) to render any Web page. Technically, this is known as a REpresentational State Transfer, or REST architecture.³ Adopting the REST architectural style is gaining popularity and is sometimes referred to as a Resource-Oriented Architecture, or ROA.

The complexity of the nearly unlimited combinations of protocols, methods and schemas required for traditional data integration creates a morass of integration protocols, access methods and data schemas that results in massive investments that lack the four fundamentals required.

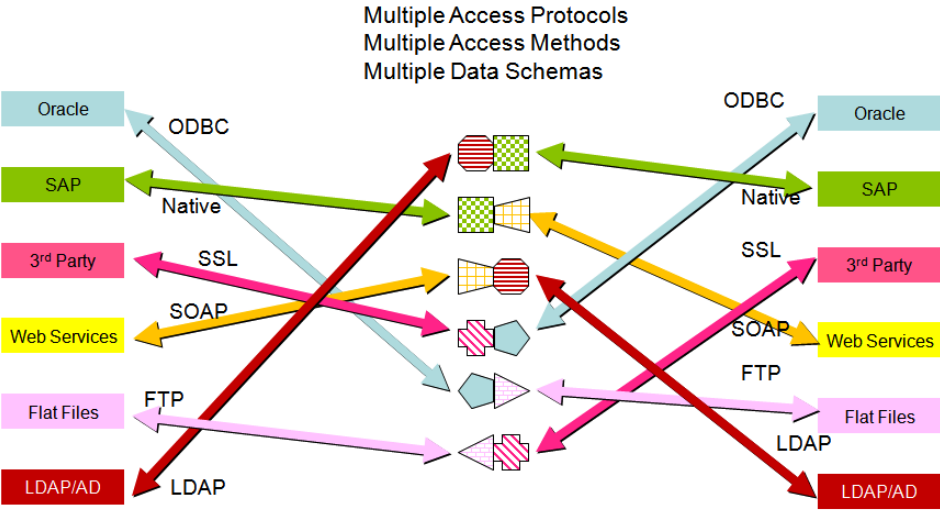


Figure 1. Traditional data integration faces three dimensions of complexity: protocols, methods, and data schemas.

Is it any wonder that data integration solutions eventually collapse under their own weight? The ongoing maintenance required to support application revisions on different platforms can eventually consume all available development resources. Often, for commercial integration solutions, the complexity of the connectivity matrix expands to the point where core product innovation suffers.

³ A canonical description of REST can be found in Roy Fielding's dissertation from UC, Irvine, *Architectural Styles and the Design of Network-based Software Architectures*. See <http://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm>

While other interoperability standards such as SOAP-based Web services⁴ exist they primarily target the requirement for application integration and often require a messaging infrastructure like an ESB. This makes current application integration techniques poorly suited for the solving the new data integration challenges presented by SaaS, cloud and web based data sources.

In contrast, simple standards like RSS and ATOM have taken off like wildfire because of their focus on solving a specific problem with a simple Web-compatible (i.e. REST) design. Historically simple solutions stacks (e.g. TCP/IP) have almost always beat out ornate ones (e.g. OSI 7 layer model).

Integrate like the Web

The loosely coupled nature of Web servers, combined with simple transport and content structure ensures every new site can be added while maintaining seamless interoperability.

Data integrations should be just as simple. Every data source should simply attach to the network where it can issue requests for the data it needs.

