Advantages of low-code on Intranet Portals

Enhancing the visualization of internal data in a major retail chain through low-code applications

Vasco José Sousa Lobo Azevedo Branco

Internship report proposal presented as partial requirement for obtaining the Master's degree in Information Management

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação
Universidade Nova de Lisboa

ADVANTAGES OF LOW-CODE ON INTRANET PORTALS

by

Vasco Lobo Branco

Internship report presented as partial requirement for obtaining the Master’s degree in Information Management/ Master’s degree in Statistics and Information Management, with a specialization in Technology and Information Systems.

Advisor / Co Advisor: Pedro Cabral

Co Advisor*:* Name

DEDICATION

To my Mother and Father, to whom I owe everything.
Thank you for all the patience and kindness.

ACKNOWLEDGEMENTS

First, I want to thank my fiancée Inês Ramadas. Without her constant support I would not be half the man I am today.
Second, I want to thank my brother JP, who has always been there for me when times get rough.
Third, I want to thank my Manager Oliver Grimm and my “Mentor” Jürg Sträuli. From the very first moment both welcomed me with open arms, and always made sure that my experience in Switzerland was the best, while guaranteeing that I grew both personally and professionally.
Last, but certainly not the least, I want to thank Catarina Gomes Madeira. Without her, this internship would never have happened in the first place.

ABSTRACT

LIDL, a major multinational retail chain with branches spread across the globe, has many internal processes that aren't centralized. In each country branch, information is dispersed within several platforms, in different formats, which in turn makes data harder to analyze, slowing down procedures that are frequently used within each division of the major retail chain.

With this problem in mind *LIDL* has decided to invest in *Low-Code*, giving the liberty to each country to develop its own internal portal to counter this problem. With this, each branch centralizes all its essential information in one place. By choosing low-code, *LIDL* has given each country the freedom of developing the necessary applications in record time, providing a way to experience *omnichannel* experiences without giant budgets and costly development teams.

The results of this study show why portal development should be done with *Low-Code*, the synergy that is built between the two concepts and the many advantages that follow. To defend these claims, the work done during my internship will be showcased and analyzed.

KEYWORDS

Portals, Data, Information, Low-Code, Application Development, Retail

INDEX

LIST OF FIGURES

LIST OF ABBREVIATIONS AND ACRONYMS

KMS	Knowledge Management System
LCDP	Low-Code Development Platform
LCAP	Low-Code Application Platform
SLA	System Log Agreement
ROI	Return of Investment

1 INTRODUCTION

1.1 LIDL History and Arrival at Switzerland

LIDL is a German international retail chain that began in 1973 in the town of Ludwigshafen. LIDL started as a small store with 3 employees and about 500 products and expanded in less than a century to a successful chain of around 11,200 stores throughout 32 countries, employing more than 310,000 people globally. Known from the beginning for its zero-waste, no-frills, "pass-the-savings-to-the-customer" approach, showing the products in their original delivery cartons, minimizing staff to a minimal. In the year of 2009, LIDL opened its first store in Switzerland, marking the beginning of the retailer's journey in this country. Focusing more on the eastern, German-speaking part of the country, LIDL has now 168 stores and is planning to continue with its expansion in the foreign country. In terms of market share in Switzerland, LIDL holds a meagre percentage. In 2019, Swiss retailers COOP and Migros lead the retail market share, each with a turnover of 27.4€ billion and 25.4€ billion respectively, while LIDL had a meager 982€ million (Grandiz!, 2019). However, is important to notice that LIDL is still recent in the mountainous nation, having only 13 years of existence, whereas leader COOP established its first store back in 1850's and Migros was founded in Zurich in 1925. Taking this into account, LIDL Switzerland has managed quite a feat in such a short period of time, and they are ready to grow even more.
In 2019 alone, LIDL Switzerland opened 16 more stores, created more 470 positions, and was elected for the fifth time in a row "Retailer of the Year" and for the third time in a row was distinguished as the "Best Training Company in Switzerland", and it doesn't stop there. Under the name of "LIDL Connect", LIDL Switzerland extended its digital offer and entered the phone market, launching also an app promoting this new service. In the summer of the same year, LIDL Switzerland custom-built the first two liquid gas service stations in the country, and achieved its environmental goals established back in 2017 for the end of 2019. LIDL is making a big bet on digitalization and on innovating while keeping its identity of a discount retail store to the consumer.

1.2 Digitalization and lack of a KMS in LIDL Switzerland

In the information age that we live in today, knowledge makes the difference. It is the final part of a three-phase process. It starts off as Data, then becomes Information, which in the end becomes knowledge. These are not interchangeable concepts (Davenport & Prusak, 1998). It is best to start with a brief description of each of these notions. Data was defined by Davenport and Prusak as a “Set of discrete, objective facts about events. In an organizational context, data is most usefully described as structured records of transactions.” One can see Data as Raw Information. It is empirical evidence of results of actions. Information can be viewed as “data endowed with relevance and purpose” (Drucker, 1996). Is data that has been processed and now carries a significance, it has meaning.
Knowledge is a whole different layer. It can be defined as a mixture of outlined experience, values, circumstantial information, and insight that delivers a framework for evaluating and incorporating new experiences and information. It can be embedded in documents and repositories, but it can also translate to organizational routines, processes, practices, and norms. One can see it as ingrained information, as experience and instinct combined, that provides to an individual or to an organization a set of tools to better predict what lays ahead. Lidl Switzerland had most of its internal information processes run on several different platforms. Many of these processes began as a file with data that would later be imported to a platform or to excel for processing. Then it would be transcribed as a file of a different format that would later be sent in an email to the next individual in the organizational chain. Besides being time consuming, having data dispersed across several platforms in different formats is redundant and can lead to mistakes, which in turn may lead to financial losses. To counter this situation, Lidl Switzerland decided to invest in a Knowledge Management System (KMS), to centralize most business processes. KMS are technologies that upkeep knowledge managing in organizations, specifically, knowledge creation, codification, and transfer (Ruggles, 1996). Besides promoting centralization, a KMS also functions as better archive, in comparison to having several files in different formats scattered throughout the organization’s chain. By investing in a KMS, LIDL Switzerland is “promoting an integrated approach to identifying, capturing, retrieving, sharing, and evaluating an enterprise’s information assets. These information assets may include databases, documents, policies, procedures, as well as the un-captured expertise and experience of individual’s heads” (Malhotra, 2004). Lidl Switzerland made the call to build the KMS as a portal, using low-code technology.

1.3 LCDP’s history and why invest on it as a retailer

We have seen in the past years several industry trends aiming at reducing the labor required to produce software and make coding easier and more accessible. There were the 4GL and CASE tools in the 1980’s (Martin, 1981), Rapid Application Development in the 1990’s (Martin, 1991), End-User Development in the 2000’s (Liberman, Paternó, Klann & Wulf, 2006) and MDE in the last 20 years (Schmidt, 2006). The term low-code was coined by Forrester in their market analysis back in 2014, where it defined low-code development platforms (LDCP) as “platforms that enable rapid delivery of business applications with a minimum of hand-coding and minimal upfront investment in setup, training and deployment” (Richards & Rymer, 2014). In 2016, Gartner named the technology with a different term, low-code application platform (LCAP) and introducing enterprise LCAP’s, focusing more on an enterprise level, where SLA’s, disaster recovery, security and high performance are required (Wong & Iijima, 2021).
For the rest of this report, it will be referred to as LCDP.
In November of that same year, Microsoft releases its own LCDP, PowerApps and became the first large cloud provider with a solution of its own. In 2017 the definition by Forrester evolved, and LCDP were now “product and/or cloud services for application development that employ visual, declarative techniques instead of programming and are available to customers at low – or no-cost in money and training time to begin, with costs rising in proportion of the business value of the platforms”.
2017 also marked the beginning of a series of acquisitions for LCDP vendors, with Appian’s initial public offering in May of that year. In 2018 its evaluation reached almost $2 billion. Outsystems received an investment of $360 million by KKR and Goldman Sachs in July of that year, and one month later its competitor Mendix was acquired by Siemens by $730 million (Rymer, 2018). 2020 was the year where the other two big cloud providers, Amazon, and Google, followed the trend set by Microsoft and arranged for LCDP’s of their own, with Google acquiring AppSheet in January and Amazon releasing HoneyCode in June of that year. Figure 1 showcases the timeline of past events of low-code history.

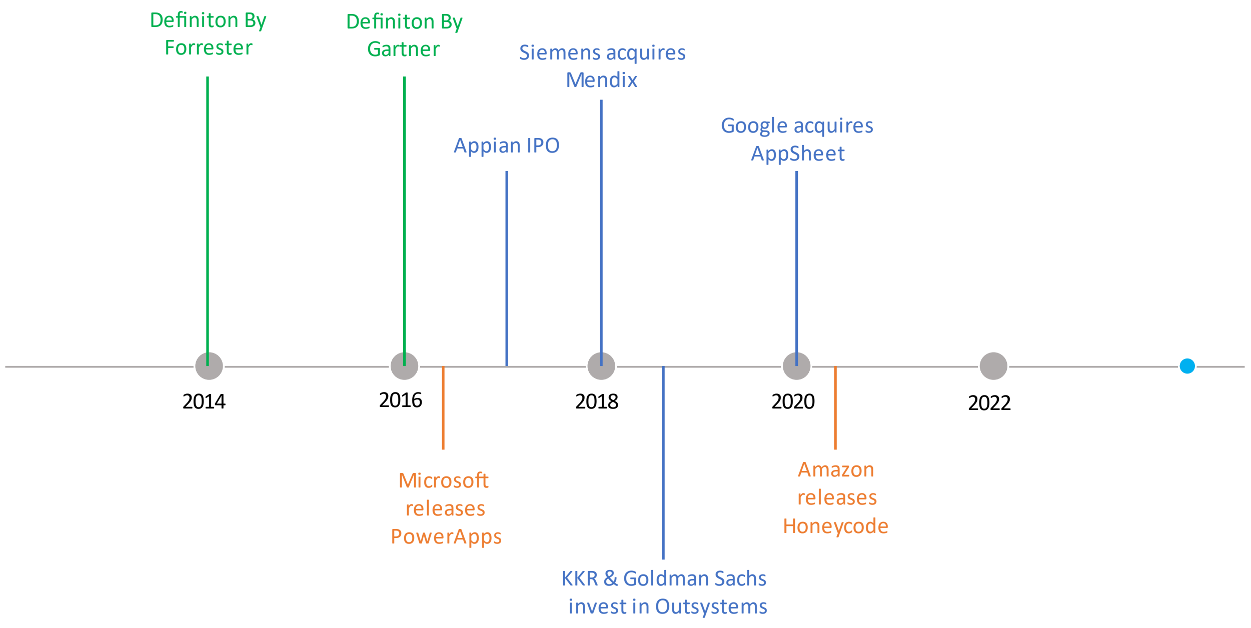

Figure 1- Timeline of LCDP acquisitions & investments. (Tisi, Lara, Kolovos, Di Ruscio, Pierantonio, and Wimmer, 2022)

Adherence to LCDP's by businesses grows day by day. It is expected by the year 2025 that the revenue of LCDP's will reach the value of $29 billion (Wong & Iijima, 2021). As an emerging technology, LCDP's have been betting on all different areas of business to showcase the versatility of the technology, and the ROI that it can provide to its clients. From Insurance, to banking or the public sector, low-code has now reached all layers. On a survey conducted in 2019 to 3300 IT professionals, 41% reported that their organizations were using a LCDP (Outsystems, 2019). Amongst them, retailers are growing and increasing their share. Retailers must leverage digital enablers to sell more and ultimately aim to grow revenue. Customer growth, retention, loyalty, and omnichannel experiences are responsibilities for the people at the senior level. This means modernizing, developing applications, automating processes, and streamlining back-end systems with some of the front-end capabilities. It means keeping up with the digital evolution. LCDP help in accelerating application development to meet the immediate need for new software. Low-code is known for its fast development and for its flexibility. LCDP apps can utilize API-based applications such as image recognition or payment gateways, offered by other companies, further enhancing customer offerings, and simplifying development. In this way, the benefits of low-code are predominantly powerful for retailers motivated to integrate capabilities in areas that would otherwise require highly specialized skills and knowledge, such as IoT solutions for supply chain and logistics visibility, product recommendation engines leveraging both technologies, and other use cases (Sanchis, Gárcia-Perales, Fraile & Poler, 2019).

