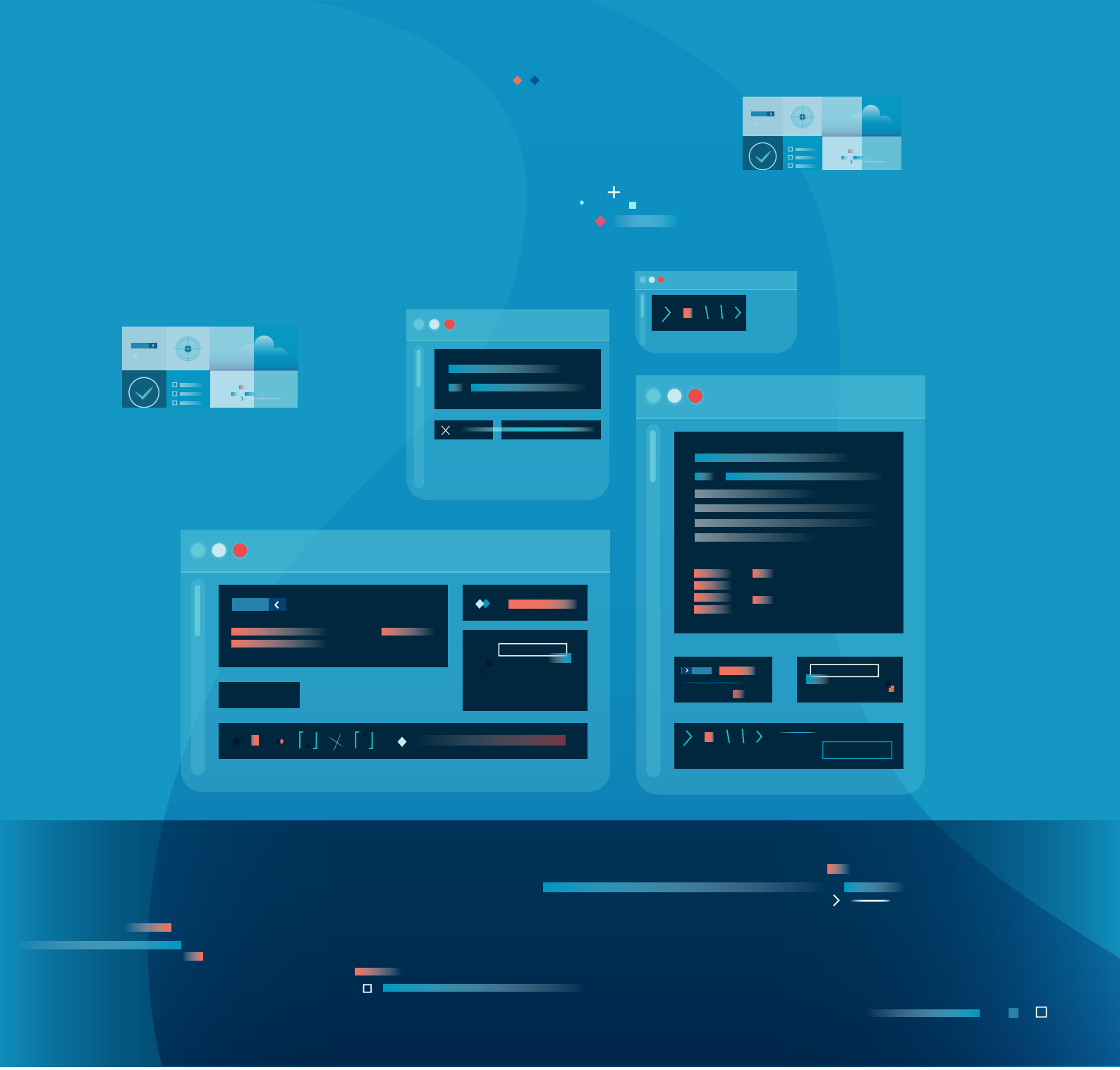

cegeka

Still future-proof after 15 years: the remarkable case of Ventouris

Software that is state of the art today, might already be a thing of the past tomorrow. Not the case for the Ventouris e-portal, which easily keeps up with the latest software developments. Quite the achievement if you know that the application handles complex administrative matters. The secret to building a software platform that effortlessly stands the test of time? Have a look behind the scenes of the application that seems immune to software rot.

