Architectural Challenges on the Integration of e-Commerce and ERP Systems: A Case Study

Fábio Santos[a] and Ricardo Martinho[b]
School of Technology and Management, Polytechnic of Leiria, Leiria, Portugal

Keywords: e-Commerce, Enterprise Resource Planning (ERP), Software Architecture, Enterprise Application Integration, Data Integration, Business Process Integration.

Abstract: Many retail companies had to go online before their Enterprise Resource Planning (ERP)-type systems were ready to fulfill all business requirements. Their overall daily operation still heavily depends on these highly customized systems often mandatory because of legal obligations, which frequently come without e-commerce "off the shelf" integration. This paper identifies main challenges derived out of the architectural and integration requirements from a case study at an e-tailer company that operates via two sales channels: online store and third-party marketplaces. These challenges led to the definition of a system architecture and implementation considerations for this common integration scenario, which was validated through its implementation. Our proposed approach allows ERP-dependent organizations to start selling online with open-source technologies, avoiding extra ERP licensing and hidden maintenance costs.

SCITEPRESS SCIENCE AND TECHNOLOGY PUBLICATIONS

1 INTRODUCTION

The growth of Internet access explains the number of consumers who order products or services online through e-commerce platforms. In the European Union (28 member states), the share of households with Internet access had risen to 90% in 2019 vs. ≈ 64% in 2009, while 60% of individuals aged 16 to 74 used some sort of e-commerce (Eurostat, 2020). This scenario led retail companies to adopt online channels in order to reach more potential consumers and consolidate their position in a global market.

Online marketplaces (e.g. eBay) and their clear dominance in the e-commerce ecosystem represents an opportunity for e-tailers to boost Business-to-Customer (B2C) sales. Well-know marketplaces already have an established high brand name amongst consumers, and companies who take advantage of this channel can benefit from that credibility and customer trust (Ecommerce Europe, 2019).

Since the late-1990s, ERP systems have become a requirement for most of the worldwide companies mainly due to the wide range of features offered by those systems. ERPs incorporate various enterprise core activities such as manufacturing, finance, human resources, among others, into a single inter-connected system shared with multiple departments across all the organization (Holland and Light, 1999). Besides the complexity and potential incompatibilities with business needs and existing business processes, ERPs are generic solutions that require a certain level of customization that could be translated into an increment of the overall enterprise's productivity and long-term performance (Davenport, 1998).

Despite the benefits of ERP systems, commercial ERP software products are often mandatory for merchants because of legal requirements related with certified software (take, for instance, the Portuguese case regarding invoicing software (Portuguese Government, 2019)) and the lack of certified open-source solutions. Nevertheless, some of these systems do not offer a complete turnkey solution for a wide range of businesses such as the absence of e-commerce modules for retailers who want to sell goods or services through the Internet. For this reason, companies are forced to use additional e-commerce software products to meet all business requirements, which leads to an increase in manual activities (and the hiring of more human resources) to address the absence of automated integration processes between ERP and e-commerce systems.

The coexistence of multiple information systems and applications to support daily enterprise opera-

[a] https://orcid.org/0000-0001-5916-7022
[b] https://orcid.org/0000-0003-1157-7510

Santos, F. and Martinho, R.
Architectural Challenges on the Integration of e-Commerce and ERP Systems: A Case Study.
DOI: 10.5220/0010494903130319
In *Proceedings of the 23rd International Conference on Enterprise Information Systems (ICEIS 2021) - Volume 1*, pages 313-319
ISBN: 978-989-758-509-8
Copyright © 2021 by SCITEPRESS – Science and Technology Publications, Lda. All rights reserved

tions and fit all business requirements presents multiple challenges. Companies should embrace Enterprise Application Integration (EAI) as an opportunity to obtain competitive advantage by optimizing their core business processes.

This article attempts to determine the software architectural and integration challenges related with the integration needed between ERP and e-commerce systems in e-tailer companies. The specific research questions are: 1) what are the main challenges on the integration of e-commerce and ERP systems; and 2) how these challenges can be addressed.

To support the research, a case study has been conducted in a Portuguese retail company that operates an online B2C business via two channels: own online store and third-party marketplaces.

The remainder of this paper is structured as follows. Next section mentions related work on the particular architectural and integration challenges regarding ERP and e-commerce systems. Section 3 presents the case study, functional and non-functional requirements and main challenges. Section 4 justifies the e-commerce platform selected. Section 5 defines the system architecture and implementation details. Section 6 concludes the paper.