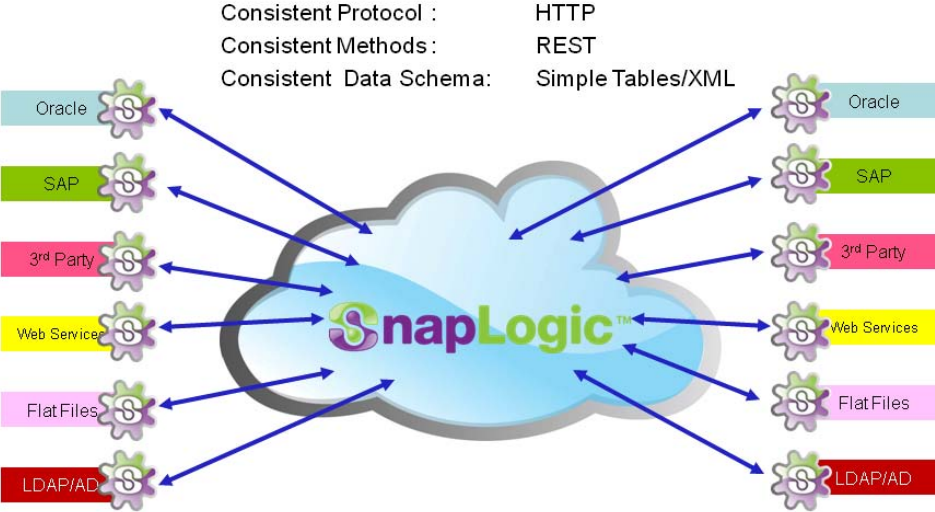


Figure 2. Applying RESTful Web interfaces to data integration simplifies the complexity of interconnecting disparate data sources.

⁴ Service-Oriented Architecture (SOA) standards including SOAP, WSDL, UDDI and others.

For that to occur, a new approach to data integration is required, one that embraces the architecture of the Web itself: freely flowing integration capabilities built into a server infrastructure that supports Web-style (i.e. RESTful) interactions for data and metadata, while supporting the execution of complex transformation logic.

This new infrastructure layer can support a new data services layer providing RESTful access, transport and transformation services for data integration. A resource-oriented data services layer simplifies integration the way the Web simplified document exchange. Every data source can easily be added to the network, creating a new Web of integration-enabled endpoints between which data can easily flow freely. This model is highly analogous to internal and external websites where *web pages* are served up by *web servers*, except in this model *data* is served up by internal and external *data servers*.

Connectivity Is Not Enough

The unique requirements of every integration, whether it is Analytic Data Integration for data warehouses and Business Intelligence, Operational Data Integration for migration, or synchronization and routine business processes, make some measure of customization a necessity and resource-oriented data services while providing a pragmatic, powerful model alone cannot address these needs.

The true potential of data services is realized when integration components can be reused for future requirements and when the data can be augmented easily by the addition of components for further inline processing.

Adopting Internet standards like HTTP and a REST provide the technology foundation for interoperability as well as the ability to take advantage of the wide range of infrastructure technologies built for the Web. Inexpensive -- sometimes even commoditized -- caching, load balancing, security, access control, logging, as well as search indexing all make this web-based approach to data integration significantly cheaper and powerful than traditional approaches.

Additionally, when these deployment technologies are combined with an open development model and a store for sharing and reselling snaps developed by an ecosystem of application and API experts, sharing and reuse multiplies exponentially as well. When a community of developers collaborate, share and re-use each other's components, and deploy them on a common infrastructure, technology and methodology reinforce one another to realize the full potential of resource-oriented data services.

SnapLogic DataFlow

SnapLogic provides a data integration platform and ecosystem that addresses the challenges of integration in a fresh new way. Like the Web itself, SnapLogic standardizes the access protocols, access methods and data structure for every interaction. These RESTful interactions allow data to flow between servers in an organized manner that makes it easy for the data to be processed and interpreted as a simple, consistent flow.

Start Small, Scale-up

The SnapLogic server is a lightweight process that serves as the container and execution environment for configurable Components called Snaps. A catalog of standard Snaps provide base integration capabilities such as database queries, file read and write, aggregate, sort, filter, join and others.

Snaps are configured with specific properties and instantiated as Resources which are accessible via unique Uniform Resource Identifiers (URIs) e.g. `http://dataserver.port.number`. Resource definitions are stored in a SnapLogic server's Repository.