1.4 Goals of this report

In the case of Lidl Switzerland, there was a need to create an internal portal that worked as headquarters for several processes. The main idea was to give the IT department the ability to fully digitalize a business process, turning it in to an app that would be later made available in the portal to the according users. Due to the emergency of the project, its dimension, and the need to have a working knowledge management system up and running as soon as possible, Lidl Switzerland made the bet to develop it with low-code. A portal, named *Service World,* was developed from scratch, with some business use cases selected at the beginning as prototypes. For each a discovery process was made, development was divided into sprints, and the platform was launched, all while maintaining scalability, security, and profiling. Two years after the start of the project, more than 30 apps are now live on the *Service World* portal.
The main aim of this report is to showcase the success of this bet, and to explain why portal development and low-code go "hand-in-hand" and illustrate this point with the developments made during my internship at LIDL Switzerland. In the literature review, the two main concepts, portals and LCDP, are thoroughly studied. For each their theoretical foundations are laid out, proceeding with their respective frameworks, and listing their respective key features, finishing with restrictions and benefits. For the methodology, it will be showed how these concepts benefit from one and other, providing examples with projects developed by myself during this past year.
On a future development for this report, the *Service World* portal will be evaluated, according *to* "Conceptual Model for measuring Portal's effectiveness" (Urbach, Riempp, Smolnik, 2009) to provide an idea of the impact that the *Service World* has made within the LIDL Switzerland branch.

2 LITERATURE REVIEW

2.1 PORTALS

2.2.1 Portal as a KMS

From the management perspective, knowledge is a process focused on applying expertise, that is, simultaneously knowing and acting (Zack, 1999). Knowledge gives the ability to act according to the situation, providing the tools to increase our odds of a better outcome.
In this perspective, knowledge management is also viewed on different lenses. In the IT view, it can be reduced to the tools and framework used to actively store and process the data.
In the organizational perspective, knowledge management is best understood by considering it as the systemic and organizationally specified process of acquiring, organizing, and communicating knowledge of employees to other employees, to improve productivity and increase efficiency (Alavi & Leidner, 1999). Knowledge management has its own lifecycle as an iterative sequence of activities (Nissen, Kamel & Sengupta, 2000). Several frameworks define this lifecycle as different phases. There does not exist a single commonly accepted definition of what a knowledge management process is. Many perceptions are similar; however, their ordering and structure differs.
For this report, we will follow the lifecycle of Generation, Storage, Distribution, Apply (Benbya, Belb & Passiante, 2004).
The KMS is the whole framework responsible to maintaining this lifecycle of knowledge generation, storage, distribution and sharing, which in turn elevates the efficiency and speed of the processes. KMS fall into four categories: *Content Management Tools*, *Knowledge sharing tools*, *Knowledge search and retrieva*l and finally, *General KMS* (Ruggles, 1996). Most of the KMS categories do what the name implies, with the *General KMS* being an overall solution for an organization's knowledge management requirements. In this category, we can place the Portal.
A portal can have several definitions. It can be viewed as a unified application, information, and knowledge management access within and between enterprises, their partners, and customers. It can also be defined as a single-point web browser interface used within organizations to promote the gathering, sharing and dissemination of information throughout the enterprise (Deltor, 2000). Portals can be categorized by their accessibility, with the possibility of being either internal or external. Internal portals can serve as home base for employees, while supporting knowledge storage and inside communications. External portals provide business-to-business opportunities and focus more on depth of content rather than extent. In any case, it is built according to the company's needs, and it is secured with required login (and in case of internal, it can require an intranet connection). More importantly, a portal is a single point of access to resources, works as an integration platform and provides a home for all the business processes. By synchronizing knowledge and applications, portals create a single view into an organization's intellectual capital (Benbya, Belbaly & Passiante, 2004).

2.2.2 Portal Framework

To be considered a portal, there is a set of core capabilities and infrastructure elements that must be established and present (Aneja, Rowan & Brooksby, 2000).

2.2.2.1 Core Capabilities and their features

- Search
  With information being generated by customers and partners, stored in several repositories, spread across several applications, employees must be able to search with comfort and speed.
  Features: Push/pull technology, profiled information, customized querying.
- Taxonomy
  Also known as "Categorization", this ability is fundamental because it gives the users of the portal the ability to separate the information into different groups, which in turn can be grouped together in hierarchies. With it, management and navigation of information becomes much easier.
  Features: Archiving, Lifecycle Management, file directory.
- Publishing
  Ability to store, render and provide information and documents in several formats. Also promotes content creation.
  Features: Information storage, File upload, file download.
- Collaboration
  With tools such as messaging, document sharing, calendars and threaded conversations, portals can create a shared community across an organization. In an age when virtual meetings are becoming more and more frequent, this gains a bigger significance.
  Features: Messaging, Application sharing, community building.
- Personalization
  According to the user that is using the platform, he must access content relevant to his role only and be notified when a relevant change occurs. Also consists of giving users the possibility to modify their settings and establish preferences.
  Features: profiled information, customized querying, configuration establishment.
- Integration
  Essential to be able to integrate different technologies, to access information, spread across different sources or repositories to provide a unified view of the organization's intellectual capital.
  Features: External Integration, Process automation, API exposure and consumption.

2.2.2.2 Infrastructure Elements

- Extensibility
  The capacity to extend and go beyond its established parameters by being able to integrate web services from different sources. It can even go beyond and extend to different hardware, adding new functionalities.
- Security
  It must be safe and secure, due to the criticality and extent of the information that is stored in it.
- Scalability
  Considering that an organization changes in dimension with time, the portal must be able to

expand to an increasing number of users. Also, if an incompatibility arises, it must be easy to modify and adapt the platform.

- Profiling

A user must only be able to access information and receive notifications according to its role.

We can see on figure 2 a representation of how a corporate portal's infrastructure connects to "outside" features.

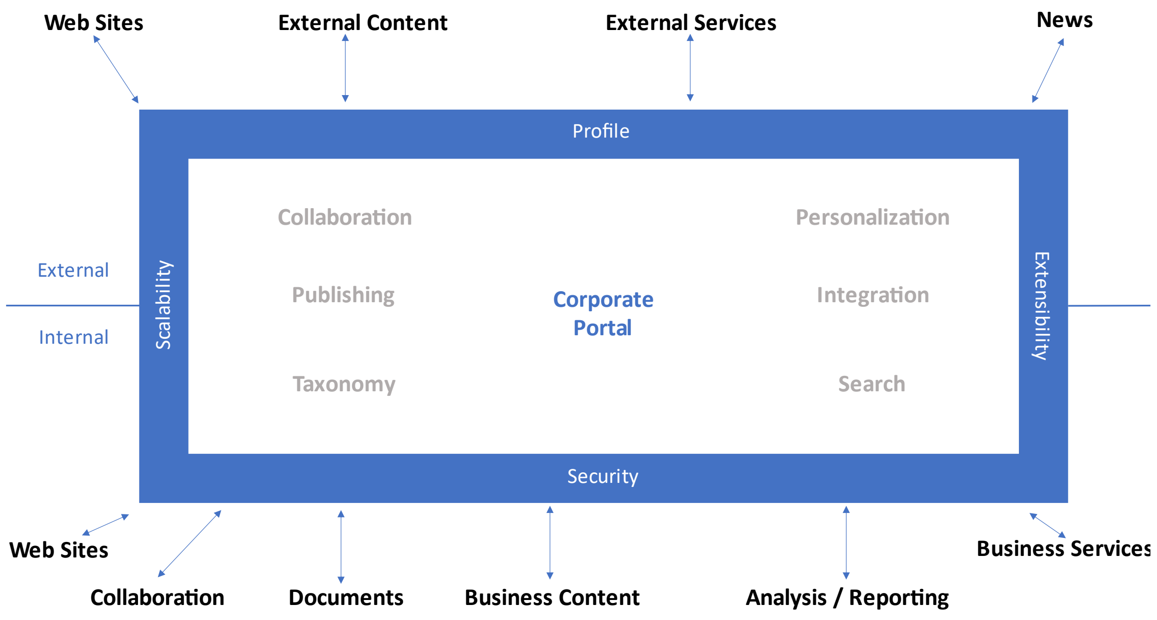

Figure 2- Portal's core capabilities (Benbya, Belbaly & Passiante, 2004).

Considering the four-phased knowledge lifecycle previously mentioned, we can establish a direct correlation between a portal's core capabilities, to each of the stages of the process, as shown on figure 3.

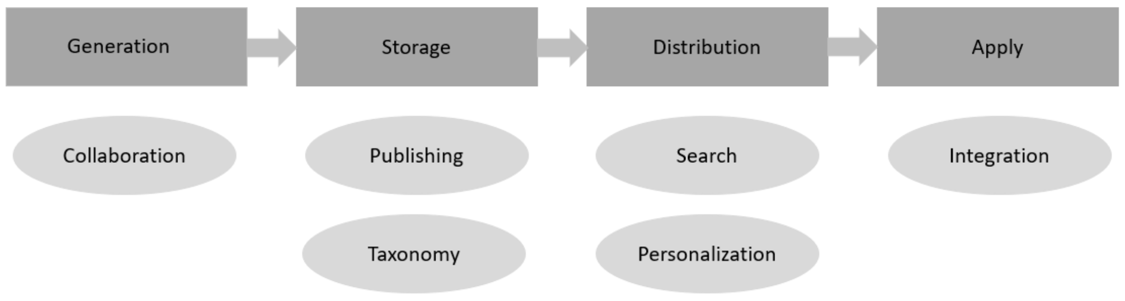

Figure 3- Knowledge generation & connection to Portal's core features

2.2.3 Considerations when developing a Portal

2.2.3.1 Constraints and benefits of building a Portal

There are several factors inhibiting the adoption of an organizational portal. First, it's an expensive investment, and as every other IT project, portals need to display return of investment (ROI). Between hardware costs, design cost, software licensing and development, external integrations, and overall maintenance, it's an investment that might reach steep values. Another cost to take into consideration is communication and training, since one common problem portals face is unawareness of users to the technology's existence and how to use it (Connelly & Kelloway, 2003). Also user unawareness may extend to the point that they do not know that a portal might contain knowledge that might be currently helpful for them.

However, lack of communication goes both ways. The main challenges that company's face when implementing portals stems from ignorance of the users' needs and practices in their processes, which in turn result in ineffective implementation. Developments are made for the work that the technologies think the users do instead of what they actually do, ending up with applications with poor design (Hickins, 1999). Another issue is deprived usability. People only end up using the portal if it provides an easy way to find the information they need. Effective interfaces and quality service delivery end up not as a luxury, but as a requirement.

Despite all of this, when done right portals end up a solid investment and a cornerstone in the company's data management by providing a shared information workspace that facilitates access to content, to group collaboration and to organizational communications.

