Case Story: How We Built a Business Information System, Lean and Agile for an SMB

Three people. Fast. Ground-up. No bugs. Operative for ten years.

24 min read · Just now

Gunnar Östberg

Introduction

This story details how we developed and implemented a customized business information system for a client company, from inception to completion, supporting all their business processes.

Within a few years, this system would help the company grow sevenfold.

I'll share the key principles and procedures behind our approach, as they are broadly applicable and may inspire others.

This example highlights the relationship between needs, requirements, and solutions. Enjoy!

Part 1 — The Business Need

Why did the company have to change

A small training company with 40 employees was growing fast — but ran entirely on binders and paper. Within a few years, we built the system that powered its nationwide expansion.

This first part of the story explains the case's business aspects and the key principles we applied.

Business Information System. Image by Microsoft Designer.

The Client

My client, a small educational company, delivered teacher-led classroom training to both businesses and through public procurement. When the project began, the company employed approximately 40 people and was targeting rapid, profitable growth.

The Task

Faced with challenging circumstances and specific requirements, we built a comprehensive custom business information system from the ground up, managing every stage.

Our process included:

1. Identifying and analyzing the need for a business information system.

2. Undertaking requirements modeling, design, development, and deployment.

3. Supporting effective use for 10 years with DevOps-like maintenance.

4. Planning and executing the system's final decommissioning at the end of its lifecycle.

The Team

Our team followed a mixed approach: I (Gunnar) was mainly the business analyst, Marcel managed development as my technical subcontractor, and Daniel offered subject matter expertise. Daniel, an employee of the client company, handled teacher-led classroom training. My company provided the information system, except for the on-premises servers, which were locally outsourced. Notably, Marcel worked remotely throughout, and we never met in person.

The System

The business information system covered all significant sectors of my client's business and provided comprehensive support for value-adding processes. In

essence, it served as a preconfigured ERP, except for accounting and finance administrative support.

> *An Enterprise Resource Planning system serves as the company's digital backbone, aiming to gather all essential processes and information in one place. This enables better control, analysis, and decision-making, while eliminating inefficient information silos within the business. Its functions, therefore, cover everything from process automation and data collection to reporting and decision support.*

More details follow below.

The Result

The business information system operated successfully for ten years, serving 100 administrators and managing tens of thousands of master records, primarily students and courses. We developed most of the two-application system in just a few months of part-time work. There were no production bugs—only a handful of user errors, which led to minor updates in the user interface or reference manual.

The Impact

The business information system enabled the client's profitable growth and expansion. Over just a few years, it helped the company expand from 40 employees at three locations to 275 employees in 17 locations.

Why this story

Our success came from making business analysis a top priority and using lean, agile methodologies. Although many IT projects fail, technical challenges are seldom the reason — poor business analysis is usually to blame.

Small and mid-sized businesses often face challenges when deploying information technology. Their limited resources and staffing constraints mean employees are busy with day-to-day tasks, leaving little time for development or significant improvements. As a result, IT projects must be highly efficient — every hour matters. On top of this, these businesses frequently lack expertise in change management and IT, increasing the risk of mistakes.

When I use the term 'SMB,' I mean Small and Mid-sized Businesses. In Europe, 'SME' (Small and Medium-sized Enterprises) is more common. In the media, startups and small businesses in the software industry often receive the most attention. However, most SMBs are smaller companies that need IT support for their daily operations. This story focuses on those businesses.

Similar stories rarely capture the complete lifecycle of a product and its impact, as this one does. I've omitted specific technology stack details to ensure the story remains current, since these frequently change. Still, the guiding principles presented here continue to be relevant.

Strong business analysis and agility are essential for success. *I am writing this story to inspire others who aim to achieve similar goals by sharing and explaining the principles we followed.* Please feel free to ask questions and share your thoughts about the story.

The Principles we followed

- Understanding the client's needs is paramount. A thorough needs analysis guides requirements, specifications, and design work.
- The information system should be designed to support business processes, which must be thoroughly examined and documented.
- Most of the work involves needs analysis and requirements modeling, rather than development. Careful preparation — including making wise decisions, doing the right things, and acting carefully — saves considerable effort later. Programming progresses quickly when objectives are clear and the architecture is established.
- Work follows agile principles with no conflict with the above.
- The project involves only essential personnel, each with expertise in their area, to maximize quality and minimize costs.
- Everything is lean, no waste anywhere. Every work minute is questioned.
- The application's interface is intuitive, providing an excellent user experience — even for coworkers without computer experience. There are no keyboard shortcuts, menus, or user manuals; the usage naturally follows the workflow processes.
- Every entry is made just once, and only when necessary.
- There is no manual data or file handling, such as Excel sheets; external input and reporting are managed through APIs wherever possible.

For the applications, we used a straightforward structure:

- One screen corresponds to one feature, implemented in one script file (which contains program code, database connections, and routing to other screens).
- Each feature covers one or a few tasks.
- Each process step consists of a set of tasks.

The following Case Story explains how we applied these principles.

The Case

This chapter provides the initial context and situational background. I then describe the specific aspects of each main phase of the project.

Initial state

The primary objective of the new CEO's directive was to grow the business profitably, expanding from 40 employees in two cities to several hundred nationwide within a few years. That's where I came in, as a management consultant, to assist him. My part-time focus was on strengthening core business processes to support the expansion, while the CEO concentrated on external stakeholders, sales, leadership, and daily operations. We collaborated closely: I translated his high-level vision into structured business plans, ensuring all aspects aligned and functioned effectively.

The first discovery

I knew that small businesses typically operate with limited resources, which are devoted entirely to their daily operations. They usually have to troubleshoot technology issues without a dedicated IT department. That was the case here — only it was even worse than I'd expected.

The company had almost no IT resources. The finance department consisted of an elderly, part-time employee who used an outdated PC primarily for accounting, invoicing, and payroll. This was the only use of computers in the organization, except for emails and printing forms. Very few employees were computer-literate, and the owner (the former CEO) disliked computers entirely.

I was genuinely amazed that a successful company could function without computers in today's digital era. Fortunately, the new CEO had a completely different mindset. On his first day, he ordered new computers for the office and began researching options like Microsoft SharePoint, website hosting, and other digital tools.

The second discovery

My first meeting with a staff member was with the woman in charge of a preparatory course designed to assess adult learners' suitability for a professional occupation. This particular business process is tied to government labor market measures. It involves theoretical tests in mathematics, language, reading comprehension, three professional subjects, and evaluations of practical skills. Interestingly, the woman was the owner's wife. Her spacious office was lined with binders from floor to ceiling along all four walls.

The first thing I asked her was, "What are all these binders?" She explained that the documents contained assessment information for thousands of people who had taken tests there. They conducted a large number of assessments each week.

She described the complicated assessment process and the complex conditions that must be met. I tried to examine how the business was actually run. They used photocopiers and overhead projectors. Much of the process was undocumented; we had only a few paper forms. I realized that we need to capture the tacit knowledge embedded here.

I explained, "Expanding nationwide to around 20 offices within 2–3 years is unrealistic without an information system to support you. Competence transfer and scalability would be unmanageable, and ensuring consistent quality across the country through manual processes is not only unfeasible but also inefficient. We need an information system equipped with a database and business process logic. This should include an assessment application to evaluate candidates' suitability for the company's training program and to measure their knowledge before students are examined."

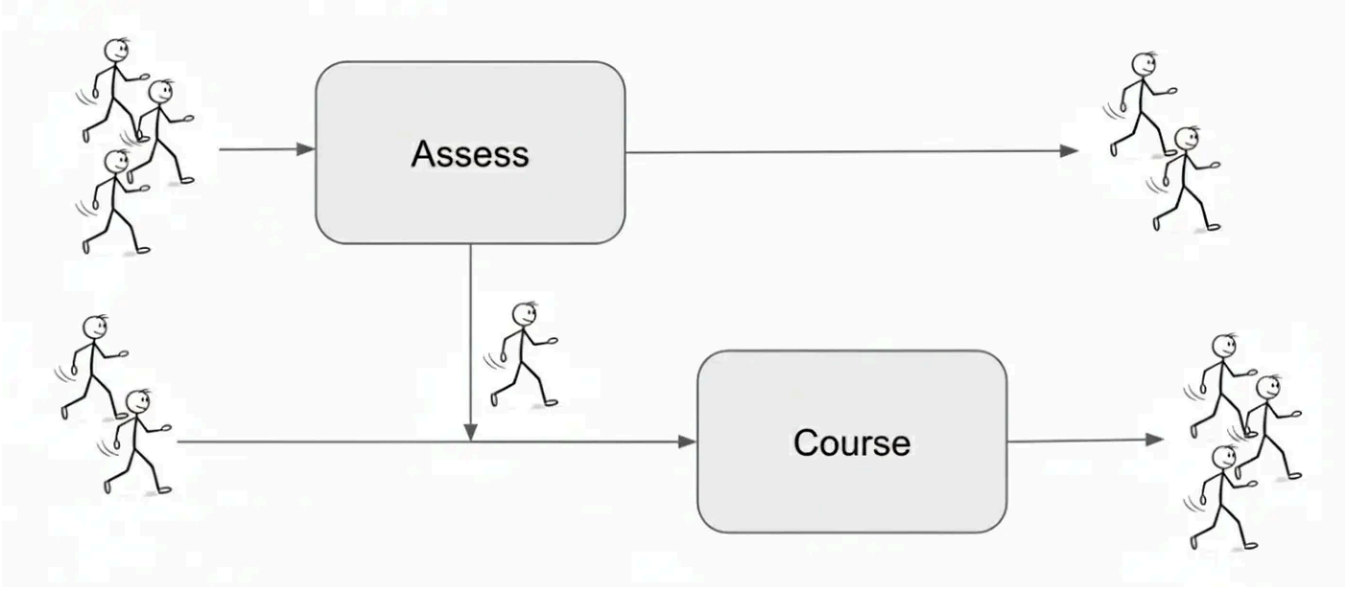
She asked me to build it. I replied, "Let's do it". How hard can it be?