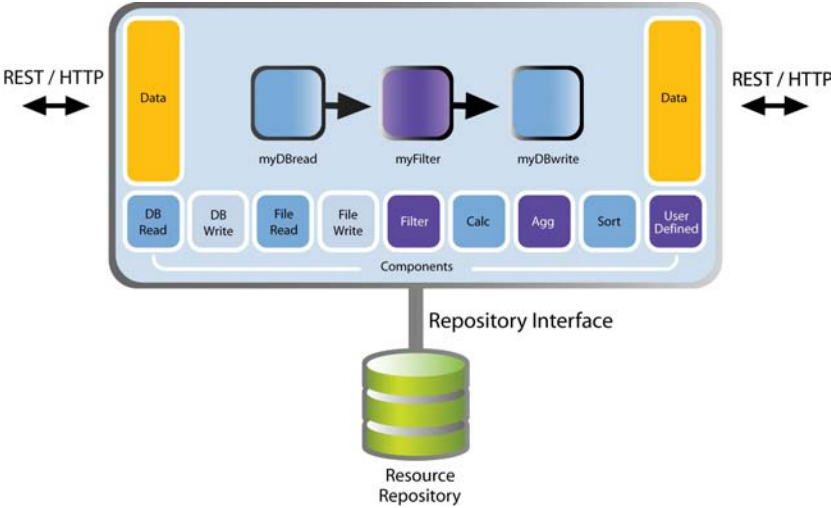


Figure 3. Architecture of a SnapLogic Server, including a repository of Components, Resources, and Pipelines. A Resource, such as myDBRead, is a generic Component, such as DBRead, configured for use with a particular application.

Data sources accessed by SnapLogic Resources present the data in a simple easily understood record-oriented format that encapsulates and hides the complexity of application-specific data schemas. This enables Resources to interoperate more easily, facilitates reuse and provides a simple, lucid and scalable flow of data between resources.

Resources can be linked to other Resources through their REST interfaces to become transformation Pipelines. Pipelines can be assembled into hierarchies to implement complex logic and transform data for sophisticated integrations or to comply with data governance.

With its “snap-together” Resources and RESTful interfaces, SnapLogic enables a Lego™ building-block approach to data integration that facilitates component re-use and allows integrators to use only the resources needed without any additional overhead. Enterprise users can quickly and easily leverage the work of their colleagues by sharing Snaps within their organization. Independent developers can leverage their domain expertise with specific APIs and data structures to create custom Snaps for sale in the SnapStore

Metadata Repository of Reusable Components

In integration change is the only constant, and hand-coded solutions are nearly impossible to repurpose. As requirements grow and personnel turn over, hand-coded integrations inevitably become an unruly, un-catalogued, and un-maintainable assortment of scripts and custom applications. One developer may write in Perl, another in PHP. Their scripts will likely have no common interfaces. Other than by word of mouth, one developer will likely have no way of knowing what another developer has written. Because the scripts are written quickly, they typically lack comments. Some of them may be inscrutable to anyone other than their author. In a sense, these scripts end up as functional silos, as difficult to leverage as the data silos they're attempting to integrate.

SnapLogic addresses this problem by including a searchable repository for Components, Resources, and Pipelines in every SnapLogic server. This enables developers to quickly find and use prior work by themselves and by colleagues. Instead of amassing a disjointed collection of scripts, developers, using SnapLogic, can create a growing body of reusable components, which benefit from testing and standardization, and which accelerate future development.

Search and Indexing

Since each SnapLogic Resource and Pipeline is accessible at a URI, standard Internet search and indexing techniques can be applied to find which Resources are available across multiple Repositories where ever they reside. Integration reuse is like information; it's only useful if you can find it.

Programmatic and Graphical Interfaces

SnapLogic supports both a programmatic interface as well as a browser-based graphical design tool, SnapLogic Designer. Using Designer's drag-and-drop interface, IT workers and business analysts can assemble Pipelines without programming.

