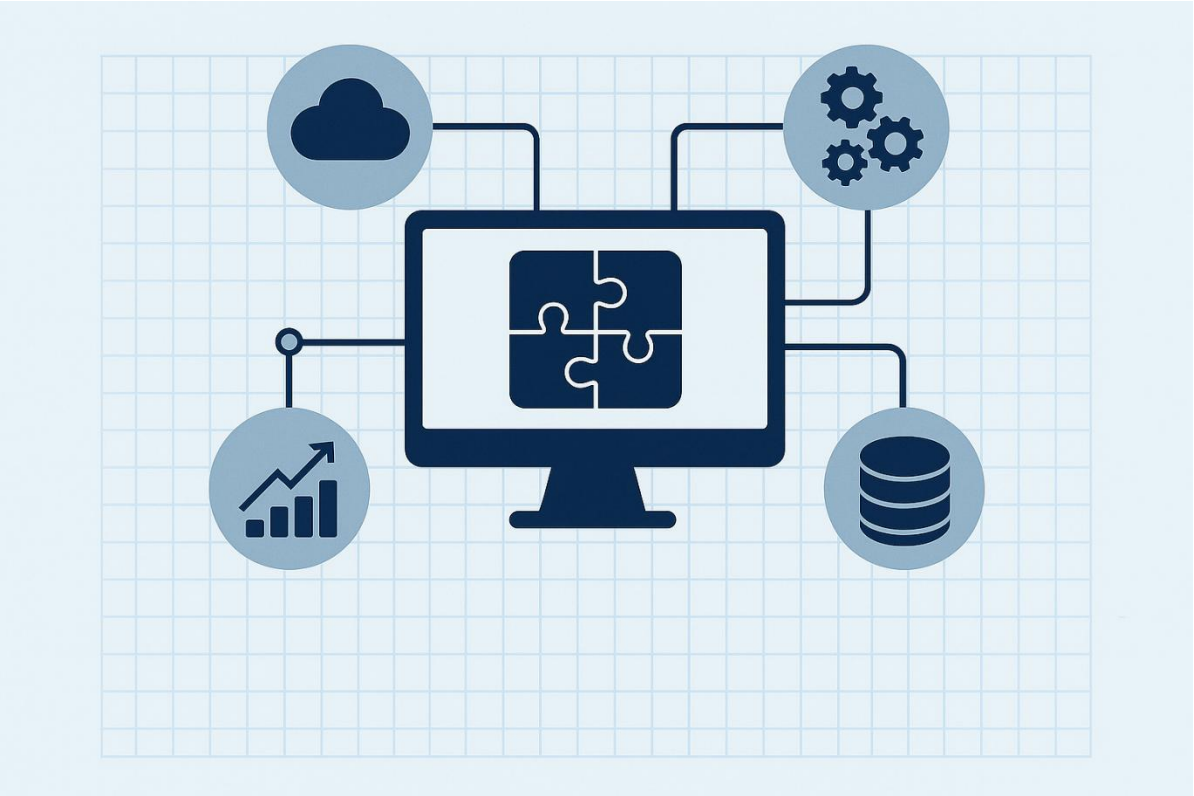


# A Short Paper on Enterprise Integration – Paper B

## (Legacy Integration)

Written by Tejinder Rai

**An Enterprise Integration Paper (B) – Legacy Integration**

This is a short paper on enterprise integration, Paper B. This paper expands on Paper A, where several integration patterns were described and depicted. This paper focusses upon enterprise integration technologies, where legacy approaches were specified in Paper A.

If you wish to download Paper A for further reading, it is available to download in the link below.

EnterpriseIntegration-AShortPaper.pdf

**Introduction**

Many organisations deploy multiple applications; the realisation sets in where the data and integration strategy does not fit the business requirements. There is no "one-size-fits-all" to enterprise integration to ensure that the relevant data and information integration is addressed as part of an enterprise strategy, especially in a hybrid environment with on-premises and cloud systems. There may be reasons for an enterprise to have a hybrid environment, especially in the case of where legacy systems still play a key role within on-premises or co-located data centres, which may not have the support for being hosted in public cloud environments.

