Impact of Business Problem Characteristics on the Architecture and Specification of Integration Framework

Monika Tsaneva, S. Kouzmanov*

Department of Information Technologies and Communications, Faculty of Applied Informatics and Statistics, University of World and National Economy, Bulgaria

Copyright © 2016 by authors, all rights reserved. Authors agree that this article remains permanently open access under the terms of the Creative Commons Attribution License 4.0 International License

Abstract Modern enterprise information environments usually host a lot of different software systems that are involved in a wide range of business tasks and operations. For a successful usage of the whole heterogeneous information environment of an enterprise, the integration framework used to connect different software systems should take into account the specifics of the business problem and scope. Different needs can vary a lot, but still they can be generalized into larger groups of common integration and data processing problems. This can be used as a foundation for researching the impact of business problems and needs on an integration frameworks used into and enterprise. This paper uses a brief, general classification of business problems and proposes a mapping between business needs and integration approaches.

Keywords Integration, Real Time Processing, Batch Processing, Services, Data Integration

1. Introduction

Nowadays the implementation and execution of an enterprise business is becoming more and more complex. With the process of globalisation, expansion to new markets, introducing new products and services and also with the advance of business oriented information technology, organisations increased their need for fast and accurate information processing. This includes an increase in data volumes to be processed and also a diversification in types of business problems and needs.

On the other hand, modern organisations have complex, heterogeneous information environments and application infrastructures. This leads to a constant need for development of an enterprise-wide integration frameworks. Since various applications that should be interconnected by the integration framework are deployed in different business problem context, the way they are integrated with other applications can be affected by the type of business needs

they solve.

2. Different Business Problems and Needs

Recipients of information within an organisation can be different and their needs can vary. This changes the way, time, format and representation of similar base data delivered to different addressees.

For example, on line customers will need accurate on an item they intend to purchase – is the item in stock, how many units of it is remaining, what is the exact current price. Operating staff, like employees who are in charge for order shipping will also need real time data for customers who are waiting their orders to be shipped. Managers or marketing staff will also need real-time data about sales, quantities and so on in order to adjust promotions, marketing initiatives and so on. All of the above examples can be generalized as core business and customer support data. If this type of data is processed in a real time manner the organisation will be more flexible, will deliver more value to customers and a new business opportunities will be enabled.

On the other side, there is also data that is not needed in a real time or near real time fashion. For example a top management member or a board of directors will not need information about today orders, but will be interested in the trend of sales for the last quarter or in a prediction for the sales of a product for the next year. Such information can be provided after collecting and analysing of large volumes of data. Such big data can be transferred to a business intelligence system in the so called batch window, when the core business systems are not fully loaded and has free resource for synchronising with the BI. Another example is some bank transactions which are not due on the same day. Data about such kind of transactions can be transferred between systems during the night, when information infrastructure is not loaded.

Generally, data integration as defined by Haughey [1] is the use of common data by multiple applications or the

exchange of data across multiple applications. According to the above examples we can suppose that most of the various business needs can be classified in the aspect of their data integration and generally most of them can be addressed as either core business, real-time data integration or non-real time, batch data replication problems.

3. Real Time and Batch Data Processing

After we identified the possible classification of different business problems we can examine the difference between batch and real time data processing as a fundament of further research.

In integrations frameworks based on batch processing, according to Long [2], a large group of transactions are collected and then the whole amount of data is processed by a software agent during a single execution. Since this usually involves a large volume of data processing tasks, it is better to run such processes in time intervals of lower system

resource load, for example during nights for systems which are loaded on the normal working hours. According to Zandbergen [5] and Walker [6] good examples are payroll and billing systems.

On the other hand, in a real-time processing based integration middleware, a single transaction or a small amount of data are processed on demand, again according to Long [2]. Instead of a synchronizing the whole amount of data in a single run, the transactions are sent periodically to a remote software agent for processing. This guarantees that the two integrated systems are always synchronized and if a new event occurs against one of the systems it will be replicated against the other one in a real-time manner. Using this data processing approach secures that the data processing load will be spread across the whole time interval of system operation and not in a moment of single data synchronization execution. According to Zandbergen [5] and Walker [6] examples where real-time processing is well applied are airline ticket reservations, bank ATMs, customer services, etc.