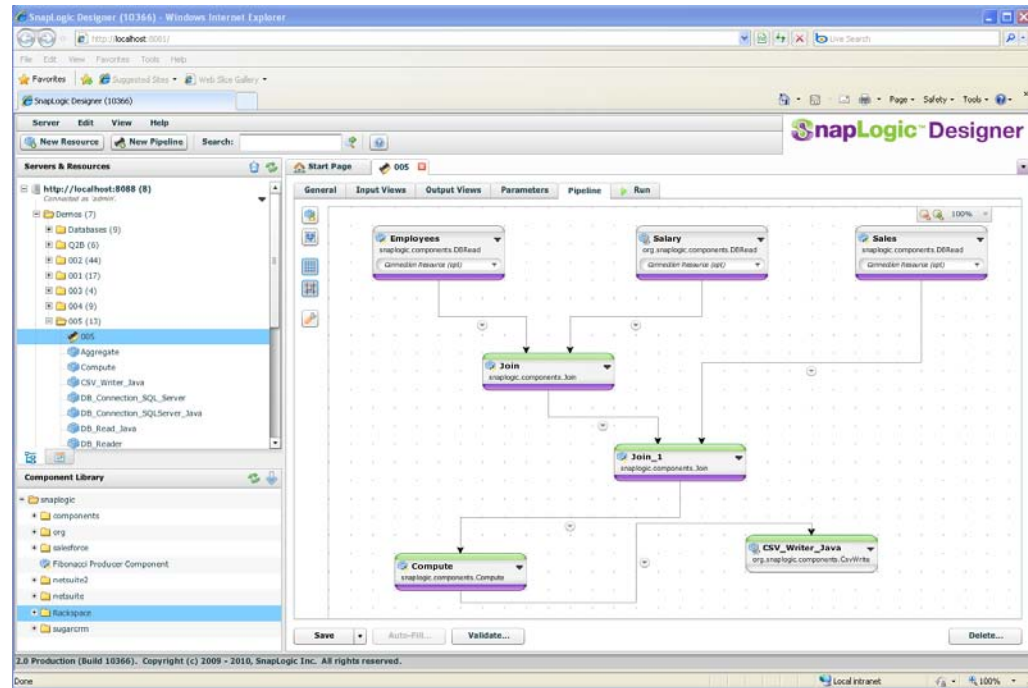
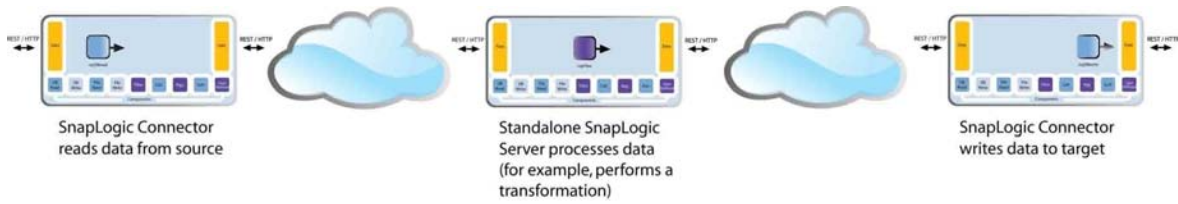


Figure 4. SnapLogic Designer's drag-and-drop interface makes it easy to assemble and configure Pipelines. Built-in search enables users to discover and access re-usable Resources and Pipelines.

SnapLogic's programmatic interface allows programmers to create new Resources, Snaps and Pipelines and manipulate them in the programming environment of their choice. The programmatic interface provides all the benefits of dynamic programming languages, within a structured framework for metadata and execution control. Together, these two design techniques provide both the novice as well as the expert the choice of whatever approach is best for them.

Distributed Execution and Snap-enabled Data Sources

SnapLogic has a built-in Web (HTTP) server that allows Resources to link across servers so that Pipelines can be partitioned to execute arbitrarily across data sources, intermediate transformation servers, or in any manner that is appropriate. SnapLogic uses an innovative record-streaming technique so that Resources can process individual records without waiting for entire data sets to become available.



Example of a Distributed SnapLogic Pipeline

Figure 5. Pipeline execution can be distributed across servers.

At any point along the way, data can be accessed by any client that requires it. Support for JSON output enables AJAX-based RIAs to directly consume SnapLogic data. A client library supports direct access from Java, PHP and Python applications. SnapLogic Resources and Pipelines can be accessed as live data feeds by any technology that can call a URI.