The third discovery

My second meeting with the staff was with the manager responsible for the company's future and leading product portfolio: teacher-led classroom training

programs focused on professional competence and vocational skills for adult learners. This meeting revealed that this part of the business was not yet properly established. We identified a desperate need for an information system — not only to ensure productivity, quality, and operational efficiency, but also to guarantee compliance. We learned that government authorities imposed stringent requirements on the company's operations; for example, reporting needed to follow specific procedures or utilize certain APIs. Both previously assessed and approved students and other participants would enroll in these training programs. We clearly need an application for course management.

He asked me to build it. I replied, "Let's do it." It can't be that hard, can it?

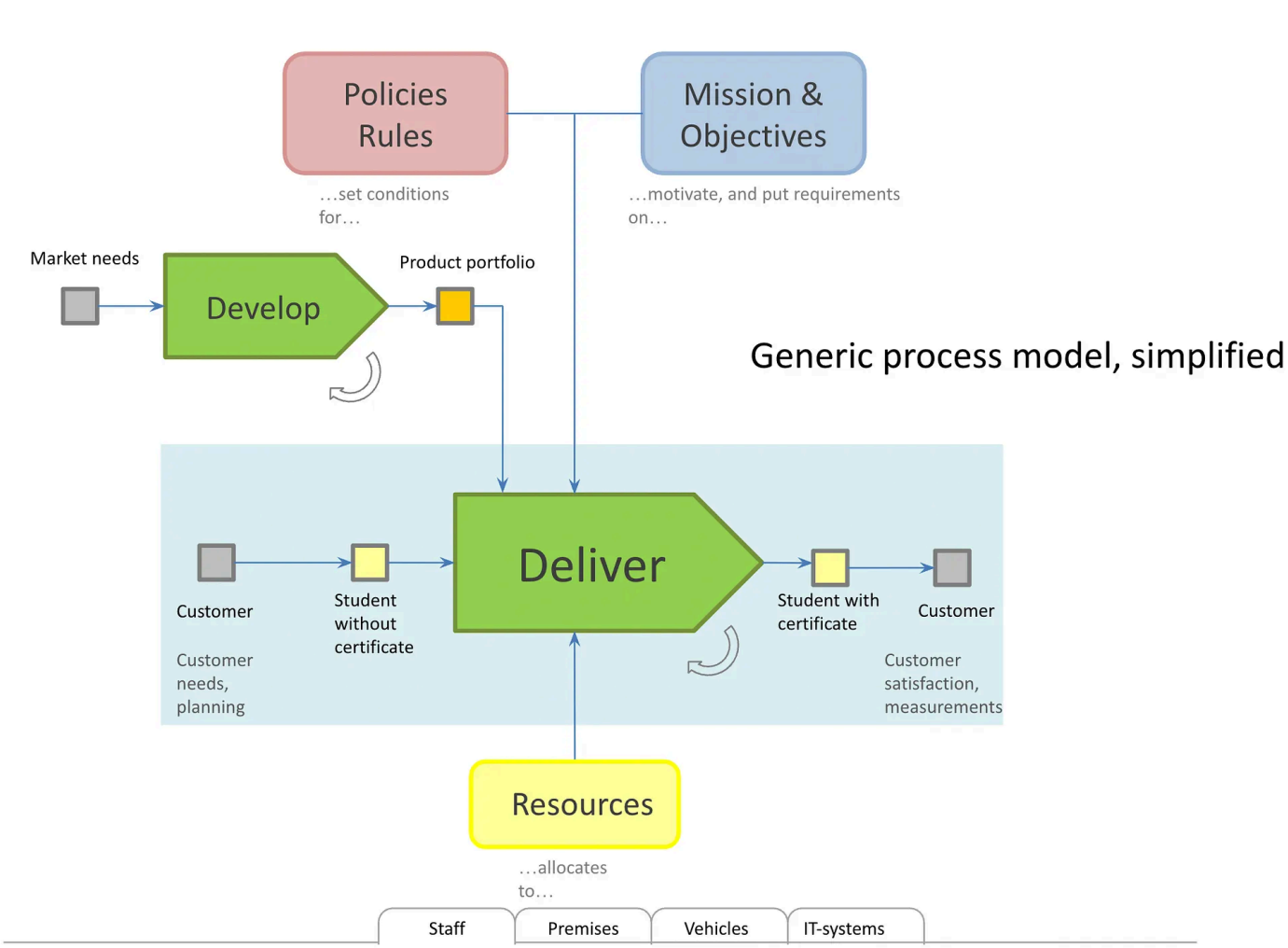


End of Part 1 — Next: How we structured the project and modeled what to build.

Part 2 — From Project Kick-off to Needs Analysis

Defining What to Build

In Part 1, we defined *why* change was needed. Now, let's look at *how* we structured it — the project, the analysis, and the models that turned chaos into clarity.



The Project

I now understand the challenge.

This project aims

- To fully digitalize the unique business processes of two significant parts of the company without any IT resources, skills, or equipment. Zero.
- To implement an administrative information system comprising two distinct but integrated, complex applications.
- For a very short timeframe of just a few months.
- For a minimal budget and with no existing resources.

Honestly, I enjoy challenges. How difficult could it be?

This project is interesting; it actually felt great! Usually, with IT projects, you have a legacy nightmare to start from, along with other constraints that aren't much better. Not every day you get a chance like this to make a difference — and from a blank sheet of paper.

We truly need this system; without it, achieving company growth and meeting client goals will be impossible. Strategy comes first, followed by structure and systems. To expand, we must embed both detailed process knowledge and effective habits into our information system.

Business agreement and legal

I served as both advisor and provider, with my company supplying the business information system and holding the IP rights. Our excellent client relationship allowed us to agree on the leanest implementation process and total transparency. The client chose to pay monthly on a current account, spreading costs and focusing on value and impact. My rates were standard — I wasn't greedy and genuinely loved this work, treating it almost like a hobby. Over the years, I advised this client and others, including significant projects at Scania. I was full of energy and courage — and I love challenges.

Needs analysis

Needs analysis addresses business strategy and processes — not technology — by identifying organizational needs, assessing gaps, and finding root causes, without prescribing specific solutions.

To perform the needs analysis, we modeled the existing processes and stakeholder relations using my usual framework, which covers goals, rules, concepts, and resources. The process involved some workshops and several interviews.

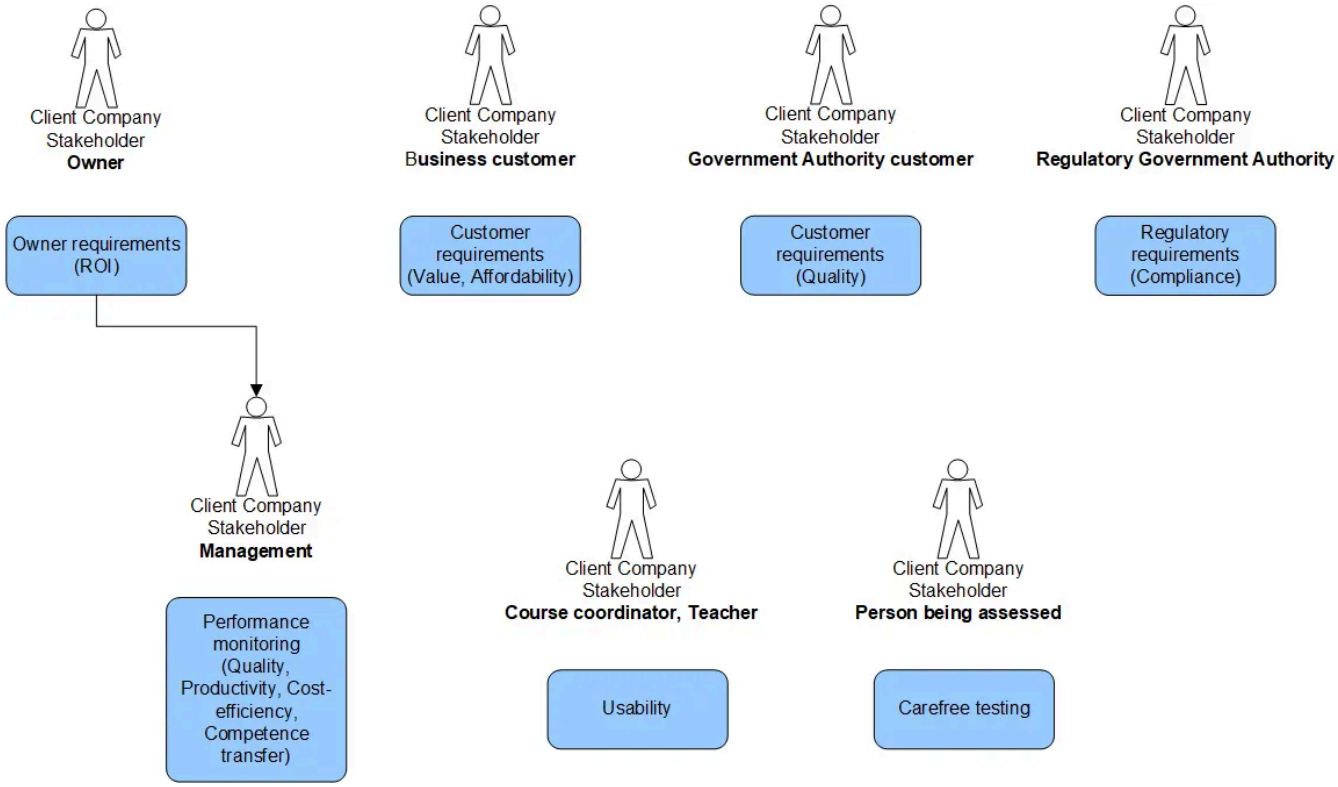
The main results of the needs analysis are an integrated collection of artifacts:

- Stakeholder diagram
- As-is business process model with resource layer
- To-be digitalized business process model with resource layer
- Concept model
- Goal model
- Rules model (ensuring compliance with relevant laws and regulations)

After conducting a strategy check and SWOT analysis, I concluded with a final gap analysis, which clarified the overall scope and priorities.

