Integration Development for SAP Business One: Internship Contributions at Be One Solutions

Master's in Computer Engineering – Mobile Computing

Carolina Pazeiro Antunes

Leiria, September 2025

Integration Development for SAP Business One: Internship Contributions at Be One Solutions

Master's in Computer Engineering – Mobile Computing

Carolina Pazeiro Antunes

Internship performed with the guidance of Professor Doctor José Carlos Bregieiro Ribeiro.

Leiria, September 2025

Originality and Authors' Rights

This internship report is original and was prepared solely for this purpose, with all authors whose studies and publications contributed to its development properly cited.

Partial reproductions of this document will be authorized on the condition that the Author is mentioned. That reference is made to the study program within which it was carried out, namely, the Master's in Computer Engineering – Mobile Computing, in the academic year 2024/2025 at the School of Technology and Management of the Polytechnic Institute of Leiria, Portugal, as well as the date of the public examinations aimed at evaluating this work (if applicable).

Dedication

This report is dedicated to my family, whose unwavering support, encouragement, and sacrifices have been the driving force behind all my achievements.

To my parents, thank you for your unconditional love, guidance, and for believing in me even during times when I doubted myself. Your strength and values continue to inspire me every day.

To my mentors and educators, thank you for instilling in me the knowledge and discipline that have shaped my academic and professional journey.

And to everyone at Be One Solutions who contributed to my growth during this internship, your guidance and collaboration have left a lasting impact.

This work stands as a reflection of the collective support, wisdom, and motivation I have received from those around me.

Acknowledgments

I want to express my sincere gratitude to Be One Solutions (BE1S) for granting me the opportunity to undertake my internship as a Junior Developer Intern. This experience has been both challenging and transformative, allowing me to grow professionally and personally in a real-world enterprise environment.

I extend my sincere thanks to my internship supervisor, Diogo Lopes, for his invaluable guidance, encouragement, and insightful feedback throughout my time at the company. His mentorship was instrumental in helping me grasp complex technical concepts and apply them effectively in real-world projects.

My appreciation also goes to the entire development team at BE1S. Their support, teamwork, and openness to sharing knowledge created an inspiring and collaborative work environment that greatly enhanced my learning experience.

I am equally thankful to my academic advisor, José Ribeiro, and the faculty of the Polytechnic Institute of Leiria for their continuous support and for making this internship possible.

Most importantly, I would like to express my deepest gratitude to my family. Their unconditional love, support, and encouragement have been the foundation of my journey. Their belief in me has been a constant source of motivation, and I am genuinely thankful for their presence in my life.

This internship has been a valuable milestone in my career, and I am genuinely grateful to everyone who contributed to making it a memorable and enriching experience.

Abstract

This report presents the activities and technical contributions developed during an internship at Be One Solutions, in the role of Junior Developer Intern. The core focus of the internship was the development of integration solutions for SAP Business One, utilizing both the BE1S Integration Platform and the Business One Integration Framework (B1if).

Various integration scenarios were implemented using the BE1S platform, which supports C#-based module development and provides secure connectivity to file systems, SQL databases, and SAP Business One. Additionally, B1if was used to develop workflows utilizing its visual interface and XML/XSLT-based logic, enabling flexible and standardized data exchange.

Beyond integration development, the internship also involved creating an installer for the BE1S platform, streamlining its deployment process, and documenting integration projects to ensure the clarity and maintainability of the implemented solutions.

The report outlines the technical architecture of the platforms used, the challenges addressed throughout the projects, and the applied methodologies. The internship provided hands-on experience in enterprise integration, software development practices, and ERP system interaction within a professional environment.

Keywords: SAP Business One, BE1S Integration Platform, B1if, Integration

Index

Figure List

Table List

List of Abbreviations and Acronyms

B1if	SAP Business One Integration Framework
BE1S	Be One Solutions
ESTG	Escola Superior de Tecnologia e Gestão
MSSQL	Microsoft SQL Server
SAP B1	SAP Business One
UDF	User-Defined Field
UDT	User-Defined Table

1. Introduction

Enterprise Resource Planning (ERP) systems play a crucial role in enabling organizations to manage their business processes efficiently and effectively. Among these, SAP Business One[1] is a widely adopted ERP solution tailored for small and medium-sized enterprises. While the system provides a robust foundation for core business operations, many organizations require custom integrations to align the ERP with their specific workflows, external systems, or data sources.

This report documents the work conducted during a professional internship, from October 1st, 2024, to June 12th, 2025, at be one solutions (BE1S)[2], an international consulting company specializing in global SAP Business One rollouts and integration services. The internship focused on the design, development, and deployment of integration solutions for SAP Business One (SAP B1), utilizing both the BE1S Integration Platform and SAP's native Business One Integration Framework (B1if).

The Integration Platform, developed internally by BE1S, offers a modular and flexible approach to integration, allowing developers to implement logic in C# while benefiting from built-in support for secure connections, structured logging, and a rule-based execution model. In parallel, the B1if tool provides a standardized environment for creating integration workflows using XML, XSLT, and visual flow design.

The internship began with a one-and-a-half-month onboarding phase focused on building a solid foundation in SAP Business One and the integration frameworks and platforms relevant to the role. This initial period was designed to prepare for the responsibilities and tasks to be undertaken throughout the internship.

Throughout the remainder of the internship, various tasks were undertaken, including the development of four integration scenarios, the implementation of a platform installer, and the documentation of projects to facilitate knowledge sharing and future maintenance.

The report examines the technical components employed, the challenges encountered, and the outcomes achieved, providing a comprehensive overview of the internship experience within a real-world ERP integration context.

While some of the integration projects carried out during the internship involved sending and receiving data from SAP to other third-party applications, others aimed to create and update information on SAP based on defined rules and information.

The first official project was a straightforward integration that retrieved data from SAP B1 and sent it in XML format to a designated service, referred to as outbound. The project took about two months (mid-November to mid-January) from start to finish. It included

[1] https://www.sap.com/products/erp/business-one.html

[2] https://www.beonesolutions.com/

development with the Be One Integration Platform and the preparation of SAP HANA procedures to fetch all necessary data.

The second was the most prolonged, lasting approximately five months (from the end of December to the end of May), and certain aspects are still being improved today. As a second project, developed using the Be One Integration Platform and SQL preparation on SAP HANA, it introduced additional complexity by incorporating inbound processes for the first time. This entailed the entry of data from a third party, which must be accurately processed and integrated into SAP B1, in addition to the standard retrieval of data from SAP B1.

The third, initiated at the beginning of February and still under development at the conclusion of the internship, represented a significant departure from previous implementations. Eliminating the conventional outbound-inbound workflow, this project was considerably more complex than earlier developments. Its objective was to automate the process of assigning batches, groups of items, to sales orders. The procedure involved retrieving data from the batch master, including order numbers and corresponding line items, and subsequently managing batch assignments with careful consideration of existing allocations and the total quantity of batches in relation to the sales order value. Developed using the Be One Integration Platform, this project also reintroduced the use of Microsoft SQL Server (MSSQL), which was primarily utilized during the initial onboarding phase. Nonetheless, the procedures involved in this integration project encountered unforeseen challenges and increased complexity beyond initial expectations.

Finally, the last project was a bit shorter, taking about five weeks to develop and deploy in the testing environment. It was the first official project being developed with SAP Business One Integration Framework (see SAP Business One Integration Framework). Its primary purpose was to update Journal Entries, which ensure the company’s financial records are accurate and complete, helping to verify and trace transactions, detect and correct errors, ensure compliance with regulations, and support internal audits. After fetching the Journal Entries, specific fields would be analyzed, and based on the defined mapping and rules, the document would be updated.

The remainder of this report is organized as follows:

- Chapter 2. Characterization of the Host Entity: This section provides background on BE1S, as well as a SWOT analysis of the company from an intern's point of view.
- Chapter 3. Background and Related Work: This section provides background information and technical details on the various technologies used during the internship.
- Chapter 4. Internship Program: This section provides a detailed description of the internship program, outlining its key components and essential details.
- Chapter 5. Integration Projects: This section provides a detailed view of the integration projects developed during the internship.
- Chapter 6. Other Projects/Tasks: This section describes any other projects or tasks that were carried out.
- Chapter 7. Critical Analysis and Improvement Proposals: This section provides a further analysis of the work developed and improvement proposals.

- Chapter 8. Conclusions and Future Work: This section provides a balance of the internship experience and possible future work.

2. Characterization of the Host Entity

BE1S is a global company specializing in SAP B1 solutions for subsidiaries of large corporations. As a host entity for internships in the Junior Developer Intern position, the company offers a dynamic and international environment that is ideal for developing both technical and professional skills. Interns benefit from exposure to real-world ERP projects, cross-cultural collaboration, and mentorship from experienced professionals in the field.

In the evolving landscape of enterprise resource planning (ERP), BE1S has established itself as a reliable and experienced SAP service partner, bringing over 15 years of expertise in delivering global ERP projects. The company focuses on supporting large multinational corporations and their medium-sized subsidiaries through a strategic two-tier ERP approach, which effectively links corporate headquarters with local branches using streamlined solutions based on SAP Business OneBE1S offers a full range of services, including the creation, implementation, and ongoing support of ERP systems tailored to the unique needs of each client. [1]

With over 170 workers, BE1S promotes a strong organizational culture founded on core values that promote collaboration, leadership, and personal development. The company highlights the importance of individual contributions to overall success, encouraging team members to initiate actions and make impactful decisions. By prioritizing mutual support and shared objectives, the organization nurtures teamwork to effectively address challenges and exceed expectations. The fundamental values – Trust, Excellence, Stronger Together, Diversity, and Wellbeing – govern daily operations and interpersonal interactions, fostering a workplace that values both individual and collective achievements.