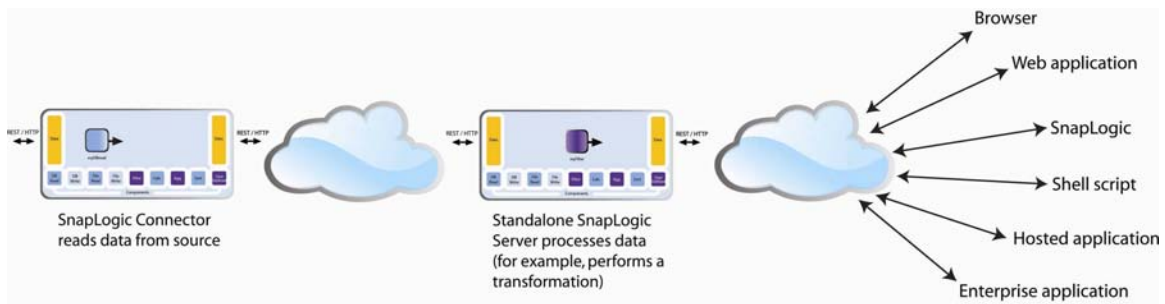


Figure 6. A SnapLogic data feed can be accessed directly by applications as varied as browsers, shell scripts, hosted applications, and enterprise applications.

By migrating SnapLogic to source and target servers, they become Snap-enabled endpoints supporting controlled access for all data in the business. As more data sources become Snap-enabled, a complete resource oriented data integration infrastructure layer emerges.

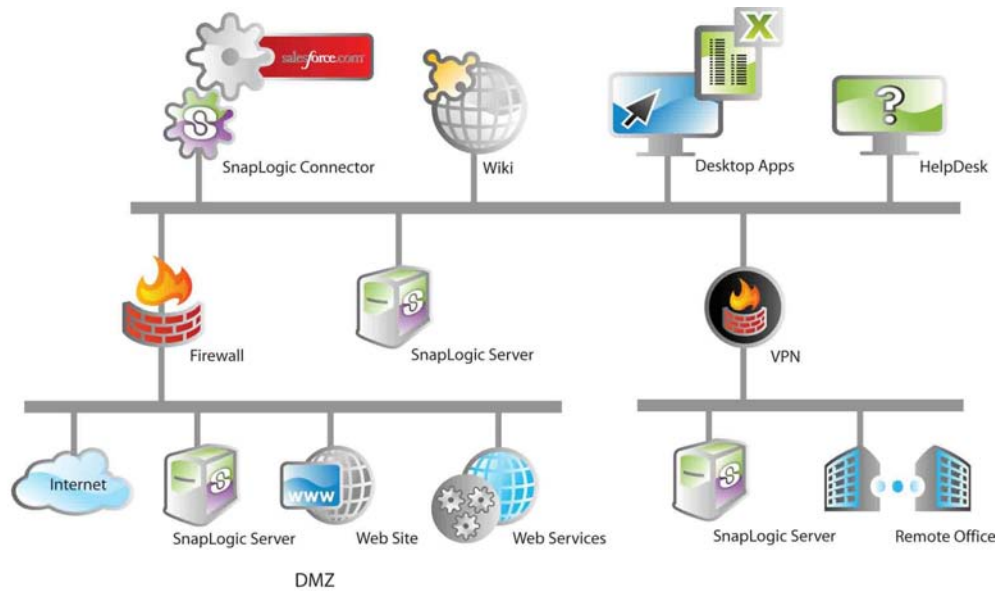


Figure 7. SnapLogic Connectors and Services can be distributed across an enterprise network. In the DMZ, they can provide integration services for public data sources. In remote offices, they allow Lines of Business to tailor applications and mashups to suit their particular analytical or operational needs.

The benefits of the data integration infrastructure layer provided by SnapLogic accelerate as more data sources become Snap-enabled and isolated networks connect with each other across firewalls and enterprises. Their Repositories can be searched for useful integration Resources so that new integrations can be assembling faster, and more reliably than any hand-coded, or any other Last Mile integration technique.

Table 1 summarizes the features and benefits of the SnapLogic framework.