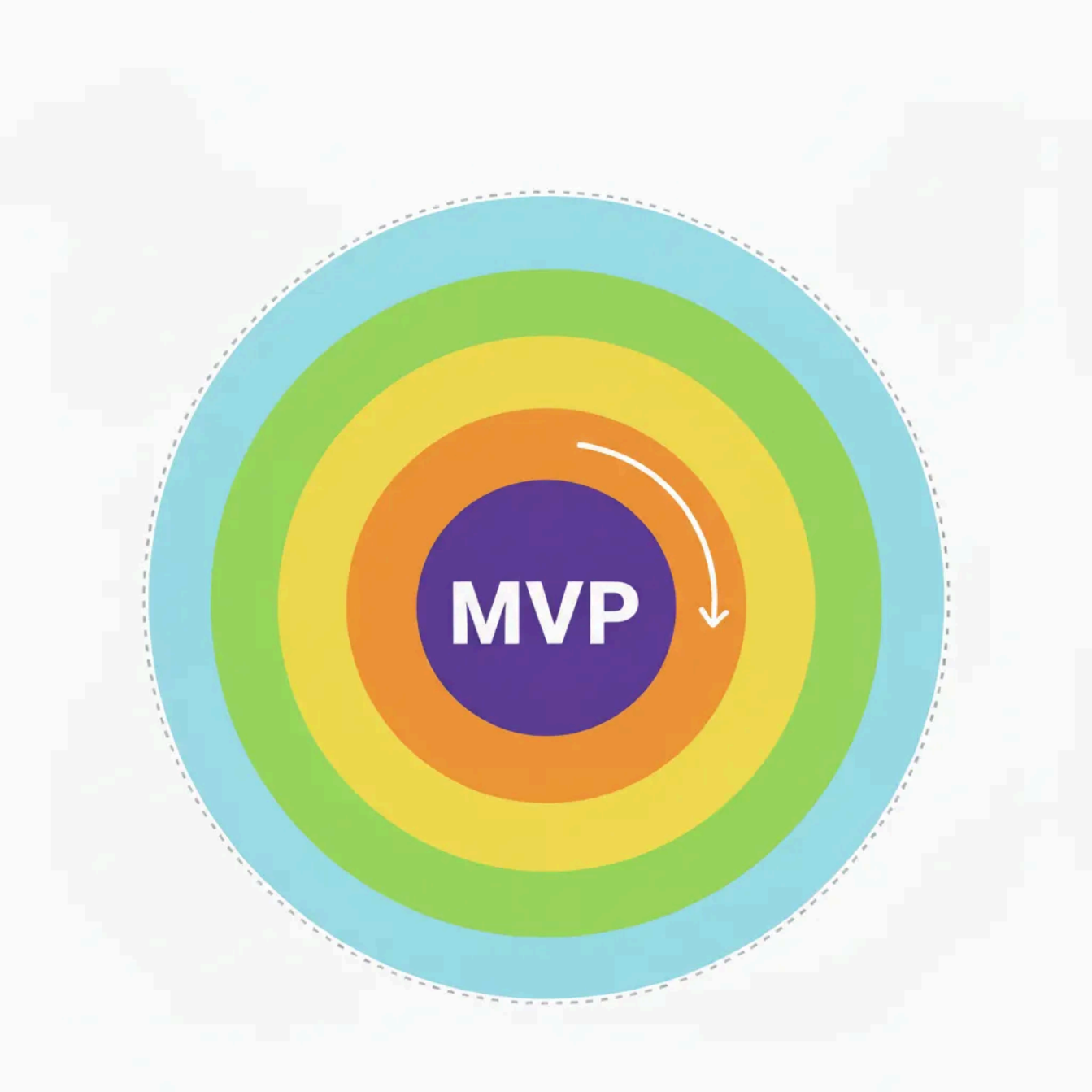
Some of the key points we identified are:

- The company's strategic objectives for its business information system include preparing for growth through standardization (to achieve quality, competence, and skills transfer), ensuring regulatory compliance (meeting control requirements), and increasing internal efficiency.



- An IT application should fulfill the demands and objectives of five main stakeholders: regulatory government authorities (control), other government authorities (customer value), business customers (customer value), management (performance monitoring), and course coordinators, teachers, and students being assessed (usability).

- We need to get things moving quickly and implement basic functionality immediately. We should work lean, applying agile principles by releasing new features in sprints and building our product layer by layer, like an onion. Therefore, we need a production-ready MVP (Minimum Viable Product) from the start — there's no time or budget for prototypes or proof of concepts. The MVP must meet all essential, must-have requirements.



> *A minimum viable product (MVP) is the simplest version of a product that includes just enough features to be usable by early users, who can then provide feedback for future development.*

- We need two distinct web applications since their processes, application security, users, and registered entities differ. Each has a unique purpose.
- Since some data and users will overlap, the applications must be integrated.
- We need a standardized system with a shared database. Both applications must have consistent authority levels, privacy protections, basic security measures, integrations, and user experiences.
- We need to handle sensitive personal information and fully comply with GDPR.

- Since everything is internal and users don't have internet access to the system, it doesn't need to be elaborate — focus on making it helpful and practical.

- The 'Assess' process will remain unchanged due to strict government regulations preventing us from fully 'digitally transforming' it. However, the process itself is solid and will continue as is in its digital form. We have still managed to introduce some innovations, such as fully automated test correction — even for assessments with fill-in-the-blank sections in continuous text. Once the 'Assess' application is in place, physical papers and binders will be eliminated, and everything will become digital.

- The new 'Course delivery' process will be entirely digital, which I will design and integrate into the 'Course' application. This process will cover course design, planning, delivery, and the management of students, their attendance, results, diplomas, and statistics. Timeliness is crucial — there must be options to act and prioritize immediately. For example, it should be possible to register course participants for a session even if the course content hasn't been finalized. The application should support delivering suitable course material for each class session, with integrated material distribution and attendance tracking.

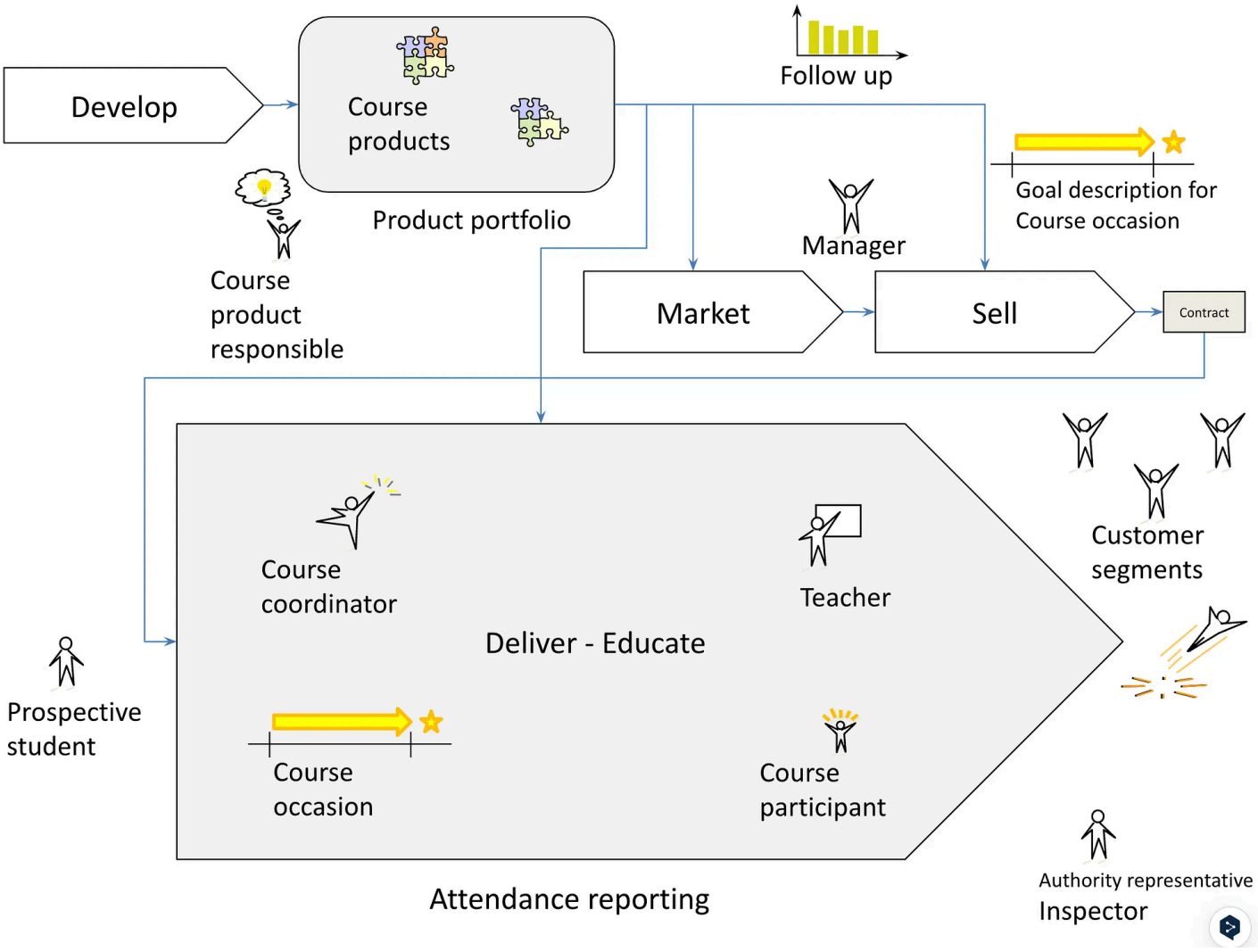
- For the 'Assess' application (called initially the Test or T system), there are specific internal requests for managing the tests. Each test should be usable only once per assessed individual, and neither the questions nor the answers may be saved or stored.