To better understand the organizational context and its potential impact on interns' experiences, a SWOT analysis of BE1S from an employee perspective is provided below in Table 1 and Table 2. This analysis highlights the company's internal strengths and weaknesses, as well as external opportunities and threats that could influence the internship experience and future career growth within the organization.

Internal Factors	
Strengths	Weaknesses
Global Exposure: Employees work on international projects, gaining cross-cultural experience and global business insights.	**Training Gaps**: Limited structured onboarding or continuous learning programs in some regions.
Specialized Expertise: Opportunity to develop deep knowledge in SAP B1 and S/4HANA, which are in high demand.	**Dependence on SAP Ecosystem**: Heavy reliance on SAP's roadmap and ecosystem could limit flexibility in adapting to non-SAP technologies.
Flexible Work Environment: Many roles offer remote or hybrid work options, supporting work-life balance.	
Diverse Team: A multinational workforce fosters inclusivity and encourages learning from diverse perspectives.	

Table 1 - SWOT Analysis, Internal Factors

External Factors	
Opportunities	Threats
Skill Development: Exposure to cutting-edge ERP technologies, cloud solutions, and AI integrations enhances employability and job prospects.	**Rapid tech changes**: Client expectations for advanced tools in cloud, AI, and automation may quickly shift product viability.
Global Mobility: Potential for international assignments or relocation due to the company's global footprint.	**Economic Volatility**: Global downturns may lead to budget cuts or hiring freezes.
	Client Dependency: Heavy reliance on a few large clients or SAP's ecosystem could affect long-term stability.

Table 2 - SWOT Analysis, External Factors

The two tables provide a detailed SWOT analysis of BE1S, examining both internal and external factors that influence the company's performance and strategic direction.

Table 1 focuses on internal factors. Among the company's strengths is its global exposure, allowing employees to work on international projects and gain valuable cross-cultural experience and business insights. The organization also offers specialized expertise in SAP B1 and S/4HANA, which are in high demand in the ERP space. A flexible work environment supports work-life balance, as many roles include remote or hybrid work options. Furthermore, the company benefits from a diverse team, with a multinational workforce that fosters inclusivity and encourages learning from varied cultural and professional perspectives. However, the company faces internal weaknesses, including training gaps, with limited structured onboarding and ongoing learning opportunities in some regions. Another challenge is the company's heavy dependence on the SAP ecosystem, which could restrict its ability to adopt and integrate non-SAP technologies, reducing overall flexibility.

Table 2 highlights external factors. Opportunities include skill development through exposure to advanced ERP tools, cloud solutions, and AI integrations, all of which enhance employee employability and technological relevance. The company's global footprint also provides opportunities for international assignments and mobility, which can contribute to professional growth. On the other hand, threats include the rapid pace of technological change, where evolving client expectations around cloud, AI, and automation may quickly render existing solutions obsolete. Economic volatility presents another risk, as global downturns could lead to budget reductions or hiring freezes. Additionally, the company faces potential vulnerability due to client dependency, where reliance on a small number of key clients or SAP's ecosystem may pose risks to its long-term stability.

In summary, BE1S presents itself as a globally oriented and professionally enriching workplace. The company offers a unique opportunity to engage with international ERP projects, providing exposure to SAP Business One, one of the most in-demand platforms in enterprise technology. The organization fosters a collaborative and inclusive environment, where values such as trust, excellence, diversity, and well-being are clearly reflected in day-to-day operations. Interns benefit from a flexible work structure, the chance to work alongside a multicultural team, and opportunities to develop skills in emerging technologies like cloud solutions and AI. While there are some challenges, such as rapid technological shifts and reliance on the SAP ecosystem, the overall experience is one of professional growth and global awareness. The company's emphasis on teamwork, innovation, and individual development makes it an ideal environment for aspiring professionals to learn and grow.

3. Background and Related Work

This section provides a comprehensive overview of SAP Business One and its surrounding ecosystem of development and integration tools, as well as any other technologies used during the development of the internship's projects of enterprise-level integration development.

Enterprise-level integration development [2] is the strategic practice of connecting disparate systems, applications, and data sources across an organization to create a cohesive and automated IT environment. The goal is to eliminate data silos, automate complex business processes, and provide a unified view of information to enhance efficiency, inform decision-making, and foster agility. This involves using specialized platforms and architectural patterns, such as API-led integration and event-driven architecture, to ensure that systems can communicate seamlessly and reliably at a large scale.

SAP B1 (see section 3.1) is an integrated enterprise resource planning solution designed for small and medium-sized businesses. It helps companies manage their core processes such as accounting, sales, purchasing, inventory, production, and customer relationships in one centralized system.

Unlike SAP's enterprise-level solutions, SAP B1 is designed to be affordable, user-friendly, and easy to implement, making it accessible to growing businesses that require more than spreadsheets or disconnected software. It provides real-time visibility into operations, improves accuracy, and supports better decision-making.

The system can be deployed on-premises or on the cloud, offering flexibility to scale as the business grows. With built-in reporting and analytics, SAP B1 not only streamlines daily operations but also helps business owners gain insights into performance and plan strategically.

Although the SAP B1 platform was not directly utilized except for project testing, all integrations were connected to it. Therefore, having background knowledge of the ERP solution was essential for better understanding, just as it was during the onboarding phase. During onboarding, information regarding SAP B1, its processes, and technologies was gathered, even if not all of it was pertinent to the integration projects.

Now, the SAP B1 Software Development Kit (SDK) (see section 3.2) is a set of tools, APIs, and templates that allow developers and partners to extend and customize SAP B1. It provides access to the system's data and business logic so you can create add-ons, automate processes, or build integrations with external applications while ensuring compliance with SAP standards.

The Integration Framework for SAP B1, B1if (see section 3.3), is a middleware platform included with SAP B1 that enables system-to-system integration and is used directly in some of the integration projects described in this report. It allows businesses to connect SAP B1 with other SAP solutions (such as SAP S/4HANA, SAP Business ByDesign) or third-party

applications (including e-commerce, CRM, logistics, etc.) using scenarios, workflows, and web services.

Together, the SDK and Integration Framework provide companies with the flexibility to tailor SAP B1 to their unique needs, whether by developing custom functionalities or ensuring seamless data exchange with other systems, without disrupting the core ERP processes.

Lastly, the integration platform developed by BE1S (see section 3.4), which is the leading platform used for integration projects in BE1S, was created to overcome the limitations of B1if when handling non-standardized file formats and complex logic. Unlike B1if, which relies heavily on structured XML, XSLT transformations, or visual tools, BE1S enables developers to build integration modules directly in C#, offering more flexibility, control, and maintainability. It provides built-in services for secure connections to file systems, databases, and SAP B1, while also simplifying low-level configuration for consistency across modules.

Beyond connectivity, BE1S features robust logging, debugging, and monitoring capabilities to ensure stability and facilitate faster troubleshooting. Its architecture is based on a Manager-Worker paradigm with a predefined rule structure enforced through a Class Factory mechanism. This guarantees that integrations follow a consistent, auditable sequence of steps, making the platform reliable, scalable, and developer-friendly for complex integration scenarios.

Further details and explanations are provided in the following sections of this report, where each topic is discussed in greater depth.

3.1. SAP Business One

SAP B1 as an enterprise resource planning (ERP) solution for small to medium-sized enterprises, continues to evolve with a focus on cloud-first strategies, the integration of artificial intelligence and machine learning [3], an enhanced user experience inspired by SAP Fiori [4], advanced data analytics capabilities, exploration of Internet of Things (IoT) integration, and the development of industry-specific solutions.

The platform emphasizes robust security and compliance measures, while also strengthening its Service Layer to support broader API functionalities and easier integration with other systems. Key benefits include its integrated nature, real-time insights, scalability, flexibility, customization options, user-friendly interface, mobile accessibility, and global reach through multi-language and localization support [5]. Recent advancements include significant enhancements to the web client, bringing it closer to the desktop client's functionality with customizable homepages, improved document management, and user-defined table support.

Powerful analytics tools offer advanced list view analysis, enhanced user-defined queries, and improved financial reporting capabilities. Inventory and distribution management benefit from better batch, serial number, and bin location handling in the web

client, alongside cargo tracking enhancements. Financial management sees improvements in VAT handling and posting period management. The platform also provides increased extensibility through Web Client UI APIs and API Views.

SAP B1 deployments often leverage SAP HANA [6] as the underlying in-memory database, providing significant performance advantages for transactional processing and real-time analytics. This columnar database architecture enables rapid data retrieval and complex calculations directly within the database layer, thereby minimizing latency for critical operations and reporting.

The application server, typically running on a separate tier, houses the business logic and manages client connections. Communication between the client tiers (desktop, web, mobile) and the application server primarily occurs via secure protocols, with increasing adoption of HTTPS for web-based access and secure proprietary protocols for the desktop client.