Figure 1. Batch Processing

Figure 2. Real Time Processing

Figure 3. Data Layer Integration.

Figure 4. Functional Layer Integration.

4. Different Integration Approaches

Based on the above classification of business problems and definitions of different styles of data processing we can map them to different integration approaches. Different integration styles are usually classified by the access of the integration interface to a layer of the application architecture. Two of the most common approaches are the data integration and the functional integration patterns.

In data integration approaches, the integration adapter or interfaces are implemented to wrap directly the data repository or data-access layer of system that should be interconnected with another system or an integration environment. As it is defined by Balasubramanian et al. [3] this approach integrates systems at the logical data layer, typically using some form of data sharing. Since this approach by-passes the business logic layer and the already implemented business rules are not involved in the transactions processing, it is usually used when the main aim of the integration is synchronizing or replicating data between the repositories of the interconnected systems and direct access to data is sufficient. However, according to Grundy et al. [4] most data oriented approaches limit the range of technologies that can be integrated and only work on low-level database interaction. According to Trowbridge et al. [7] a good example for this approach is the product data integration between an order entry application and an ERP system.

In functional integration environments, the integration

interfaces are implemented with wrapping the business logic layer of the stand-alone systems. This can be done by invoking an existing API or if access to the system implementation is available. As it is defined by Balasubramanian et al. [3] that kind of approach integrates systems at the logical business layer, typically using distributed objects/components, service-oriented architectures or messaging middleware. Such integration approaches are usually used, when direct access to data repositories is not appropriate and the existing business rules, like input validations, pre-save processing and so on must be explicitly used. The functional integration is often recognized as the most adopted and wide spread integration approach. According to Trowbridge et al. [7] example for functional integration approaches are the distributed objects technologies, message-oriented middleware and service-oriented integration.

5. Mapping between Different Problems and Integration Approaches

After knowing the basic classifications of business problems and integration approaches, we can map the both sets and research which mappings are more appropriate than others. For this aim we can use four models: real-time functional integration, batch processing functional integration, real-time data integration and batch processing data integration.

Figure 5. Functional integration model for batch processing.

Figure 6. Functional integration model for real-time processing.

In a sample functional integration model for batch processing, the target system has an integration interface implemented as a web service and the initiating system has a proxy which is based on the target system integration service. In this type of integration, the initiating system calls the remote web service when the whole transaction batch should be processed, for example on scheduled intervals. The whole batch of transactions is sent in the service request, and then the web service iterates the transactions and calls the wrapped business logic in order to process them. After the processing finishes, the web service returns an asynchronous response to the calling system, which is not blocked during the batch processing.

Since the integration adapter is only responsible for iterating the batched transactions, its implementation is easier and this is the main advantage of this approach.

But while the wrapper implementation is easy, the method has significant disadvantages, for example that calibrating an integration middleware is not a simple task, which makes the integration between systems harder. The method works well only for small data volumes and can not be controlled synchronously, which is a serious drawback for functional integrations.

In an example for functional integration scenario with real-time processing, the same integration adapter setup can be used – a web service adapter wrapping the target system business logic layer and a web service proxy on the initiating side. The difference from the previous model is that the initiating system calls the target web service every time when a transaction event occurs. The web service request contains only single transaction data and the target system process the data in a real-time manner. After the processing is completed, the target system returns synchronous result to the initiating side, which is blocked during the processing period.

This method has significant advantages. Since the integration adapter operates in a real time manner, the integration itself is easier, every transaction is processed on demand and the data volume is theoretically unlimited, because the processing load is distributed along time. The remote calls can be synchronously controlled also, which is an important advantage in functional integration designs.

This disadvantage of the method is the harder integration adapter implementation, which is related to on-demand processing, synchronous execution and session and context management.

Figure 7. Data integration model for on-line processing.

Figure 8. Data integration model for batch processing.

