Arcanys

10 DEADLY SOFTWARE DEVELOPMENT TEAM EXTENSION MISTAKES (EXPLAINED)

The trend of extending a software development team with offshore engineers has steadily been gaining steam as more tech-enabled businesses discover the advantages of maintaining one, **particularly since the Covid-19 pandemic** Because really, once a tech business realizes the potential impact of an augmented team, what's not to like? With the ability to fill crucial missing roles in your core team, you can boost your tech output and get projects off the ground faster, all while keeping costs lower than if you were to hire locally.

But while the right software team augmentation partner is a valuable asset to your business, approaching the team extension model with misaligned expectations or rushing into it unprepared can introduce significant risks of failures, additional costs, and loss of time.

Let me be clear: I'm not claiming to provide a foolproof recipe for success here. But I believe that the knowledge and experience that our IT team augmentation company, Arcanys, has accumulated over the last 13 years in the software development space can be helpful to anyone interested in assembling and maintaining an outstanding offshore development team. After all, the more you know about common pitfalls and miscalculations, the better equipped you'll be to avoid them. Enjoy reading!

Frederic Joye, co-founder of Arcanys

10 Deadly Software Development Team Extension Mistakes (Explained)

1. Thinking a fixed-price project is a safe move
2. Thinking that the requirements can be built on the fly with no clear roadmap
3. Building your software product on a proprietary framework or platform that is developed & owned by the provider
4. Not having access to your source code on a regular basis
5. Looking for an outsourcing vendor that will blindly execute software development
6. Believing that doubling the team size will double the output in a linear fashion
7. Treating the outsourced resources as a different team separate from your company
8. Confusing price with value
9. Accepting bureaucratic processes and not having direct access to your engineers
10. Funding special: Thinking that there is such a thing as a free lunch - Work for equity deals

Let's take them in turn.

1 THINKING A FIXED-PRICE PROJECT IS A SAFE MOVE

If a service company offers an attractive fixed price for a complex and poorly specified project, it is probably too good to be true.

A very common request we hear is: "Read the requirements (actually just a few lines in a Word document), and please send us a timeline and a cost." I hate to say this approach is bound to fail. Here's why:

A. Fixed-price projects require clear-cut requirements leaving no stone unturned (spoiler: it doesn't exist)

In theory, a fixed-price project could work if the project scope and requirements are extremely clear and set in stone. In practice, however, this is rarely the case, particularly with startups and innovative projects. Due to the complexity of software systems and the uncertainty surrounding user needs, requirements are often somewhat vague, with scenarios that are not fully conceptualized and subject to change as the project progresses. The thing is, when working on a fixed-price project, **any unclear requirements leave the developer to fill in the gaps, often resulting in outcomes that do not align with the client's vision.** This is especially true if the provider operates in a different country and culture than the client. The scope and requirements often have to evolve based on new knowledge gathered during the course of development.

B. Estimating a project with a margin of error below 20% requires a crystal ball

If you insist on pushing your contractor to go for a fixed price, they'll have no choice but to try and estimate the risk and add significant padding to the project cost and pray it wasn't too underestimated. Since the risk is capped, clients are usually less involved in the project because it's more or less the burden of the developers to figure things out. As the project moves along, the developers start discovering more and more unforeseen issues, pretty much to the point that they will start going over his estimations and buffer combined. The only solution left for the provider is to attempt to limit his losses.

C. Everyone is losing in a fixed-cost project

How does the provider limit their losses? By cutting corners, and cutting on quality. They may remove their best team members for more interesting clients, and try to handle the project at the lowest cost. Code may not be as well handled as it should, documentation could be skipped, and important details can just be looked over. Undoubtedly, deadlines and quality don't matter anymore to the developers; their only worry is how to crash land their project while still meeting whatever was agreed in the contract. Demos may be buggier, and it'll depend on the client to spend more time testing and finding bugs, and then junior developers will try to go about the project.