Portals offer means of gathering all the various technologies that populate the corporate landscape into a single system that enables employees to find information regardless of its physical location. With it, users gain access to a wide range of information sources from internal databases and legacy systems to web file servers and API's that can reside both within and outside the company. Advanced portals might even provide more specialized functions, enabling users to read, write and update data directly through the portal's interface (Deltor, 2000). From a knowledge management system perspective, a portal gives users information channels that lead them to engaging conversations and negotiations with others, which may result in new shared interpretations that turn into new perspectives that can end up in innovation that derives from the portal and that can be stored directly back into it. A portal's biggest advantage is its convenience, which is a factor that can promote acquisition and use of information throughout the entire organization as individuals tend to use information characterized by high accessibility (Allen, 1984). All of this has led to an increased interest among information managers due to the technology's ability to improve the flow and exchange of information through all layers of an enterprise (Newell & Scarbrough, 1999).

2.2.3.2 How to promote a successful portal development

To successfully implement a portal within an organization and fully benefit from its advantages, the key is to place the priority on people when designing and building the portal and its applications, to promote information seeking instead of information retrieval. The distinction here is that seeking is more human oriented and open-ended. Retrieval means that the object was known in the past at some point. Often people in an organization who "knew" organized it for later "knowing". Seeking on the other hand implies the process of acquiring knowledge, it is more problem oriented as the solution may

or may not be found (Marchionini, 1996). By highlighting and focusing on the users, corporate portals can be better understood as information seeking systems rather than systems that merely support the retrieval of information. Users and their respective environments are critical and necessary ingredients to the understanding and improvement of systems in general. Developers and designers must look at the user and their uses of information, to the contexts in which the users make their choices about what information is useful to them at what times. The choices that the users make are based not only on subject matter, but on other elements of the context within which a user lives and works with (Taylor, 1986). Designers and developers must fully delve in and understand the contexts that draw the users to use portals and how the information must be displayed and presented, to make it meaningful for them. They cannot assume that the users know what information they want and that they can search for it directly, the developers must build the portal with the idea in mind that the employees more often use a portal not to find a specific answer, but to help them make sense of their environment, resolve their problems, and learn new ideas. Only by fully understanding how individuals work; how they seek, share, structure and make sense of the information in their work settings, can the information providers maximize the system's effective use (Davenport, 1997).
The portal must present an ideal environment to integrate business processes and synergize it with knowledge generation, actively supporting the worker in using and adding knowledge resources to the system by establishing standards for information collection, processing, and presentation that in the end foster the motivation to actively share knowledge. To do this, managers and the developers must not assume that they know what the users want, and as stated before, they must understand what motivates people to apply their expertise. Only by doing this can they avoid the failure of an unsuccessful portal implementation and ending up with a system that no one uses.

2.2 Low-Code

2.2.1 LCDP as a Development Tool

LCDP's are software platforms that are hosted on cloud environments and enable developers of different domains of knowledge and expertise to build fully-fledged applications ready for production (Rymer & Richardson, 2016). Applications are developed through model-driven engineering (MDE) principles and harness the capabilities of cloud infrastructures, automatic code generation and graphical abstractions to develop entirely functioning applications (Wong & Iijima, 2021). LCDP's take advantage on recent developments in cloud computing technologies and models such as Platform-as-a-service (PaaS), and proven software design patterns and architectures to ensure effective and efficient development, deployment and maintenance of the applications.
Generally, an application is made up of several components, from a database where records are stored, to code compilers, deployment and maintenance systems to the backend and frontend code. What the users see and interact with is only a small portion of what the application truly is. These components, each its own technology, are carefully coordinated, and together enable the running of an application. When it comes to the architecture, there are several ways of laying out the different components. The most well-known is the four-tier architecture with the presentation, data service, business logic land data access layers (Cao, Wei & Qin, 2013). Most of the LCDP take a similar approach, also adopting a four-layer architecture (Sahay, Indamutsa, Di Ruscio & Pierantonio, 2020).
There is a deployment layer where, depending on the LCDP, applications can be deployed on-premises environments or on dedicated cloud infrastructures, followed by the data integration layer, which handles data integration from several data sources. The service integration layer is responsible to connect to different services through API's and handle their authentication mechanisms, but also to assist with the containerization and orchestration of applications together with the Data integration layer. At the top, the application layer resides, consisting of the graphical environment where users directly interact with to specify their applications. This layer provides modelling constructs to specify the behavior and logic of the application (similar to what we see in Figure 4 which is the logic modeler of Outsystems), widgets and toolboxes to build the user interface. Here we can also find authenticated and authorization mechanisms.
By expanding this architecture, we can establish a three-tier setting of the components that make up the LCDP, as demonstrated on Figure 5. The very first tier is made of the application modeler, whereas previously mentioned, the developers build the application with the use of widgets and modeling constructs. Some LCDP's enable the application to run locally before deploying it. Once the model is finished, it can be sent to the platform's backend for further analysis and manipulations, including the generation of the full-fledged application, which can be tested and deployed on the cloud. To this end, the middle-tier takes the application model and performs model management operations such as code generation, optimizations by considering the involved services including database systems, micro-services, API connectors and model repositories of reusable artifacts (Forsyth, 2021). Regarding the database servers, the developers and the application users are not concerned about the type of employed database, or the mechanisms ensuring data integrity and query optimization. All the required micro-services are created, orchestrated, and managed in the back-end without any kind of user intervention, relieving developers from the responsibility of manually managing technical aspects like load balance, business logic consistency, data integrity, security, and authentication. LCDP's also provide developers with repositories that can store reusable modelling artefacts and provide versioning.

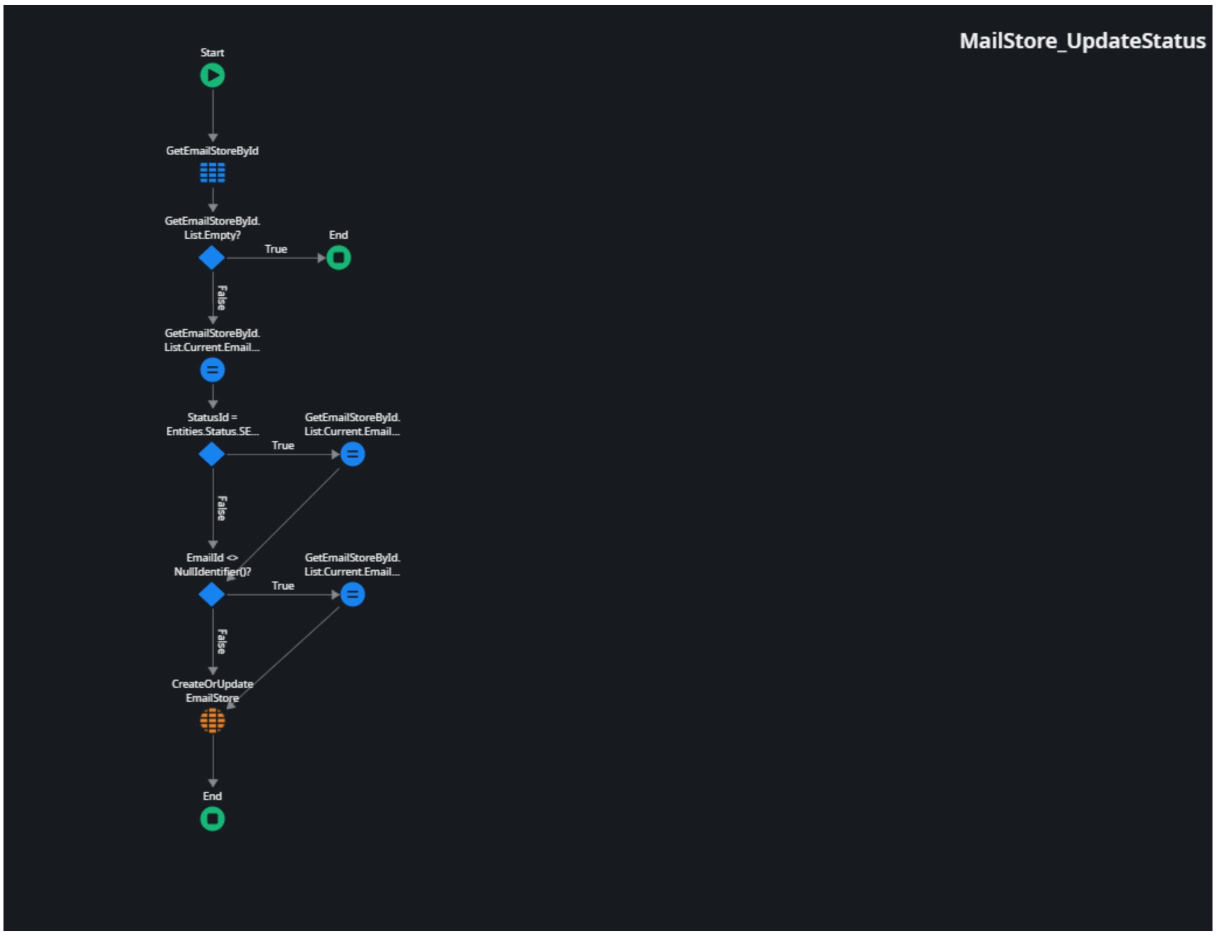

Figure 4 Logic Modelling in Outsystems. Taken directly form tool

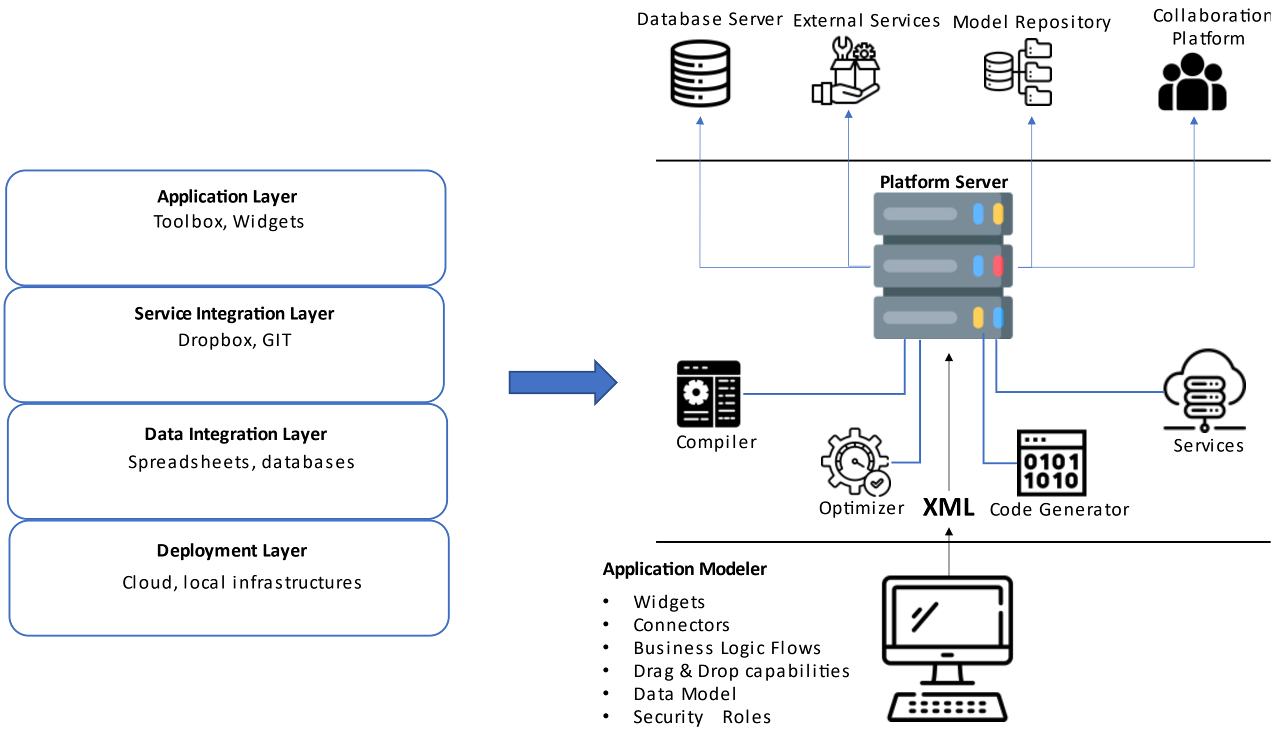

Figure 5- LCDP Layers and hardware structure. (Sahay, Indamutsa, Di Ruscio & Pierantonio, 2020).