The birth of Ventouris: building a timeless application

The roots of Ventouris go back to the 1970s, when Cegeka – then called Orda B – was asked to build a core application for the insurance funds for self-employed workers. "The initial platform was built using first Assembly and then Cobol. As Belgium's social security system evolved, so did our software. We had to update the functionality so frequently that the Cobol backbone had become difficult to maintain. In 2004, we decided that it was time to rebuild the platform from scratch, moving from a mainframe to a server-based system," says Johan Lybaert, Vice President of the Social and Government division at Cegeka.

Developing a mainstay platform

Johan was present at the birth of Ventouris as program manager. Today, he still considers Ventouris as one of his biggest accomplishments. "In 2004, we decided to start fresh, building software with really clean code. The idea behind the new platform – which we called Ventouris – was equally straightforward and challenging. We needed to develop an application that would stand the test of time. Building a new software architecture is expensive and time-consuming, so we combined a reliable business model and new methods of software development to make Ventouris a mainstay. The objective was to reduce the infrastructure cost by 50%."

> *"The objective when rebuilding Ventouris was to to develop an application that would stand the test of time."*
>
> **– Johan Lybaert, Vice President Social & Government Division**

The roots of Ventouris

Back in 1968, the Belgian government made major reforms to the social security system. It was the start of social security contributions by self-employed workers, and the child benefit system was set in place. The government gave mandates to a limited amount of companies to collect these contributions.

"The reform introduced the social security system as it is still known in Belgium today," Johan explains. "It's an area that is continuously under scrutiny: the system is constantly being changed and tweaked by a government striving for the most ethical solution. That's what makes the Ventouris case so interesting. We needed it to be very flexible so we could adapt to these changes quickly."

The business model: cut and share costs

Ventouris had to be built in a way that it would be easily maintained for years to come. The cost of building the application was non-recurring and shared by seven companies at the time. Today, nine mandated companies use Ventouris. They pay a predetermined yearly maintenance fee.

Reducing the infrastructure cost

Ventouris is a massive e-platform, used by 500 accounts managing the files of 900,000 active self-employed workers in Belgium. Cegeka factors in about 4,000 man-days every year to maintain the application. "Admittedly, the maintenance cost is significant. But since we don't have to create a new architecture every few years, the cost in the end is much cheaper. Sharing the platform reduced the cost, which is a win for all stakeholders," Johan explains.

Sharing costs within an ecosystem

Since fewer than a dozen companies are mandated to collect social security contributions in Belgium, it makes sense to build a software package for the entire ecosystem. Cegeka has a track record of cost-sharing projects, where several stakeholders combine investments to use one back-end application. This principle was perfect for Ventouris, as the legal stipulations for collecting contributions are identical for all users.

Audit recognition

The first contract was a nine-year term, lasting from 2005 to 2014, followed by another five-year deal continuing until 2019. The current third contract expires in 2026. That amounts to an impressive 21-year term for the same application. "We had the third deal audited by two external firms: Software Improvement Group and Inno.com. The results of the audits were clear: we've been consistently keeping the software state-of-the-art, maintainable and offering it at a correct price, which justifies the contract extension," Johan says.

> *"The results of two independent audits were clear: we've been consistently keeping the software state-of-the-art, maintainable and offering it at a correct price."*
>
> **– Johan Lybaert, VP of the Social & Government division**

Technical acumen

"Traditionally, software like this lacks technical depth after about seven to ten years, which means you then need to build an entirely new architecture. We convinced our clients that we could do it differently," Johan explains. "If we are guaranteed a yearly maintenance fee so we can dedicate a team to this project, we can permanently keep the architecture components state of the art, for instance, by using open source components. This doesn't add anything functionally, but from a technical point of view, the software is on par with the components you would use if you built the software today."

> *“Traditionally, software like this lacks technical depth after about seven to ten years. By relying on a yearly maintenance fee, we permanently keep the components state of the art, so we don’t need to build a new architecture every few years.”*
>
> – Johan Lybaert, VP of the Social Impact division

DevSecOps

“Before Ventouris, the industry standard was to have three new releases every year. Because we have a dedicated team working on the application full time, there is a new release every two weeks,” says Johan. “This has several advantages. The first one is obviously the flexibility to quickly implement changes. But perhaps more importantly, every release requires only a minimum amount of testing. If you have one big update every four months, you need a huge user base to test whether there is software regression. By releasing small updates and implementing minor features constantly, the testing is minimal, and we can assure the users that there is no software regression.”

> *“By releasing small updates and implementing minor features constantly, the testing is minimal, and we can assure the users that there is no software regression.”*
>
> – Johan Lybaert, VP of the Social Impact division

Software development: keep it clean

It seems logical to keep the software up to date through small, regular releases. “But many companies fail to implement this successfully,” explains Natalie Vanderhasselt, Cegeka sales manager for Ventouris.