Table 1: SnapLogic Summary

Feature	Benefits
RESTful interface	<ul style="list-style-type: none"> • Leverages existing consumer Web investment in hardware, software, and training • Supports enterprise security controls • Simplifies development through loose coupling • Scales across every enterprise
Data access from any source: enterprise applications, hosted applications, SOA, databases, data warehouses, MDM/CDI, Web sites, RSS, etc.	<ul style="list-style-type: none"> • Eliminates needs for costly data extracts, duplicate databases, time-consuming custom development, etc. • Provides a consistent framework for data integration (ubiquitous access to data)
Searchable distributed repositories of reusable components	<ul style="list-style-type: none"> • Accelerates development • Supports best practices, security policies, etc. • Increases ROI • Reduces errors
Index-able distributed metadata and browser-based design tools	<ul style="list-style-type: none"> • Supports self-service by IT engineers and business analysts (ubiquitous access of data) • Accelerates delivery of data to business users • Reduces IT workload, while still providing control and governance

SnapLogic Deployments

Integrations take a variety of forms from building a data warehouse to creating ad hoc reports and analyses, to Enterprise Mashups or deploying Rich Internet Applications.

Analytic Data Integration

The SnapLogic solution can play two different roles in the world of Analytics and Business Intelligence.

First, it can provide the lightweight, flexible, and affordable data services access layer that enterprises desperately need, in order to take full advantage of their substantial investments in data warehouses for Business Intelligence, Customer Data Integration, and Master Data Management projects.

In many organizations, IT has worked hard to create centralized data warehouses with “clean,” up-to-date customer records and other vital data. Unfortunately, they often find that there’s no affordable, efficient way to distribute the data out to departments and divisions, where it needs to be used, limiting the utility and return on their investment. Lacking a downstream data services access layer, organizations begin duplicating data and investing in duplicate infrastructure: local databases built from extracts of the core data records. In some large organizations, IT workers rely on tens of thousands of local databases and thousands of daily FTP transmissions to make up for the lack of real-time, downstream data services. By providing secure, real-time access to core data warehouses, SnapLogic can improve the quality and timeliness of data, while reducing the need for duplicate IT resources and time-consuming management tasks. SnapLogic can:

- Provide secure data services from centralized data warehouses, such as MDM data warehouses
- Transform data as needed for consumption by downstream applications
- Eliminate the need for costly infrastructure and processes, such as duplicate databases and daily extracts and FTP transmissions