2.2.2 LCDP Framework

2.2.2.1 Core features

LCDP are characterized by being able to offer most of the following set of features (Bock & Frank, 2021).

- Conceptual Data Modelling Component

It provides the developers the ability to define data structures in UI-based dialogs or lists, either by drawing out Entity-Relationship models or via a simplified proprietary language. It is displayed on Figure 6 the domain modeler of the Mendix LCDP.

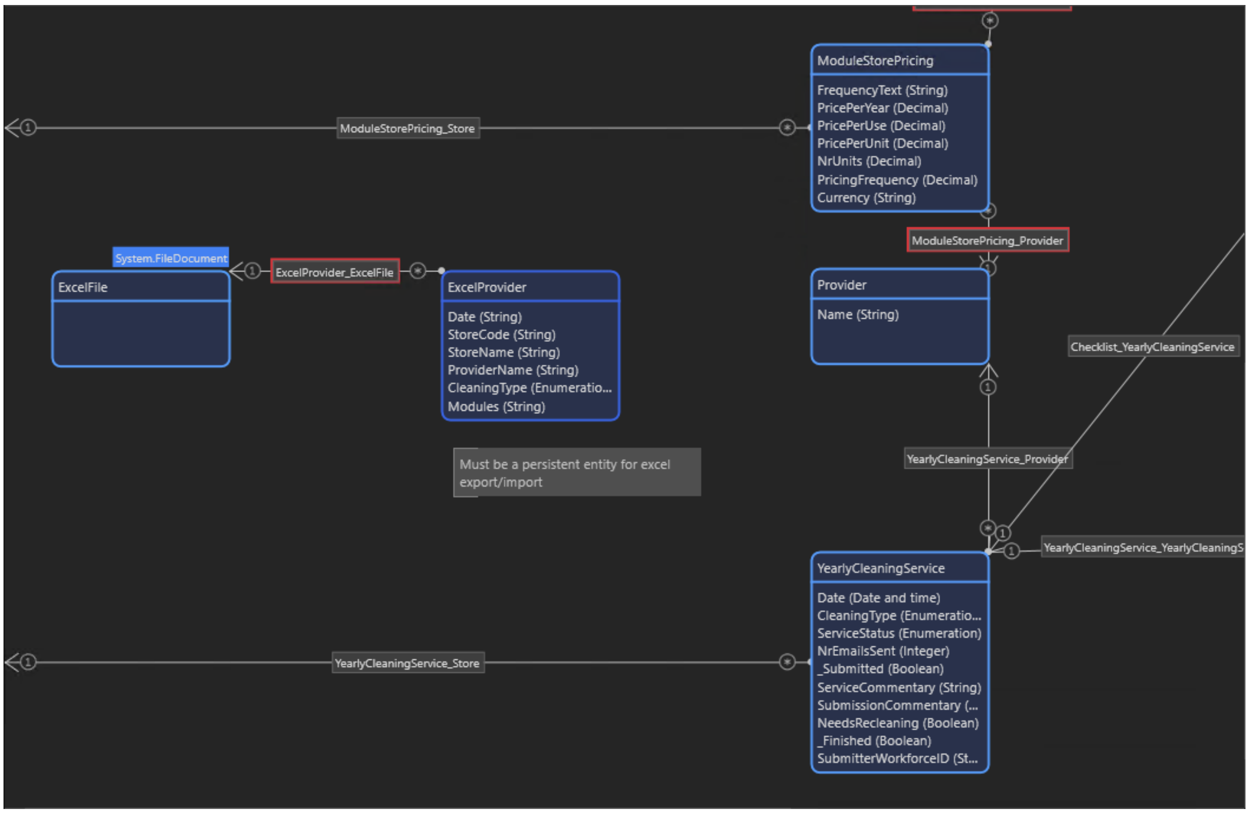

Figure 6 Domain Modeling on Mendix. Taken directly from tool

- Internal database management system

Where the instances of data that are generated through the apps of the platform are stored. No need to setup or create the database. It also offers the possibility to curate the data.

- Access to external data sources

Through API's or other connectors, LCDP's can access external data sources. Some LCDP's also offer the possibility to set an external data source as the primary database where generated instances are stored.

- Conceptual Logic Modelling Component

LCDP's enable the definition of decision and business rules through simple expression languages and dialog-based ways of specifying flow conditions.

- Library of standard operations

Access to a library with generic operations that are commonly used when developing, such as mathematical functions or string operators.

- Access to external services/functions

In varying ways, each LCDP provides the possibility to invoke and integrate external functions via API's or by web services.

- GUI Designer

Ability to develop graphical user interfaces through the usage of pre-defined widgets and building blocks. Most of the drawing of pages are done through the dragging and dropping of these widgets.

- Intelligent GUI mechanisms

LCDP's offer ways to automatically couple the GUI's and the existing data structures, some in one-click, and offer the ability to adapt the applications on different devices.

- Deployment and export mechanisms

Depending on the LCDP, this key feature can take quite different forms. Some systems allow to deployment of the developed solutions as self-contained applications on devices. Other platforms require the installation of the environment on a web server.

- Roles and user rights system

Developers can easily establish application roles and rights. This component is usually contained in the governing architecture of the LCDP, and it will be deployed together with the custom application.

- Availability of traditional coding components

Some LCDP's involve one or more explicit components where procedural specifications can be made using traditional programming languages, most often them being Java or Javascript. All LCDP's grant access to traditional programming code albeit at some more or less hidden level of the architecture.

- Workflow management system

Included at the architectural core, usually only present at the most developed LCDP's. Made possible with a conceptual modeling language such as Business Process Model and Notation (BPMN) or a proprietary representational structure.

- Building block-like application units

Configurable units made available through the LCDP online vendor that offer a limited scope of pre-functionalities for areas such as RPA (Robotic Process Automation), AI (Artificial Intelligence) or BI (Business Intelligence).

2.2.2.2 Development stages

When developing in a LCDP, the process has typically an order (Sahay, Indamutsa, Di Ruscio & Pierantonio, 2020).

1- <u>Data Modelling</u>
Developers establish the data schema of the application by creating entities, establishing the relationships and its dependencies, and defining constraints by using the conceptual data modeling tools, as shown on figure 7 with the domain modeler of the Outsystems LCDP.

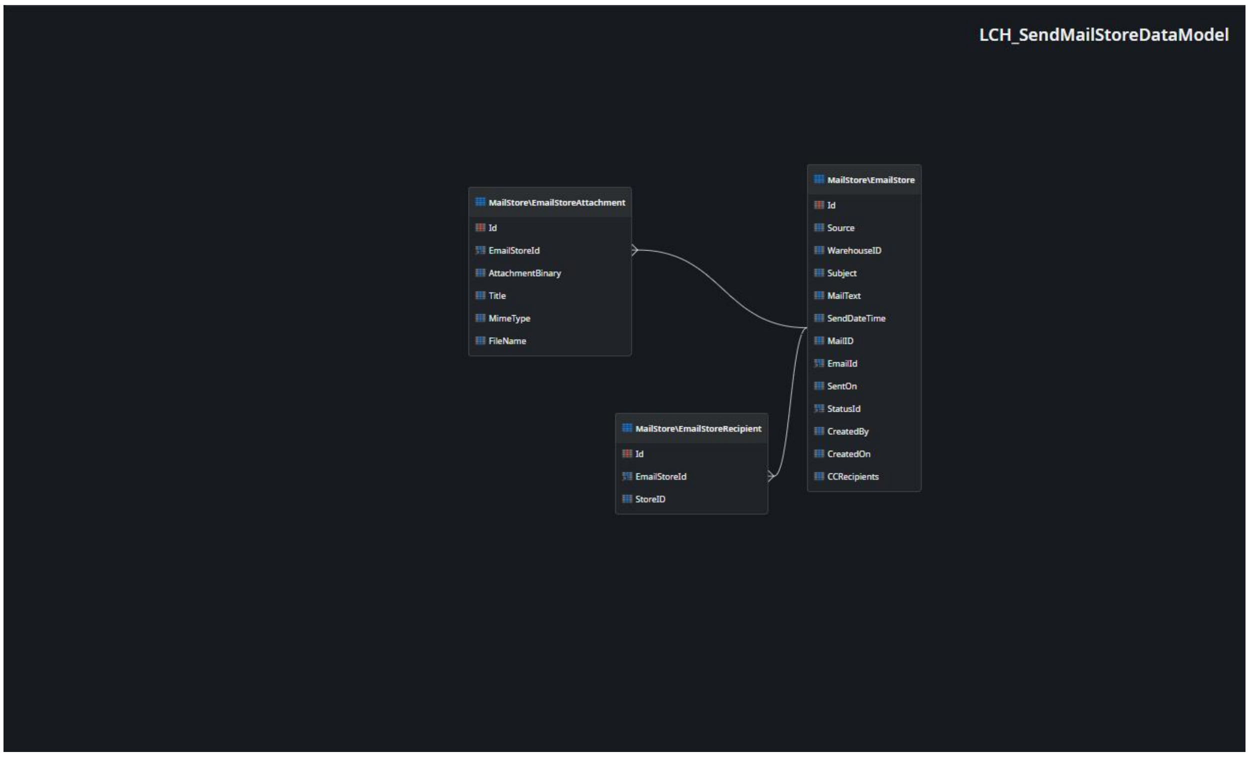

Figure 7 Domain Modelling in Outsystems. Taken directly from the tool

2- <u>User Interface Definition</u>
The second step is the configuration of forms and pages that define the application views, according to the user roles. Here the drag-and-drop capabilities play a major role to speed up development and render the screen quicky, thanks to the GUI designer.

3- <u>Specification of business logic</u>
With the aid of the logic modelling component, developers define the data flows and the business logic required for the application. Logic is established in BPMN-like notation (Business Process Management Notation) in visual-based workflows, as seen is figure 8 with the logic modeler of the Mendix LCDP.