From a deployment perspective, SAP B1 embraces flexibility, offering both on-premise installations and cloud-based deployments, including private cloud options managed by partners or public cloud infrastructure providers. The cloud offerings often leverage virtualization technologies and managed services to ensure scalability, high availability, and automated maintenance.

The core business logic within SAP B1 is structured around a comprehensive data model that encompasses master data (business partners, items, and G/L accounts), transactional data (sales orders, purchase orders, invoices, and journal entries), and configuration data. Data integrity is enforced through database constraints and application-level validations. Security is implemented through a granular role-based authorization system, allowing administrators to define precise access rights to different functionalities and data objects. Audit trails are maintained to track system changes and user activities, crucial for compliance and security monitoring [7].

Real-time analytics are a key aspect of the modern SAP B1 experience, particularly when running on SAP HANA. Embedded analytics capabilities allow users to generate interactive reports and dashboards directly within the application, leveraging in-memory processing for rapid data aggregation and visualization. Integration with SAP Analytics Cloud provides more advanced BI and planning functionalities. The platform also supports predictive analytics through embedded machine learning capabilities within SAP HANA, enabling features like demand forecasting and anomaly detection.

3.2.SAP Development SDK

The SAP Development SDK (Software Development Kit) for SAP B1 offers a multi-faceted approach to extensibility and integration, reflecting modern software development practices. The DI API [8], while based on COM technology, provides low-level access to the SAP B1 data model and business logic. Its object-oriented structure mirrors the business objects within the application, allowing developers to manipulate data and trigger business processes programmatically. Understanding the underlying database schema and the

interrelationships between tables is often necessary for efficient DI API development, especially for complex data manipulations. Transaction management within the DI API ensures atomicity of database operations.

The UI API [8], primarily utilizing .NET assemblies, allows for the creation of event-driven user interfaces within the SAP B1 desktop client. Developers work with UI objects, such as forms, grids, buttons, and text boxes, responding to user actions through event handlers. Threading considerations are essential when developing UI add-ons to ensure the responsiveness of the leading SAP B1 client. The UI API interacts with the DI API to persist and retrieve data manipulated through the custom UI.

The DI Server enhances the DI API by providing a stateless, multi-threaded architecture optimized for handling concurrent requests, making it suitable for building scalable integration solutions. It often communicates with client applications via SOAP or RESTful web services. Connection pooling and session management are key technical aspects of the DI Server.

The B1iF, which will be described in more detail in a later topic (see section 3.3), is built on a Java-based architecture and utilizes XML for message transformation and process orchestration [9]. Its core components include the Atom Management for managing integration scenarios and the various integration packages (IPOs) that define the steps and transformations involved in an integration flow. B1iF supports multiple communication protocols (HTTP, SOAP, JDBC, etc.) and data formats (XML, CSV, IDoc). Understanding XSLT for data transformation and XPath for navigating XML structures is crucial for B1iF development. Monitoring and logging capabilities within B1iF are essential for managing and troubleshooting integration scenarios.

The Service Layer, built upon RESTful principles and OData standards, exposes SAP B1 entities as resources that can be accessed via standard HTTP methods [10]. Its metadata document describes the available entities, properties, and relationships, enabling client applications to understand the API structure dynamically. Authentication often relies on OAuth 2.0 or basic authentication. Understanding HTTP status codes, request/response headers, and JSON payload structures is fundamental for Service Layer development. The stateless nature of RESTful services simplifies scalability. Webhooks provide a mechanism for server-sent events, allowing external applications to subscribe to real-time notifications of business events within SAP B1. The Service Layer often interacts directly with the SAP HANA database, leveraging its performance for data retrieval and manipulation.

3.3. SAP Business One Integration Framework

B1if stands as a highly evolved, Java-based middleware specifically designed to facilitate robust and complex integration scenarios for SAP B1. While the Service Layer offers a modern RESTful approach for direct API consumption, B1if excels in its capacity for orchestrated, message-based integration, providing a more comprehensive platform for managing diverse and often complex data flows between SAP B1 and disparate systems, both within and outside the SAP ecosystem.

One of the standout features of B1if is its preconfigured integration scenarios, which significantly reduce development time and complexity. These templates cover everyday use cases such as e-commerce (e.g., Shopify, Magento), logistics (e.g., DHL), marketing platforms (e.g., Mailchimp), and retail (e.g., SAP Customer Checkout). For more tailored needs, the framework supports custom scenario development using a combination of XML (eXtensible Markup Language), XSLT (eXtensible Stylesheet Language Transformation) 1.0, and SQL scripting, offering developers a high degree of flexibility [9].

Therefore, B1if's technical capability lies in its scenario-based approach. The integration logic is set in "scenarios," which are essentially workflow definitions that consist of a series of "scenario steps" or "atoms". Each atom performs a specific function, such as connecting to a system, transforming data, or executing a business operation. This modularity enables the reuse of common integration patterns, simplifying the management of complex integration flows [11].

At its core, B1if operates on an XML-centric architecture. All messages processed within B1iF are transformed into an internal XML format, regardless of their source or destination. This universal XML representation allows for highly flexible and powerful data mapping and transformation using XSLT. Developers leverage an integrated development environment (IDE) within B1if to design these transformations, defining how data elements from a source system's XML structure map to a target system's XML structure. This approach ensures data consistency and enables the application of complex business logic during the transformation process.

XSLT 1.0 is a language designed for transforming XML documents into different formats such as HTML, plain text, or other XML structures. It follows a declarative programming paradigm, where transformations are defined through templates and pattern matching rather than procedural logic. XSLT 1.0 played a foundational role in enabling interoperability between systems that exchange data in XML, and it remains in use in many legacy environments and integration frameworks such as SAP’s B1if (Business One Integration Framework).

Compared to later versions [12], the most significant differences lie in its limited feature set. For example, XSLT 1.0 does not support user-defined functions, grouping mechanisms, or advanced data types beyond strings, numbers, and Booleans. More recent standards, such as XSLT 2.0 and 3.0, have introduced significant improvements, including native support for grouping, powerful string and date/time functions, schema-awareness, and the ability to work with sequences rather than only node sets. These additions greatly simplify complex transformations and reduce the amount of workaround logic developers need to implement. However, working with the more recent standards is not possible when working with B1if.

The most challenging aspects of working with XSLT 1.0, especially in integration platforms like B1if, stem from these limitations. Implementing grouping or aggregations requires the use of verbose and error-prone "Muenchian grouping" techniques [13], which are difficult to read and maintain. Handling large XML documents can also be performance-intensive, as XSLT 1.0 lacks efficient mechanisms for streaming or incremental processing. Furthermore, debugging and testing transformations are often cumbersome, since the

declarative style can make it hard to trace execution flow. In B1if specifically, the learning curve is steep, as developers must not only master XSLT syntax but also understand how transformations interact with the framework's message processing, mappings, and integration scenarios.

B1if provides a wide range of built-in adapters for connecting to different systems and protocols, like JDBC, for direct database connectivity; HTTP/SOAP, for integrating with web services; file adapters, for processing flat files (CSV, XML, etc.); FTP/SFTP, for secure file transfer; mail adapters, for integrations triggered by or sending emails [14].

The framework contains message routing, error handling, and retry mechanisms. Messages flow through inbound channels, are processed (including transformations and business logic execution), and are sent out through outbound channels, ensuring a reliable and traceable data exchange by providing features like message queues, guaranteed delivery, and correlation IDs.

B1if has a System Landscape Directory (SLD) that centrally manages connection parameters and system definitions for all integrated applications [11], ensuring consistency in connectivity configuration and simplifying the deployment of integration scenarios across different environments (development, testing, or production).

B1if is optimized for cloud computing environments and supports integration with cloud-based applications and services. Its architecture and use of standard web protocols make it a suitable choice for connecting SAP B1, whether on-premise or cloud-hosted, to other cloud platforms.

With the introduction of B1if 2.0, there's enhanced support for multi-tenancy, which is crucial for cloud-hosted environments. This feature enables the management of integrations for multiple SAP B1 company databases from a single B1iF instance, offering greater flexibility and resource optimization for cloud deployments.

3.4. Be One Integration Platform

The BE1S Integration Platform was developed to address a critical challenge in working with non-standardized file formats within the B1if (see section 3.3).

Traditional solutions using B1if often require dealing with structured XML data and rely heavily on XSLT transformations or a visual interface for integration logic. However, this approach can become unmanageable and limiting when dealing with less structured data or when advanced logic needs to be implemented.

To overcome these limitations, the BE1S Integration Platform introduces a more flexible and developer-friendly approach. It functions as a framework capable of executing different modules or scenarios, much like B1if. However, instead of relying on XSLT or graphical tools, the modules on BE1S are developed directly in C#, allowing for greater expressiveness, control, and maintainability of complex logic.

The framework itself provides several essential services and tools to support module development. It offers secure connections to key systems, including file systems, SQL databases, and SAP B1. These connections are managed internally, thereby removing the burden of low-level configuration from developers and ensuring consistency across modules.

Additionally, the platform features robust logging and debugging capabilities, enabling developers to effectively trace issues during both development and runtime. This ensures that integration processes can be monitored closely, contributing to higher stability and faster troubleshooting.

Perhaps most importantly, BE1S enforces a predefined rule structure, implemented through a Class Factory mechanism based on a Manager-Worker paradigm. This architectural pattern requires each module to follow a structured sequence of steps, ensuring that processes are executed consistently and in a manner that is auditable and reproducible. Each step in this workflow can be independently monitored and validated, promoting reliability and scalability across integration scenarios.

