How to Share Data Securely with Business Partners

A look at Business-to-Business data sharing options and best practices

ENZO UNIFIED

POWERFUL DATA SOLUTIONS TO GET THINGS DONE. ™

April 2021

Contents

Introduction

Sharing data between two or more business partners to improve business process automation and quality of service can be challenging, time consuming, and costly. Many organizations exchange flat files for sharing information with business partners due to the simplicity of the approach, or build complex solutions leveraging expensive integration platforms or custom development. But is there a better way?

This white paper explores some of the key areas that hinder organizations from adopting simpler and more flexible Business to Business (B2B) data sharing solutions, the options that are currently available with their respective advantages and limitations, and identifies best practices that modern data integration platforms should offer in order to deliver maximum flexibility and speed of implementation. This white paper also looks at some of the main factors that drive the total cost of ownership of B2B data exchange solutions.

Drivers for B2B Information Sharing

While companies generate revenue streams as a direct result of their primary sales operations, many organizations operate as part of a larger network of companies that have an interest in sharing information, such as product inventory levels, pricing information, general customer demographics and more. The need to engage in Business-to-Business (B2B) data sharing is not new; it has been around for decades in basic forms, such as flat file exports, or using more advanced Application Programming Interfaces (API) for example. Some businesses depend on B2B data sharing so much that stopping these processes could directly hinder critical aspects of their operations such as online sales, customer support, or product delivery.

B2B data sharing can simply provide more accurate data to business partners, increases business opportunities, or even create a competitive advantage in a larger partner ecosystem. For example, in the case of the distribution industry, allowing suppliers to access distributor inventory levels would allow them to adjust their production levels accordingly, better plan for route distribution, and better estimate expected contractor costs.

As important as B2B data sharing is, the available technologies for doing so haven't evolved significantly over the last decade; in fact, many existing critical business processes still rely on classic implementations that leverage the exchange of flat files because of the simplicity of the solution. However, the demand for a better/faster data exchange topology is growing because many businesses also depend on information that changes rapidly. The ability to share data once a week, or even once a day, is rarely sufficient in a dynamic environment such as online sales. In fact, many business processes depend on accessing information almost in near-time, making a flat file exchange ill-suited and creating significant technical challenges for organizations of all sizes. Let's review some of the key considerations that make near-time B2B data exchange challenging.

B2B Data Sharing Challenges

As discussed previously sharing data between organizations can be critical; yet, the ability to innovate in this space has been slow due to multiple factors, from the level of complexity of APIs to security.

It is worth noting that when both source and target systems are already in the cloud, running as a Software as a Service (SaaS) platform, certain options exist that can facilitate faster data integration projects, and some of the challenges identified in this section may not apply. However, data integration is usually considered a lower-level data exchange pattern that offers significant flexibility which point-to-point SaaS connectivity doesn't provide.

Let's review some of the most important factors that are preventing many organizations from adopting a near-time data integration model:

Area	Consideration
Source/Target differences	Information needed by various business partners is stored in vastly different systems, including entirely different database engines, schema, and data types.
API Development	Building integration solutions by leveraging APIs can be very expensive to build and maintain, and requires advanced software development skills.
Network bandwidth / Connectivity	Moving data usually requires synchronizing large sets of data which could be an issue when dealing with limited bandwidth; this is in addition to connectivity challenges that may require using proxy servers in Demilitarized Zones (DMZ) environments.
Identifying data changes	In many cases, identifying changed records, including new, updated and deleted records can be hard. This depends on the source system's ability to provide a Change Data Capture (CDC) stream.
Batch vs Near-Time Integration	Most modern B2B data exchanges require near-time information and cannot rely on legacy batch jobs scheduled daily, weekly or monthly.
Security / Encryption	Changes in security standards, authentication protocols and complex security algorithms usually require the involvement of senior resources or consulting partners for deep API integration.
Data Masking	Sharing sensitive data may also be controlled by complex regulations and could drive data masking requirements for Personal Identifiable Information such as Social Security Numbers, Phone Numbers and more.