- For the 'Course' application (called initially the Course or Y system), authorities impose strict requirements regarding attendance tracking, teacher signatures, certificate reporting, and performance evaluation of course delivery. These requirements are specific and differ from those in other educational settings. The courses themselves are highly complex and demanding — unlike standard high school or university classes. The curriculum is split into a syllabus, which incorporates government mandates, and a teaching plan. The teaching plan represents the client company's strategy and detailed lesson outlines for implementing the curriculum. Both elements place constraints on the process and the application's design.

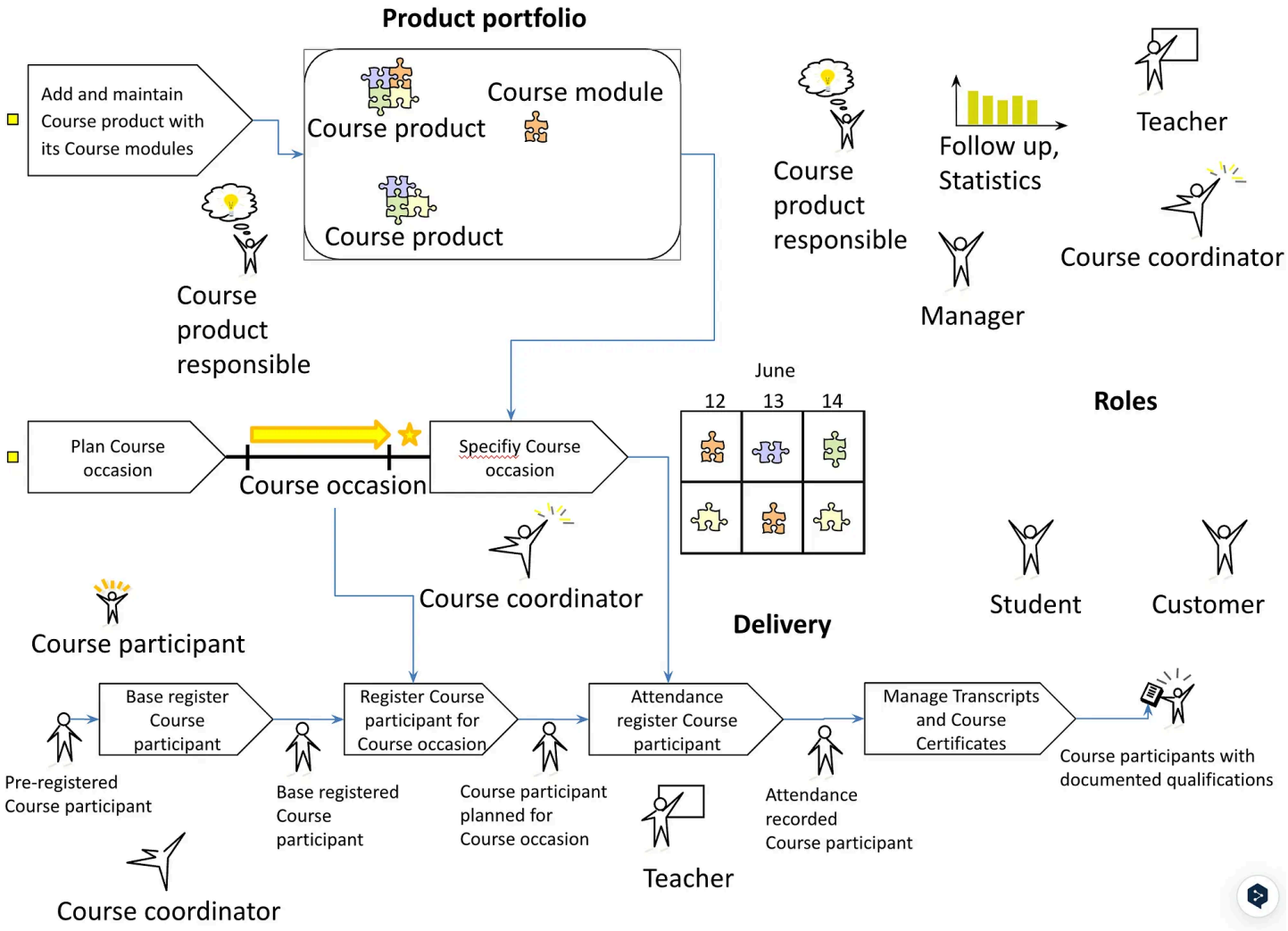
- We need to get the concepts right. For example, when you are assessed, you can be, but need not be, a student. A *student* is participating in, or has completed, a *course occasion*. *Course participants*, not necessarily students, are registered on

course occasions. *Course products* are comprised of *course modules*. Course occasions are delivered in *course moments* by *class sessions*. Course occasions, along with their course moments, involve the delivery of course products and their course modules. You see, it's a bit complicated, but it is essential to get the concepts right to make correct requirements on information architecture and database design. Otherwise, both development and the use by professional teachers will be a complete mess.

Here are some initial process charts we created for the 'Course delivery' process.

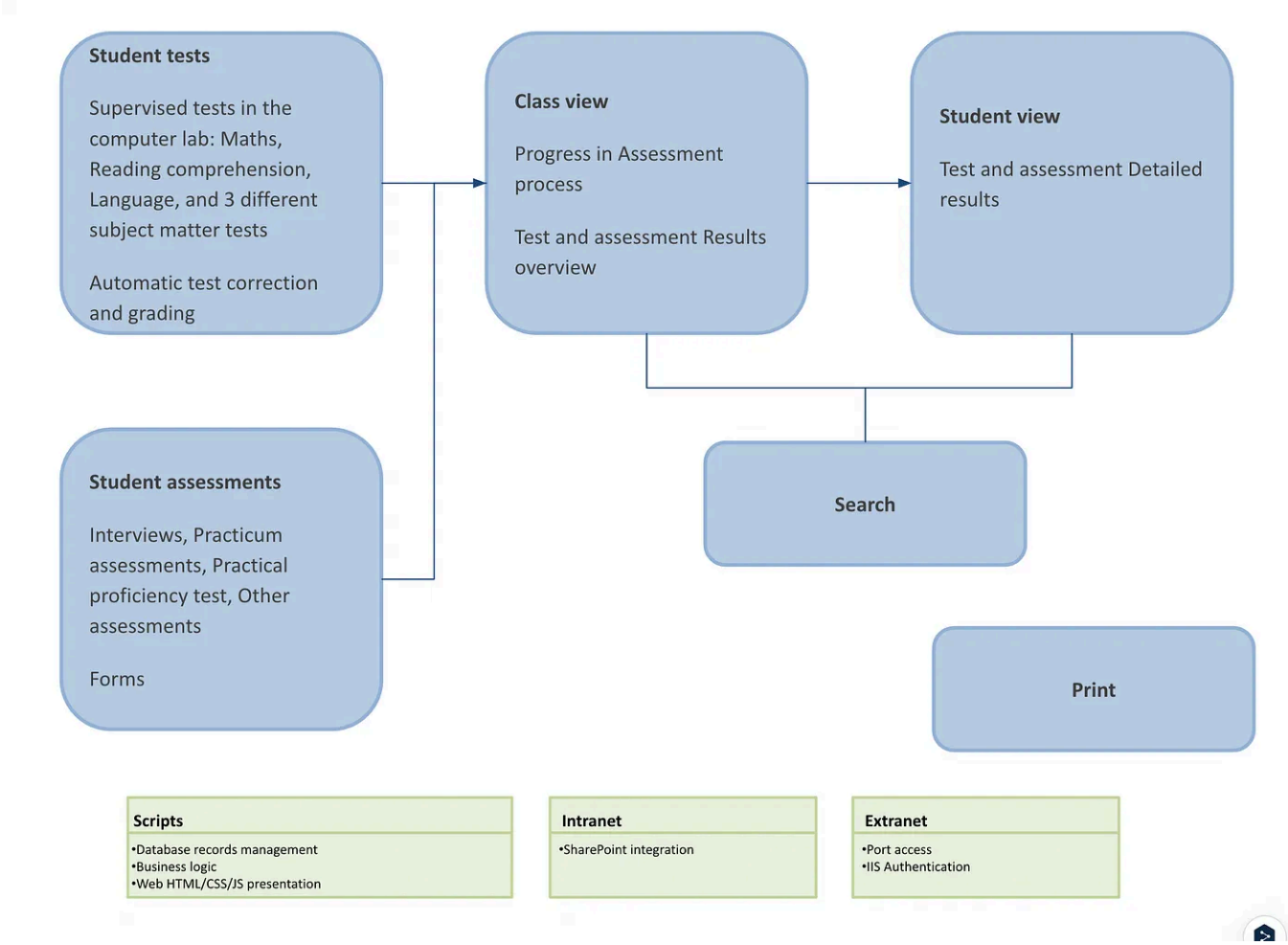


These illustrations show how to use the 'Course' application.



The classroom delivery process should be further divided into class sessions and specific class moments, including the delivery and use of course material and the registration of absences.

The ‘Assess’ application is at a similar level, but is otherwise completely different.



Sketch of the ‘Assess’ application, later at the Design phase.

The shared base system requires an advanced authorization structure, a database, a web content management system, robust security measures, and various integrations.

The deeper you explore, the more complexity you discover. We’re still in the needs analysis phase. No IT yet. Properly conducting this analysis is essential for success. What I’ve outlined here provides the general idea; many details remain.

End of Part 2 — Next: How we turned requirements into a working MVP and extended it for a decade.