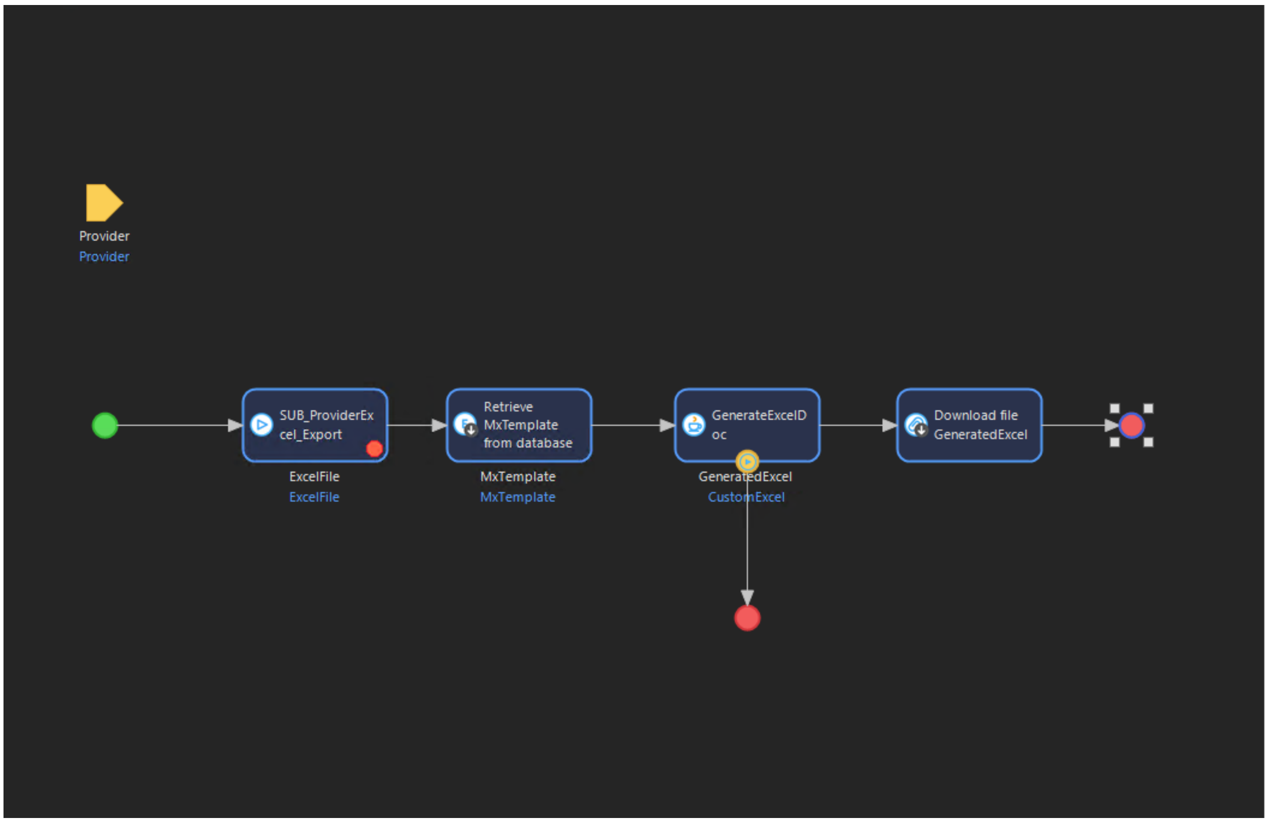

Figure 8 Logic Modelling on Mendix. Taken directly from tool

4- <u>External Services Integration</u>

Establishing the connections and consumptions of external data sources and integration of third-party API's.

5- <u>Deployment & Maintenance</u>

With the assistance of deployment and export mechanisms, developers can easily deploy applications with just a few clicks and quickly preview the applications on a development environment, and just as fast have it on productive.

2.2.3 Considerations when developing with a LCDP

2.2.3.1 Constraints and benefits of low-code development

Low-code is a recent technology, so it still has many trials to face.
The main point that LCDP's still must prove is regarding scalability. LCDP are mainly used for the development of small business applications, there are few cases of large-scale projects and mission critical enterprise (Tise, Mottu, Kolovos, Lara, Guerra, Di Ruscio, Pierantonio & Wimmer 2019). Being preferably based on cloud, LCDP should be able to handle intensive computations and manage big amounts of data, that are created at high volume, variety, and speed (Di Rocco, Di Ruscio, Iovino & Pierantonio, 2015). Still, we cannot be certain of this due to the lack of open standards of LCDP's, which in turn makes it very challenging to assess the scalability of these platforms. Security is another reason that might make companies hesitate on adopting low-code (Radain, Alshammari & Fakieh, 2021). Few are the LCDP vendors that invest in security certificates such as HIPPA, FedRAMP or SOC2. The remaining vendors rely on their partner's infrastructure certificates (Rymer, 2021). Still, LCDP's offer a great variety of features in security support, such as authentication mechanisms, adopted security protocols and user access control infrastructures, to guarantee confidentiality, integrity and authentication at the platform level. Another constraint is the LCDP's lack of interoperability and fragmentation. Most platforms are proprietary and closed sources, which means that there is no exchange of information between them, such as architectural design or of developed services (Sahay, Indamutsa, Di Ruscio & Pierantonio, 2020). It does not help that each LCDP proposes its own low-code development, associated with a particular programming model (Tise, Mottu, Kolovos, Lara, Guerra, Di Ruscio, Pierantonio & Wimmer 2019). This leads to another setback that businesses face when adopting low-code, vendor lock-ins. Adopting a LCDP is a commitment, mainly because abandoning or changing from one vendor to another might translate into an expensive operation, depending on the number of developments made. Lack of extensibility is another issue, once again due to the proprietary behavior of these platforms (Sahay, Indamutsa, Di Ruscio & Pierantonio, 2020). Adding new features to the product itself is possible only on some LCDP's, however extensive coding is necessary, and it must adhere to the architectural and design restraints of the platform being extended. Lastly, despite being aimed for adoption by citizen developers, there still exists a learning curve to go through to confidently use these tools, and the adoption process might require software development knowledge.
In spite all of this, LCDP is on the rise.
The main reasons are speed and cost reduction. The key strengths of low-code development. With the reduction of complexity and with so many services being offered "out-of-the-box", low-code accelerates application development up to 10 times (Forsyth, 2021), which results also in a reduction in cost of building, especially if the developments were previously done by external companies (Richardson & Rymer, 2014). Relieving the outsourcing of third parties also results in more privacy, keeping information and knowledge within company ranks. By also simplifying deployment and logging, LCDP's offer easy maintenance, which helps on keeping a permanent alignment between the service offered and the business requirements. An increase of involvement of business profiles in application development is another big benefit. By creating a context where less technological knowledge is required, LCDP's enable a bigger engagement of business users, and in some cases, they might end up on becoming the developers themselves (Waszkowski, 2019). This in turn leads to the minimization of unstable or inconsistent requirements. Low-code provides the opportunity to quickly build minimum viable products to validate ideas and customer requirements before wasting resources on functionalities that customers may not value (Richardson & Rymer, 2016b).

2.2.3.2 How to promote a successful LCDP adoption

The first step on low-code development is choosing which platform to develop in.
There is a whole plethora of LCDP’s nowadays, so businesses must be careful. They must identify and prioritize the right feature set for their enterprise. To obtain maximum value out of the platform, what features, and tools are needed besides the basics for application building? Companies should pay attention to application management, application lifecycle management, change management, integration and user experience features and compare to their current needs. (Richardson, Rymer, 2016a). Another factor to consider is the vendor’s ability to sustain innovation and value.
Only a handful of today’s LCDP will graduate as the market consolidates and the years go on. Few LCDP are publicly held, so financial performance is speculation (Sahay, Indamutsa, Di Ruscio & Pierantonio, 2020). A good way to measure this is to keep an eye on the vendor’s ability to sustain innovation (Richardson, Rymer, 2016a). Investigating previous work, partnerships, achievable roadmaps, all these factors are things to look out for. After making their choice, companies should make a strong initial investment on training. LCDP’s are an expensive investment which is paid off by quick application building. If developments are not fast and end up taking as much time as regular coding, then the adoption of the platform is just a huge expense on the company. Lastly, management should consider productive governance policies for low-code platforms. All the customers using LCDP’s with sustained success recommend setting up shared services, application frameworks and conventions. Setting up these “guardrails and road rules” will require additional investment, in both time and money, but it will pay off in management changes, allowing them to happen fast without compromising application integrity (Richardson, Rymer, 2016b).

3 METHODOLOGY

3.1 Why choose Low-Code for Portal Development?

3.1.1 From a Technical Perspective

One cannot help but notice that all the portal's core capabilities and respective features are easily provided by a LCDP. The main characteristics of LCDP's allow for a portal's easy creation, generating synergy between the business concept and the technology.
When it comes to search, through their roles and user rights systems and internal database management system, LCDP's can guarantee customized querying and profiled information for each Portal user. Regarding taxonomy, once again we can rely on the internal database management system that LCDP's provide from the "get-go" to allow for archiving and file directory. Also, in junction with Data and Logic modeling components, LCDP's allow for the easy creation of a tailor-made lifecycle management system. When it comes to publishing, the same LCDP mechanisms that allow taxonomy make this feature possible, with the addition of intelligent GUI mechanisms that facilitate and dynamize the viewing and sharing of files. Collaboration is easily made possible thanks to once again the ingrained user right system that make role specific application sharing a reality. Messaging and notification mechanisms can either be created from "scratch" or in some cases, depending on the LCDP, are already built-in and made available from the start or as building application unit in the LCDP online vendor. The GUI designer and respective intelligent mechanisms, paired with the role system and the database system allow for full personalization of the portal, styling it according to the business's preferences. The deployment and export mechanisms of some LCDP's allow for configuration establishment which further enhance the portal's personalization. Last, integration is key for portals and thanks to the access to externals services, functions, and data sources that LCDP's provide this is easily achieved.
Figure 9 illustrates the connections established previously.

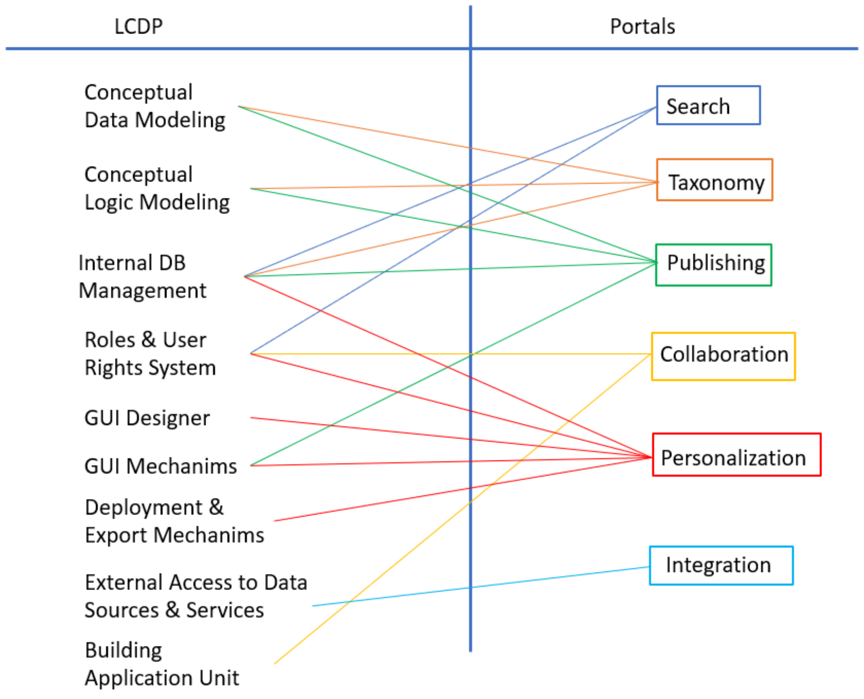
LCDP
Portals
Conceptual Data Modeling
Conceptual Logic Modeling
Internal DB Management
Roles & User Rights System
GUI Designer
GUI Mechanims
Deployment & Export Mechanims
External Access to Data Sources & Services
Building Application Unit
Search
Taxonomy
Publishing
Collaboration
Personalization
Integration