Coupling	Decoupling the source from the target system is a challenging task that emphasizes the need to separate the identification of the changes from applying them, so that the target system is unaware of the source system.

When any of these conditions are present, organizations wanting to share information with other business partners in near-time will most certainly need to build a custom solution, or use an integration platform to build a custom workflow that can be both expensive and complex.

B2B Data Exchange Patterns

Understanding which challenges apply to which data exchange pattern, and whether or not batch and near-time integration can be achieved, is important in weighing the pros and cons of implementation speed against other factors, such as the total cost of ownership and reusability. Let’s review at a high level the main B2B data exchange patterns that are available today: Flat File, Integration Platform, and Database Replication.

Flat File

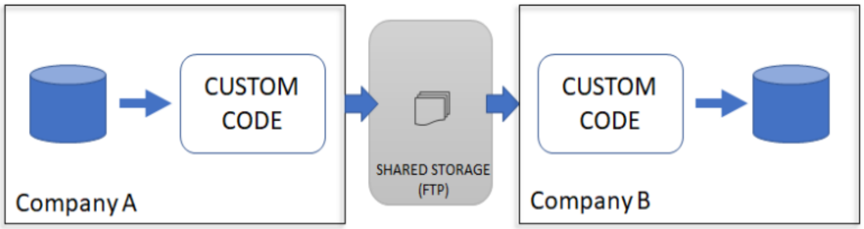

Figure 1: Flat File Data Integration Pattern

As discussed previously, flat file integration is still very common for sharing information in many environments due to its simplicity, decoupled architecture, and relatively low cost of maintenance. The source and target systems may be entirely different, and the same file can be sent to multiple customers if needed. The security implementation is usually implemented at the protocol level on the medium shared by both parties (such as an FTP site for example); for more secured implementations, the file itself may be encrypted using Pretty Good Privacy (PGP) or other technology.

A flat file integration pattern can either provide a full copy of the data, refreshed daily for example, or a change file made available on a periodic basis (for example weekly) with a file naming convention that indicates the period. It is worth nothing that a change file allows simulating a CDC integration pattern, which can be very difficult to implement depending on the source system. Custom development is usually needed to generate the file and to apply the changes on the target system, either in the form of custom software development, scripting, or as in integration workflow using an integration platform.

The Flat File pattern is best when the frequency of the data exchange is relatively low and doesn't require strong operational oversight.

While the flat file integration pattern can be simpler to implement, certain considerations may quickly make this implementation considerably harder and more expensive to use. For example, if the flat file contains changes only, custom code may be needed to implement the complex logic required to identify changes in the source system in order to emulate the CDC pattern. If additional functionality is needed, such as data masking on Social Security Numbers, additional code will be needed at the source. Finally, this pattern usually lacks an overall management framework that provides visibility into the replication topology, log files, consumption history, retry capability and more. As a result, this pattern is best when the frequency of the data exchange is relatively low and doesn't require strong operational oversight.

Integration Platform

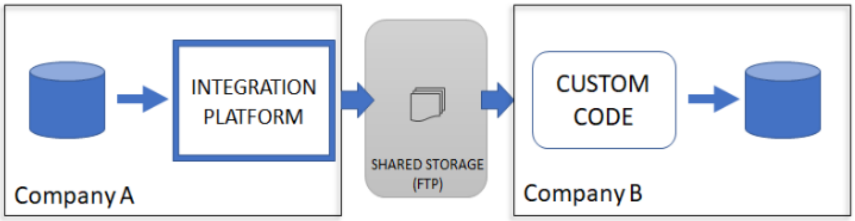

Figure 2: Integration Platform Data Exchange Pattern

In the context of B2B data integration, various platforms can be used to reduce or eliminate the need for custom code, such as Enterprise Application Integration platforms (EAI), and Extract Transform and Load (ETL) tools. Integration platforms can also offer other shared storage mechanisms, including cloud drives and blobs. Normally this pattern also leverages flat files for data exchange as the preferred medium when multiple business partners need to participate in the exchange, although that's not always the case. Some implementations are done through more complex APIs, creating tight coupling and increasing the total cost of ownership significantly as a result of API development needs. It should also be noted that ETL tools are usually a better fit than EAI platforms for replicating data as they can be somewhat simpler to use and are designed to deal with databases and data sources.

