Technology is evolving at such a high pace, that we need to put efforts to keep up with innovation.
KEEPING UP WITH INNOVATION

Technology is evolving at such a high pace, that as we are witnessing how the future becomes present, we also need to put efforts to keep up with it. This especially concerns industries like high tech, energy, automotive, logistics, industrial automation and manufacturing.

New systems are deployed daily, computer languages keep growing in numbers and complexity, products are becoming more multi-dimensional. Many businesses still struggle to catch up with these developments.

Old systems don’t have to be broken and non-functional to be modernised. But software modernisation is not a quick fix either. It’s a continuous process that ensures old systems don’t put a strain on the business and that enhances the lifecycle of a product.

Introduction of new code on old systems is more difficult and it can result in bugs and unpredictable changes elsewhere in the code, even if a monolith is rock solid. Companies that fail to modernise their legacy systems might still serve current business needs but might not be able to attract new clients or stay relevant as they need a responsive UI or a complete code rewrite.

WHEN SOFTWARE MODERNISATION IS MANDATORY

Legacy systems don’t have to be broken to be modernised

Many businesses wait until their operations can’t continue as usual to realise they need software modernisation but this shouldn’t be the case. Software modernisation is driven by different business and technological needs and can have many aspects, forms, aims, and benefits.

• The code is too difficult to maintain and the archive needs better modularity
• Every new feature that is added causes code changes elsewhere
• A 32-bit system needs to be updated to a 64-bit system to improve the overall performance
• The UI is outdated or becomes non-responsive

At Strypes, we use a modular approach which results in a painless transition and quick, efficient development. Our years of experience working with high-profile clients led us to create our own strategies and work models. We use automation to deliver on time while we also make sure that the final product has the maximum quality when it comes to looks and functionality.
THE STRYPES APPROACH TO SOFTWARE MODERNISATION

We enable our clients to evolve their business and prepare for the future

Software modernisation is not a one-off service. It’s a partnership that we nurture throughout the years while renovating our clients’ legacy systems and adding value to their business by supporting innovation, increasing efficiency, and reducing costs.

Choosing the right partner who brings expertise, focuses on the solution, takes full ownership of the project and cares for the long-term growth of your enterprise is essential. This is why we have made it our priority to do exactly that.

We enable our clients to evolve their business and prepare for the future. Strypes’ unique work model allows us to have seamless communication, transparency, and a top-down unwinding of the issue from the very beginning.

ARCHITECTURE UPGRADES AND INTERFACE IMPROVEMENTS

We managed to automate 50% of the changes in the work process, while the rest of them were domain-specific and required our client’s involvement

Our client (under NDA) is a major industrial manufacturer. Their software group has been developing and maintaining legacy software with a lifetime of over twenty years. A large number of interfaces were provided but they never had the resources to dedicate a team that can manage this whole structure so they approached Strypes.

Our challenge was to achieve better modularity of their archive as the code was too difficult to upgrade, change, and maintain. Every new feature caused unpredictable changes somewhere else and so our client needed a better encapsulation of the whole code. We had to segregate modules of the code so that they can be managed more efficiently by the dedicated teams.

First, we had to come up with a strategy to execute multiple packages in parallel, without disturbing the workflow of the client. Not in all cases, automated tests with full coverage were available. Due to the high complexity of the system and the multitude of hidden dependencies, there also were changes of functionality outside of our changes that heavily affected our work. This meant that it was very difficult to plan for the timeframes of the changes and there was a real risk of scope creep – an expansion of the project’s goals while the work is in progress.
To solve all these issues, we first had to dedicate resources to understand the client’s needs and outline the exact scope of the project, breaking it down into logical packages. We then set priorities, separating the critical packages from those that can be executed last (nice-to-have). We also had to analyse the major risks and the dependencies between the separate packages. Aligning with the client on these questions allowed us to form the right subteams for the separate items, depending on complexity.

To achieve maximum quality of our work, we regularly sent teams on-site for periods of a few weeks to ensure that complex changes were clearly defined and agreed. Later on in the project, we identified an opportunity to increase the overall quality of the final product and added more people to the team, which also ensured that our work won’t be disrupted by scope creep.

We managed to automate 50% of the changes in the work process, while the rest of them were domain-specific and required our client's involvement. Relying on continuous development and deployment for each package allowed us to avoid disturbing the client’s processes as the project progressed. We created a roadmap that predicted the impact and timing of each change and also defined regression test cases, executed reference runs, and defined progression tests to supplement the migration.