Part 3 — Building the MVP and Working Truly Agile

From requirements to ten years of uptime

With the scope clear and models ready, we moved fast. Within weeks, the first features were in production.

Here's how we built, tested, deployed, and improved — with zero bureaucracy and true agility.

Providing the information system

> *An information system is used to collect, store, process, and distribute information, usually with the support of IT, to support communication and work within and between organizations. It encompasses not only software and hardware, but also users and organizational processes. The main functions of an information system are to act as a memory, a source of information, and a support for action.*

Here, we refer to an IT-based information system comprising two applications supporting digitalized business processes. The system, which includes a database,

business logic, and user interface, was developed based on a needs analysis.

Overall requirement specification

The requirements specification serves as the bridge between needs and technology. Here, we assess available and relevant technologies, considering both technical possibilities and limitations, to derive realistic requirements that address those needs. The requirements specification is a technical and user-centered document that outlines what the system must do, including its functionality, performance, security, and other quality expectations.

We had already established that economic bookkeeping and resource planning were outside the scope.

We then quickly made additional decisions regarding further limitations.

We specified this system to integrate with Microsoft infrastructure for simplicity and control.

We also specified that this system would be entirely internal, running on the new intranet I was building in parallel. The extranet was only to be used temporarily for emergency support.

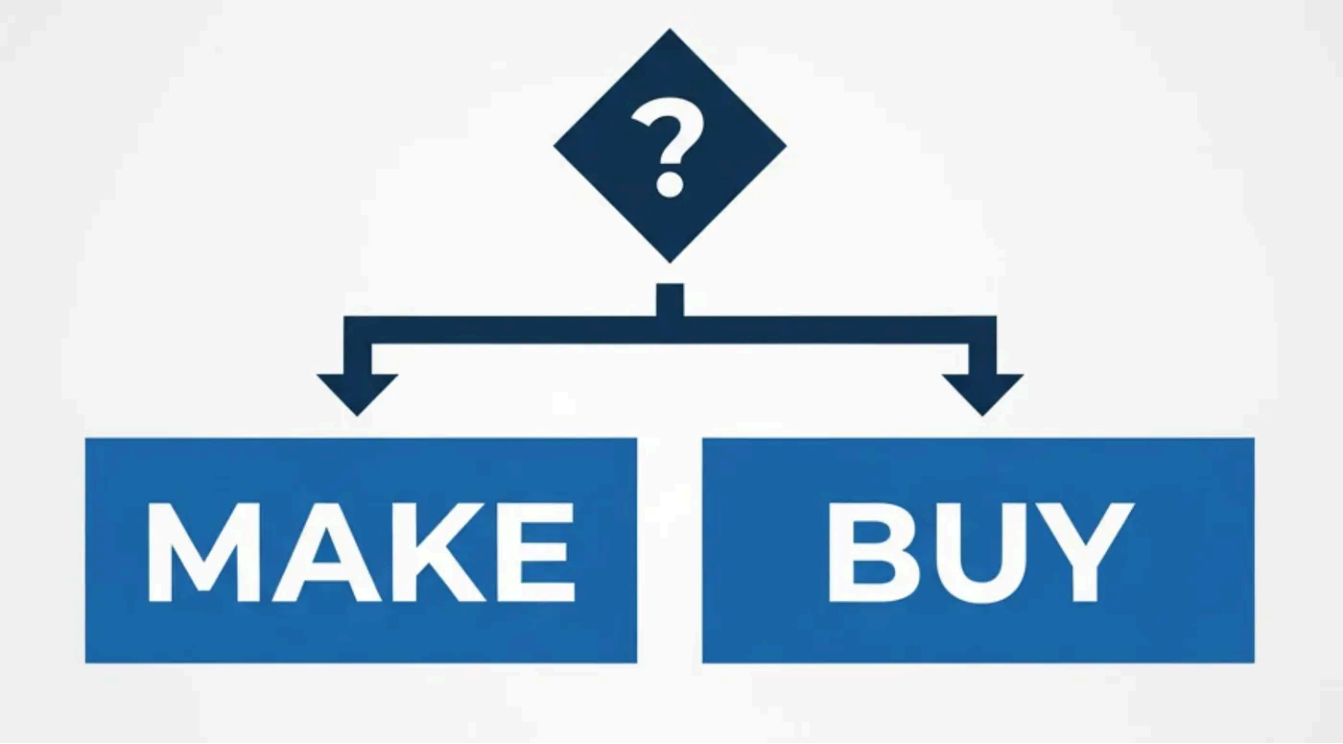
Everyone registered in our system would come to us through various channels, such as email, telephone, or in person at the reception desk. Changing this was not appropriate from a business standpoint.

By doing this, we get some things for free. Only intranet use meant low-level requirements on information security and graphic design.

Make or Buy

Before I could detail the requirements, I needed to understand how to provide the necessary business information system. Some parts of the requirements vary depending on the chosen platform.

So, the next big question was: make or buy?



I considered the available options in order.

1. Find a SaaS service.
2. Purchase and configure an off-the-shelf course management system.
3. Develop the solution on SharePoint.
4. Use an open-source platform or middleware as a development base for our applications.
5. Develop everything ourselves (as a last resort).

1–2. We thoroughly scanned the market and conducted an in-depth evaluation of available alternatives, but none met our needs. Not at all. What you can buy and configure may suit schools or universities with standardized processes.

2. Even if it did work, purchasing and configuring an off-the-shelf system isn't always better, more fruitful, or advantageous. In some cases, like ours, it's not even possible. Off-the-shelf systems are closed. A bespoke system is necessary for us. We require access to the source code to develop software that supports the client's unique business processes.

3. The SharePoint alternative was complicated and would have been expensive and time-consuming due to the shortage of skilled consultants. We hired consultants to

tailor SharePoint for our intranet, but we faced only problems. At the time, SharePoint clearly was not a feasible solution for us.

4. After researching, I found an open-source platform intended for research surveys. It had all the needed features and seemed easy to use, adapt, and extend to fully suit our purposes. I reached out to the platform's Head of Support to learn more, which formed the basis for a reasonable hypothesis.

Now that I knew there was a possible solution and a way forward — a light at the end of the tunnel — I could proceed with the requirement specification work.

Detailed requirement specification

The requirement specification translates

- concepts into specific terms,
- goals into measurable criteria, and
- regulations into logical rules

The artifacts produced during requirement specification are typically much more detailed than those generated in the needs analysis phase.

These artifacts include:

- Information models
- User stories
- Use cases
- Feature descriptions, with business logic
- Performance requirements
- Security and compliance specifications

I leveraged my experience to create shortcuts for working faster and more safely, noting everything significant or out of the ordinary. While complete documentation is crucial for larger teams or handovers between teams, our small team of three managed with a lighter, agile approach.

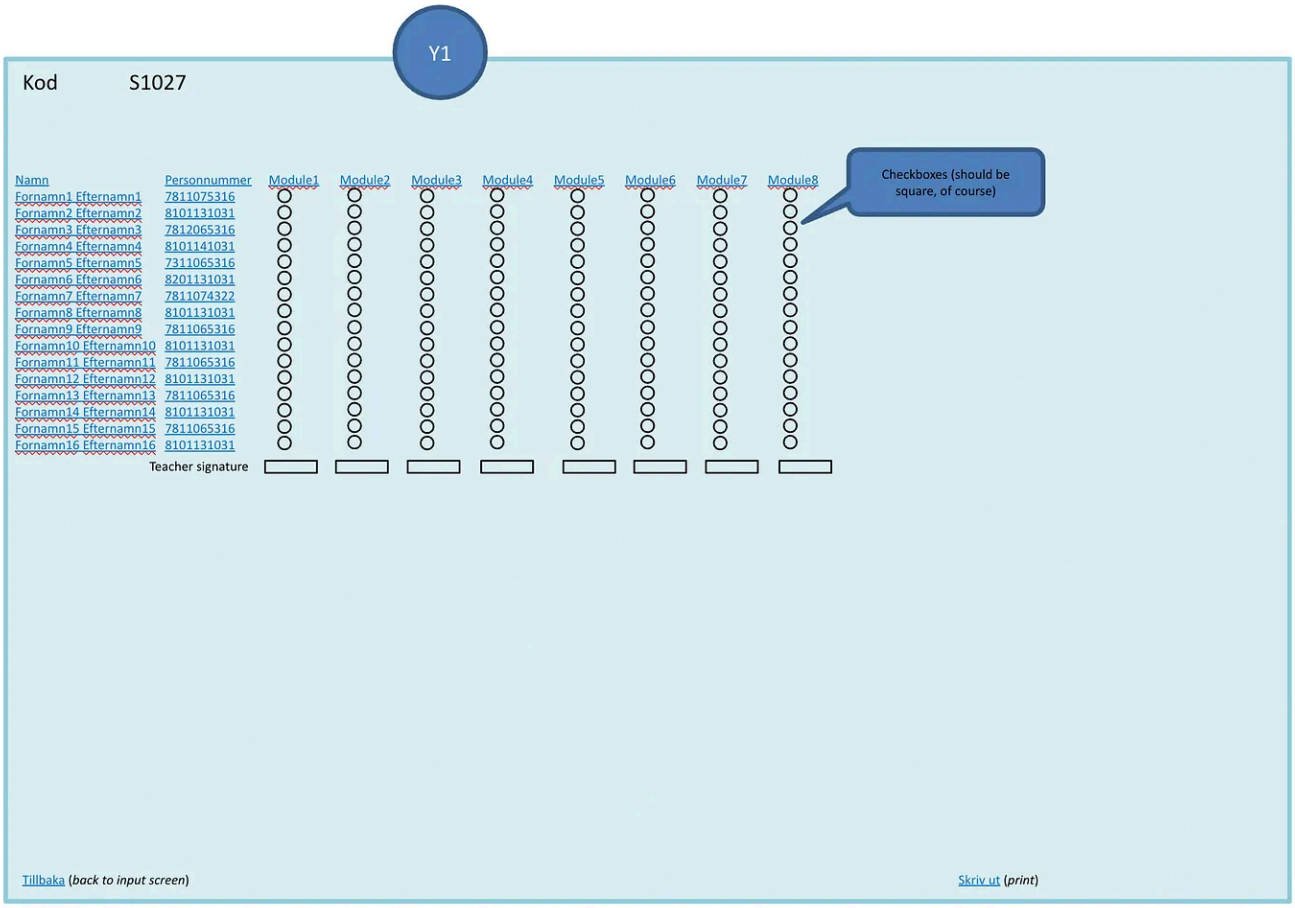
One issue was the Authorization structure, which is often a concern for company information systems. This is always an important area to consider when specifying. We used the same AD for the authorization tables for both this business information system and the intranet, based on SharePoint and email. However, in this business information system, we specified an additional authorization table, providing a much simpler way to control the highest level of access. Overall, it should be easy to manage authorizations or permissions.