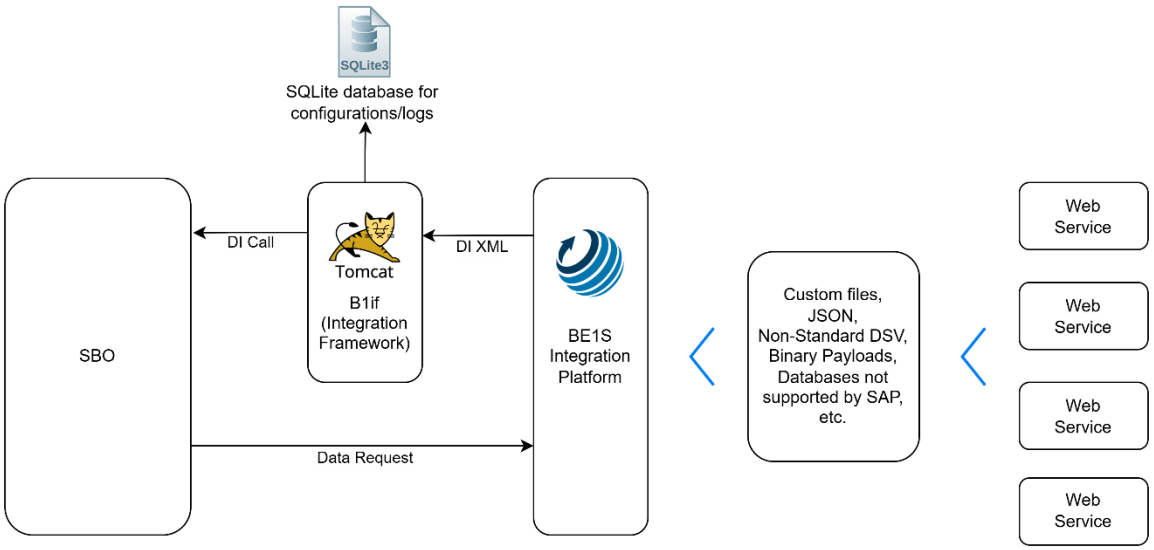

Figure 1 - BE1S Integration Platform

In Figure 1, the architecture of the BE1S Integration Platform can be seen, highlighting the core components and the interactions between modules, the framework, and external systems, including file storage, SQL databases, and SAP B1.

The diagram illustrates a bidirectional data integration workflow between SAP B1 (SBO in the diagram) and external systems, enabled by the BE1S Integration Platform and the B1if.

On the inbound side, data originates from web services or non-standard sources such as custom files, JSON payloads, delimiter-separated values (DSV), binary payloads, or databases not natively supported by SAP. These inputs are first processed by the BE1S Integration Platform, which normalizes and transforms the data into DI XML format. The

messages are then passed to B1if, hosted on Tomcat, which translates them into DI Calls for consumption by SAP B1, where the business transactions are executed.

On the outbound side, SAP B1can initiate a Data Request that is passed to the BE1S Integration Platform. From there, the platform translates and distributes the information to external targets, enabling integration with web services, non-standard data formats, and databases that are not supported.

An SQLite database supports both processes by providing lightweight storage for configuration and logging.

In this way, the architecture ensures seamless and flexible two-way communication between SAP B1 and a wide range of external systems, regardless of format or protocol.

4. Internship Program

The Internship Program serves as a structured plan designed to guide the training and development of interns within the organization. It establishes the overall objectives, scope, and expected outcomes of the internship while providing a clear roadmap for implementation. To ensure transparency and effectiveness, besides an explanation of the onboarding period, this section is organized into two subsections: the timeline, which outlines the phased schedule and key milestones of the program, and the methodology, which describes the strategies, processes, and resources employed in its design and execution.

The internship program was divided into two essential parts:

- Onboarding, which lasted one and a half months, aimed at acquiring and consolidating knowledge about SAP B1 and the various integration frameworks and platforms to be used throughout the internship. This phase served as preparation for the tasks and responsibilities that would follow during the rest of the internship and will be discussed in more detail in the following paragraphs.
- Participation in different integration projects, which took place during the remaining period of the internship. This part of the program will be described in the next chapter of the report (see section “Integration Projects” on page 19).

The onboarding process for the role to be performed at the company began with completing various Learning Journeys from the SAP portal. It started with a consultancy perspective, aiming to identify the multiple functionalities of SAP B1, as well as the different business processes and their corresponding workflows. This information proved highly relevant in consolidating knowledge and later in carrying out various integrations, providing a deeper understanding of how SAP B1 functions at a functional level before delving into its development aspects.

Among the Learning Journeys completed, the following stand out:

- Learning Journey: SAP Business One - 10.0 Implementation (English)

This Learning Journey [15] provided a comprehensive overview of the implementation process of SAP B1 version 10.0. It focused on key functional areas, including financials, sales, purchasing, inventory, and reporting. The course covered the best practices, configuration steps, and the general workflow of a typical SAP B1 implementation, offering a consultant’s perspective on how to set up and tailor the system according to business needs.

- Learning Journey: SAP Business One: Developer

This Learning Journey [16] was oriented towards technical users and developers. It introduced the SAP B1 SDK (see section 3.2), focusing on tools like the DI API, UI API, and Service Layer. Through this course, a solid foundation was built for extending, integrating, and customizing SAP B1 solutions. It played a crucial role in understanding how to interact with the system programmatically and laid the groundwork for developing integration projects during the internship.

After the initial learning period about SAP B1 and its processes, some simple exercises were conducted using B1if [14] to acquire skills and gain familiarity with the framework.

The learning process with this framework primarily focused on creating various scenarios and steps to automate simple processes, such as extracting documents from SAP B1, as well as importing and updating records, managing inventory transfers, and other tasks.

The final part of the onboarding process with the BE1S development team involved acquiring skills in developing various integrations utilizing the BE1S Integration Platform and B1if. Based on multiple examples, predominantly from previously executed projects, several straightforward exercises were undertaken to illustrate the procedures for extracting and importing numerous documents to and from SAP B1.

With onboarding complete, there was sufficient knowledge to start participating in and working on various integration projects, which will be detailed in the following sections (see section 5). Some projects utilized the BE1S Integration Platform, while others were developed with B1if.

4.1. Timeline

In Figure 2, a time-lapse of the internship's duration can be seen, from the beginning of onboarding on October 1, 2024, to the end of the internship on June 12, 2025. In this time-lapse, the duration of onboarding, as well as all the projects, is visualized using different colored bars, alongside the time dedicated to other tasks simultaneously with the integration projects.

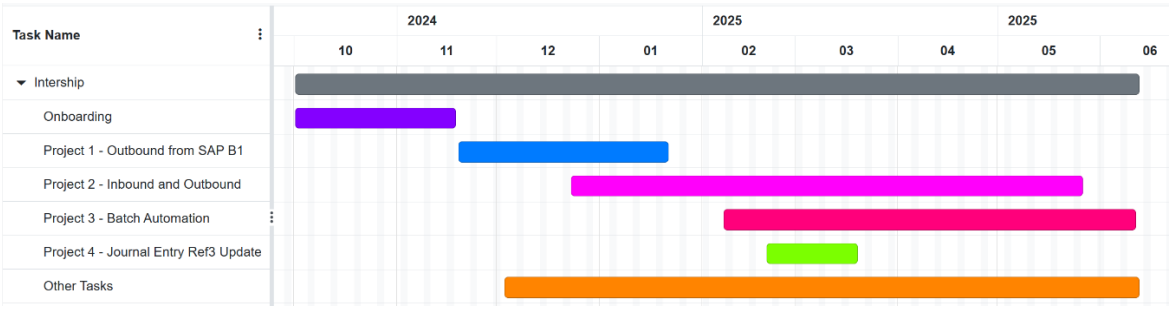

Figure 2 - Internship Timeline

As seen in Figure 2, there were multiple occasions during the internship where I worked on different projects and tasks.

While the onboarding took place for about the first month and a half of the internship, that was probably the most linear moment in the internship. That time was used to learn the SAP B1 foundational modules and processes, as well as its Development SDK and integration framework. In the later weeks, the attention moved to different exercises both on B1if (see section 3.3), where hands-on experience was gained building B1if scenarios, similar to real-life projects, and for the BE1S Integration Platform (see section 3.4) C#-based modules were developed, leveraging its built-in services, and applying the Manager-Worker

paradigm for structured processes. By the end of that period, it would be possible to design, implement, and troubleshoot simple integrations across these platforms.

After that initial period, the involvement in real integration projects started. In Figure 2, all four projects developed during the internship can be seen, respectively, Project 1 – Outbound from SAP B1 (see project 5.1), Project 2 – Inbound and Outbound (see project 5.2), Project 3 – Batch Automation (see project 5.3) and Project 4 – Journal Entry Ref3 Update (see project 5.4).

During the whole duration of the internship, after the onboarding, besides the participation in different integration projects, there was a diverse range of tasks done (see section 6Other Projects/Tasks), such as documenting existing add-ons, providing support and fixes for previously developed projects, and developing an installer solution for the integration platform.