D. Fixed cost is limiting for all parties

Truth be told, companies asking for fixed-price are rarely strong technically, and instead are usually coming from the business side of things. This is one of the reasons why they are pushing for fixed-price, as a way to cap their risk for something they don't understand well. This means they also rarely foresee the biggest drawbacks of potentially huge delays, large time investments by themselves in quality control, and the absolute impossibility to change their product during development (the developers will certainly dig their feet into the ground before taking on any change orders). At the end of the day, you end up with a project that doesn't fit your needs, that is grossly delayed, and that took you way more time than you expected, but yes… the bill was cheap.

E. The safer road to success is controlled flexibility through a trusted relationship

In a time and material-based project, it is the burden of the client to make sure the developer has everything they need to succeed, as quickly as possible — pretty much like in any company employing its own developers: it is just one team working together.

With everyone sharing a common goal of pushing a project to completion, the collaboration is much closer and usually brings the best results. There is more pressure and time spent at the beginning to plan everything well, conceptualize the best product, help your outsourced team to do the best job they can – but the results are most often way more satisfying.

For a successful innovative project, make sure you:

- Avoid fixed-cost projects.
- Polish your requirements as much as possible so the provider may be able to evaluate your project as accurately as possible (keeping in mind he could still be 25-50% away from actual numbers).
- Ask for help from your provider or hire a professional agency if you have trouble with the requirements.
- Consider building non-working prototypes to get the validation of all potential users way before entering the beta phase.
- Be ready to ditch non-essential parts of your project to lower the scope of your first phase.
- Request regular demos every sprint, which typically occur every 1 to 2 weeks.

2 THINKING THAT THE REQUIREMENTS CAN BE BUILT ON THE FLY WITH NO CLEAR ROADMAP

Adhering to lean development principles is great, but there are limits to how flexible a team can be. Failing to prepare is preparing to fail.

As an integral part of the Agile software development methodology, lean software development is a concept that emphasizes optimizing efficiency and minimizing waste in the development of software. At Arcanys, everyone is trained in Agile practices, and it shows in our partnerships with clients: we are flexible, use a creative approach, and can provide incremental results every two weeks. **We work in a way where waste is avoided and removed, and we prefer to measure the value in terms of fitness for use rather than strict conformance to requirements**. We see short iterations as opportunities to speed up the learning process, communicate small sets of plans upfront, and allow us to adapt to unforeseen circumstances.

But while it's true that organizations with the ability to complete fast and simple improvements in the shortest time frame can gain powerful decision-making benefits and put value into the hands of their customer faster, rushing into a project headlong and building all requirements on the fly is not something we would also advocate. **Just because you don't need to think too far in advance about future requirements doesn't mean you don't need discipline, focus, and a solid roadmap to make things work**. You need to make sure that you have enough work piled up for your team for the next one to two sprints at the very least, with a clear plan in mind – which should be the role of your tech lead or CTO. In some cases, you can get a UI/UX designer to clarify the vision (usually in highly visual projects or at the start of a new project), or utilize the expertise of a business analyst or the product owner (either on the provider's side or on your side) to help the team establish clear requirements and the specific work to be accomplished on sprints.

3 BUILDING YOUR SOFTWARE PRODUCT ON A PROPRIETARY FRAMEWORK OR PLATFORM THAT IS DEVELOPED & OWNED BY THE PROVIDER

Unless there is a very, very good reason for your provider to choose a proprietary or custom framework, you'll always be better off running your business on a common framework.

We're used to rescuing failing projects. Unfortunately, a fair share of our clients have been struggling with their previous providers, and come to us with a hot potato that needs immediate attention. As we try to get control of the situation and carry on with development, we also assess what should've been done differently.

The hardest projects to rescue are the ones where the previous provider has started developing their custom framework, or platform to run the project. Rather than using well-established industry standards like Angular or Next.js, they have built their own soon-to-be outdated framework.