Design

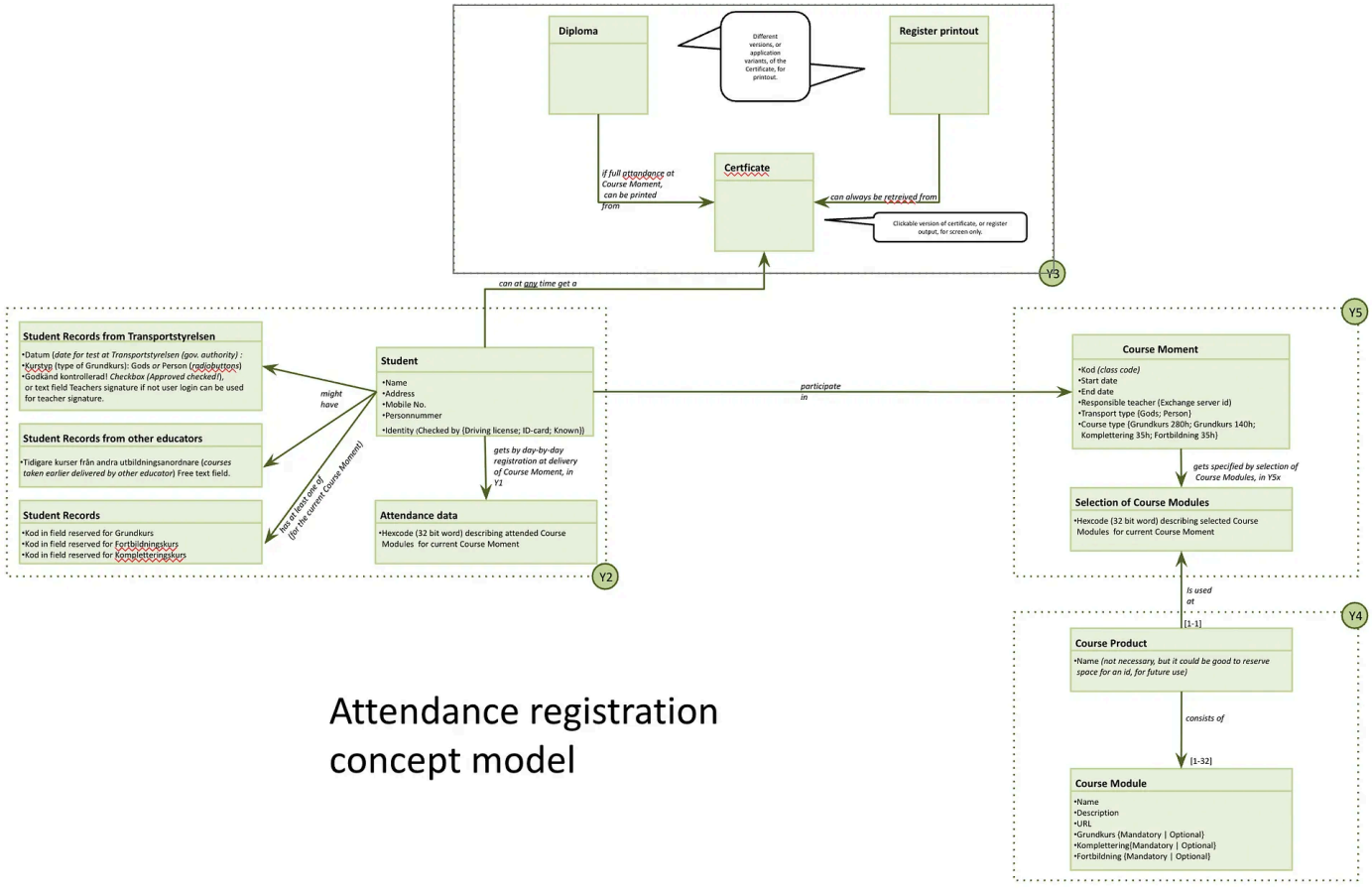
Design involves sketching and modeling, transforming requirements into goals for realization, functions into experiences, utilities into joy, and usability into a desire to use. An information system with user applications means modeling information and technical solution architectures, databases, and user interfaces. In general,

a) Application UI (layout, interaction design)
b) System architecture (components, integrations)
c) Information architecture and Database (data model, structure, relational schema)

In our case, we need to design the user experience based on the business processes, standardize the user interface across various screens, and prepare the user manual (primarily for reference or documentation in case of unexpected events) and, to some extent, the database. Given our strict constraints, we prioritize utility over beauty.



Feature design sketch in the works.



Attendance registration concept model

One of the initial Concept models, later modified.

While working on the design, I contacted and contracted Marcel, an expert on the open-source platform I found. I agreed with him to set up development, test, and production environments and develop the applications together.

Development and test

The development of the information system initially involved adapting the open source platform to the target environment (Microsoft) and performing certain configurations. As this was a survey system, it already had built-in functionality for conditional text registration, logical validation, storage, and retrieval in the database.

Developing the applications involves writing software code, including scripts for handling database requests, business logic, UI presentation, and routing. It consists of writing the user reference manual, which is a component that also needs "development."

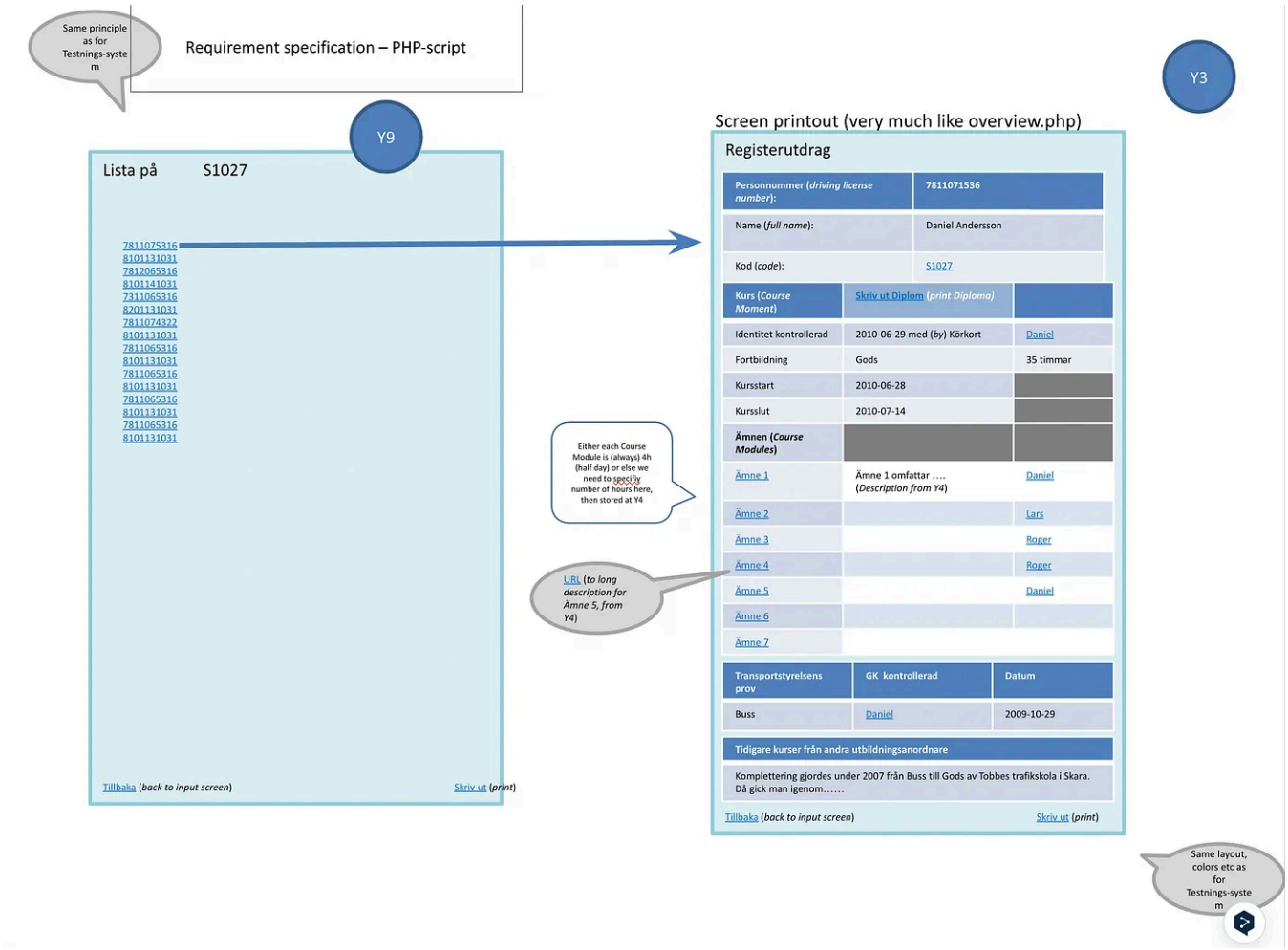
We want to adopt an agile mode of operations as quickly as possible, regularly implementing effective practices and building upon them to extend the application with more features. As soon as the entire process is supported and the applications have been implemented, we want to move into a DevOps mode.