Another sample integration framework is data integration for real-time processing. In this model the initiating system begins a transaction and delivers data to the connected system as parameters of a stored procedure, which belongs to the target system and is accessible via DB link. The target system completely performs the processing and then returns some result to the source system. If no exception had occurred, the system which initiated the transaction commits work, otherwise rolls it back.

The main advantages of this approach is the ease of the implementation of a single transaction processing stored procedure, used as an integration adapter, and the availability of synchronous control.

On the other hand, the method has significant drawbacks, like the large amount of stored procedure calls between databases, and the higher rate of network DBMS processing overload. These disadvantages make it inappropriate for large data volumes.

The last sample approach is a data integration framework for batch processing. In the proposed model the initiating system prepares all the data in suitable structures (tables and/or views) which are accessible to the target system via DB link. Then the source system begins a transaction and invokes a stored procedure, which belongs to the target system and is accessible via DB link. The target system performs some basic validations (for example checks if needed data are available) and then returns some result to the

source system. If no exception had occurred, the source system commits work, otherwise rolls it back, and the target system proceeds with the data processing. That way two of the basic requirements are satisfied – the transaction is as short as possible, and some quality of the service is achieved, because at least the successful startup and the most important prerequisites of the synchronization are guaranteed.

This approach involves a single stored procedure call and since this leads to low network and DBMS overload, it can be defined as a serious advantage. Based on these characteristics we can make the conclusion that the method is better for large data volumes.

On the other hand, implementation of transaction iterating and batch processing stored procedures is harder due to the query orientation of SQL. The stored procedure has limited synchronous control also and these limitations can be qualified as drawbacks of the approach.

6. Conclusions

After we have identified some different business problems and needs, we made a quick comparison of real time and batch processing and we looked through different integration approaches. Using a mapping between different business problems and integration approaches we proposed a couple of integration scenario samples and researched the

advantages and disadvantages of various integration framework types.

Based on our comparison and analysis, we can conclude that data integration frameworks are better applied in business scopes with batch oriented problems, because of their lower network and processing overload and better scalability for large data volumes. On the other hand, functional integration frameworks are well adopted in real time processing problems, because of their better synchronous control, good data validation and convenient implementation of the integration middleware and respectively easier integration between systems.

REFERENCES

-
- [1] Haughey T. TDAN.com - The Data Administration Newsletter. [Online].; 2003. Available from: HYPERLINK "<http://tdan.com/data-integration-and-sharing-part-one/5156>" <http://tdan.com/data-integration-and-sharing-part-one/5156> .
  - [2] Long E. jumpmind.com. [Online].; 2013. Available from: HYPERLINK "<http://www.jumpmind.com/blog/when-should-you-use-real-time-data-integration>" <http://www.jumpmind.com/blog/when-should-you-use-real-t>
  - ime-data-integration .
  - [3] Balasubramanian K,SDC,MZ,LÁ. System Integration Using Model-Driven Engineering. In Tiako PF. Designing Software-Intensive Systems: Methods and Principles. Hershey, PA: IGI Global; 2009. p. 474-504.
  - [4] Grundy J,BJ,BJ,HJ,AR. An Architecture for Efficient, Flexible Enterprise System Integration. In Proceedings of the 2003 International Conference on Internet Computing; 2003; Las Vegas.
  - [5] Zandbergen P. Study.com – Transaction Processing Systems (TPS): Batch and Real-Time Systems. [Online].; Available from HYPERLINK "<http://study.com/academy/lesson/transaction-processing-systems-tps-manual-and-automated-systems.html>" <http://study.com/academy/lesson/transaction-processing-systems-tps-manual-and-automated-systems.html>
  - [6] Walker M. Data Science Association - Batch vs. Real Time Data Processing. [Online].; 2015. Available from HYPERLINK "<http://www.datascienceassn.org/content/batch-vs-real-time-data-processing>" <http://www.datascienceassn.org/content/batch-vs-real-time-data-processing>
  - [7] Trowbridge D,RU,HG,MD,NEG MSDN.com – Integration Patterns. [Online].; 2004. Available from: HYPERLINK "<https://msdn.microsoft.com/en-us/library/ff647309.aspx>" <https://msdn.microsoft.com/en-us/library/ff647309.aspx>