4.2. Methodology

For most projects, the same methodology was used, as seen in Figure 3. At the beginning of each project, a High-Level Document was provided. This document had already been prepared by the relevant stakeholders or planning teams and contained all the requirements, along with the proposed solution. It clearly defined the overall scope of the project, outlined what was expected in terms of both functional outcomes and technical specifications, and served as the central reference point throughout the development process.

Once the High-Level Document was received, the next step involved structuring the project. This included determining whether to assign different managers to specific areas of responsibility, identifying the most efficient workflow for task execution, and assessing potential risks associated with these assignments. Measures were implemented to minimize the likelihood of errors or issues, including reviewing task dependencies, selecting suitable communication and coordination practices, and establishing mechanisms for quality assurance and error prevention.

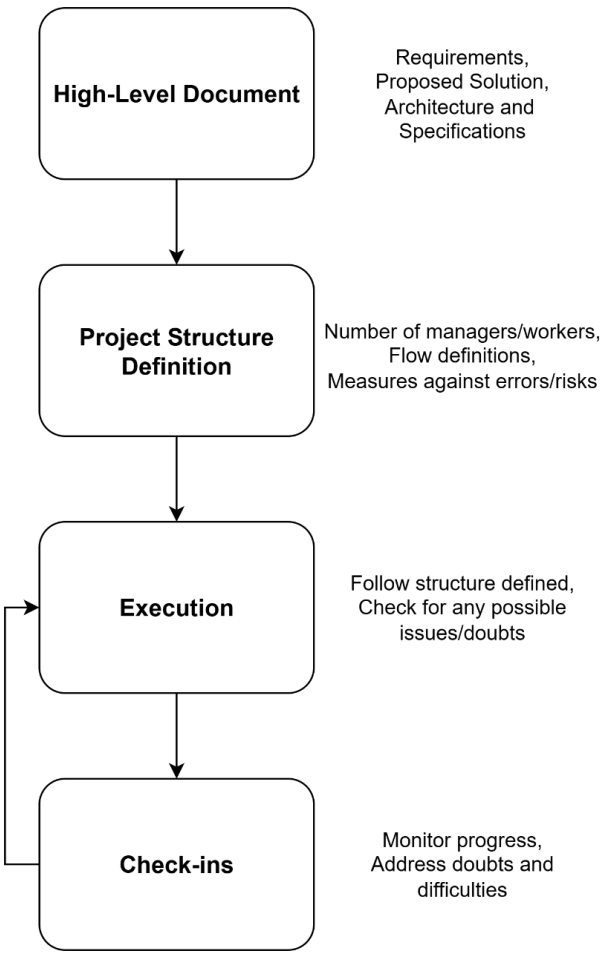

Figure 3 - Project methodology

Throughout the development phase, the project's progress was monitored by an assigned supervisor. Periodic check-ins were conducted to verify alignment with the original requirements, confirm that milestones were being met, and address any questions or challenges that had arisen. These supervisory reviews provided ongoing oversight and support, helping to ensure that the work stayed on track without micromanaging daily activities.

5. Integration Projects

The participation in various integration projects was a fundamental part of the internship, where all knowledge acquired during onboarding proved invaluable, and much more still needed to be learned.

There are different types of integration projects to be developed. Still, during the internship, the focus was on integrations that required data exchange between SAP B1 and a third-party application, and vice versa. This implies that data was fetched and manipulated to fit the corresponding standard need for the receiver of the data. However, not all integrations developed were about data exchange. There were also projects (see sub-section 5.3, for example), where action was needed, based on SAP B1 information, and that was the goal of the integration.

In the following sections, the various projects developed during the internship will be described in more detail.

The first project (see sub-section 5.1), was a simple integration, consisting of an outbound flow of information to a third-party webservice. For this, the main challenge consisted of learning how to navigate and work with procedures on SAP HANA (see section 3.1), since it had only been used with MSSQL up to that point. Besides that, the integration was very consistent with the exercises carried out during the onboarding experience.

For the second project (see sub-section 5.2), besides an outbound flow like described for the project before, there was a new layer of complexity added, not only for the insertion of a new inbound flow, which hadn't been worked with yet, but also for the existence of a central database connected to all the different companies used in the integration. The use of the central database also necessitated new measures to ensure accurate data insertion in other companies, as some of the data received was specific to individual companies.

The third project (see sub-section 5.3) wasn’t about sending or receiving any data; the primary purpose of this integration was to act based on information and fields of SAP B1. This involved retrieving specific data from SAP B1, verifying its value, and taking action in accordance with a predefined set of rules to achieve the desired objective. For this project, the procedures developed were more complex, as a significant amount of data from multiple tables was required to update the records correctly.

The fourth and final integration project (see sub-section 5.4) described in this report, same as the previous one, was about acting based on SAP information. The objective of the project was to retrieve the most recent 100 journal entries and, based on their data and a mapping, update them and fill in the necessary fields.

5.1. Outbound from SAP B1

The first project involved extracting and sending various data (Master Data and Transactional Data) from SAP B1 in XML format. For this project, it was decided to use the

BE1S Integration Platform (see section 3.4), in other words, the development of a DLL in C# that would meet all the requirements provided by the client.

These requirements included the extraction of Business Partners, Items, and Sales Orders, all according to a predefined mapping. The extracted data then needed to be converted into XML format and sent to the client's designated endpoint.

The approach defined for this project involved extracting data into separate XML files for each data type (Business Partners, Items, and Sales Orders) and then sending each XML file to a client endpoint specified in the integration settings, which were stored in a SQLite database.

For the data extraction, different procedures were developed in SAP HANA (see section 3.1) to return the fields required by the mapping for each data type.

Throughout the development process, some additional requirements were introduced, which resulted in a few differences in how the integration functioned, although the overall approach remained unchanged. Additional requirements included extracting only the Sales Orders from the last three months and creating a separate XML file for each object, rather than generating a single file per data type.

Overall, this first project provided a valuable introduction to real-world integration work, combining both technical implementation and adaptation to client requirements. On the one hand, it reinforced the importance of structured data extraction and transformation processes, as well as the need for explicit mappings and endpoint configuration. On the other hand, it demonstrated the dynamic nature of integration projects, where evolving client requirements can lead to changes in scope and technical approach. Despite these adjustments, the project successfully delivered the expected outcome, showing that the chosen architecture, using the BE1S Integration Platform and custom C# development, was both flexible and effective.

From a learning perspective, several key takeaways can be retained. First, the project introduced new knowledge of SAP HANA, particularly in creating queries and stored procedures tailored to integration scenarios. Second, it strengthened abilities in C# development for middleware components, especially in handling XML formatting, data transformation, and endpoint communication. Third, it enhanced practical skills in integration design, such as separating data flows per object type, configuring endpoints, and managing integration parameters through a database. Finally, the experience highlighted the value of adaptability and problem-solving, as unanticipated requirements, like filtering sales orders by date range or splitting files at the object level, needed to be addressed without altering the overall architecture.

Therefore, the project not only delivered a functional solution for the client but also served as a foundation for developing both technical expertise and professional skills that will be essential for more complex integration challenges in the future.

5.2. Inbound and Outbound

The second project to be developed introduced a higher level of complexity, offering the opportunity to work with an Inbound perspective for the first time, in addition to the Outbound approach that had been used in previous projects and exercises.

The requirements for this project included the inbound processing of Business Partners, the outbound transmission of Cost Centers and Bank Details, and ensuring that only new or updated data would be sent.

For the development of this integration, carried out using the BE1S Integration Platform, simple queries and procedures to retrieve the data were not sufficient. It was necessary to introduce a certain level of complexity by creating and using auxiliary tables to store and prepare the received data before it could be inserted into the correct database.

Developed to incorporate several different companies, the project required the use of multiple databases. Therefore, the integration had to be designed to work independently of the number of companies configured in the local integration database (SQLite).

The development of the outbound (information sending) part, being similar to the previous project, did not present a significant challenge. It involved creating a few procedures in SAP HANA to retrieve the necessary information, according to the defined mapping, and sending it to a client endpoint.

The process of receiving inbound data and correctly creating or updating the Business Partner became more complex with the use of a central database to receive this information, which would then be replicated to each target database. Since specific Business Partner details, particularly banking information, are particular to each company and country, they could not be inserted directly into the central database. This created the need to store such data in an auxiliary table and to develop a worker that runs at specific intervals, verifying the creation of the Business Partner in its respective company database and adding the corresponding banking information accordingly.

For the first time during the internship, it was necessary to create a User Defined Field (UDF) to store the original ID provided in the Inbound data, as well as a User Defined Table (UDT) to maintain a log of the Business Partners that were inserted and those that failed.

UDFs are additional fields that appear on existing SAP B1 screens, such as Sales Orders, Deliveries, Business Partners, etc., which help track integration-specific details. At the same time, UDTs are custom tables created to store structured data that doesn’t belong in standard screens. They are mainly used for tracking, logging, or storing settings for the integration.

In Figure 4 and Figure 5, the flow used for the Inbound process can be seen.