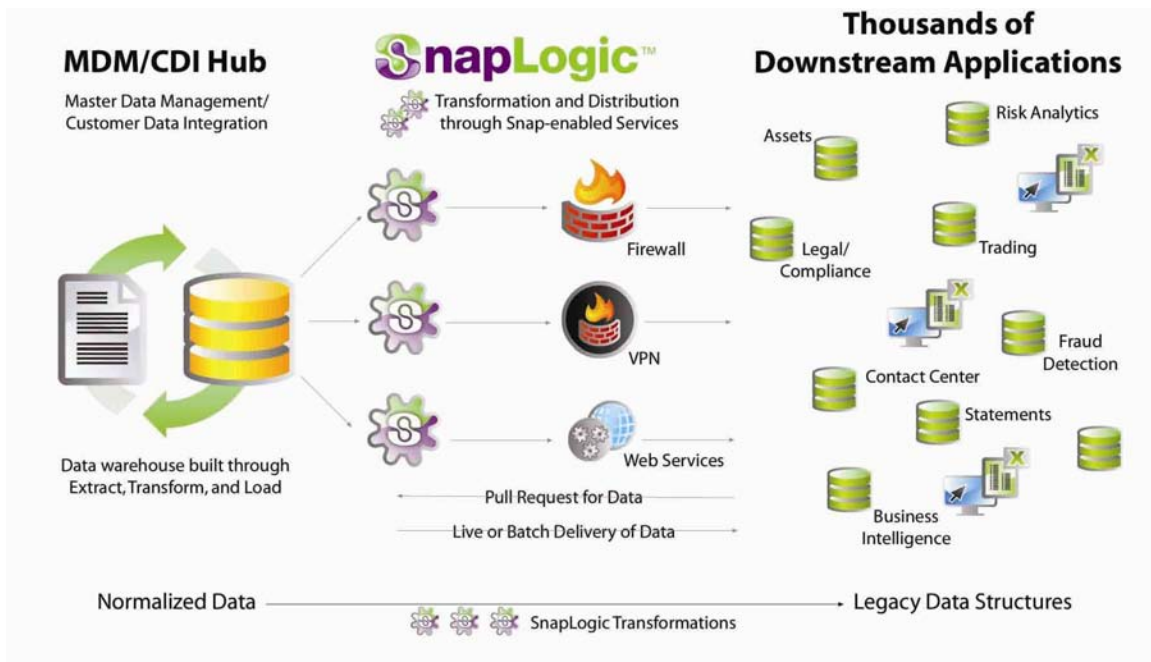


Figure 8. SnapLogic enables MDM and CDI data to be easily consumed by downstream applications. Core IT teams can let Lines of Business serve themselves to create data integration solutions using live or batch-fed data from MDM or CDI data warehouses.

Second, when appropriate, the SnapLogic framework can be deployed as an Extract, Transform, and Load (ETL) solution itself. SnapLogic can be deployed as a single instance to behave as a stand-alone ETL server. As an enterprise grade ETL solution, SnapLogic can extract, transform and load (ETL) data from enterprise applications as well as files, hosted applications, Web Services and other data sources.

Operational Data Integration

Most businesses run on a collection of single purpose, hand-coded scripts and programs for data access and manipulation. These programs are used for migrating and synchronizing data sets between applications or across remote offices, and are difficult to maintain.

SnapLogic's distributed, Web-based approach is perfect these kinds of operational data integrations. Its modular approach facilitates re-use across remote and distributed data sources and the metadata repository provides a central, shared database of Resources and Pipelines that are easily searched and reused or repurposed for any integration need.

Enterprise Mashups

Enterprise Mashups are becoming an increasingly popular solution for reporting and visualization. Mashups combine data from different sources in a browser or on a server, typically with a minimum amount of scripting. The idea is to quickly and efficiently combine different data in a way that provides value by correlating data and creating a whole that is of more value than the sum of its parts. For example, a logistics mashup might plot a list of delivery addresses on a map of live traffic reports and weather reports, enabling drivers to find the fastest, safest route to a destination. Or it might map an address selected from a list of 50,000 addresses, as shown in the next figure.

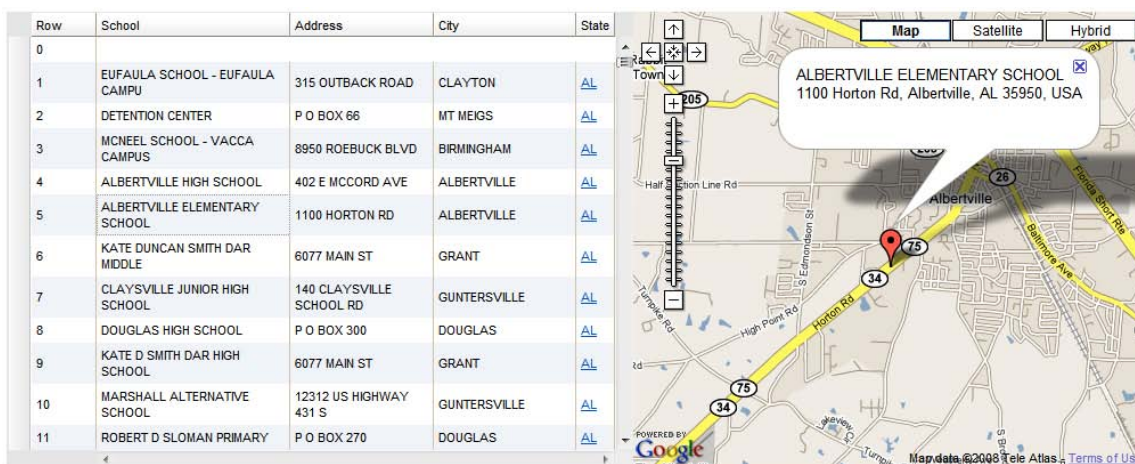


Figure 9. In this example of a browser-based mashup using the Dojo Data Grid and the SnapLogic Dojo Data Store, a user clicks on an address record—one of 50,000 records efficiently handled by the data store—and sees that address instantly rendered on a map.