When both companies use an integration platform, it may be possible to use a different shared medium, such as leveraging an event-based integration model through messaging, or leveraging an iPaaS platform for cloud-to-cloud application integration. However, event-based integration models usually answer different integration needs and are not the preferred implementation pattern for replicating larger data sets across organizations. In addition, leveraging more advanced integration platforms may limit the ability to share data with future partners that may not have the resources or skills to implement an integration platform.

The Integration Platform pattern is usually limited to larger enterprises with the skills and financial backing required to implement and support a complex solution.

Most of the integration platforms available on the market provide options that facilitate data masking, data encryption, retry logic, and other key capabilities. Some of these platforms can also identify changes from source systems, although this feature is usually limited to specific database platforms. The majority of these platforms do not have a mechanism to emulate the CDC pattern, although they usually provide the management framework that the flat file pattern can't offer on its own.

The primary drawbacks of integration platforms, when used in the context of B2B data replication, include a significant total cost of ownership. This is a direct result of the specialized skills needed to build the integration flows that require custom coding and complex configuration settings, the cost of acquisition and yearly support of the integration platform itself, and the cost of workflow maintenance when data integration requirements change over time. With the exception of a few simpler ETL tools, this pattern is usually limited to larger enterprises with the skills and financial backing required to implement and support a complex solution. However, even ETL tools usually involve some form of custom development and workflows that must be maintained. Many organizations underestimate the effort required to implement ETL solutions beyond simple use cases. In the case of B2B data replication, solutions using ETL tools can be difficult to implement and may require complex upgrades over the years.

Database Replication

Database level replication is another integration pattern that is normally used within a company's network infrastructure; however, it may be possible to leverage this pattern through a VPN tunnel so that two companies can exchange data. The primary benefit of database replication is the speed of replication, allowing near-time data integration without writing code and leveraging the internal CDC capabilities of the database engine.

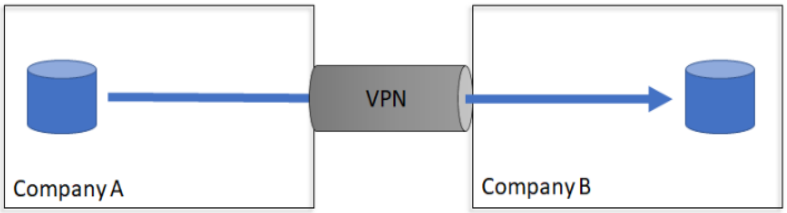

Figure 3: Database Replication Pattern

However, this pattern has many drawbacks, including tight coupling of the database platform itself (both source and target databases must be the same type, for example Oracle to Oracle), single purpose implementation (data cannot be shared easily with other business partners), inability to mask data, and invasive networking configuration changes to implement a VPN tunnel. Database replication is of course also limited to database systems. Replicating other non-relational databases or hosted platforms is usually not possible. Finally, while this solution doesn't require any custom code, the maintenance and monitoring of the end-to-end solution may be very complex. This is particularly important when the replication breaks and troubleshooting is needed.

Security Considerations

Let's review a few security principles that are relevant in the context of data replication between business partners. Since security is a vast topic area, we will keep the discussion at a high level considering the areas of the CIA triad: Confidentiality, Integrity, and Availability.

Confidentiality

This area of security is concerned about ensuring that information is not made available to unauthorized individuals, companies, or processes. In the context of B2B data replication, this concern can be addressed through different means, such as using a common storage area in the cloud (such as an Azure Blob Container, or an AWS Bucket), ensuring that information is encrypted while in transit, and that access to the change log requires authentication. FTP sites leveraging SSL for encryption may be sufficient as long as an authentication mechanism is applied. PGP encryption is also another generally accepted technology that provides confidentiality since it requires having the private key and a passcode to decrypt data. The primary areas of confidentiality hence require strong encryption in transit, encryption at rest, and an authentication mechanism.