Developers rarely like to use the code of others and they sometimes don't agree with architecture decisions taken by some public frameworks. But in-house frameworks make it very difficult for other software providers to take over. Essentially, they have to learn a new way of programming that could take weeks to learn, and then remain permanently trapped in it throughout the future development of the project. **Custom frameworks are hardly flexible, very scarcely documented, hard to maintain and upgrade, and lack the necessary community around them for continuous improvements which results in them using outdated technologies**. In addition, the developers working on those custom frameworks tend to get bored because they barely learn anything at all that will be helpful for their careers. Usually, it isn't even worth putting on your resume. They can handle this for a few months but soon enough, they are likely to lose interest and performance may drop.

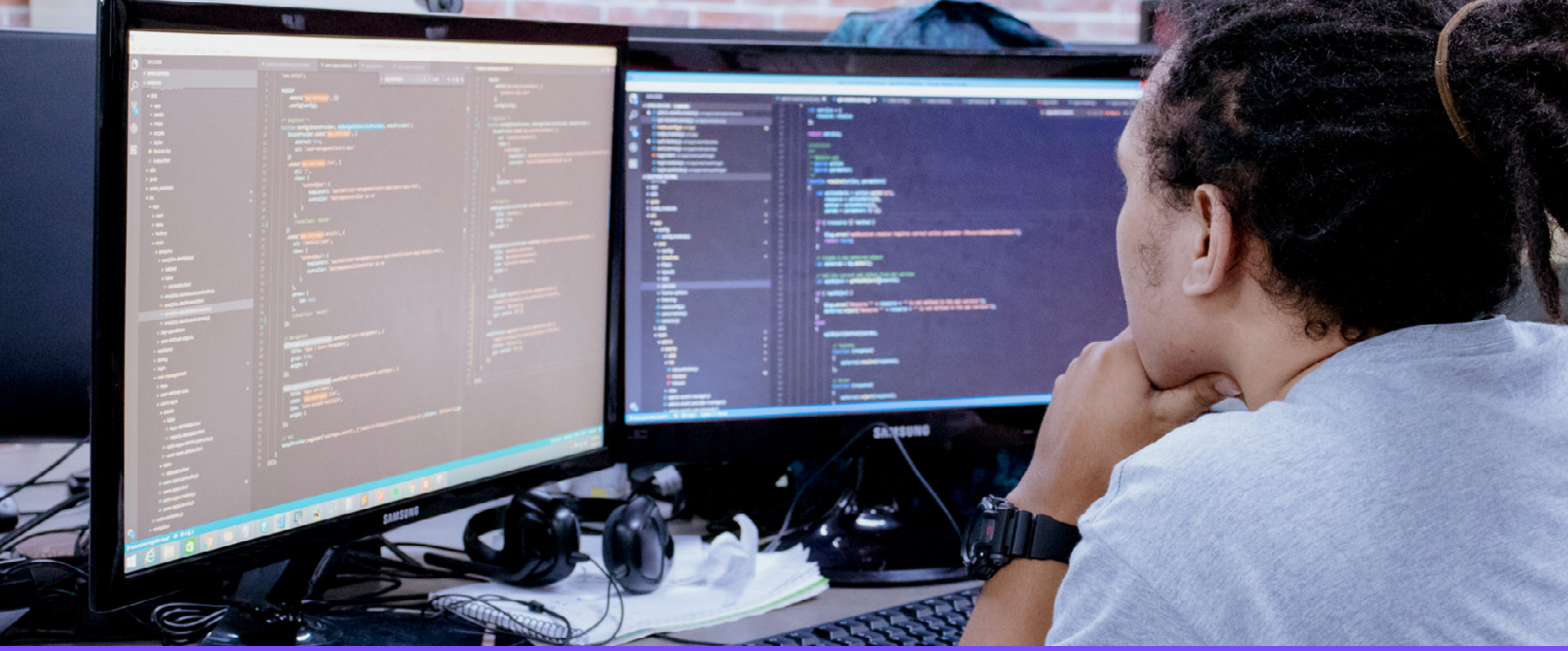
When the project is not too advanced and there is still a lot more to accomplish, we usually recommend starting all over again on an open-source, recent, maintainable, well-documented framework such as Angular, Next.js, or Vue.js. It is very easy to