2 RELATED WORK

While an online store can operate without a real-time connection to accounting systems, ERPs or others, the cost of manually managing sales and product catalogs in these multiple systems is often higher than initially expected. Here, EAI enables stocks, orders, deliveries, returns and invoicing (among other related sales data) to be automatically exchanged between independent systems (Farzaneh, 2014). Existing literature on the integration of e-commerce and ERP systems point to potential issues that could be encountered.

For instance, in (Kaya and Aydın, 2019) it is concluded that the automated and direct data transfer from an e-commerce platform to an ERP database could cause accounting problems because ERP systems are often in-use by the whole company everyday. An intermediate database layer could be created to mitigate this issue, so data is temporarily stored prior to importation into the ERP.

In (Nikitović and Mahmutović, 2019), extensive ERP customization is also referenced as a generator of hidden costs, since some requirements elicitation is only concluded during the implementation process.

Despite integration problems, e-commerce solutions face also relevant challenges that should be taken into account. Noted in (Thaw et al., 2009) is information security, which plays an important role because of consumer's trust in online transactions (crucial for the continuous growth and development of any e-commerce activity). The usage of technology enablers such as firewalls and security protocols (e.g. 3D Secure) could improve the overall security of the system (Nili et al., 2019).

Maintaining an efficient product inventory is seen as another challenge. It is important to prevent and account for stock outs to avoid losing sales and improve customer experience (Patil and Divekar, 2014).

e-Commerce platforms and ERP systems are heterogeneous and complex since they are often developed independently in potential different technological ecosystems. Consequently, the industry adopted technologies and standards which offer cross-platform support such as eXtensible Markup Language (XML) and Simple Object Access Protocol (SOAP) (Wang and Shi, 2017).

Moreover, the integration of ERP and e-commerce can be achieved by using specific agents to filter, organize and transfer information efficiently through web services (Kong et al., 2007).

Regardless of the integration dimension/strategy (presentation, functional or data, as referenced in (Ruh et al., 2002)), the development of inter-application middleware is a common architectural approach to enable connections between legacy systems, databases, servers, web applications, among others (Lee et al., 2003). Figure 1 illustrates the comparison between the traditional vs. middleware approaches.

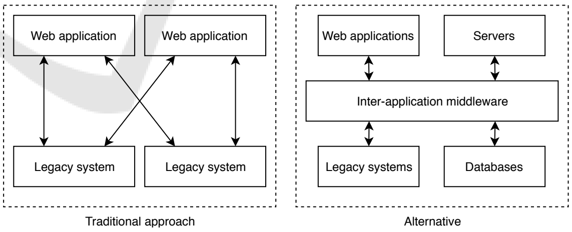


Figure 1: Comparison of two software architectural approaches for EAI.

The traditional approach promotes the implementation of custom changes in each existing node of the software application ecosystem to allow the communication between them. It will eventually concentrate the same business logic on distinct applications, which leads to complex maintenance tasks and repetitive validation procedures for integration purposes.

Adopting inter-application middleware as a software architectural solution makes possible to mitigate these problems, since it is focused on mapping processes, workflows and data between diverse applications, systems and databases. This was, indeed, the overall architectural solution adopted for the case study described in the next section.

3 CASE STUDY

The case study of this research is focused on a Portuguese retail company that has established to sell biking gear and cycling clothing through the Internet. The firm intends to execute B2C operations via two sales channels:

- Online Store: company's e-commerce platform consisting of a web application with product catalog, shopping cart and automatic payment system;
- Marketplaces: third-party (marketplace) platforms where the company's products can be listed, advertised and sold online.

ERP PRIMAVERA Professional was the system already in place used by the company to support stock management, invoicing and human resources, among others. It runs on an on-premises environment composed by Microsoft Windows Server and Microsoft SQL Server (PRIMAVERA, 2020).

Therefore, any e-commerce solution would have to be fully integrated with this system, due to its supporting role on the company's main business processes, and since it did not offer any e-commerce module to create an online store and to easily integrate with existing online marketplaces.

3.1 Architectural and Integration Software Requirements

Before the design of the overall architecture and implementation phase, the requirements were elicited through multiple in-person meetings with the board of the retail company. We resume below the main functional and non-functional requirements derived out of these meetings.