Figure 9 - LCDP's & Portal's Synergies

3.1.2 From a Business Perspective

Management has all the reasons to want portal development to be made with low-code.
Besides all the technical synergies, there are many to be explored from the business perspective. All the infrastructure is guaranteed right from the start. Being hosted on cloud allows for the platform to be extended more easily, making it less dependent on hardware. The ability to integrate external data sources and services also contributes to guarantee extensibility of the portal. The user and rights system provided by LCDP assure the portal's necessary profiling, guaranteeing that each user only has access to information according to its role. The two motives regarding infrastructure that might deviate management in betting on a LCDP for portal development are security and scalability, but the pros outweigh the cons.
In terms of security, as previously stated, depending on the LCPD vendor, some are certified when it comes to security. Besides that, most of the LCDP's vendors offer security support that guarantee confidentiality, integrity, and authentication. Scalability is a whole different issue. As it was mentioned before, despite being hosted on cloud, due to their lack of open standards, companies cannot be sure if LCDP's are able to handle intensive computations and manage big amounts of data. However, LCDP have proven up until now to be up to the challenge, and LIDL is one prime example of that. Being a huge retail multinational spread across several countries with thousands of users, LCDP's have been able to handle and run applications without any history of fail or of a fatal issue. Another pro that LCDP's provide regarding extensibility is the ability to quickly and easily fix incompatibility or bugs that might appear, thanks mainly due to the pairing of the easy programming models present in the logic and data modelling components with the fast deployment and configuration mechanisms. These two features allow for hotfixes to be applied swiftly and without major consequences.
Nevertheless, the main reason to have a portal developed with low-code is the fast-development time that it provides, which in turn allows for developers and designers to have more time to look at the user, their uses of information and their respective contexts. By simplifying and making development taking up to 10 times less, technical and business developers can invest more time in fully understanding the process and user needs. By decreasing development time, LCDP's allow for more time to be invested in the discovery processes, making these bigger and more thorough. If a portal application would take, for example, 30 days developing with traditional coding tools, and if with a LCDP it would reduce the development by a factor of 5, that would translate to only 6 days of development, leaving a 24-day difference. That remaining difference could be used for further understanding the process and fully learn the user needs. This alone proves to be the biggest synergy that it is built by pairing LCDP with portal development.
For proper implementation of Portal applications, there needs to be a bigger investment of time in understanding the business process and the user requests. By cutting development in more than half, this allows for a bigger time investment in the discovery process by the technical and business developers. In some cases, since low-code provides a GUI designer and fast deployment mechanisms, there can be joint sessions with developers and clients where the product is developed and corrected on real time, according to the feedback provided.
In sum, by choosing to have a portal developed in a LCDP, management assures that the infrastructure is set from the start, all with only one provider, and in addition the time advantage that low-code provides promotes and improves the chances of the portal being properly developed, increasing usage and user satisfaction.

3.2 Developments made during the Internship

3.2.1 Snow Cleaning Application

My first project in LIDL Switzerland was to develop an application to digitize the process regarding the confirmation and invoicing of snow cleaning in the stores. When winter season arrives, snow piles up in the parking lots of the LIDL stores. So, the facility management team in LIDL Switzerland hires an external service provider to clean up the parking lots, and salt it when necessary. As it was before, an employee of the facility management team was responsible to schedule the snow cleanings with the external providers and notify the stores of the arranged dates. After the cleaning, the employee would have to reach out to the stores again to check if the cleaning had actually happened and if the information invoiced by the provider is correct (if it was only cleaned or salted as well and the square meters of area cleaned).
This whole process was time consuming and represented a hindrance to productivity in every winter season. To solve this issue, facility management reach out to our team to design and develop an application and have it published on the Service World portal.
The first step was to identify the major issues to solve. The conclusion reached was that the most time-consuming activities circled around communication to the stores. Both notifying and validating information with the store employees consumed too much time, for the facility management team and for the stores. With that in mind, I proceeded to learn on how the provider established the dates. The facility team provided the freedom to the provider to establish the dates on their own. They send the dates on a CSV file to the facility team, via email.
The stores have in their back office a computer, which has access to the *Service World* portal. It was clear that an application had to be built, with main roles established: Store User and Facility user. Each user would have its own distinct view.
On the facility management view, the users would be able at first to upload the CSV file, to process and validate the data (checking if the stores written in the file are valid, if there is no mistake in the dates or in the areas, so on and so forth) and automatically schedule the services. After the file is uploaded and validated, the snow cleaning services are officially scheduled, by being created with the status of "Pending". As soon as they are created, an email is sent to the store, notifying them of the date of the cleaning. When the fateful day arrives, a new email is sent, reminding the store manager of the cleaning. While the facility user can upload the CSV file and see all the scheduled services of all the stores, the store user only has access to check services assigned to his own store. The user can see all services that are pending to happen and is able to either approve or reject services that have already occurred. If approved, the status changes accordingly. To reject, the store user must write down the reason of rejection. The facility user can then proceed to filter all the services according to date, store, or status, and open each service and see if it was approved or in the case of rejection, the motive that led to it. Lastly, it was requested the functionality of generating a pdf report per month, listing all the cleaning services, their respective status and key information, which would proceed to be attached to the invoice of the provider, for information management purposes.
A process that was dependent on reading files and having to actively (and manually) schedule and checking on stores transformed to a fully digital process, with the most time-consuming activities being automated. The project was established as done in one month, with development time being 12 days and it is estimated to save around 160 hours of work each winter season.

3.2.2 File Manager

The file manager application was not built around a specific business process. Our team identified many advantages in several business processes if it were possible to store files on a Sharepoint/Teams folder. Besides working as backup for data, it also opened a new channel for information sharing connecting *Service World* to the most used resource in the headquarters, Microsoft Teams.
Seeing the synergies that could be built if this integration was made available in *Service World*, I was assigned to build an application that provided the means to access and store file in Sharepoint folders. The first challenge was to learn about Sharepoint integration and how could our LCDP access it. To put it simply, to access a Sharepoint folder externally via REST calls, it is required to have ClientID and a respective Client Secret, both which can be obtained in the respective folder settings. These two parameters would have to be stored, for the applications to have access to the folder.
The approach taken was to build a central application, with only one role, Admin. Only our team members would have this role. In this central application, Client ID's and their respective secrets could be stored. After storing these records for a specific Sharepoint folder, three keys had to be set to fetch the record with the ClientID and Secret.
One key was application ID, so the ID value that identifies a specific application. A second key was one value out of a collection of static records that we established (e.g Requests, Analysis, Report), and finally the third key was free to be defined by us. The second part of the developments was to create a file explorer widget that could be used in any application. When developing a new application in which the file explorer widget is going to be used, he is inserted in the application, with the three keys as input parameters. With these keys, the widget fetches the Client ID and Secret stored in the database and proceeds to list the files present in the folder.
The file explorer widget was developed with a set of boolean input parameters, to establish the permissions that the users have. Flags such as *canDelete, canUpload, canDownload, canCreateFolder, canMoveFiles, canExplore,* allow developers, when implementing the widget, to establish if applicational users can delete, upload, or download files through the explorer, open them, so on and so forth.
Another advantage that this widget provides is direct access to Microsoft files. Before this widget, whenever a user had to edit an excel or a word file that was stored on a Service World application, the user had to first download the file, proceed to edit it, and then upload it back to the storage of the application. With this widget, when clicking on a file that is Microsoft file (e.g .xslx, .pptx, .docx) depending on the value that it was established on the *CanOpenOnBrowser* input flag, the file would open either on the designated software or on the online version of it, if the flag was set to true.
For example, a user wants to edit an excel file. He clicks on the file in the widget. If the flag is set to false, as soon as the user clicks on the file, Microsoft Excel opens the file. Keep in mind that excel is opening the file <u>online</u>, which means that the file was <u>not download</u>, and that all the changes that the user makes are done directly on the file that is stored on the Sharepoint folder, saving a couple of minutes to the end user on each edit. If the flag is set to true the exact same thing happens, however instead of opening the computer's Microsoft excel program, it runs its online version on a new window. This was made for store usage since the back office computer of the stores do not have excel installed into it. This project was defined as done in a little over a month, with development time being around 10 days, and it is already being used in several business processes, from finance applications to HR.

3.2.3 Customs Application

The customs department in LIDL Switzerland has a critical role in the whole company. As a retailer it is essential that the flow of goods does not stop, and every minute counts. The customs departments handle the arrival of trucks, the customs inspection, the checking of the cargo, and so much more. Switzerland, not being part of the EU, also adds an extra layer of red tape that the customs handle and solve each day. The customs department approached our team with a request to digitize the main process in their whole department: the cargo arrival and processing.
This process proved to be more complex and with several factors that weighed in. For starters, the process did not only involve the customs department, the goods receiving team at the warehouses also played a key part in the process. Discovery meetings were held, where each part of the process was studied and dissected, to fully understand and find out which would be the best way to build the application. It was not a whole single process, more several little processes that happen in a specific order. Files that had to be uploaded were identified and small "nice-to-haves" were noted down.
From the start, two main roles were established. The role of the goods receiving and the role of the customs. Each role had its own view. Both views shared the same style. A four bucket view and in each view the title of the buckets is different. In both views, the first bucket is new, on the customs the buckets go by the following order: Prepared, Goods Receiving and Transfer. On the goods view, the second bucket is Registering, the third is Counting and the last one is also Transfer.
The main element in the whole process are the declarations.
process starts when customs learn of the arrival of a new truck. The customs user proceeds to create a declaration, where they establish what kind of cargo it holds, in which warehouse it will arrive, date of arrival and assigns a user of the customs to be responsible for this declaration. Once the declaration is created, an entry is registered in the new bucket, of both views. This way both departments are made aware of the date of arrival of the truck.
Customs then proceeds to prepare some documentation. When it is all done, they mark a checkbox in the declaration, signaling that it is prepared. When this happens, on the customs view, the declaration proceeds to the second bucket, Declaration prepared. On the Goods receipt view it remains on the first bucket. One day before the arrival of the truck, they also signal as prepared. By doing this, the declaration moves on both views: on the customs views it goes on to the third bucket, Goods Receiving, and on the Goods view proceeds to the second bucket, Registering. Now the truck arrives, the Goods team receive the cargo and proceed to check it and analyze the documentation with the driver.
If the documentation is ok, they check it on the declaration, which proceeds to go on to the third bucket of the goods view, Counting. Now they must check and count the cargo. When this is finished, they check the count checkbox on the declaration and on both views, it moves to the final bucket, Transfer. To finish the process, Customs checks the done checkbox, which removes the declaration from their respective view. On the Goods side, after the customs departments flags it as done, only when the truck leaves do the Goods department check the done box on their side, removing the declaration from their view and effectively closing the process. There are many more extra functionalities, such as a comment box present in each declaration detail page, where all the users can comment, providing as an alternative channel of communication between both departments, and enhancing it by compartmentalizing it in the respective declaration.
This application also takes advantage of the file explorer widget, providing integration with Sharepoint. For each declaration that is a created, a Sharepoint folder is also created on the Customs Channel. This functionality was integrated, mainly since being a critical process in the company, in case of failure of

the LCDP and for some reason the users are unable to access the application, the process must continue and the files that were stored must be able to be accessed. The goal at the end is that both Customs and the Goods team have always the bucket view turned on, on their respective headquarters main big screen, providing an overview of all the arrivals and processes that are currently happening.
The application features a heavy investment on the UX/UI, by having several icons of different colors that can appear on the bucket's views, signaling to the users if the truck is late, if its being held for inspections or if it has pending tasks. This project took around four months to be defined as done, due mainly to the discovery process consuming so much time. Several joint sessions were held, to fully understand the process thoroughly, and to guarantee that no gap was left, due to its critical nature. Development time itself was around 30 days. The time that it saves remains to be calculated exactly, however we estimate that might be around 200 hours per year.