find developers everywhere on the planet who know these widely-used technologies and can take over a project with minimal hassle. Whether you need to start from scratch or not is usually a funding-based decision: if you can't afford to start over, then try with what you have, until your project starts picking up. But just keep one thing in mind: **even if your project succeeds with that custom framework, one day, you will need to start over, no matter what**. It may be a few years down the road, but eventually, various reasons will get you to start over, and **the longer you wait, the more it will cost.**

When developing a new software product, make sure you check (i.e: put them in writing in the contract):

- The type of technologies involved
- The frameworks or platforms and their version number
- The programming language (niche programming languages usually don't have a large pool of developers you can hire)
- How widespread the community is
- If the technologies involved in your project are suitable for your project

Sometimes, having an independent consultant to check the assumptions of your provider can be a good option. Two pairs of eyes are always better than one.



4 NOT HAVING ACCESS TO YOUR SOURCE CODE ON A REGULAR BASIS

Some outsourcing companies keep the source code until full final payment is made. Here's why you want to avoid that.

Will you have access to the source code once completed? The answer to this question should always be yes when you're paying for some software to be done specifically for you. But another important question you want to ask is: When?

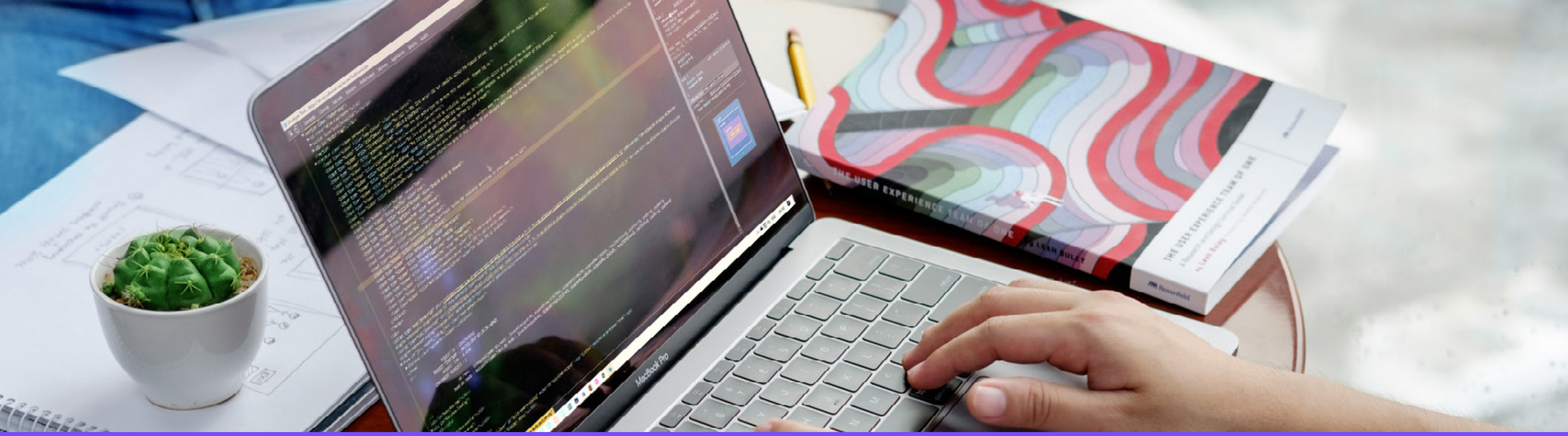
Some outsourcing companies deliver the code only when full and final payments have been made for the entire project. But problems in projects usually arise before the product is entirely finished, and sometimes the final payment is never made, or the cost just ends up much higher than initially expected (for various reasons, but most often due to poor requirements).

