Resolving the question of Building or Buying your enterprise software

TECHNICAL PAPER

Buying or developing your company's business software is a genuine question. To answer it from a long-term perspective, taking into account profitability and sustainability, one should ask himself many other questions which we will try to answer in this article.

Offer Sadey
Chief Technical Officer
Tinubu

Focusing on your primary business

When deciding if you should develop your business software in house, or rather purchase it from a specialized editor, you should answer this first question for yourself: Are you willing to spend time and money on building software when you are not a specialist in this field, or would you rather focus on your core business, namely Credit Insurance or Surety?

True, you can probably think of the many advantages you would get from building your own software, especially the capacity to tailor your software to your own company and its very needs.

However, to achieve and maintain those advantages in the long run, more questions should be investigated before making this crucial decision and be sure you made the right one.

How rich do you want your content to be?

You probably want — as well as the rest of the company management and staff — your Credit Insurance or Surety software to gain new wanted — and needed — features at each new version release: increased security and productivity, integration of the latest business best practices...

If it seems reasonable to first invest in customizing your software so it aligns with your business processes and workflow, it is also necessary to renew that effort each year thereafter in order to keep your software evolving, release after release, by bringing new functionalities that will result in productivity gains, an acceleration of decision-making capacity, and a better user experience.

Adding those new functionalities into your software is really an anticipation job. For instance, if you would wait for users' demands to arise to start integrating a Business Intelligence solution, an electronic signature system, or a prediction using a neural network, you would be way too late and would lag behind competition for about 1.5 year — and by the time you finally catch up, there would be new evolutions you would also have to take into account.

The right approach is then to constantly watch market trends and technological changes or innovations.

Indeed, being able to integrate today an existing state-of-the-art electronic signature system into your software means you started working on it 1.5 year or even 2 years ago updating in real-time each aspect of your code to current global technical evolutions it relies on — making sure your software doesn't outdate — so by the time the development of e-signature is complete and hit the market you are ready to implement it right away.

The technical debt

That brings us to your potential technical debt. The technical debt of a software must be reduced continuously. If your software is not regularly 'refactored', this technical debt will accumulate over the years until it becomes too much of a burden and will prevent the software to run properly. Indeed, the best practices used to build your software are evolving quickly.

The software also embeds thousands of open-source dependencies — called "libraries" — that continuously advance technically to provide new features and improve security.

These upgrades of the open-source libraries sometimes impact compatibility with your software or introduce the use of new technology. If the gap between the earlier library version — used with the software — and the most recent version becomes too important, there is a strong risk that the earlier version becomes obsolete in favor of the latter one.

The refactoring effort that is not produced today rapidly becomes difficult to catch up with tomorrow. Absorbing such a backlog consumes a lot of time and resources and can very well lead to the need to rewrite the software entirely, or abandon it.

Another type of effort must also be taken into consideration: The current technology standard is based on the use of Microservices/Containers of which the technology is also constantly progressing. Designers must keep themselves tightly up to date or they risk seriously lose ground.

This endeavor is naturally planned by a software publisher, but what about a company whose main business is not software development?

Security first

Every change, no matter how small, can potentially introduce security vulnerabilities in your code. Detecting and fixing them must be part of the software lifecycle or these vulnerabilities will accumulate and create a technical security liability, making it easier for hackers to exploit them. Detecting and correcting flaws such as CSRF and XSS requires a significant effort.

Failure to do so then leads to multiple risks: introduction and takeover of company systems, theft or destruction of data, or ransomware.

The infrastructure that hosts the solution is also a key element. It should for instance be able to withstand a massive DDoS attack without having to interrupt the service.

Cloud hosting providers are investing heavily in security and attack detection. A software publisher has a privileged relationship with a cloud hosting provider and generally benefits from his expert support, which has the positive effect of enriching the software company's internal competencies which can then be used to the benefit of their customers.

Technical support

The availability of your software is critical for your company, especially if it is accessed by thousands of users on a global scale. Very often, companies that build their own software are tempted to outsource their 24/7 support to an external provider located abroad who does not always know well enough the customer's business, or even the application itself.