3.2.4 Forms Core & Template

Our team noticed that most of the requests for digitalization that we received were for processes that focused around two main concepts, the submission of a form request by a user and an approval by either his manager or someone else, depending on the business process. A third of our live applications and many requests in our team backlog followed this procedure type.
Having noticed that, our team decided on two projects to be developed as soon as possible. First, a central application for all forms, where according to user roles, form entry points and access to their respective administrator dashboard appear. This central application also presents a dashboard, with information regarding all of form submissions. Second, a dummy form-based application, to serve as a template for future projects. By doing this, we offer a centralized solution that collects information and works as a portal regarding all form based solutions, and with a template we further accelerate and reduce development time for upcoming projects in the backlog. The first step was analyzing each form application and identify the similarities that they shared. Several key features were identified.
First, the ability to create and submit a form. Second, manager or responsible receives an email with information regarding the submission, who then proceeds to approve or reject. Third, to have a comment box on each submission where information could be shared or discussed. Last, the ability to attach files to each submission.
The form template application has two roles, the submitter, and the approver. The submitter only has access to the form request screen, where he can create a new submission or check his pending requests. The approver has access to the form request screen, where they can check the information and either approve or reject the respective submission, and to a dashboard where they can see all submissions and filter accordingly. This template application serves now as a “starting point” for form based solutions, where in most of the cases our team only has to change or add questions. The main functionality is guaranteed from the get-go.
Everybody has access to the Forms Fore, however if the user has a role of admin, then has access to the dashboard mentioned before and to a back office where new form entry points are created. This project was developed with no specific business process in mind, its main purpose was to accelerate our own development process, and it was built in one week.

3.2.5 Cleaning Application

At the beginning of this year, facility management approached our team once more, this time with a far more demanding request. To fully digitize the cleaning invoice process.
LIDL Switzerland rolled out this year a new device for the stores named *Mobile Office*, an android device whose goal is eventually to fully replace the back office computer. This device opens many possibilities, since it can take pictures, access geolocation, read QR codes, and many more functionalities.
Facility management caught wind of this and proposed to take advantage of it for the cleaning process.
There exist two types of cleanings, the weekly cleanings of the general aspects of the stores (floors, registers), and the monthly cleanings of specific modules (freezers, showcases). The processes are quite distinct. A joint decision was taken to first focus on the yearly cleanings.
The processes centers around modules. Most of the stores have the same modules, however there are some exceptions. Each store has a provider, with whom a price is established per module. Modules can be measured by different units, from square meters to pieces. All these pricings were managed in one single excel. The facility management reached out to the different providers to schedule the cleanings, which were noted down in another excel. Stores were then manually notified of when these yearly cleanings were to happen. The yearly cleanings always happen overnight. The next morning, the store manager had to validate if the cleaning had happened, and if it had been done properly.
The store manager proceeded to print out a checklist, with sample images of how the modules should be after the cleaning, and went around checking, writing done if the module had been properly cleaned or not. After doing this, the store manager scanned the checklist and sent it out to the facility management. In several cases, where the cleaning had been deemed as not proper, the provider demanded a picture, which forced the facility management to reach to the region manager, to drive out to the specific store and take a picture. Facility management had then to send the picture and create a report of their own to attach to the invoice.
As before with the snow cleaning application, the focus point to improve in this process is the communication between the headquarters and the stores.
First, there might be time lost in the stores by having a static checklist, which might induce the store employees in error by asking information regarding modules that are not present in their respective stores. Another time consuming factor is the fact that the checklist must printed and scanned. Last, forcing the region manager to go out of his way to take a picture, might vary from minutes to hours lost, depending on how far the region manager is to the store.
These three issues were the main points to solve.
A similar approach was taken as with the snow cleaning application, with two main roles: facility user role and store user role, each with their own view. The facility user has access to a back office, where they can see the module pricings established. The excel with the pricings is stored on a network folder, which the application access via timer every day to download the prices and create the providers on the database. With that information the facility user can export to an excel a schedule for each provider, listing the stores that they must clean. They proceed to send this excel to the provider, for them to fill out with the dates (as with the snow cleaning, facility gives the freedom to the provider to establish their own dates). When they receive the excel filled out, they can upload it into the application, and by doing this, the cleanings are scheduled and the stores informed of it, via an email sent to the back office computer.
On the morning after the cleaning, it is now time for the store employee to validate if the cleaning was done properly. With the mobile office in hand, they can open the application, which opens on their

view, which is quite simple. If the cleaning has already happened (the system does this check via the date), then a button appears signaling them to start the checklist. By having access to the pricings established with the provider, the list is now dynamic, only showing modules the store has.
They can check the example images on the device, and if not properly cleaned, they can proceed to take a picture on their own, which is automatically attached to the checklist. After submission, facility management has direct access to the reports, and can proceed to forward the images to the provider and schedule a recleaning again on the application. The project remains to be defined as done, since is entering its testing phase.
By doing this, an entire process that was run on excels, papers, emails and phone calls became centralized in one application for all users. It will go live in the beginning of the next year and is estimated to save to up to 250 hours of works every year. In the future, the application will be extended, to encompass the weekly cleanings which will take advantage of the mobile office to produce a digital signature.

4 RESULTS

Following the development and deployment of my projects, I proceeded to gather with the users and study their feedback. Similar to the selected portal evaluation model (Urbach, Riempp, Smolnik, 2009), I asked the users information regarding their satisfaction regarding the quality of service, collaboration, process, information and system, as seen on figure 10, observing to see in the end if there were individual benefits brought with the development of these applications.
The result was significantly positive.

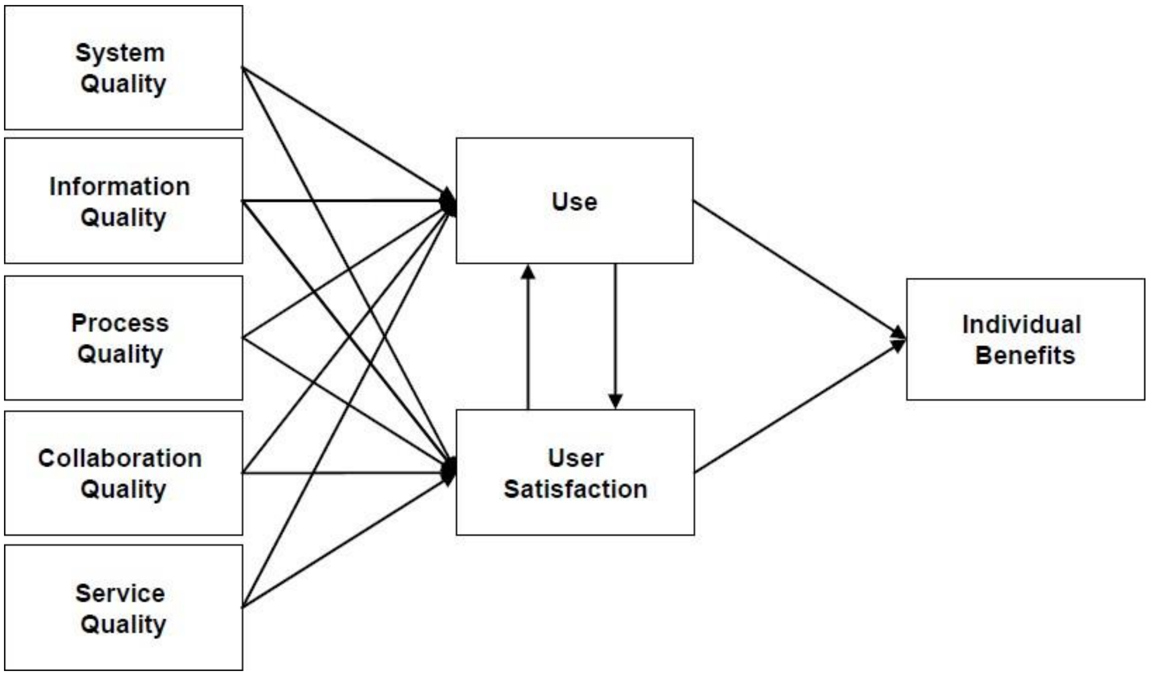

Figure 10 The Conceptual model for measuring the effectiveness of employee portals. (Urbach, Riempp, Smolnik, 2009)

Starting with the snow cleaning application, users reported a high degree of satisfaction, since it has allowed for such a big amount of time saving. According to the key users, information retrieval and process has been constantly reliable, allowing for an increase in the quality of the process, speeding it up and making it easier to manage the services on the stores. The biggest strength that they have reported however is in collaboration, since the application removes all the hassle in communicating with the stores. Store employees have reported the same since they can now rest assured regarding approving these seasonal services. Application roles were automatically and easily established, guaranteeing for a personalized experience for each user, which overall increased the satisfaction and the individual benefits. For the company as an entity, it was a case of success, both in terms of worker happiness as financial, since it led to the saving of work hours, which translates to money saved.
It was a certified win all around.
Although the File Manager was not developed as a specific business case, it has already been integrated into three different applications since its conception. Customs, Goods Return and Price Confirmation. In all three cases, users reported that the synergy brought by the integration allowed for a significant increase in both collaboration and process quality. The integration allows for the users to directly edit the files online, which has been the feature that the key users have noted as the most beneficial when it comes to collaboration improvement. Besides storage function, the ability to have the data stored on teams allows for easy access in the Microsoft infrastructure, which has been the factor that the users have reported that increased the process quality. The ability to store from the application directly to the Sharepoint has allowed also to keep the overall system quality at a high level, since it centralizes

storage functionalities into one chosen place. In all three cases, the file manager has provided with small increments of time saves. With time, this application might end up becoming one of the biggest “money savers” to the company on our portal. The benefits ripped from this integration have been abundant as previously stated, and user satisfaction has been high.
By having digitalized the whole Customs process, the users have told us that the impact is great on all fronts. Collaboration has improved dramatically, since now they have an omnichannel to communicate with the Goods In department. They find that the quality of the information has improved, since having the support of digital has led to less errors occurring during the process. This also binds into the increase in process quality. They also added that by having a customized overview for each department has improved the service and the system in general, since they now “see” clearly the flow of the process, when before it was not clearly established. Users are satisfied so far, application has usage and the benefits have been great, on both the Customs and Goods receiving department, and for LIDL as a whole.
For the Forms Core, users have been grateful that they now have a central hub for form-based applications. Since these processes represent almost 50% of the processes available in the *Service World*, by having a centralized point, a portal in the portal, the users have reported that it has made managing the forms much easier. By having a centralized dashboard for all types of form requests, users have reported a severe increase in both information and in process quality. Users that have to approve pending form requests have given the feedback that the centralized dashboard has increased both the quality of the service, since it allows them to have a general overview and approve requests faster, and the quality of the system, since it makes it more robust. With a high degree of usage, the overall benefits that it has brought has turned this project into a success case.
In the case of the Form template, my own team are the key users since the main purpose of the template is to use it as a starting point for new form based applications. It has worked as a propeller of several projects now, allowing for faster development time. Overall, all of us are satisfied with this template, since it has improved the process quality of developing a new form based project, and having it made available straight from the get-go into the core has allowed for a more robust solution, similar to what was previously mentioned, which in turn led to an increase in the system quality.
These two elements in conjunction have led to an increase in satisfaction throughout all the users and has allowed for the reduction in working hours, in both for the users and for us as developers of applications. The individual benefits that it has brought are clear for everyone.
Finally, the cleaning application. Since it has not been deployed yet, it is harder to present concrete results. However, a small demo rollout has been made, and if the feedback collected until now is an indicator, we are off to a good start. The facility management team stated that the process is improved in all the questioned aspects. Collaboration is greatly improved, due to the integration of the Mobile Office device. By having the pictures taken the quality of the information collected is on a whole new level. Having a synchronization mechanism that allows for dynamic checklist has significantly enhanced the process and the system. With all these enrichments, the key users stated the quality of the service will be on a whole new level when this eventually is rolled out to all the LIDL stores in Switzerland. The store users that have tested the application are happy and the facility team is eager to roll out. It is safe to assume that the benefits that this application will bring will have quite a positive toll on the whole use case.