Integrity

From an integrity standpoint, multiple concerns are in scope and should be evaluated depending on the security requirements of your organization. At a minimum, integrity in the context of B2B data replication refers to the quality of the data being exchanged, so that there is a form of guarantee that the information hasn't been tampered with. In the case of the various patterns discussed previously, the level of integrity varies widely based on the level of encryption used, the trust associated with the integration platform, and the mechanisms by which the source and target can authenticate that the change log is indeed trusted.

Availability

From a business standpoint, data is usually only useful if it is available in time. In the context of data replication, companies receiving data should be able to identify whether expected changes are not made available in time. This requirement may vary widely based on the business process and downstream decisions being made. However, B2B data replication implies by design a certain delay that could be a few minutes to a few hours (or even days). Consuming systems should assume the data is potentially delayed and should include a safeguard to ensure critical business processes do not trigger downstream decisions without a warning or an alert.

B2B Data Exchange Pattern Summary

As a summary of the patterns discussed previously, the data replication pattern is considered highly rigid and complex (due to the level of skill required to support the solution), and can be somewhat costly due to the networking infrastructure needed and the replication software required for the database platform.

The integration platform pattern is the most flexible since many vendors offer a large number of connectors for various scenarios, and can allow the implementation of complex workflows with limited coding. However, many implementations require the use of expensive platforms and/or dedicated servers and skilled

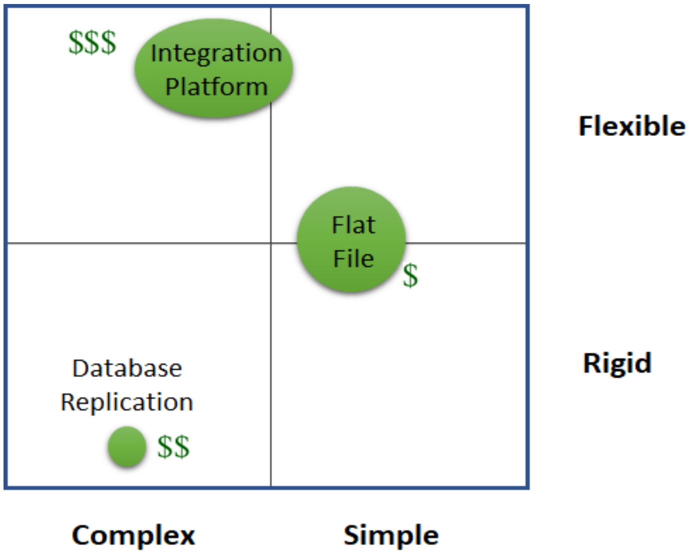

Figure 4: B2B Data Integration Quadrant

engineers to build and maintain the solution. While the total cost of ownership can vary widely, this pattern is usually considered the most expensive.

The flat file pattern is normally considered simple to implement, offers some flexibility in terms of implementation options, but is not as flexible as the integration platform pattern without significant custom development. Generally speaking, the flat file data sharing pattern is considered one of the cheapest implementation methods as long as little code customization is required and the security requirements are not too stringent.

There is an opportunity to define a new B2B data sharing pattern that can provide both high flexibility and simplicity with low or no-code implementation.

When placing each pattern in a quadrant that measures simplicity versus flexibility, we observe that there is an opportunity to define a new B2B data sharing pattern that can provide both high flexibility and simplicity with low or no-code implementation. In order to achieve flexibility and simplicity, this data integration pattern should leverage the loosely coupled pattern of flat file integration, provide the necessary management framework offered by the other patterns, and address the complexities discussed earlier such as data masking without writing custom code. We will call this pattern Plug-and-Play Data Integration.