So, while incremental payments are released, **make sure that the source code is delivered throughout the project at regular intervals, preferably after each demo, or even on a weekly basis if someone on your side can review the code.**

Additional tips:

- Negotiate that your provider must deliver code on a regular basis, linking payments based on proper code deliveries.
- Verify that the code gets released on a regular basis.
- Create your own source code repository, for example, on Git, Bitbucket, or other providers. It's also acceptable to let your provider host the repository, as long as you always have access to it.
- Make sure you see the progress of developers on a regular basis.
- Pay on regular intervals when the provider is honoring his obligations

  (e.g., monthly payments just like you would for your in-house team). This is a two-way street and being a bad/late payer will also hurt the motivation of your provider.



5 LOOKING FOR AN OUTSOURCING VENDOR THAT WILL BLINDLY EXECUTE SOFTWARE DEVELOPMENT

Technical skills alone won't cut it: you want to partner with a tech talent provider who understands your challenges and objectives well enough to act as a genuine business partner and advisor.

Companies stepping into the process of team extension often fail to realize at the outset that outsourcing providers are not created equal. Software development firms actually vary in terms of function and intention: some act more as software vendors geared towards creating the actual software solutions; Other staffing providers are focused on meeting the required manpower and then leave it at that. For many companies, though, this is not the partnership they envisioned.

Instead, what most companies need is a healthy collaboration with a real software development team extension provider. What sets software development companies apart is their **deep understanding of the broad picture. Being in the industry themselves, they know the whole development life cycle, can step in when issues arise** or when they see you taking the wrong path, and are able to give expert advice. And that changes everything.

It's important to partner with a team augmentation provider that maintains the same high standards as you do to align expectations when running day-to-day operations, assessing skill levels, and evaluating solution quality. Also, don't hire a firm that only offers software development. You want to make sure you choose a full-service outsourcing company that has the capability to understand your business and technical needs and turn them into requirements with business analysts, or into intuitive and user-oriented designs with UI/UX experts. **Look for people who will not blindly execute what you've written down on paper, but rather help your software evolve into the product that you have envisioned.**

Other key services that you should consider in your provider search are code reviews, manual QA, automated QA, SysOps, UI/UX design, Scrum master, among others. Keep in mind that developers are creative people, but they are not the best people to design a new interface or to understand the full picture of the business logic behind your choices. For these, you would need higher levels of IT talent. If you don't have the expertise of senior software architects, business analysts, or UI designers in your core team, make sure that the company you choose is able to add these crucial skills to your extended team.

6 BELIEVING THAT DOUBLING THE TEAM SIZE WILL DOUBLE THE OUTPUT IN A LINEAR FASHION

Let's debunk this misconception once and for all: adding more programmers to a project falling behind schedule won't get it back on track.

While having more people on board may sound logical for other industries, it is not the solution in the software development space. In fact, it may even be counterproductive, delaying the project further. This observation was explained by Frederick Brooks in his 1975 book, The Mythical Man-Month. According to Brooks, there is an incremental person who, when added to a project, makes it take more time, not less. This is similar to the general law of diminishing returns in economics, which states that increasing one factor of production while holding all other factors constant will, at some point, result in lower returns.

In software development, at least three factors can explain this:

- There is a "ramp up" time for any new team member, meaning it takes some time for the people added to a project to become productive in any software project.
- Communication overheads increase as the number of people increases. When team members need to communicate with more people, they get less productive work done.
- Most software development tasks are not divisible. This is in contrast to other tasks like cleaning a room, for example, where adding more people to the job hastens its completion. The man-month model does not work here.

In software development, the ideal team size is about 4 to 5 people. Past this threshold, the work process would no longer be as efficient. The best solution in projects where more individuals are involved is to create multiple independent teams of 4 to 5 people. Thus, four 5-person teams, as opposed to a single team with 20 members, would have higher productivity with fewer communication bottlenecks.