After a few scripts, we had a fundamental application up and running, with the basic system features integrated and working.

Marcel and I worked a few hours daily using a chat application. Then, we had everything in print, and it was asynchronous. I sent the design sketches and explained the features, while Marcel brought them to life by scripting and testing. This was full-stack development, that is, both front-end and back-end programming.

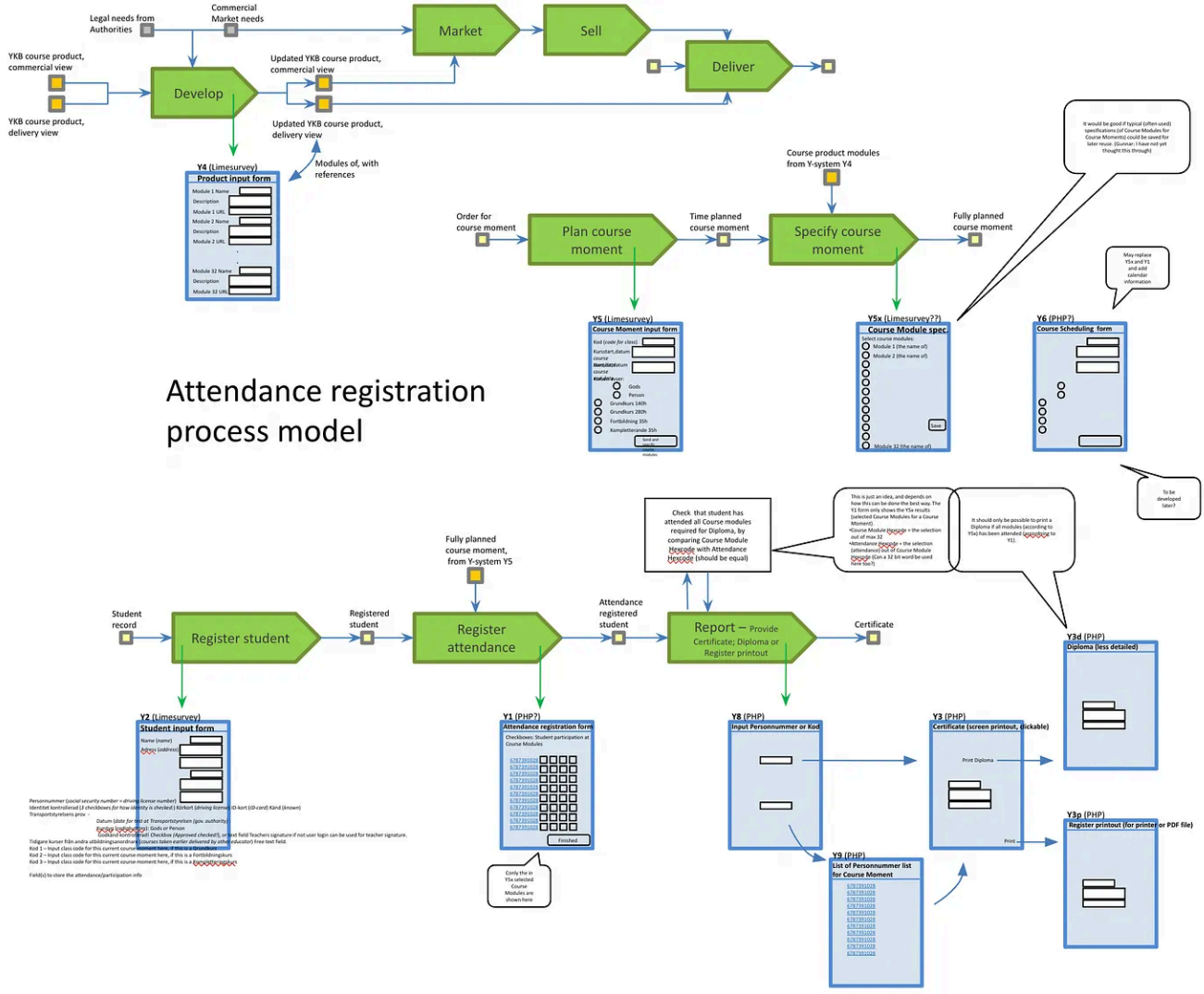
Design and development required interactive, iterative, and sometimes intense collaboration. Creating fully comprehensive design models that are complete and detailed enough to be handed off to development for flawless implementation without ongoing communication is not possible. Teamwork is crucial throughout the process. This teamwork is most effective when, like in our case, team members share the same skills but have different areas of expertise.

The applications followed the business process, where each process step involved a small number of user tasks. We defined a feature of the application to support one or a few tasks. Then, we used the principle One feature = One screen = One script file. This design approach worked fine in development, providing an easy way to progress and a user experience that made sense. Users don't execute commands by looking for menu options; instead, they follow the value stream from screen to screen, entering data, reading data, and clicking on buttons to either proceed or execute external actions (for instance, printing a certificate on paper or sending a report to authorities).



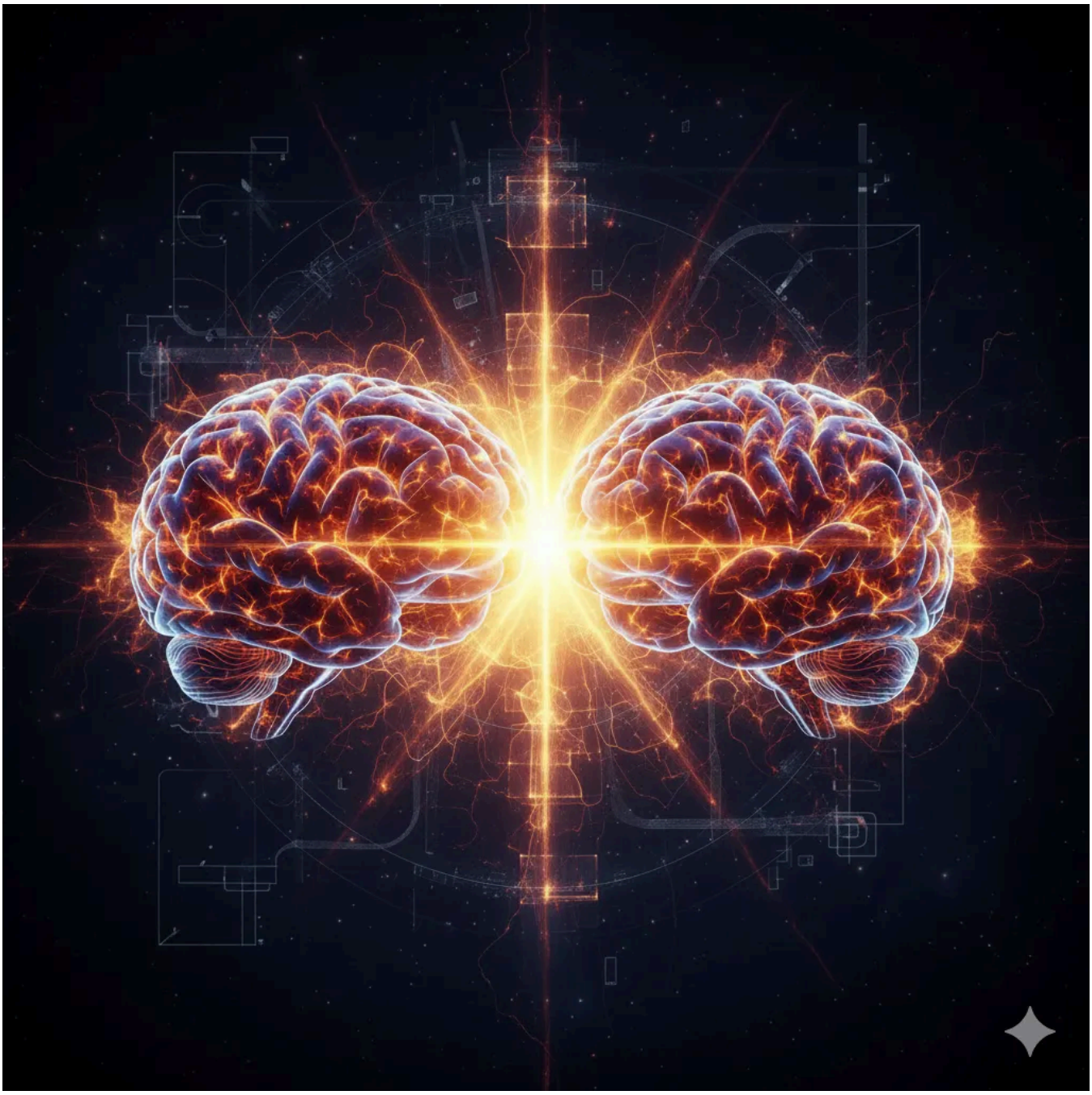
Feature design in the works, Student record. A basis for development.

Development and testing for each feature took around three days of part-time work. Marcel and I worked almost the same number of hours per feature due to extensive communication with the client company (apart from communicating with Marcel and testing). We worked consistently in an agile manner.