Plug-and-Play Data Integration

Overview

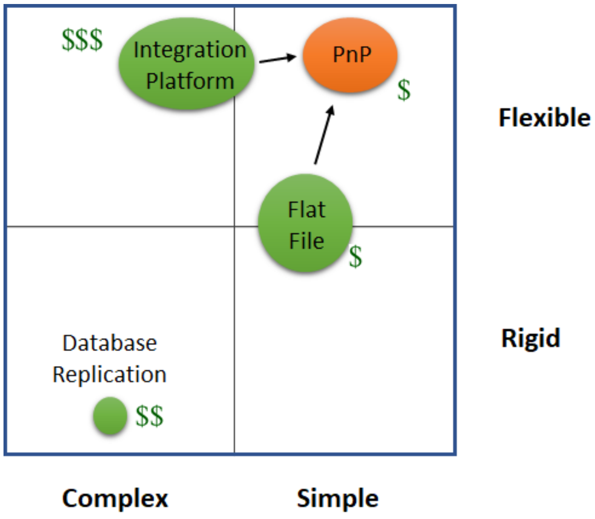

Figure 5: PnP Pattern Quadrant Position

The Plug-and-Play (PnP) integration pattern leverages the simplicity of the flat file pattern, while incorporating the flexibility of the integration platform pattern. Essentially, the PnP pattern ventures to solve the complexities discussed previously, while keeping the implementation cost low by eliminating virtually all coding and platform-specific expertise.

By its very nature, the PnP data integration pattern falls in the top right corner of the B2B Data Integration Quadrant. The ability to address the challenges identified earlier without writing code allows this pattern to offer the lowest total cost of ownership. With a PnP platform, the implementation cost is largely limited to licensing as long as the necessary capabilities are provided by the platform.

B2B Data Integration Best Practices

In order to achieve a "Plug-and-Play" pattern, the integration platform needs to implement a managed and secured data integration solution that follows a number of best practices aligned with the challenges previously identified:

- Any Source/Target
  To qualify as a "Plug-and-Play" solution, the integration platform should be able to forward changes to any target system, regardless of the system where the changes come from. For example, capturing changes in a SaaS platform and forwarding them to an Oracle database, or vice-versa.

- Codeless Implementation
  The codeless implementation means that no code should be written specifically to make the "Plug-and-Play" aspect of the integration tool work. Any platform-specific changes need to be handled by the integration platform itself.
- Replication Monitoring
  The platform should provide some form of monitoring and execution log, so that administrators can proactively manage the data replication environment.

- Micro-Batch Integration
  The frequency of the replication should support traditional batch changes (weekly, daily, hourly) or more frequently (near-time), such as every 5 minutes.

- Synthetic CDC
  Because not all source systems are able to provide a change stream, the PnP platform should be able to construct a change log based on the last known state of the source records, and identify inserted, updated and deleted records automatically, without additional custom code.

- Strong Security and Encryption
  The PnP integration platform should implement a standard security model such as PGP to ensure the data replication is encrypted and can only be read by the intended target consumer.

ENZODATAZEN

An example of a PnP Data Integration Platform is **ENZO DataZen**, a secured Corporate & B2B Data Replication technology.

DataZen allows organizations to quickly implement data replication, ingestion, and Change Data Capture (CDC) across an array of heterogeneous databases, platforms, and virtually any data source accessible through ODBC drivers or a data virtualization platform (such as ENZO Server).

With ENZO DataZen, organizations can securely move data across data centers or share data selectively with business partners, without requiring custom code.

For more information, visit: https://www.enzounified.com/Solutions/DataSync/ or contact us at: info@enzounified.com

- Data Masking & Filtering
  The platform should allow secondary filtering and masking to protect sensitive information, such as Personal Identifiable Information (PII) or other critical data without requiring custom code.

- Loosely Coupled
  The ability to generate the change log from any source system should be entirely independent and disconnected from the target system (or systems) to which the changes are applied.

- Eventually Consistent
  Due to the loose coupling need and the synthetic CDC aspect of this solution, the replication engine needs to ensure an eventually consistent data replication model. This means that data found in the target system will eventually match data at the source, normally at the next replication window.