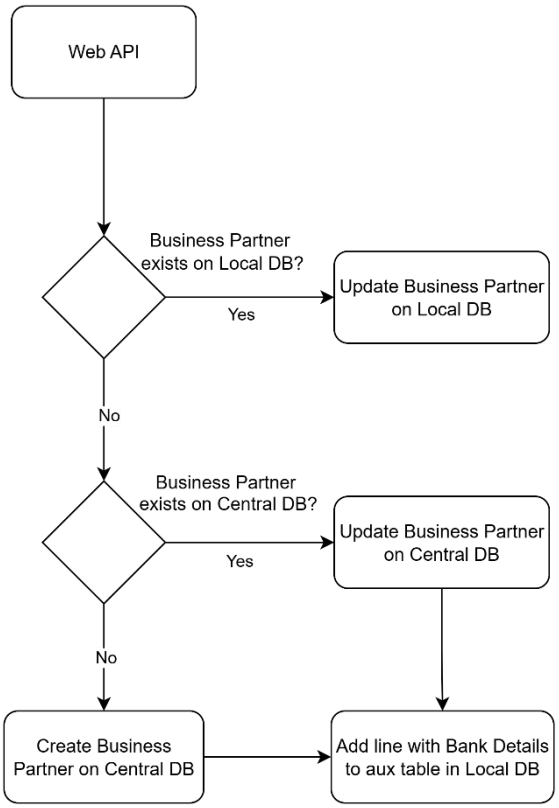

Figure 4 - Inbound Process part 1

The process begins when data is received through a Web API, triggering the logic to manage a Business Partner's master data. The system first checks whether the Business Partner already exists in the local database. If it does, the record is updated accordingly with all the information. If not, the system then checks for the existence of the Business Partner in the central database. If found, the record is updated there; if not, a new Business Partner record is created in the central database.

Regardless of the path taken, whether updating or creating records on the central database, an additional step is always performed at the end: a line containing the Business Partner’s bank details is added to an auxiliary table in the local database. This ensures that no bank details are added to the central database; instead, they are saved in a local table until the correct Business Partner is replicated to its corresponding company, allowing for later updates with the data.

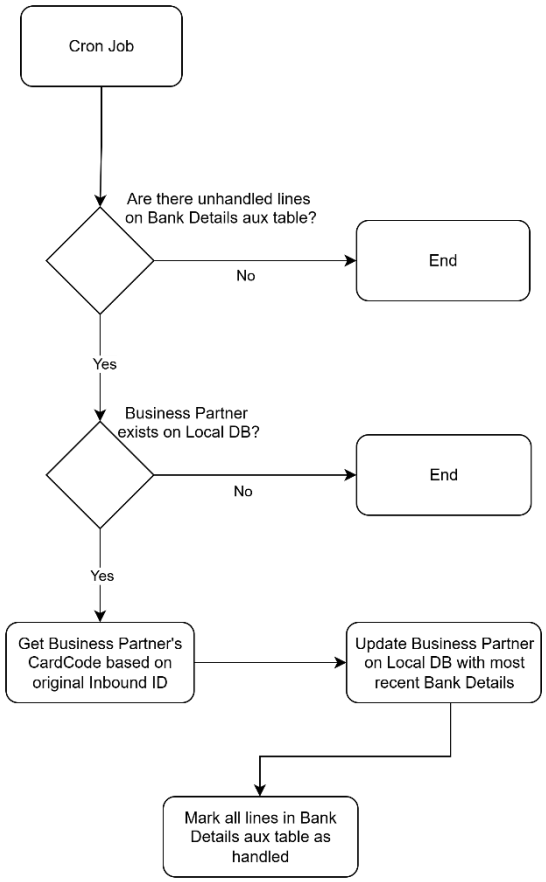

Figure 5 - Inbound Process part 2

This process is triggered by a scheduled cron job, which checks for unhandled lines in the bank details auxiliary table. If there are no such entries, the process ends. If there are pending lines, the system verifies whether the associated business partner exists in the local database. If not, the process ends. If the business partner does exist, the system checks whether it can determine the business partner's card code using the original Inbound ID. Then, it updates the business partner's master data in the local database using the most recent bank details from the auxiliary table.

Once the update is complete, all lines in the auxiliary table corresponding to that Inbound ID are marked as handled, ensuring they won't be processed again. This automated flow helps keep bank information current and consistent, eliminating the need for manual intervention.

This second project represented a step forward in terms of both scope and complexity, as it introduced the Inbound perspective in addition to the more familiar Outbound approach. It required dealing with multiple company environments, handling auxiliary data, and utilizing more advanced features of the BE1S Integration Platform. By combining outbound flows for Cost Centers and Bank Details with inbound processing for Business Partners, the project offered a broader and more realistic view of the integration challenges faced in enterprise systems.

From a technical standpoint, the project made it clear that simple query-based extractions were no longer sufficient. The need to incorporate auxiliary tables, scheduled workers, and central vs. local databases added a new layer of complexity, requiring more careful design of both data structures and process flows. The development of a cron job for asynchronous bank detail updates, along with the use of UDFs and UDTs, provided valuable experience with customization features in SAP B1. These elements demonstrated how flexibility in the platform can be leveraged to handle data logging, identification tracking, and integration-specific metadata.

In terms of learning, several essential abilities were strengthened. First, the project deepened understanding of inbound integration flows, particularly the handling of data consistency when information is distributed across multiple company databases. Second, it improved skills in database design and orchestration by requiring the creation of auxiliary structures and background workers to manage deferred updates. Third, it built competence in SAP customization through UDFs and UDTs, showing how these tools can extend the system to meet integration-specific needs. Finally, the project reinforced the importance of data governance and reliability, since ensuring that only new or updated records were transmitted was a key requirement.

Overall, this project highlighted both the technical challenges and the architectural considerations of multi-company integrations. Beyond expanding technical expertise in SAP HANA, C#, and the BE1S platform, it fostered critical problem-solving skills for designing robust inbound and outbound processes and maintaining data consistency across distributed environments. These insights are directly applicable to more advanced integration scenarios, where system complexity and business rules often evolve in parallel.

5.3. Batch Automation

This project focused on automating the batch (group of items) allocation process for Sales Orders and was developed using the BE1S Integration Platform. The requirements for this project consisted of the creation of multiple UDFs for the batches and users, validation of batches in comparison to the sales order’s lines, and creation of a UDT for logging.

As mentioned before, it was necessary to create different UDFs with the sales order’s number, the order line to allocate the batch, as well as a status flag to check if the batch allocation was already handled or not.

Some validations were necessary before the allocation of the batches to the order could be done, from checking if the line associated with the batch actually exists on the order, to making sure that the item of the batch and the one on the order’s line match and are in the same warehouse, all of those checks are necessary to make sure the batch can be allocated the order.

Besides the previous validations, comes the rule given by the client in terms of quantities: if the quantities of the batches correspond to the quantity of the line then the batch can be assigned; if the quantities are different but the difference is less than or equal to one then the batch can be assigned, however the order’s line quantity must be updated before the

batch allocation can be done; if the quantities to allocate are higher than the line quantity but the difference is bigger than one then the batch allocation is not possible and an error message should be sent; if the quantity to allocate is lower than the line quantity but the difference is bigger than one then the batches can still be allocated, however the quantity update is not necessary.

At the end of each order that is handled, in case of any errors or failed allocation, an SAP System Message is sent to all users with the notification flag saved to Yes (UDF created on the user master data).

For this project, there was also the necessity to change SAP B1's Transaction Notification procedure to ensure that whenever a batch was removed from a sales order, its status UDF would be updated to "not handled".

This project introduced a significantly higher level of complexity, particularly in the creation of stored procedures, as it involved the use of numerous different SAP B1 tables and more complex logic to obtain the precise information required for this solution. It also turned out to be a lot more complicated to test, in comparison to the integrations developed before.

The tests had to consider all the different scenarios possible. That meant considering the number of batches to be allocated, if there were already batches allocated to the order, whether the order had been partially delivered already, and if it went beyond the 1m3 range that was permitted.

In the following paragraphs, some of the tests that were conducted will be described in more detail.

- Order quantity has enough space for the 'to be allocated stock'. Quantity: 10, to be allocated: 5.
    - Expected result: Success, batches allocated without issue.
- Order quantity has enough space for the 'to be allocated stock'. Quantity: 10, to be allocated: 9.2 (margin +/- 1 m3).
    - Expected result: Success, batches allocated after order quantity is updated.
- Order quantity does not have enough space for the 'to be allocated stock'. Quantity: 10, to be allocated: 10.2 (margin +/- 1 m3).
    - Expected result: Success, batches allocated after order quantity is updated.
- Order quantity does not have enough space for the 'to be allocated stock'. Quantity: 10, already allocated stock/batches against order line: 5, to be allocated: 5.2, RDR1.Quantity to be increased to 10.2 (margin +/- 1 m3)
    - Expected result: Success, batches allocated after order quantity is updated.
- Order quantity does not have enough space for the 'to be allocated stock'. Quantity: 8, already delivered quantity: 6.75, to be allocated: 2. 6.75 + 2 = 8.75 (margin +/- 1 m3)

    - Expected result: Success, batches allocated after order quantity is updated.
- Order quantity does not have enough space for the ‘to be allocated stock’. Quantity: 10, already delivered quantity: 6.75, already allocated stock: 2, to be allocated: 2.
  6.75 + 2 + 2 = 10.75
    - Expected result: Success, batches allocated after order quantity is updated.
- Order quantity does not have enough space for the ‘to be allocated stock’. Quantity: 10, already allocated stock: 6.45, to be allocated: 5 (1 batch).
    - Expected result: Failure, add error message to UDT with “Outside 1m3 range”
- Order quantity does not have enough space for the ‘to be allocated stock’. Quantity: 10, already allocated stock: 6.45, to be allocated: 5 (2 batches of 2.5).
    - Expected result: One batch is allocated without updating quantity (6.45 + 2.5 = 8.95) while the other batch is not allocated, and an error message for that batch is added to UDT with “Outside 1m3 range”

This was the primary structure that underwent quality testing, which had to be completed before the integration could be considered for the production environment.