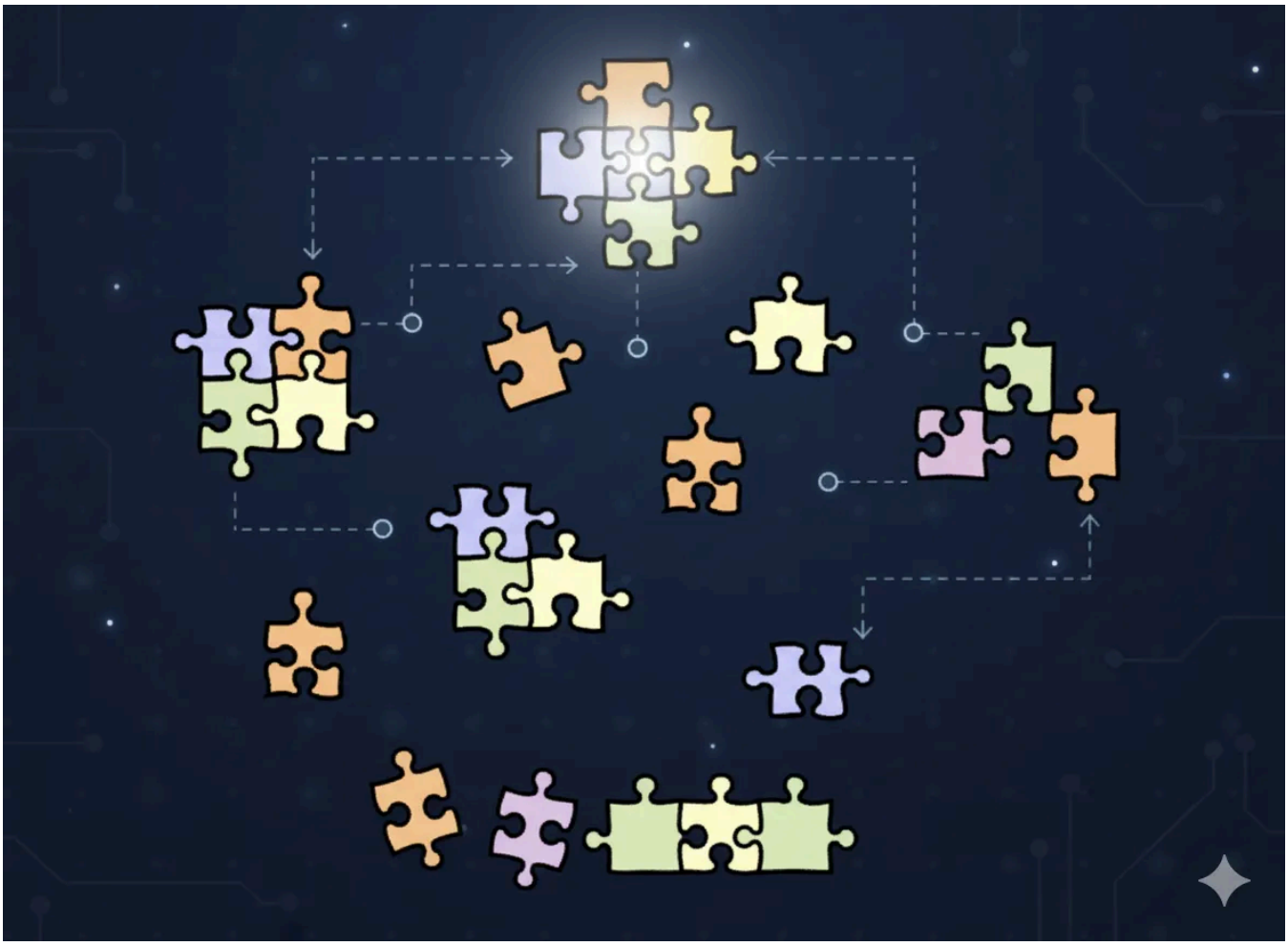


As soon as we had something useful for the client organization, the MVP, we started with a small user group in production. We then extended the functionality of business process support and the user group step-by-step.

We started by implementing the ‘Assess’ application. After it was up and running, we moved on to the ‘Course’ application.



In the ‘Assess’ application, we encountered some tricky features that required significant ingenuity. Marcel and I have Master of Science degrees in Engineering or a PhD, so we are used to demanding problem-solving. Challenging aspects included automatic test correction with language fill-in questions, handling multi-answer questions, enforcing one-time testing per student, and securing questions and answers. While modern AI could have made things easier, we solved these issues ourselves.



A few weeks later, while working on the ‘Course’ application, we faced several challenging features, such as configuring course modules for specific course products and defining course occasions linked to those products. Registering absences during class sessions, retrieving course materials from SharePoint, and compiling and printing student records proved more demanding than expected.

Application user test

After we had developed almost all of the applications — excluding advanced features such as management statistics, which were added later — I allowed students to assist us in testing the applications thoroughly. Together, the two applications comprised over 50 different screens and included several integrations.

Implementation

The organizational implementation, which included the deployment, ie. putting the system into production, launching the applications, training the staff, and transitioning the organization from its previous processes to exclusively using the new applications, proceeded surprisingly smoothly.

DevOps — Maintenance — Support

For us, maintenance meant continuous delivery, monitoring, problem-solving, bug fixes, improvements, and enhancements based on new needs that arise. It was DevOps-like, but didn't take us much time.

I set up a webpage on the intranet for easy user reporting. The applications and system functionality must always work smoothly and carefree. If a user experiences a hiccup, they can easily report it, and we'll solve it almost instantly.

Phase-out

After our baby, the business information system described in this story had been in service for ten years, and the business had evolved; it was time for a controlled decommissioning, data migration, and transition to the new system, in which Marcel and I participated. And this was the end of a successful journey, bringing this story to a close.

A comment on agile

In large software development companies, adopting agile methods often results in a flipped or overlay organization where epics are run instead of projects. While this structure may allow you to predict costs and delivery dates, you lose clarity about what will be delivered in each iteration. On a broader scale, you might see the final product but not know when it will be ready or what it will cost, which can lead to losing control over the process. This flipped organization does not increase effectiveness or agility. Instead, it simply highlights a lack of strong project management.

We genuinely worked in an agile way.

For a software development company making standard ERPs, where all customers use the same code base and need to make configurations to adapt to business needs, a smaller change request, for example, adding a condition to a business rule, might take the customer months or even years due to complicated processes and immense communication between people in-house and with many customers. Issues are posted on Jira boards, transferred to other boards, become anonymous, and then transferred to Azure boards. They must be prioritized, moved back and forth between customer teams, business analysts, product owners, development teams, and so on. It's a mess.

In our case, when we receive a demand to add a condition to a business rule, or, more accurately, when we discover the need ourselves because our client lacks the necessary competence, it takes us hours to implement it in production.

> *Hours, not years. Minutes, not months.*

We changed the code, tested it carefully, and then pressed the button.

That is being agile.

In parallel, I also developed the intranet on SharePoint, which includes information about the organization and its activities, document management, and staff collaboration. In both cases, I worked as agilely as possible, for example, by prioritizing the product backlog to avoid waste and focus on doing the right things.

My main point is that the most critical work is the business analysis, needs analysis, and specification of technology-independent requirements. Technical development and solutions can often be implemented, but may vary over time. Modern IT can offer several advantages over older techniques, providing numerous benefits to the final information system. However, knowing what is needed and preferred, as well as how the business processes will perform with the support of the information systems, is essential for success.

If we had worked in the waterfall paradigm, the requirements specification would have been sufficient for any developer to use their preferred IT to make a quote. The same qualities are valid for lean-agile teamwork.

If you want something done, you must define the scope, as this is one of the ideas behind the MVP agile approach.

Summary of effects and impact

The business information system we developed was used by over 100 admins for 10 years, supporting more than 10,000 learners, hundreds of classes, dozens of courses, and additional resources. It enabled the expansion that made the Gazelle company an actual gazelle.

The applications offered considerable benefits in terms of

- efficiency and resource utilization,
- providing a standardized approach to working, and
- ensuring consistent, high-quality processes.

Learners accepted their assessment results without question.

The business information system was stable and well-functioning for ten years, with low operational and maintenance costs.

Course coordinators and teachers appreciated the ease of use.

I enjoyed leading the digitalization process, from needs analysis to implementing modern web solutions.

How I would do this again

I would repeat this approach using the latest technology. Specifically, I would use the tech stack that AI tools like Claude Code utilize by default. In fact, I leverage such AI tools to write code and enhance the overall product. AI would assist me in every step and task of the project, dramatically improving both the process and the result. I would use a team of AI agents for the specification, design, development, and testing of features. To make this effective, I would use someone with experience in similar projects.

A few years ago, before tools like Copilot, Cursor, Claude Code, or Gemini CLI existed, my response would have been different:

I prefer declarative functional languages, such as Elm (which is specifically designed for web apps and uses pattern matching), or logical languages like Prolog, over imperative programming languages. This is because my focus is more on business and operations development and solution architecture than systems

development. I model business needs through process models (with resource layers), target models, stakeholder models, rule models, and concept & information/object models — different perspectives on the business requirement. This business architecture can then be naturally translated into program architecture in languages like Elm. The value objects in the process have different states, which are modified by the functions. When the programming language aligns closely with business process analysis or the user journey, development becomes faster and easier.

Key Lessons for SMB IT Projects

- “Business analysis matters more than technology.”
- “Lean expertise beats large teams.”
- “MVP must be production-ready.”
- “Agility is a mindset, not a method.”

💡 Like?

Please clap, comment, or share if you found this story helpful. Your actions help spread the insights to more readers.

About the Author

Gunnar Östberg, Business Transformation Architect, CEO, Business Clarity.

Gunnar is an M.Sc. computer engineer with 40 years of experience in **business and IT-driven transformation**. He has worked with global giants like Ericsson and Scania and nimble high-tech firms. Passionate about sound architectural principles and systems thinking, Gunnar writes to expand perspectives and deepen insight. Through his consultancy, Business Clarity, Gunnar helps organizations evolve ideas into sustainable systems and products, transforming their business through structured, principle-based innovation. His focus areas include pragmatic architecture, business modeling, and delivering real impact.

Small Business | Erp Software | Information Systems | Implementation | Web Development

Edit profile

Written by Gunnar Östberg

170 followers · 120 following