3.1.1 Functional Requirements (FR)

Users such as administrative and warehouse workers should be able to perform the following actions on the architectural (integrated) solution:

FR1. View online orders by status (billing, shipping, return, etc.);

FR2. View orders history;

FR3. View documents issued by the ERP;

FR4. Create accounting documents (e.g. invoices, credit notes);

FR5. Control visibility of articles in the online store and marketplaces;

FR6. Manage attributes and characteristics of articles (e.g. bar codes, title, descriptions, images);

FR7. View blocked orders and causes;

FR8. View sync failures and causes;

FR9. Force sync of product makes/brands, attributes and catalogs.

In turn, the solution should also account for the following automated features:

FR10. Sync the makes/brands lists;

FR11. Sync attributes (e.g. clothing size charts);

FR12. Sync products with/without variations;

FR13. Sync images of products;

FR14. Sync stock as soon as an order is placed;

FR15. Adopt prevention error mechanisms (e.g. rollback transactions);

FR16. Reserve stock in ERP after an order is placed;

FR17. Create accounting documents via ERP.

3.1.2 Non-Functional Requirements (NFR)

The final integrated solution should also comply to the following non-functional requirements:

NFR1. PRIMAVERA ERP integration;

NFR2. Marketplaces integration with Amazon and Cdiscount;

NFR3. Multi-language support;

NFR4. Search engine optimization;

NFR5. Use only open-source components, libraries and frameworks for potentially needed software development.

3.2 Challenges

The identification of the requirements above and related work determined the following challenges:

C1. Choose a viable e-commerce platform;

C2. Integrate proprietary code with open-source components;

C3. Automatically synchronize data of heterogeneous systems;

C4. Manage stock efficiently;

C5. Minimize manual steps required to integrate information (e.g. customers, orders, product catalog, etc.);

C6. Avoid direct overuse of the ERP for core daily operations (e.g. invoicing);

C7. Focus on information security practices;

C8. Optimize development workload and minimize future maintenance effort;

C9. Handle an increase in transactions.

4 e-COMMERCE PLATFORM DEFINITION

Considering the development effort of building an e-commerce platform with all the usual base features (e.g., catalogs, payments, checkout, customers, among others), a comparative study was conducted in order to evaluate existing open-source solutions (NFR5) and choose a viable platform (C1). These solutions were retrieved from their market share of the top 1 million sites (BuiltWith, 2021). The selected platform has direct impact on the process to define the overall architecture as it depends on features offered by the e-commerce solution to support systems integration.

The table 1, presents a weighted scoring analysis of various open-source platforms, based on criteria to meet the case study requirements (e.g. multi-language support (NFR3) and search engine optimization (NFR4)). Weights were distributed according to the relevance of each topic to the case study.

The setup and configuration are not only related with the initial e-commerce platform setup (e.g. connecting to a database) but also how simple and easy an operator can change any configuration via the back-office (e.g. setup payment providers).

Any existing integration interface (Application Programming Interfaces (APIs)) (C2) together with a shallow learning curve of the platform and an extensive marketplace of extensions and themes, are relevant to decrease the development time-frame (C8).

Future maintainability efforts could be minimized through an active community and support, well-documented project but also by a clear development roadmap to anticipate features or bug-fixes (C8).

None of the solutions has native integration with the PRIMAVERA ERP (NFR1) or with online marketplaces (NFR2). However, the integration with marketplaces can be easily achieved via third-party extensions if no customization is required.

All evaluated solutions were developed with the PHP and JavaScript programming languages and relational database management systems such as MySQL or PostgreSQL. These technologies allow the usage of free Operating Systems (e.g. Ubuntu Server) which reduces overall implementation costs.

WooCommerce is a plugin of WordPress, a widely used content management system. However, its adoption would cause more complexity and potential security breaches since it would be necessary to install additional third-party plugins due to the lack of some core features (e.g. search engine optimization (NFR4)).

Regarding Zen Cart, it has a very small number of maintainers and noticeable low contributions or new versions. Therefore, we considered it not to be a good option to take for a long-term scenario, once there is no clear roadmap definition and it lacks relevant extensions in the marketplace and multi-language support (NFR3).

OpenCart providers do not maintain a change log in order to anticipate negative impacts of upgrading versions. Additionally, the native API does not provide products and attributes' synchronization in an easy and efficient way.