Should an incident occur, technical support must be able to comply with SLAs and ensure a contractually defined availability of the system, such as being able to respond within 15 minutes to any major incident. However, when an incident requires a code correction, support can only offer workarounds, without being able to act on the code.

When buying

- ⊕ You can focus on your core business
- ⊕ Continuous innovation and best-of-breed design are delivered to you
- ⊕ You benefit from 24/7 end-to-end support involving experts in various domains (IT, business, security, cloud, etc.)
- ⊕ You obtain reliable and industrialized delivery allowing application scalability

When building

- ⊖ You need to bear high maintenance costs
- ⊖ You have to permanently manage the security vulnerability
- ⊖ You have to produce massive efforts in managing the technical debt to prevent your software from becoming outdated
- ⊖ You need to anticipate changes or seriously catch up in implementing innovation

Premium support, therefore, requires significant human and financial investments to be able to manage any incident from start to finish, including correcting the code and deploying the software again within minutes.

Yet, the optimal support level should go one step further and advise the customer on the most fitted configuration or use of their software based on the analysis of the resolved bug and the customer's business processes or operational changes.

Last but not least: An easy deployment

Finally, you most certainly want your software to be easy to deploy, while also rationalizing your cloud consumption.

Progress made in the deployment of digital solutions has been tremendous in recent years. Engaging in this technological race to be compliant with changing standards and requirements of cloud architectures is a must, should you choose to build your own software.

Only a few years ago, the setting up of new environments and the deployment of applications required a manual configuration of the infrastructure. On top of that, there was a likewise manual configuration of the deployment system. Not only were these operations time consuming, but they also increased the risk of errors and discrepancies between the different newly created environments.

For instance, the operator had to ensure that the UAT and pre-production environments were allocated an equivalent storage capacity and computing power.

The execution context of the application within the infrastructure also had to be configured manually, a task that again embarked its share of errors and time consumption — and the resulting negative impacts on the reliability, security, and availability of the deployed application.

For all these reasons, the technical modus operandi has been changing over the years leading to today's trend: IaC, Infrastructure as Code. IaC means relying entirely on code for each phase of the deployment, from configuring the infrastructure to the execution context, and automatically deploying your application with its tailored security rules and permissions settings.

You do that using "scripts", such as Terraform and Kustomize. While it would take a week or more to manually configure a new environment and deploy an application, it now takes less than an hour using these scripts. This new technological trend has resulted in a significant increase in coding skills requirements among DevOps teams. Although a company that builds its own software would naturally tend to continue to configure everything manually, a cloud solution provider will have made the necessary technological and competency investment to streamline and automate its infrastructure and deployment processes.

Overall, using a cloud-based solution presents many advantages; however, its potential cost is a factor you should pay attention to. Leaving your server running day and night is not anymore an option today when you use cloud services. Indeed, 24/7 use of the memory, disk space, and processor of your cloud services provider will for sure get your bill to ramp up. But this extra cost can be avoided by shutting down your development and test environments when they are not in use. That is why a software publisher will routinely seek to optimize its infrastructure and environments, and it is now possible to automate their shutdown or even remove them if they are no longer needed. ■

About the Author

Offer SADEY
Chief Technical Officer of Tinubu

Artificial Intelligence enthusiast, passionate about technologies, Offer leads the technological vision, the innovation and the cloud transformation of the company. Twenty years of experience in managing IT teams, he has a record of organizing and scaling large teams (120+ staff). His technical expertise and experience as software company founder allow him to leverage cutting-edge technologies to build SaaS platforms in the most efficient way.

About Tinubu

Tinubu is the business facilitator and exchange enabler that delivers fluidity and simplicity to the insurance industry by using the strength of collective performance.

Our company is an alliance of technology software and insurance expertise offering the best combination to its clients. It covers the entire value chain of credit insurance & surety with one end-to-end platform, connecting every part of your business with one digital highway.

Established in 2000 and headquartered in Paris, France, Tinubu is an independent software provider and employs 170 people, located in Paris, London, New York, Orlando, Singapore, and Montreal. Its clients represent 30 of the top 60 Credit & Surety underwriters worldwide.

Follow us

Contact us contact@tinubu.com

Call us EMEA +33 (0)1 74 31 20 70
Americas +1 (213) 290-9628

tinubu.com