This third project marked another increase in complexity compared to the previous integrations, as it focused on automating the batch allocation process for Sales Orders. Unlike the earlier projects, which were primarily centered on the extraction, transformation, and delivery of data, this one required working intensely with SAP B1 internal logic and database tables. The project combined the development of UDFs, UDTs, stored procedures, and modifications to SAP B1’s Transaction Notification, resulting in a solution that extended beyond data flow and addressed operational business rules.

From a technical perspective, the project strengthened knowledge of database logic and validations, especially by introducing a broad set of conditions and rules for batch allocation (matching items, verifying warehouses, handling partial deliveries, and applying client-specific tolerance ranges). This complexity required designing stored procedures that pulled information from multiple SAP B1 tables and orchestrated checks across them. The challenge was compounded by the need to create error handling and logging mechanisms, ensuring that failures were recorded correctly in the UDT and communicated to end users through system messages.

In terms of learning, this project contributed significantly to the development of advanced problem-solving and debugging skills, as testing had to account for a wide variety of scenarios with different quantities, allocations, and delivery statuses. It also improved proficiency in quality assurance, since designing and executing test cases became a central part of ensuring the integration’s reliability. Moreover, the project enhanced familiarity with SAP customization and procedural automation by leveraging UDFs and UDTs in conjunction with triggers to create a system that could operate seamlessly within the client’s business rules.

Overall, the project provided a solid foundation for understanding the intersection of integration and business process automation. By balancing technical development with thorough testing and client-driven requirements, it demonstrated how integration projects can extend beyond system connectivity to support and enforce operational workflows directly.

5.4.Journal Entries Update with B1if

This was the first project fully developed with B1if (see section 3.3), bringing a need to have a better understanding of the framework's primary language, XSLT 1.0.

The main requirement for this project was the update of a specific field for each Journal Entry (JE) in SAP B1, based on a defined mapping and rules.

In SAP B1, a Journal Entry is the fundamental accounting transaction that records financial events in the company's general ledger [17]. Each entry consists of at least two lines, reflecting the principles of double-entry bookkeeping, where one account is debited and another is credited for the same amount, ensuring that total debits always equal total credits. Journal Entries can be created directly by users for adjustments and corrections or automatically generated by system transactions such as sales, purchasing, payments, and inventory movements. They are essential for maintaining accurate financial records, enabling audit trails, supporting compliance with accounting standards, and providing the basis for financial reporting and analysis within the organization. Every financial transaction (sales, purchases, payments, inventory movements, etc.) ultimately creates a journal entry in the background.

A JE contains much more information and fields, like the `ShortName`, `Geography`, and `Ref3` fields, which are the ones analyzed and processed for this integration.

Each time the integration runs, it retrieves 100 Journal Entry IDs. For this purpose, we created a UDF in the JE to check if it has already been updated.

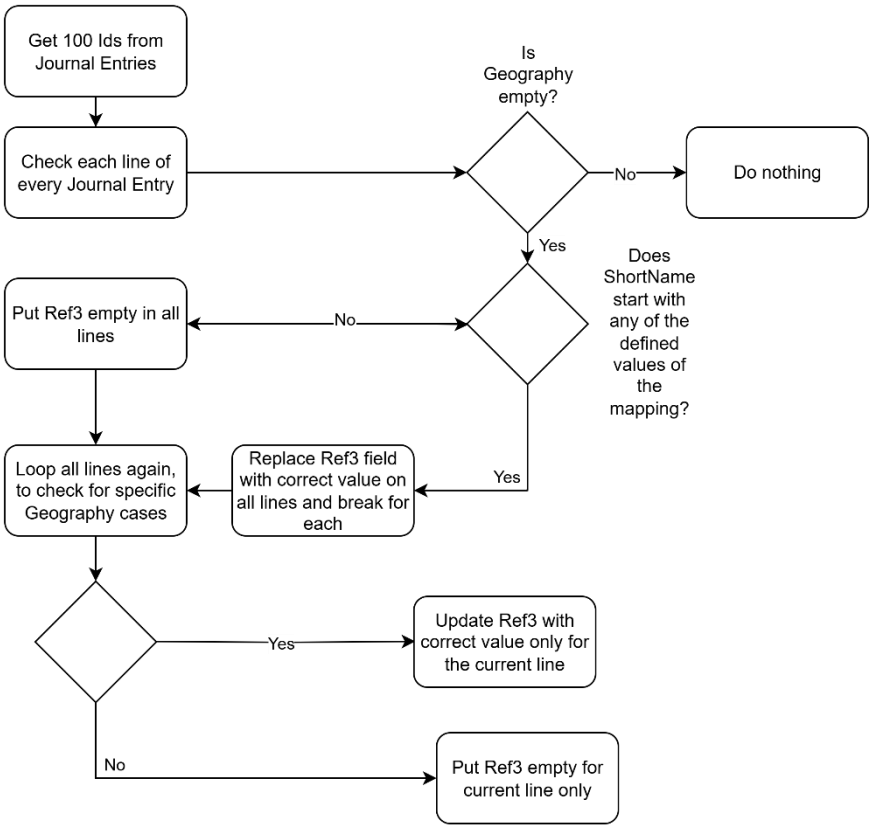

Figure 6 - Flow diagram for Ref3 Update

In Figure 3, it's possible to see the flow of the integration in a more detailed way.

This process is designed to analyze and update the `Ref3` field in journal entry lines based on two key factors: the `ShortName` and `Geography` fields. It begins by retrieving a batch of 100 journal entry IDs, and then it iterates through each line of every journal entry to evaluate the data.

The first step is to determine whether the `Geography` field is empty. If the `Geography` field contains data, the system takes no action on that line, assuming it doesn't need correction. However, if the field is empty, the process then evaluates whether the line's `ShortName` begins with any of the predefined values in a mapping configuration.

If no match is found between the `ShortName` and the defined mapping, the system clears the Ref3 field for all lines of the journal entry, treating it as a case without valid reference data.

On the other hand, if a matching `ShortName` prefix is detected, the process updates the `Ref3` field with the correct mapped value across all lines in that journal entry and exits this loop early as a way to prevent redundant checks once a match has been confirmed.

After a second loop begins to address specific Geography-related cases, each line is re-evaluated to see whether the `Geography` field is not null and corresponds to any defined values from the mapping.

If both conditions are met, the line's `Ref3` field is updated with the correct mapped value, but only for that specific line. If the `Geography` does not match a known value, the system empties the `Ref3` field for just that line, ensuring only valid mappings populate the reference field.

In the following few tables, there are a few examples of how the integration is supposed to work.

BP Code	Ref. 3	Geographic
LV1546	LAT	
1234	LAT	
1235	LAT	
1236		PHAME

Table 3 - JE Update Scenario 1

In Table 3, the `Geography` field is empty, so when the `ShortCode`, which corresponds to the "BP Code" in this scenario, is checked and starts with one of the prefixes found in the mapping (LV, in this case), then the `Ref3` field is updated to "LAT" in all of the lines. After that, a second loop is done, checking only the Geography field, since it found a value that did not exist in the mapping (LIT/LT, LAT/LV or EST/EE), then the `Ref3` field is forced to be empty.

BP Code	Ref. 3	Geographic
EE1546	EST	
1234	EST	
1235	LAT	LAT or LV
1236	EST	

Table 4 - JE Update Scenario 2

In the second scenario, seen in Table 4, while the first loop worked in the same way as Table 3, by updating all the `Ref3` field to "EST" based on the `ShortCode`, the second loop found a `Geography` field that corresponded to the mapping (LIT/LT, LAT/LV or EST/EE), therefore the value of the `Ref3` field for that specific line was updated to its corresponding value.

This project marked a turning point in the internship experience, as it was the first integration fully developed within B1if, relying entirely on its native language, XSLT 1.0. Unlike previous projects where C# and SQL-based procedures were the main drivers, this one demanded a deeper understanding of XSLT transformations and the B1if framework's processing logic. This represented both a technical challenge and a valuable opportunity to broaden skills in working with SAP's official integration tool.

The project requirements, updating the Ref3 field of Journal Entries (JE) according to a predefined mapping of `ShortName` prefixes and `Geography` values, highlighted the importance of building structured, rule-based transformations. It was also the first time that

data handling had to be approached through iterative loops, which required precise control over conditions, filtering, and early exits to optimize performance. The introduction of a UDF to track which Journal Entries had already been processed also demonstrated the need for governance in integration solutions, ensuring efficiency and preventing unnecessary reprocessing.

From a learning standpoint, the project contributed to the development of proficiency in XSLT 1.0, particularly in handling nested conditions and multi-step evaluation flows. It also reinforced the concept of data integrity and consistency, since the integration had to decide when to update fields, when to clear them, and when to leave them untouched. The dual-loop mechanism provided a practical lesson in designing multi-phase logic, where initial mappings are applied globally before more specific conditions refine the result. Furthermore, by working directly with Journal Entries, the financial backbone of SAP B1, the project provided valuable exposure to the accounting dimension of ERP systems and the importance of precision when handling critical financial data.