The analysis determined that Magento and PrestaShop were the best options to consider on the current case study. Both platforms provide web services to facilitate the integration with other enterprise applications and an extensive documentation which promotes the acquisition of technical knowledge by users.

While Magento offers both open-source and commercial solutions, developers who maintain PrestaShop are totally focused on open-source development. Furthermore, Magento's learning curve is considerably higher and could demand for paid training and support. Therefore, PrestaShop was considered the best option for this case study.

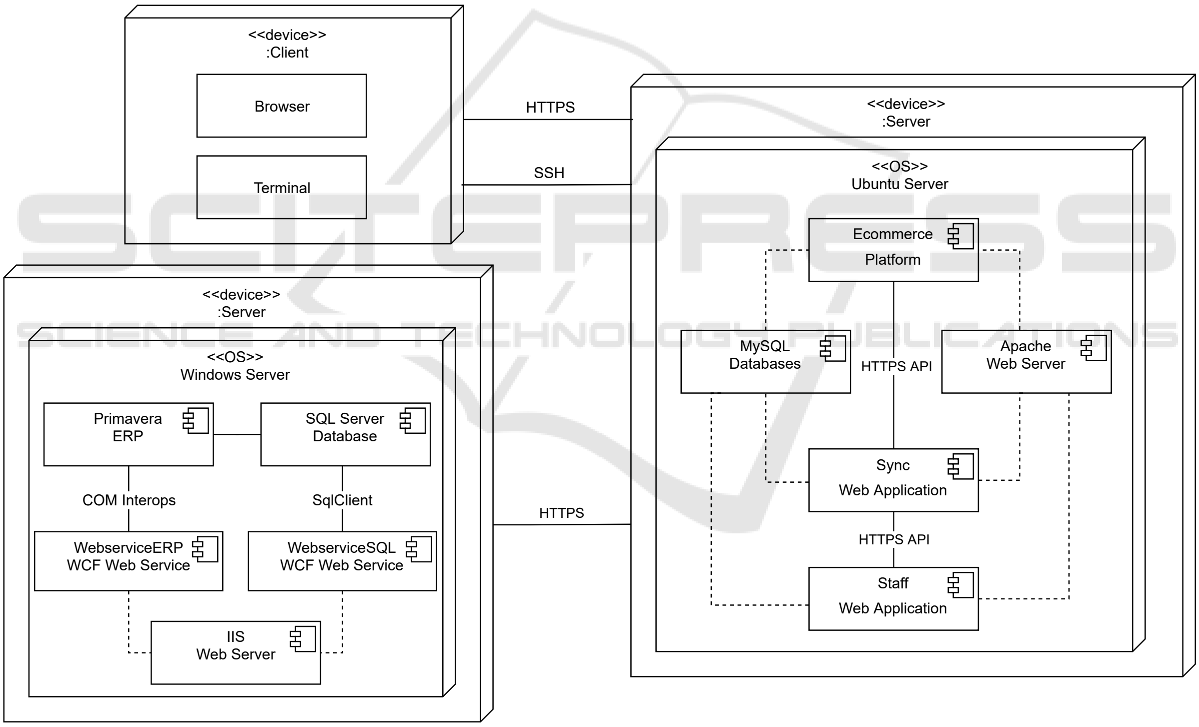
5 ARCHITECTURE & IMPLEMENTATION

The proposed architecture for the integrated solution is illustrated in Figure 2, using different nodes and associations according to the Unified Modeling Language (UML) standard, allowing to represent the physical view of deployment and various component dependencies (Patig, 2010).

The Figure shows an UML deployment diagram composed by 3 nodes: Client, Windows Server and Ubuntu Server. The selection of software and technologies took into account open-source options (NFR5).

Table 1: Comparative analysis of e-commerce platforms with scoring from 1 to 5.

	Weight (%)	**OpenCart**	**PrestaShop**	**Magento**	**WooCommerce**	**Zen Cart**
Setup and configuration	5	4	4	2	4	3
Learning curve	10	4	5	2	5	3
Multi-language support	10	3	5	5	5	1
Search engine optimization	10	4	4	4	3	3
Marketplace of extensions and themes	5	4	4	2	4	2
Community and support	15	3	4	3	5	2
API	25	3	5	5	4	1
Documentation	10	5	5	5	5	1
Development roadmap	10	1	5	3	3	1
Total (%)	**100**	**66**	**93**	**76**	**85**	**34**



SCITEPRESS SCIENCE AND TECHNOLOGY PUBLICATIONS

Figure 2: Simplified system architecture.