5 CONCLUSIONS

On only one year, applications that with traditional code would take months, were developed in a matter of days with only one resource as the designated developer, me. Thanks to this significant time reduction in development, I was able to invest more time in fully understanding the business thoroughly and spend more time with the key users, to see how we could further enhance their experience. Applications that did not require much contact with the business managers, such as Forms Core or the Sharepoint integration, were developed in quite a short time frame and mainly for our use, however these applications enhanced other processes and took many other applications to the next level, in both in quality as in development time.

Being deployed on our company portal, *Service World*, these applications are visible only to the relevant users, have their safety guaranteed by the LCDP's innate structure, with deployment and maintenance being easily provided by the platform. Applications that are used by up to 4000 users run smoothly, and departments with critical processes such as the customs, trust the portal enough to have one of its core processes digitized and running on it. It also shows that low-code has earned its place among the hearts of LIDL Switzerland employees, who frequently approach our team with new ideas on how we can improve internal processes and include them in *Service World*.

LIDL as a whole has seen the advantages of low-code. Switzerland is only one country. Germany, Portugal, Spain, Greece, Netherlands, Hungary, Great Britain, Ireland and more, each country has its own company portal being held on the same LCDP, and all of them report the same as Switzerland: the ability to develop applications so quickly has provided the opportunity to the IT departments to produce high quality apps in a short time frame when compared to traditional coding, due to the opportunity that it provides to the developers to really "dig into" the process and understand the struggles that users face, the limitations around the process and how they can improve their work.

In most cases, developers have reported that by having more time to spend with the users, they can also spend more time in showcasing what they are able to build, providing the business managers with more ideas on how they can improve their procedures. Low-code also made development more "personal". I was able in several meetings to prototype procedures with the business managers "right on the spot", which in turn made feedback flow more easily, and made being able to implement user changes at a much faster rate. Even when changes were requested by users after the go-live, I was able to swiftly apply hotfixes or correct the bugs and have them deployed in a matter of hours in most of the cases, all thanks to the functionalities that the LCDP provide.

Low-code has only just arrived, and has already reached big clients such as LIDL, and it has demonstrated, quite successfully, that is perfectly able to handle and provide demanding solutions while guaranteeing scalability and safety. Company Portals are a perfect match with low-code, since it crosses all the boxes of technical requirements, while also countering the biggest advantage that developers face when developing portal applications, low-quality delivery due to not enough time investment in a proper discovery process. LIDL can easily provide the example that portals should be implemented with low-code, since has been able to provide in several countries since 2019 with high quality portals, built at a record pace with a high degree of user satisfaction.

BIBLIOGRAPHICAL REFERENCES

Alavi, Maryam, and Leidner, Dorothy. (1999). *Knowledge Management Systems: Issues, Challenges, and Benefits*. Communications of the AIS, Vol 1 (2ed).

Allen, Thomas. (1984). *Managing the Flow of Technology*. MIT Press (2nd ed).

Aneja, Atul, Rowan, Chia, and Brooksby, Brian. (2000). *Corporate Portal Framework for Transforming Content Chaos on Intranets*.

Benbya, Hind, Belbaly, Nassim, and Passiante, Giuseppina. (2004). Corporate Portal: A Tool for Knowledge Management Synchronization. *International Journal of Information Management.* https://doi.org/10.1016/j.ijinfomgt.2003.12.012

Bock, Alexander and Frank, Ulrich. (2021). Low-Code Platform. *Business & Information Systems Engineering.* https://doi.org/10.1007/s12599-021-00726-8

Cao, Jiexian, Jiayin Wei, and Yongbin Qin. (2013). Research and Application of the Four-Tier Architecture. *Proceedings of the 2013 the International Conference on Education Technology and Information System (ICETIS 2013)* https://doi.org/10.2991/icetis-13.2013.173

Connelly, Catherine, and Kelloway, Kevin. (2003). Predictors of Employees' Perceptions of Knowledge Sharing Cultures. *Leadership & Organization Development Journal* https://doi.org/10.1108/01437730310485

Davenport, Thomas. (1997). *Information Ecology: Mastering the Information and Knowledge Environment*. Oxford University Press.

Davenport, Thomas, and Laurence Prusak. (1998). *Working Knowledge: How Organizations Manage What They Know.* Harvard Business Press.

Deltor, Brian. (2000). The Corporate Portal as Information Infrastructure: Towards a Framework for Portal Design. *International Journal of Information Management* https://doi.org/10.1016/S0268-4012(99)00058-4

Di Rocco, Juri, Di Ruscio, Davide, Iovino, Ludovico and Pierantonio, Alfonso. (2015). Collaborative Repositories in Model-Driven Engineering. *IEEE Software* https://doi.org/10.1109/MS.2015.6

Drucker, Peter. (1996). *The Landmarks of Tomorrow*. Transaction Publishers.

Forsyth, Alexander. (2021, February 18). *What Can You Build With Low-Code?* Outsystems. https://www.outsystems.com/blog/posts/what-can-you-build-with-low-code/.

Grandiz! (2019, March 28). *Swiss Supermarket Giants Coop and Migros Are Doing Great*. https://www.retaildetail.eu/news/food/swiss-supermarket-giants-coop-and-migros-are-doing-great/.

Hickins, Michael. (1999). Xerox Shares Its Knowledge. *Management review, Vol 88, Issue 8*

Lieberman, Henry, Paternò, Fabio, Klann, Markus, and Wulf, Volker. (2006). *End-User Development: An Emerging Paradigm***.** Springer. https://doi.org/10.1007/1-4020-5386-X_1

Malhotra, Yogesh. (2004). *Why Knowledge Management Systems Fail? Enablers and Constraints of Knowledge Management in Human Enterprises.* Springer. https://doi.org/ 10.1007/978-3-540-24746-3_30

Marchionini, Gary. (1996). Information Seeking in Electronic Environment. *Journal of Education for Library and information Science*

Martin, James. (1981). *Application Development without Programmer"*. Prentice Hall-Canada.

Martin, James. (1991). *Rapid Application Development*. *MacMillan Publishing Company.*

Newell, Sue, and Scarbrough, Harry. (1999). Intranets and Knowledge Management: Complex Processes and Ironic Outcomes. *Proceedings of the 32nd Hawaii International Conference on System Sciences -*

1999

Nissen, Mark, Kamel, Magdi, and Sengupta, Kishore. (2000). *Toward Integrating Knowledge Management, Processes and Systems: A Position Paper [Dissertation Navy Postgraduate School Monterey, CA, USA]*. https://www.aaai.org/Papers/Symposia/Spring/2000/SS-00-03/SS00-03-005.pdf

Outsystems. (2021). *State of Application Development*. https://www.outsystems.com/1/state-app-development-banking/.

Radain, Dhefaf, Alshammari, Wahj and Fakieh, Bahjat. (2021). Factors That Affect the Utilization of Low-Code Development Platforms: Survey Study. *Revista Românã de Informatică și Automatică* https://doi.org/10.33436/v31i3y2021

Richardson, Clay, and Rymer, John. (2014 June 9). *New Development Platforms Emerge For Customer-Facing Applications*. Forrester. https://www.forrester.com/report/New-Development-Platforms-Emerge-For-CustomerFacing-Applications/RES113411.

Ruggles, Rudy. (1996). *Knowledge Management Tools*. Routledge.

Rymer, John, and Richardson, Clay. (2016a). *The Forrester Wave^{TM}: Low-Code Development Platforms, Q2 2016*. Forrester.

Rymer, John, and Richardson, Clay. (2016b January 2015). *Vendor Landscape: The Fractured, Fertile Terrain of Low-Code Application Platforms***.** Forrester. https://www.forrester.com/report/vendor-landscape-the-fractured-fertile-terrain-of-lowcode-application-platforms/RES122549

Rymer, John. (2018, August 2). *Siemens Snaps Up Mendix; Low-Code Platforms Enter New Phase*. Forrester. https://www.forrester.com/blogs/siemens-snaps-up-mendix-low-code-platforms-enter-new-phase/.

Rymer, John. (2021). *The Forrester Wave: Low-Code Development Platforms For AD&D Pros.* Forrester.

Sahay, Apurvanand, Indamutsa, Arsene, Di Ruscio, Davide, and Pierantonio, Alfonso. (2020). Supporting the Understanding and Comparison of Low-Code Development Platforms**.** *46th Euromicro Conference on Software Engineering and Advanced Applications.* https://doi.org/10.1109/SEAA51224.2020.00036

Sanchis, Raquel, García-Perales, Óscar, Fraile, Francisco and Poler, Raul. (2019). Low-Code as Enabler of Digital Transformation in Manufacturing Industry. *Applied Sciences by MDPI* http://dx.doi.org/10.3390/app10010012

Schmidt, Douglas. (2006). Guest Editor’s Introduction: Model-Driven Engineering**.** *IEEE Computer Volume 39, Issue 2* https://doi.org/10.1109/MC.2006.58

Taylor, Robert. (1986). *Value-Added Processes in Information Systems*. Ablex Publishing Corporation.

Tisi, Massimo, Mottu, Jean-Marie, Kolovos, Dimitrios, Lara, Juan, Guerra, Esther, Di Ruscio, Davide , Pierantonio, Alfonso, and Wimmer, Manuel. (2019). *Lowcomote: Training the Next Generation of Experts in Scalable Low-Code Engineering Platforms*. [Dissertation IMT Atlantique, Université de Nantes, France, University of York Helsington, UK, Universidad Autónoma de Madrid, Spain, Università degli studi dell’Aquila, Italy, JKU Linz, Austria]

Tisi, Massimo, Lara, Juan, Kolovos, Dimitrios, Di Ruscio, Davide, Pierantonio, Alfonso, and Wimmer, Manuel. (2022). Low-code development and model-driven engineering: Two sides of the same coin? *Software and Systems Modelling* https://doi.org/10.1007/s10270-021-00970-2

Urbach, Nils, Riempp, Gerold, Smolnik, Stefan. (2009) A Conceptual Model for Measuring the Effectiveness of Employee Portals *Proceedings of the Fifteenth Americas Conference on Information Systems, San Francisco, California August 6th*
-9th 2009

Waszkowski, Robert. (2019). *Low-Code Platform for Automating Business Processes in Manufacturing.* [Dissertation Cybernetics Faculty, Military University of Technology,
2 Kaliskiego str., Warszawa, Poland]

Wong, Jason, and Iijima, Kimihiko. (2021 20 September). *Magic Quadrant for Enterprise Low-Code Application Platforms***.** Gartner. https://www.gartner.com/en/documents/4005939

Zack, Michael. (1999). Developing a Knowledge Strategy. *California Management Review, Vol 41 No 3* https://doi.org/10.2307/41166000