> *“From developers to analysts, everyone applies the same high standards day in, day out.”*
>
> – Natalie Vanderhasselt, Sales Manager Ventouris

“We’ve invested heavily in automated testing to be able to do this. Automated testing ensures a high-quality standard and it offers us the flexibility to release a new version biweekly. If the law is adapted, for example, we can quickly switch our priorities to implement needed changes. For small changes in functionality, we can plan them immediately and have them in production a few weeks later.”

Best practices combined

This is just one of many best practices Cegeka applies in the exemplary Ventouris case. “Many companies apply some of these principles,” explains Natalie. “But what sets Cegeka apart is that we combine them in the most disciplined way throughout the project team. From developers to analysts, everyone applies the same high standards day in, day out.”

“We constantly have 40 people working on this project. Of course, there’s been some rotation over the years,” Johan adds. “The fact that the way of working has not changed proves that the code is highly maintainable. If we add new people to the team, they are working on code that is 20 years old. Without clean and readable code, this would be impossible.”

> *“A team of 40 people is working on code that is 20 years old. Without clean and readable code, this would be impossible.”*
>
> – Johan Lybaert, VP of the Social Impact division

Pair programming

One of the techniques Cegeka swears by, is pair programming: every line of code is immediately reviewed by a second programmer. This is a technique that requires the code to adhere to an Olympic standard. “Without pair programming, there are always small differences in how the programmers write the code. By using this technique, every line of code is immediately reviewed by a second programmer. This ensures that everyone in the team uses the exact same way of writing code.”

Again, this is an example of how Cegeka uses best practices in a way that few other companies do. “There have been quite a few studies on pair programming,” says Johan. “As it turns out, less than 10% of companies actually apply this technique. The main reason is because managers assume it is twice as expensive when you have two people working on the same code. In reality, pair programming offers many advantages in the long run, including better cost efficiency.”

> *“Pair programming offers many advantages in the long run, including better cost efficiency.”*
>
> – Johan Lybaert, VP of the Social Impact division

Scrum pioneers

The scrum method is omnipresent in software development today, but Cegeka already applied this way of working in 2005. “We were pioneers in Belgium back then,” Johan says. “Team members take ownership of the tasks and work cross functionally as a team. There’s no project leader delegating the tasks: we take collective ownership. This assures continuity, as you’re not depending on one or two crucial people in a team.”

Quality parameters

The solid combination of best practices leads to the level of excellence Cegeka is well-known for. “We have many visitors who come and check our way of working in our Digital Factory,” Natalie confirms. “There are not that many firms with the same level of discipline when it comes to these best practices. All this leads to high-quality software that is still up to date today. We can count the number of bugs we’re fixing on two hands at any given time. Severe bugs are what we call ‘incidents with business impact’. In all those years, we have never had any incidents. Moreover, the performance speed or functionality have never been questioned by users. These are quality parameters proving the Ventouris platforms reliability.”

Specializing in complex large-scale projects

To stay ahead, Cegeka is continuously planning for the long-term as well. As the current Ventouris contract ends in 2026, the team is already looking ahead to 2040. “We need to carefully consider what we will need to do in the future to stay best in class,” says Johan. “For instance, the software trend now is a component-based structure comprised of microservices. By contrast, Ventouris is built as one monolith. Instead of rebuilding from scratch, we are slowly reshaping the architecture to have a more component-based structure, step by step.”

“Few companies can deal with this kind of complexity. It’s all about taking a complex project, dividing it into domains and developing all these subtasks into a high-quality product.”

– Natalie Vanderhasselt, Sales Manager Ventouris

Engaging new talent

There’s another advantage to our way of working. “We continue to add young talent to the 40-person Ventouris team. Investing in continuous training and keeping the in-house knowledge up to date is a priority at Cegeka. Challenging work is the key to maintaining their engagement. Our way of working ensures that the people on the team develop state-of-the-art software even after 20 years, and at the same time, we offer the best quality to our clients.”

It’s this approach that defines Cegeka as a software company. We have a vast proven track record of building software with a combination of best programming practices in a cost-sharing business model.

“Few companies can deal with this kind of complexity,” says Natalie. “We’re currently working on another application for a payroll system. It’s equally complex, but our clients know that we are one of the few companies that can deal with these large-scale projects. 70 to 80 people will work simultaneously on this application, equivalent to eight or nine scrum teams. It’s all about taking a complex project, dividing it into domains and developing all these subtasks into a high-quality product. We have the knowhow, experience and organizational capacity to do this,” Natalie concludes.

WWW.CEGEKA.COM