Both “Windows Server” and “Ubuntu Server” nodes were deployed on-premises, within the company’s local area network, and assuring that only the “Ubuntu Server” node had Internet connection. Network isolation and firewalls minimized the risk of occurrence of security breaches (C7).

ERP licensing and maintenance costs could represent a major slice of expenses in any enterprise application integration. An inter-application middleware was adopted to prevent hidden costs since the middleware will host most of the integration business logic (C8), leaving the opportunity to define which type of technologies should be used (C2).

The performance of synchronization operations across different enterprise applications and databases could be critical in terms of scalability. As the catalog and order volume increases, processes slow down. Thus, synchronization processes were optimized by placing software applications and databases under the same data center, so that applications could communicate using a local network and avoid extra-latency linked to Internet usage (C9).

5.1 “Client” Node

This node represents the device used to access the system through the company’s internal network or over the Internet. It is used to access the e-commerce platform and the “Staff” web application.

Some user profiles can execute routines of the “Sync” web application component via a computer terminal (FR93) and Secure Shell (SSH) connection (C7).

5.2 “Windows Server” Node

This node provides the environment for the ERP system execution and it is composed by a database management system – MS SQL Server – and two Windows Communication Foundation (WCF) web services. Both web services are executed on the Microsoft web server Internet Information Services (IIS).

The development of those web services using Microsoft technologies allows to take advantage of the ERP libraries, but also promotes the integration with any other enterprise application through widespread standard and protocols such as REST/JSON and Hypertext Transfer Protocol (HTTP) (C2).

“WebserviceSQL” acts as a middleware which allows secure operations on the ERP database by external applications (NFR1). Incoming connections are filter based on a set of permissions composed by a list of allowed IP addresses for certain types operation (e.g. create, read, update and delete) (C7).

On the other hand, “WebserviceERP” facilitates the access to core features of the ERP (NFR1). ERP system’s APIs were leveraged in order to execute operations such as create, void and retrieve documents, from any other application, avoiding administrative workers to access directly to the ERP and saving licensing costs (C6).

5.3 “Ubuntu Server” Node

This node is composed of a database management system – MySQL – and a web server – Apache – to support the correct functioning of the e-commerce platform and of the two web applications: “Sync” and “Staff”.

The “Sync” web application is an inter-application middleware which supports data synchronization and automatic operations related to e-commerce activity (FR10 to FR16) (C3, C5). It connects to “WebserviceSQL” to execute database operations on the ERP system, “WebserviceERP” to perform core ERP operations (e.g. create invoices) (FR17) (C2) and also the e-commerce platform API. Exposes an API to provide the list orders and the ability to create/void invoices or change orders statuses (C5).

In contrast, “Staff” web application allows administrative and warehouse workers to perform daily operations (e.g. view order statuses) without accessing ERP directly (FR1 to FR6) (C6). It connects to the API exposed by “Sync” and provides interfaces for order management, manual invoicing, product management and errors details (e.g. synchronization failures, blocked orders, etc.) (FR7 and FR8) (C5, C6).

5.4 Implementation Considerations

The synchronization of e-commerce data occurs through scheduled routines which can be executed manually (C3). These routines are susceptible to fail due to multiple reasons, including temporary network issues and missing data in the ERP system, among others. A retry mechanism was implemented to bypass temporary issues and every failure is logged by the system for post-analysis.

Regarding payments, most of the e-commerce platforms already support third-party providers such as Paypal via extensions widely used and tested. Security protocols such as Secure Socker Layer (SSL) and Transport Layer Security (TLS) improved the security of communication channels used not only between applications and users, but also in transactions without human interaction (e.g. automated synchronization processes) (C7).

The cost of the software components implementations was optimized by using open-source technologies, frameworks and libraries. Furthermore, the technical background of the development team has been also taken into account since past experiences were useful to build a robust solution (C8).

Existing extensions to integrate payment services, online marketplaces, among other features, were important to reduce the development workload but also future maintenance costs since these extensions are maintained by authors or the community (C8).