A fundamental challenge with mashups is getting access to the data in an easily consumable form. Often these integrations are performed by someone without deep technical expertise, and often outside of the functional areas of the data's origin.

SnapLogic's Web-based approach creates a consistent data services layer that mashups can tap into for accessing, mixing, and matching data for any purpose.

When the data sources are also SnapLogic-enabled, mashing up data is as easy as assembling Resources that perform the desired function. The graphical browser-based design environment makes assembling mashups easy enough for beginners.

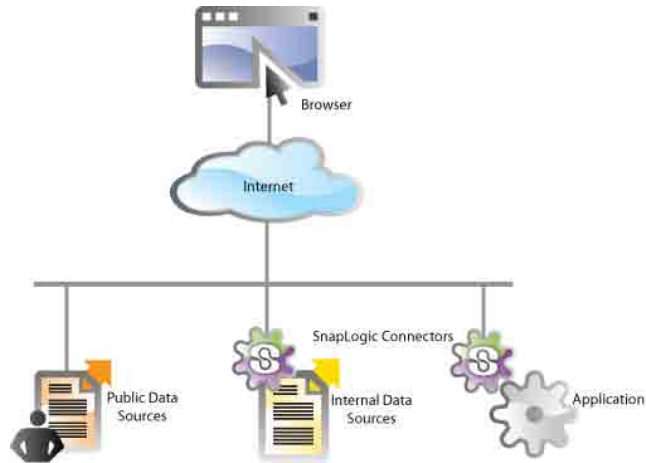


Figure 10. SnapLogic Connectors make data services available to mashups.

Rich Information Applications

Rich Internet Applications (RIA) based on AJAX technologies are rapidly replacing old click-and-wait static Web pages. RIAs frequently program presentation logic (and sometimes even business logic) in JavaScript that runs in the browser for a more compelling user experience. AJAX-based applications are now available that closely resemble full-fledged desktop applications.

An emerging class of RIAs requires access and presentation of large quantities of data for visualization and reporting. These Rich Information Applications increasingly rely on data services to simplify application development and off load the application server. For these kinds of RIAs, SnapLogic is ideally suited to provide the data without burdening the application server with the data processing task.

Some RIAs program both presentation and business logic in the client and do not require an application server, only the data. For these kinds of applications SnapLogic can be used to access, transform and expose data directly.

To further support this style of RIA development, SnapLogic has become active in the Dojo community and has contributed to the development of their grid widget as well as created a Dojo SnapLogic data store. The SnapLogic data store abstracts the details of interacting with data service so that application developers do not have to worry about the details of data access. Dojo and SnapLogic can be a powerful combination for developers building Rich Information Applications.

The full impact of RIAs is only beginning to be realized as the technology advances and developers establish new application partitioning schemes SnapLogic can be an essential element for successful deployments.

Conclusion

The limitations of traditional integration solutions in addressing new data integration have forced developers to continue to build hard-coded, brittle, and expensive to maintain integration solutions. The success of the Web and the consumer internet, built on inexpensive commodity infrastructure serves as a model for how a businesses of all sizes can apply a resource-oriented approach to data integration and achieve interoperability and reusability to drive down cost while speeding up the adoption of new cloud computing solutions.

SnapLogic DataFlow thrives on the new demands of data integration between the enterprise, the cloud, and the consumer web by providing:

- An open API data integration framework and ecosystem that allows dataflow pipelines to be configured with only the resources needed to accomplish their intended task
- A developer network and a store for the resale of Snaps developed by individuals, consultants and independent developers
- An architecture that allows data to flow unimpeded by the restrictions of a proprietary, inextensible architecture. .

For more information, please visit www.snaplogic.com.