- Schema, Initial Load & Replay
  A full initial load and replay capability, along with the ability the carry the source schema are essential aspects of a data replication technology, which the PnP data integration should implement.

The table below summarizes each B2B data integration pattern and shows to which degree they implement the best practices identified previously.

	Flat File	Integration Platform	Database Replication	PnP Data Platform
Any source / target	🟢	🟢		🟢
Codeless implementation		◑	🟢	🟢
Replication monitoring		🟢	🟢	🟢
Synthetic CDC	🟠	🟠		🟢
Micro-batch replication	◑	◑	🟢	🟢
Strong security and encryption	▨	🟢	◑	🟢

Data masking & filtering	🟠	▨		🟢
Loosely coupled	🟢	🟢		🟢
Eventually consistent	🟢	🟢		🟢
Schema, initial load & replay		🟠	▨	🟢

🟢	Implements the capability by design, or through minimal configuration settings
▨	May partially implement the capability, or can implement the capability through extensive custom code implementation or complex configuration
◐	May implement the capability by design in specific/limited scenarios, or may be achieved with development or configuration changes
🟠	Does not usually implement the capability, but could be done through extensive custom development

Sample PnP Implementation Architecture

A sample PnP implementation architecture is depicted in Figure 6. The ability to read from any data source (for example SharePoint Online or an Oracle database) and write to any target system (such as an AWS RDS database or SalesForce) is implemented using ODBC drivers or a data virtualization platform. Essentially, the PnP platform can generate a synthetic CDC change log and propagate the changes to relational databases (both schema and data), or any other target system including hosted platforms.

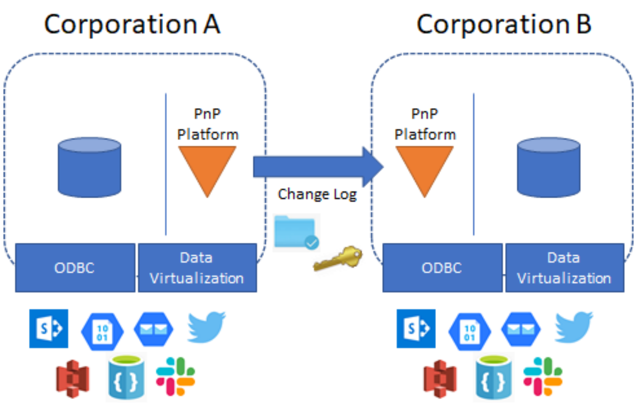

Figure 6: PnP Pattern Implementation

In some cases, the PnP platform can create the target database based on the schema information provided in the change log. This allows Corporation B to build any number of targets at any time, as long as the change logs are available.

Finally, Corporation A may choose to reuse the change logs with multiple businesses, and/or for its own needs, allowing cross data-center replication from any source to any target system, using an eventually consistent replication model.

Although this white paper focuses primarily on B2B data exchange, a PnP integration solution can assist with the implementation in a wide array of related use cases that are variations of the same problem, in which flat files, EAI or ETL tools are not well-suited, including:

- Implementing lower environment data refreshes with data masking
- Sub-setting data sets for targeted reporting environments
- Cross-database data replication and legacy database migrations
- Cloud data movement and replication
- SaaS hosted data extraction and ongoing replication

For a sample TCO calculator and more information on the factors that influence the total cost of ownership of integration platforms, visit:

https://www.enzounified.com/Solutions/TCOCalculator

Summary

This white paper exposes the primary data exchange patterns currently implemented and explores the key areas that hinder organizations from simplifying B2B data sharing, including source and target differences, custom development needs, network bandwidth, change data capture capabilities, near-time integration, strong security, data masking, and loosely coupled implementations.

Due to the complex nature of the challenges of exchanging information between business partners, including speed to market, flexibility, and security, the need for a new solution has emerged called a Plug-and-Play (PnP) integration platform. Finally, this paper identifies best practices for data replication that a PnP integration platform should offer in order to address the needs of B2B data exchange.