Finally, this project underscored the challenges of working with XSLT 1.0 in a production-like scenario, where its limitations (such as a lack of grouping functions and reliance on verbose logic) became evident. Overcoming these challenges not only improved technical skill with the language but also strengthened problem-solving abilities, adaptability, and the capacity to optimize within constraints.

6. Other Projects/Tasks

During the internship, while the focus was indeed on SAP integrations utilizing both the BE1S Integration Platform and B1iF, the experience also included a broader range of tasks.

A detailed analysis and documentation of various existing integration projects was conducted, involving a careful review of their configurations and code to understand how they were set up for both deployment and testing, as well as the impact each configuration had on the integration's performance. This process was essential for ensuring clarity, maintainability, and reproducibility of these complex integration setups, providing insights into system stability and future scalability.

Another distinct and technically challenging task involved developing a custom installer for the Integration Platform. After some investigation into suitable installer technologies, it was decided to use Inno Setup, mainly for its script-driven approach, allowing for highly customized installation logic [18]. This decision kicked off a dedicated learning curve to master its capabilities, which included understanding scripting for custom actions, file extraction, registry modifications, and user interface customization within the installer itself. This project wasn't just about delivering a practical deployment tool; it also provided valuable hands-on experience in software packaging, distribution strategies, and managing the entire deployment lifecycle, from initial setup to uninstallation. Although it still has some way to go and requires further thorough testing, the experience was valuable during the internship.

Beyond these other tasks, contributions were also made to several diverse projects. This included helping support other integrations by fixing missing components or identifying the cause of bugs, as well as working on specific parts of different integrations and configuring new companies on SAP for clients.

7. Conclusions and Future Work

The internship program was structured as a gradual learning process that combined theoretical training, hands-on exercises, and active participation in real integration projects. The initial onboarding period, lasting approximately one and a half months, provided a strong foundation in both the functional and technical aspects of SAP B1. Through dedicated Learning Journeys, it was possible to acquire a consultant's perspective of business processes and a developer's understanding of the SAP SDK tools. This stage was further reinforced by practical exercises in both the B1if and the BE1S Integration Platform, which prepared the ground for tackling real projects with confidence.

Once the onboarding phase was complete, the focus shifted towards contributing to different integration projects. The first project introduced the fundamentals of outbound data extraction, mapping, and transformation into XML, laying the groundwork for understanding the integration workflow. The second project expanded this knowledge by introducing inbound data flows, central vs. local database synchronization, and the use of auxiliary tables, which required more advanced logic and a broader view of integration scenarios. The third project raised the technical challenge even further, focusing on automating batch allocation for sales orders. It required in-depth work with SAP B1 tables, validations, and robust testing procedures to ensure accuracy across different business cases. Finally, the fourth project marked a significant milestone by working entirely with B1if and XSLT 1.0 to update Journal Entries according to predefined mapping rules.

During the internship, although the primary focus remained on SAP integrations through the BE1S Integration Platform and B1if, the scope of work extended well beyond project development. Part of the experience involved analyzing and documenting existing integrations, reviewing their configurations and code to gain a deeper understanding of deployment logic, testing strategies, and performance considerations. This exercise not only strengthened knowledge of integration design but also contributed to improving clarity, maintainability, and scalability for future work.

Another relevant aspect was the opportunity to explore software packaging by developing a custom installer for the Integration Platform. After evaluating alternatives, Inno Setup was chosen for its flexibility, which required learning how to handle scripting, file operations, registry modifications, and interface customization. While still under refinement, the project provided valuable insights into deployment processes and broadened technical versatility.

Finally, the internship also included contributing to various day-to-day activities such as troubleshooting bugs, adding missing features to existing solutions, assisting with client setups, and supporting other developers. These smaller but diverse tasks reinforced adaptability and provided further opportunities to apply problem-solving skills in authentic contexts.

Taken together, these projects and additional tasks represent a gradual progression in both technical and problem-solving abilities. Each new challenge built upon the experience

of the previous one, encouraging autonomy while still benefiting from guidance and supervision. The combination of working with BE1S, SAP HANA, MSSQL, B1if, and even auxiliary tools such as Inno Setup provided a diverse technological environment that contributed to a comprehensive understanding of integration development.

Overall, the internship provided an opportunity to acquire a comprehensive range of skills, encompassing technical competencies such as stored procedure development, data mapping, XML handling, framework-specific logic, and software deployment strategies. Additionally, it fostered transversal abilities, including structured problem analysis, testing methodologies, and adapting solutions to client requirements. Equally important was the experience of working within a professional development team, gaining insights into approaching projects methodically, communicating effectively, and managing tasks within designated timelines.

From the initial onboarding phase, which established a solid foundation in SAP B1 and its integration frameworks, to the execution of various integration projects, each stage contributed to honing technical expertise and problem-solving skills. The projects provided exposure to real-world challenges, from data extraction and transformation to process automation and financial updates, while encouraging adaptability and analytical thinking. Additional tasks, such as documenting existing integrations, creating a custom installer, and supporting ongoing projects, further enhanced the experience by expanding the scope beyond core integrations. Collectively, these activities not only solidified knowledge of integration platforms but also improved abilities in collaboration, troubleshooting, and delivering structured solutions, resulting in valuable insights and skills applicable in future professional settings.

In conclusion, the internship served not only as an academic requirement but also as a meaningful professional experience. It allowed the consolidation of theoretical knowledge in cybersecurity and informatics with practical application in enterprise systems integration. The balance between structured training, progressively complex projects, and supportive supervision created an environment that allowed both learning and contribution to be possible. The skills and insights gained throughout this program form a strong basis for future professional challenges in the areas of systems integration, enterprise solutions, and IT development.

Looking ahead, there is considerable room for further work and professional growth. Some of the projects initiated during the internship, such as the custom installer, could be refined and extended to improve reliability and usability. Additional documentation and user onboarding materials for the BE1S platform could help enhance the ramp-up time for future contributors. In addition, the growing importance of cloud-based ERP deployments highlights the need to expand knowledge into hybrid and cloud integration solutions, ensuring adaptability to evolving client requirements. These directions represent valuable opportunities to continue building upon the foundation established during the internship.

Bibliography

[1] "About Us," [Online]. Available: https://www.beonesolutions.com/about-us/.

[2] J. W. Carter, "A Brief History of Enterprise Integration," in *The Enterprise Integration Handbook*, 2011.

[3] "SAP Business One and Artificial Intelligence," [Online]. Available: https://community.sap.com/t5/enterprise-resource-planning-blog-posts-by-sap/sap-business-one-and-artificial-intelligence/ba-p/13509478.

[4] "SAP Fiori," [Online]. Available: https://www.sap.com/products/technology-platform/fiori.html.

[5] "Technical information for SAP Business One," [Online]. Available: https://www.sap.com/products/erp/business-one/technical-information.html.

[6] "What is SAP HANA?," [Online]. Available: https://www.sap.com/products/data-cloud/hana/what-is-sap-hana.html.

[7] "Exploring Master Data and Documents," [Online]. Available: https://learning.sap.com/courses/managing-logistics-in-sap-business-one/exploring-master-data-and-documents.

[8] "SAP Business One SDK - Help Center," [Online]. Available: https://help.sap.com/docs/SAP_BUSINESS_ONE/a2c93a1c88a24f1faeee3d3f62fbde7a/a0c09b4a503a47a5b3a3dd839cc28890.html.

[9] "Integration Framework for SAP Business One," [Online]. Available: https://help.sap.com/doc/a0eb55e3a0644f6da74d97946db9823c/9.0/en-US/B1if%20Processing%20-%20Call%20B1%20Object.pdf.

[10] "Service Layer API Reference," [Online]. Available: https://help.sap.com/doc/056f69366b5345a386bb8149f1700c19/10.0/en-US/Service%20Layer%20API%20Reference.html.

[11] "Integration Framework for SAP Business One Technical Overview," [Online]. Available: https://help.sap.com/doc/saphelpiis_hc_b1_image_repository_consultant_training_additional_b1if_overview_pdf/9.0/en-US/B1if_Overview.pdf.

[12] "What XSLT 2.0 and 3.0 can do better than XSLT 1.0," [Online]. Available: https://fmforums.com/topic/110806-what-xslt-20-and-30-can-do-better-than-xslt-10.

[13] "Grouping Using the Muenchian Method," [Online]. Available: https://www.jenitennison.com/xslt/grouping/muenchian.html.

[14] "In Action - Integration Framework for SAP Business One," [Online]. Available: https://learning.sap.com/courses/in-action-integration-framework-for-sap-business-one.

[15] "SAP Business One - 10.0 Implementation (English)," [Online]. Available: https://help.sap.com/learning-journeys/05bee675e533438989c5e140ba2b90ad.

[16] "SAP Business One - Developer," [Online]. Available: https://help.sap.com/learning-journeys/9c396d18a04e479196b60b3cbf67c631.

[17] "General Ledger Accounting (FI-GL)," [Online]. Available: https://help.sap.com/docs/SAP_S4HANA_ON-PREMISE/651d8af3ea974ad1a4d74449122c620e/b050d7531a4d424de10000000a174cb4.html.

[18] "What is Inno Setup?," [Online]. Available: https://jrsoftware.org/ishelp/index.php.