7 TREATING THE OUTSOURCED RESOURCES AS A DIFFERENT TEAM SEPARATE FROM YOUR COMPANY

Nothing derails a software development project like keeping your outsourced team in the dark.

Depriving the outsourced team members of knowledge about your business creates a dangerous gap between you and them. One crucial ramification of this is that your extended team would not have a comprehensive view of the business and the project at hand, potentially resulting in suboptimal or inappropriate solutions. In addition, **the perceived lack of trust and feeling of alienation could also lower their morale, motivation, and productivity**.

That's why it is important to take care of the extended team members in the same way that you would your own people. Your efforts in this aspect will greatly factor in the success of your projects and company over the long term. Ensure that you are heavily involved in the day-to-day updates with a senior lead on your side, provide the work with the necessary information, and answer questions to guide the developers. This is what you would typically do with your core team, and I cannot overemphasize the advice that you should treat your extended team no differently than yours.

Lastly, visiting the outsourced team members every year or two (if possible) can have a very positive impact on the overall output. We have experienced this ourselves, and so have our clients.

8 CONFUSING PRICE WITH VALUE

Measuring the benefits of your outsourcing investment goes way beyond the price tag. Get to grips with assessing the full impact of your outsourcing contract.

It's common for company executives to automatically look at the bottom line when faced with a business proposition: How much will this cost? While keeping tabs on expenses is a given in any business, it becomes a problem when companies fixate on the cost and not on the overall value of a deal. It should come as no surprise that many organizations choose the less expensive option when given a choice. But because of the unfortunate results they often end up with, lower rates don't always mean you're getting the better end of the bargain.

This is where the approach of looking at value for money comes into play. Yes, a certain provider may not offer the lowest rates, but considering their significant portfolio and the level of expertise of their developers, perhaps the higher costs would translate to higher chances of successfully bringing your software to where it needs to go. And then perhaps, you'll find an even better bang for your buck when you get a dedicated team based in a nearshore country with a lower cost of living. Chances are, you'll find developers whose skills match those of their counterparts in developed countries, but at lower rates. Just take a look at the Arcanys track record and portfolio, and see for yourself that these are not empty claims.

So how do you measure value for money? **Consider your potential teams using the three E's as guide factors in your assessment: Economic—how competitive are the rates? Effective—how high are the chances for project completion? And Efficient—given the provider's capacity and range of services, will you get your investment's worth back and then some?** It's all about balancing the quality of work with the cost, rather than simply looking at the price at face value.

9 ACCEPTING BUREAUCRATIC PROCESSES AND NOT HAVING DIRECT ACCESS TO YOUR ENGINEERS

If you're not directly communicating with your developers and have to go through a project manager or middle person, and if the people around the table are mainly project managers, Scrum masters, or process managers, your outsourcing collaboration might be in trouble. This setup can lead to several issues, like miscommunication (because that middle person may not always be able to effectively convey your requirements and vision to the developers, which can lead to misunderstandings and misinterpretations that result in a product that doesn't meet your expectations), delays in feedback, increased costs (someone has to pay for these extra people's time), and the feeling of losing control.
After all, you want to make sure your project meets your expectations and doesn't become a headache.

So, to maximize your chances of success, insist on having direct access to the people who are actually doing the work on your development team.

Simple as that. By doing this, you'll be able to implement changes quickly, ensure that everyone is on the same page, and avoid unnecessary delays.

We cannot stress the importance of communication and collaboration in software development enough. As Conway's law suggests, the way your organization communicates internally will impact the software systems you create. That means that if your communication structure is fragmented, for example, your software system may also end up being fragmented and difficult to integrate. That's why it's crucial to optimize your communication structure for collaboration and to make sure everyone is working together effectively.