Maintaining an efficient product inventory is important to prevent stock outs and improve overall customer experience. Although invoicing is carried out later manually by operators, it was essential to make a stock reservation of articles immediately after receiving the payment confirmation as it prevents erroneous stock in both e-commerce platform and ERP system. A virtual warehouse was used for this purpose (C4), allowing for temporary stock on both ERP and e-commerce systems as new orders are placed, and preventing inconsistent stocks due to synchronization processes.

6 CONCLUSIONS

This paper identifies the software architectural challenges behind a common integration scenario between ERP systems and e-commerce platforms. These challenges are derived from functional and non-functional software requirements elicited for a particular case study.

The proposed architecture for this case study includes the development of several components to comply with the elicited requirements, and addresses the various challenges. These also include the selection of an open-source e-commerce platform through a weighted scoring model, in order to reduce the development effort of the integration. Its successful implementation validated the presented architecture as a solution to address the case study and identified challenges.

Further research and case studies should be conducted for the identification of architectural and integration challenges that can be generalized for EAI between ERP systems and e-commerce platforms. Nevertheless, software architects and managers can benefit from the case study presented and solutions adopted as this kind of scenarios is quite common within enterprise information systems.

REFERENCES

BuiltWith (2021). ecommerce usage distribution in the top 1 million sites. https://trends.builtwith.com/shop. Last checked on Jan 23, 2021.

Davenport, T. H. (1998). Putting the enterprise into the enterprise system. *Harvard business review*, 76(4).

Ecommerce Europe (2019). European ecommerce report: 2019 edition. https://www.ecommerce-europe.eu/wp-content/uploads/2019/07/European_Ecommerce_report_2019_freeFinal-version.pdf. Last checked on Jan 23, 2021.

Eurostat (2020). Digital economy and society statistics - households and individuals. https://ec.europa.eu/eurostat/statistics-explained/index.php/Digital_economy_and_society_statistics_-_households_and_individuals. Last checked on Jan 23, 2021.

Farzaneh, M. K. (2014). Evaluation of use of erp in e-commerce: Methods and strategies. *research Journal of Applied sciences, engineering and technology*, 7(20):4171–4174.

Holland, C. and Light, B. (1999). A critical success factors model for ERP implementation. *IEEE software*, 16(3):30–36.

Kaya, A. and Aydın, Ö. (2019). E-commerce in turkey and SAP integrated e-commerce system. *International Journal of eBusiness and eGovernment Studies*, 11:207–225.

Kong, Z., Wang, D., and Zhang, J. (2007). A strategic framework for enterprise information integration of ERP and e-commerce. In *Research and Practical Issues of Enterprise Information Systems II*, pages 701–705. Springer.

Lee, J., Siau, K., and Hong, S. (2003). Enterprise integration with ERP and EAI. *Commun. ACM*, 46(2):54–60.

Nikitović, M. and Mahmutović, A. (2019). Hidden costs of ERP implementation. In *2019 42nd International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, pages 1314–1318. IEEE.

Nili, A., Barros, A., Johnstone, D., and Tate, M. (2019). Technological enablers for preventing service failure with e-commerce websites. In *27th European Conference on Information Systems (ECIS 2019)*, Stockholm & Uppsala, Sweden.

Patig, S. (2010). Modeling deployment of enterprise applications. In *International Conference on Advanced Information Systems Engineering*, pages 253–266. Springer.

Patil, H. and Divekar, B. R. (2014). Inventory management challenges for B2C e-commerce retailers. *Procedia Economics and Finance*, 11:561–571.

Portuguese Government (2019). *Decreto-Lei n.º 28/2019*. Diário da República n.º 33/2019, Série I de 2019-02-15. Regulamentação das obrigações relativas ao processamento de faturas.

PRIMAVERA (2020). PRIMAVERA profissional. https://pt.primaverabss.com/pt/software/software-de-gestao/professional/professional/. Last checked on Dec 05, 2020.

Ruh, W. A., Maginnis, F. X., and Brown, W. J. (2002). *Enterprise application integration: a Wiley tech brief.* John Wiley & Sons.

Thaw, Y. Y., Mahmood, A. K., and Dominic, P. (2009). A study on the factors that influence the consumers trust on ecommerce adoption. *arXiv preprint arXiv:0909.1145*.

Wang, Y. and Shi, Y. (2017). Analysis on the integration of erp and e-commerce. *AIP Conference Proceedings*, 1864(1):020137.