Our client gave us positive feedback as we made an effort to continuously push for tasks that were beyond the scope of our work. We even got to some of the low priority “nice-to-have” issues, after fulfilling all essential changes. The result was a modularised archive, the parts of which can be exchanged and maintained with real efficiency and predictability.

NEXT STEPS

The client gained an internal team of professionals who understand their needs by applying years of knowledge. We approach modernisation from the perspective of both our client and their supplier, treating the project as our own venture. The result is complete transparency, predictability, and stability for our client.
The goal was to migrate our customer’s software so that it is 64-bit compatible. A 64-bit system allows for more RAM so that it can run new third-party applications without complications. On top of that, many of the new third-party packages are now only available in 64-bit. This type of software modernisation also improves the overall performance. The solution required us to ensure seamless communication as we had to align with more than 50 stakeholders to achieve the different goals during each phase.

Our team had to follow the client’s timeframe and deadlines in which they required us to migrate their system. Within a set period of time, we had to migrate more than 5000 files. There was no room for errors. We had to be efficient and ensure thorough compatibility with the 64-bit environment.

What is more, the workload was growing continuously throughout the project, as our client’s over 1000 developers were constantly introducing new features that were only compatible with the 32-bit system, but not with the 64-bit one. The complexity of the whole project was challenging but we managed to deliver the work, relying on the classic project management methodologies for the initial stages, and then on Agile, Scrum, and Kanban for the stages that followed.

The second case study is also under NDA. Our client was looking to convert their system from 32 bit to 64 bit as technology improvements were hampered by the 32-bit systems 4GB process space limit. The client needed us to come up with a solution and implement it.
CROSS-PLATFORM MIGRATION

We began by producing a comprehensive impact analysis and identified the files that needed to change. This was followed by a design document that set out the specifics of each phase and how they were to be achieved.

In regards to the challenge of communication, Strypes tailored a detailed plan that envisaged catch-up and kick-off meetings at every crucial stage of the project. We also had a dedicated person on-site with the client, who was responsible for carrying out all discussions and meetings. This practice is at the core of our own Nearsurance model, which has proved to be the most effective way of smoothing out communication hardships in virtually any project we have completed.

In order to meet the client’s deadline, we deployed a high degree of flexibility and responded to the specific issues of each phase as the project progressed. Strypes involved larger teams of up to 20 people for the most labourousome phases, where we could do the work in parallel. We also drafted clear priorities for the teams to follow and thus ensured maximum time efficiency. We largely relied on our previous experience with software modernisation projects for the planning and execution phases. Our developers worked closely with the QAs, the architect, and the project manager to ensure completion of each phase of the project. Strypes also brought our scrum masters to ensure that all teams were synchronised with the goals and that we are all working in the right direction.

We overcame the complexity challenge by deploying a dedicated competence team to get in-depth knowledge of the subject matter. Together with the client, we attempted to address the question of efficiency through automation. This proved impossible in the given case. Thankfully, we mitigated the negatives to a great extent by leveraging on our communication plan.

Strypes researched several options for avoidance of introduction of the new non 64-bit compatible code. After the automatic detection proved not possible in this case, we used a special communication strategy to minimise the introduction of new issues. We communicated with the client’s developers regarding the most common 64-bit non-compatible cases and asked them to avoid them and to check for them during the reviews. All critical features of the software were tested and fixed before the switch from 32-bit to 64-bit. After the switch, we tested all non-critical parts.

NEXT STEPS

The client was extremely satisfied with the final product of our work – a system that runs smoothly in the 64-bit environment. All aspects of the migration were completed. The client’s developers had to make minor customisations. Thanks to the implemented modernisation, our client can work with more modern software, requiring more than 4GB of process space, taking advantage of the technically demanding systems and applications, enabling the future development of the enterprise.
REQUEST A CONSULTATION

The examples that we reviewed showcase a small fraction of how to apply Nearsurance as an outsourcing model that guarantees a solution.

The best way to show what we can do for your company is to offer a free consultation so that we can discuss your needs and show you what solutions we can build.

To request a demo, reach out to us at business@strypes.eu or call us to discuss your project needs.

Visit our website
Visit https://strypes.eu to read more about our services.