10 (FUNDING SPECIAL): THINKING THAT THERE IS SUCH A THING AS A FREE LUNCH - WORK FOR EQUITY DEALS

For companies that have yet to secure funding, using stock options to "pay" for initial software development may seem like an easy alternative. However, this approach comes with risks.

At Arcanys, we are very often asked to do development for equity, or to massively lower our rates because, well… They don't have a lot of funding to start with. That's a reality. Before they get to the first few-hundred-thousand-funding round, it's normal for these new companies to save as much cash as possible and make sure that the available financial resources are wisely spent. In this situation, however, you need to **be careful in choosing the software development partner you are going to work with, and how you will structure the deal**.

Firstly, it is important to remember that everyone needs to earn money to survive (and for companies, to pay their staff). Software service providers working "for free" will rarely prioritize these jobs when paying customers come knocking at their door. This could significantly slow down the development process or lower the quality of the product as the outsourcing partner entrusts the project to junior developers. This practice is risky for entrepreneurs because **it puts the project in a state of uncertainty, particularly in terms of time to market** and perceived quality for the early adopters of the product

Through its tech resources investment program, Arcanys Ventures, our company partners with promising companies to fuel their software delivery teams in exchange for equity stakes. However, just as funding companies must be cautious when choosing a software development partner, we also do our due diligence. When both sides have been working together for a while and delivering on their respective promises, it may be time to consider work for equity deals if the opportunity makes sense for both parties. Even then, work for equity

deals are always partial; funds still need to come in to cover some of the costs, as this is the only way to have sustainable "co-owned" projects.

Ideally, before an investment occurs, both parties have to work together through a regular client-provider relationship. This would help the partners determine if there is an alignment in values and if the teams can collaborate efficiently. We call this "dating before getting married". This is what we did with Register Now (which ended up in a successful exit in 2021), ThinkItTwice, Payment Logic, or Medulla, among others.

Bottomline: Any deal that seems too easy for any of the parties is usually not good, as there may be surprises down the road. Having a software development outsourcing firm onboard is key to faster and cheaper product building, but this requires serious consideration. What the outsourcing partner provides goes way beyond cash investment; it's an incredible added value for the founding teams to work with technology experts with proven track records and experience. With the right outsourcing partner, startups and scaleups can accelerate their journey to success.

Learn from these mistakes

Nobody ever said that managing and onboarding an extended software development team was easy. But with all relevant factors considered and the most common mistakes learned before you engage in a partnership, then you're in the best position to succeed. Find the right outsourcing partner for your unique needs, manage your expectations, maintain great communication, and go places (even literally) with your extended team.

Don't let your business ambitions die before you give them a chance to become a winner on the market. Use a professional software development team extension firm to hire the best talent for your needs and take your software product to greater heights. If you are interested in knowing more about how to make your outsourcing initiative a success story or simply get a free strategic analysis of your project, contact me at fred@arcanys.com or on LinkedIn.

About the author

Fred had been working on IT and operational projects in the finance and software industry in Switzerland for 10 years before co-founding Arcanys in 2010. He carefully negotiated with and managed outsourcing suppliers as a corporate buyer on the one hand, and led delivery teams as an internal provider to operational departments at an executive level of listed companies in Switzerland, on the other. With over 20 years of experience in the industry in Switzerland, Hong Kong, and the Philippines, Fred is now leading the worldwide sales and marketing efforts of Arcanys.

View on Linkedin

About Arcanys

Arcanys is a Swiss-led software development & IT staff augmentation firm based in the Philippines. We work with ambitious companies that are ready to enhance their technology output and prepare for accelerated business growth. As a long-term strategic partner, we reinforce their software development operations by supplementing their existing teams with the top 5% of IT talent in the Philippines. With a successful placement rate of 95%, we fill positions 10 times faster, and we offer an average cost savings of 56% compared to local hiring. At our core, we prioritize talent retention, boasting an attrition rate that never exceeds 10%.

Arcanys