**Why not work the old way**

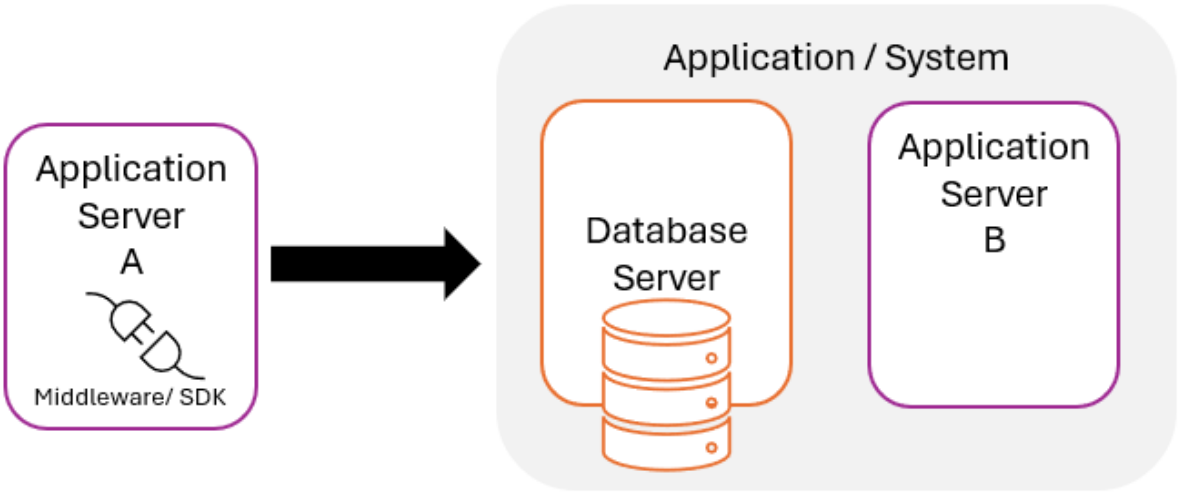
In Paper A, there were some specific integration patterns that covered the following integration methods below.

| Integration Method | Middleware | Purpose |
|---|---|---|
| **Direct Database Connection (Data Extraction)** | Database Driver / Integration Package | Extract data from an applications database to feed into a data warehouse or business process. |
| **File based integration** | Extract / Job from core application capability | Extract data or import data using a file-based approach using built-in application capabilities e.g. scheduler. |
| **Programmatic** | Application Executable | Extract / Import data from applications using built-in capabilities provided by the application software vendor. |

Whilst I reference this as the old way of integration, this is specific to a legacy approach in most cases. So why is this no longer a good practice?

Generally, these approaches are described as legacy approaches because they use specific aged patterns with approach to integration. This does not necessarily mean that they are no longer valid integration options, but the following sections describe the shortfalls in each approach.

**Direct Database Connection (Data Extraction)**

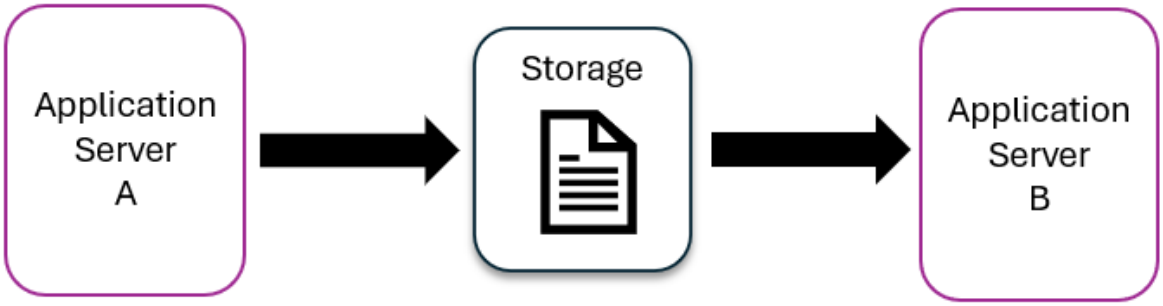



As depicted in the diagram above, application server A has a need to access the information held within application server B and accesses the information directly from the database server. Whilst this is a simple method to integrate with application B, the shortfall with this approach are as follows:

1) The schema must be known for this integration at a database table/view/field level; the application vendor may not completely publish the schema. This takes effort and time to compile, prior to performing any specific configuration.
2) The middleware on application server A must be maintained inline with the version of the database server for application server B.
3) If other applications require access to the information in application server B database, then another point-to-point connection must be configured
4) An integration identity with the relevant access to read the tables and views must be created on the application server B database and maintained.
5) Schema changes in the application server B database may break the integration between application server A and application server B database.
6) Application server A bypasses all business logic within application B; this could lead to audit information being maintained separately at a database level vs at an application specific level.
7) Unless there is a way to throttle the requests from application server A at the database level, application server A could impact the transactions in application B.
8) Any impact to application B operations, could impact the support and architectural elements of application B.
9) Depending on the number of queries and schedule of the queries that need to be executed by application server A, on the application B database, the system architecture will need to be monitored and may require an uplift in hardware.
10) There are likely latency concerns to be addresses depending on the distance of the database server from application server A to the application server B database server.
11) Point-to-Point network connectivity and database protocol and port network access must be maintained between the systems.
12) The import and export scheduler needs to be maintained.

The above list indicates the shortfalls with this approach to integration.

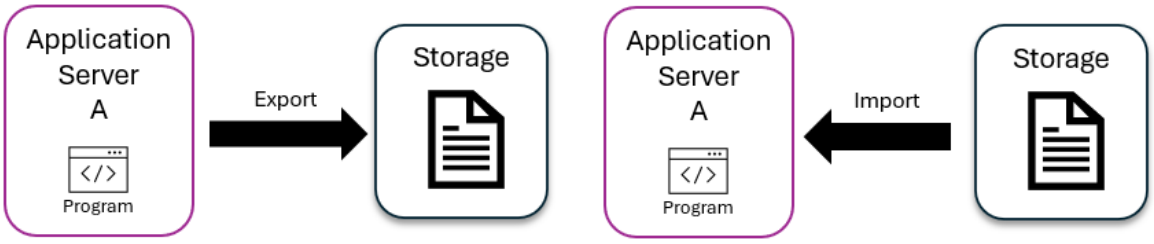
**File based integration**




Depicted in the diagram above, application server A and application server B have a file-based integration. Both applications can interchange files as required, based on the business and process requirements. This is a quite common approach to integration where files in a specification and format, which defines the contract between the applications, is agreed by both system owners. The shortfalls with this approach are as follows:

1) Point-to-Point network connectivity and storage protocol and port network access must be maintained between the systems and storage medium.
2) There are likely latency concerns to be addresses depending on the distance of storage medium between the systems.
3) The storage medium needs to be managed, including the access, governance, audit, and retention of data e.g. to meet legal requirements.
4) Export and/or import mechanisms need to be supported and maintained on both systems.
5) If the data exchanged between both system is sensitive, then both the storage of each file and transmission of the files needs to include encryption capabilities and be managed accordingly. This may include certificates for encryption/decryption and certificates for the storage medium transmission.
6) The storage medium hardware and software needs to be maintained.
7) It is typical that the data integration will need to be idempotent, so there needs to be a mechanism for sequential approaches to file processing defined between both systems.
8) The import and export scheduler needs to be maintained.

The above list indicates the shortfalls with this approach to integration.




Depicted in the diagram above, programmatic integration provides integration with applications based on supported software provided by the software development vendor. Application executables are included as part of the software architecture, so that data extraction and ingestion can be achieved via the application executable to interface data with the applications logic. The shortfalls with this approach are as follows:

1) Upgrades of application software on either system may change the behaviour of the import or export programs, resulting in additional effort for re-writing the import/export routines and/or specifications to import or export the data.
2) There is a chance that the software vendor may choose to deprecate the import or export methods in future releases of the software resulting in additional design and implementation effort to support new methods.
3) The storage medium hardware and software needs to be maintained.
4) It is typical that the data integration will need to be idempotent, so there needs to be a mechanism for sequential approaches to file processing defined between both systems.
5) The import and export scheduler needs to be maintained.
6) If the data exchanged between both system is sensitive, then both the storage of each file and transmission of the files needs to include encryption capabilities and be managed accordingly. This may include certificates for encryption/decryption and certificates for the storage medium transmission.
7) Export and/or import mechanisms need to be supported and maintained on both systems.
8) Point-to-Point network connectivity and storage protocol and port network access must be maintained between the systems and storage medium.
9) There are likely latency concerns to be addresses depending on the distance of storage medium between the systems.

**Conclusion**

The short falls in legacy integration options in this paper highlight the challenges with the integration methods. Whilst these are technology agnostic, they offer some key highlights to consider when adopting the integration approaches. I will cover modern enterprise integration options detailed in Paper A, in the next paper, Paper C, in this series.