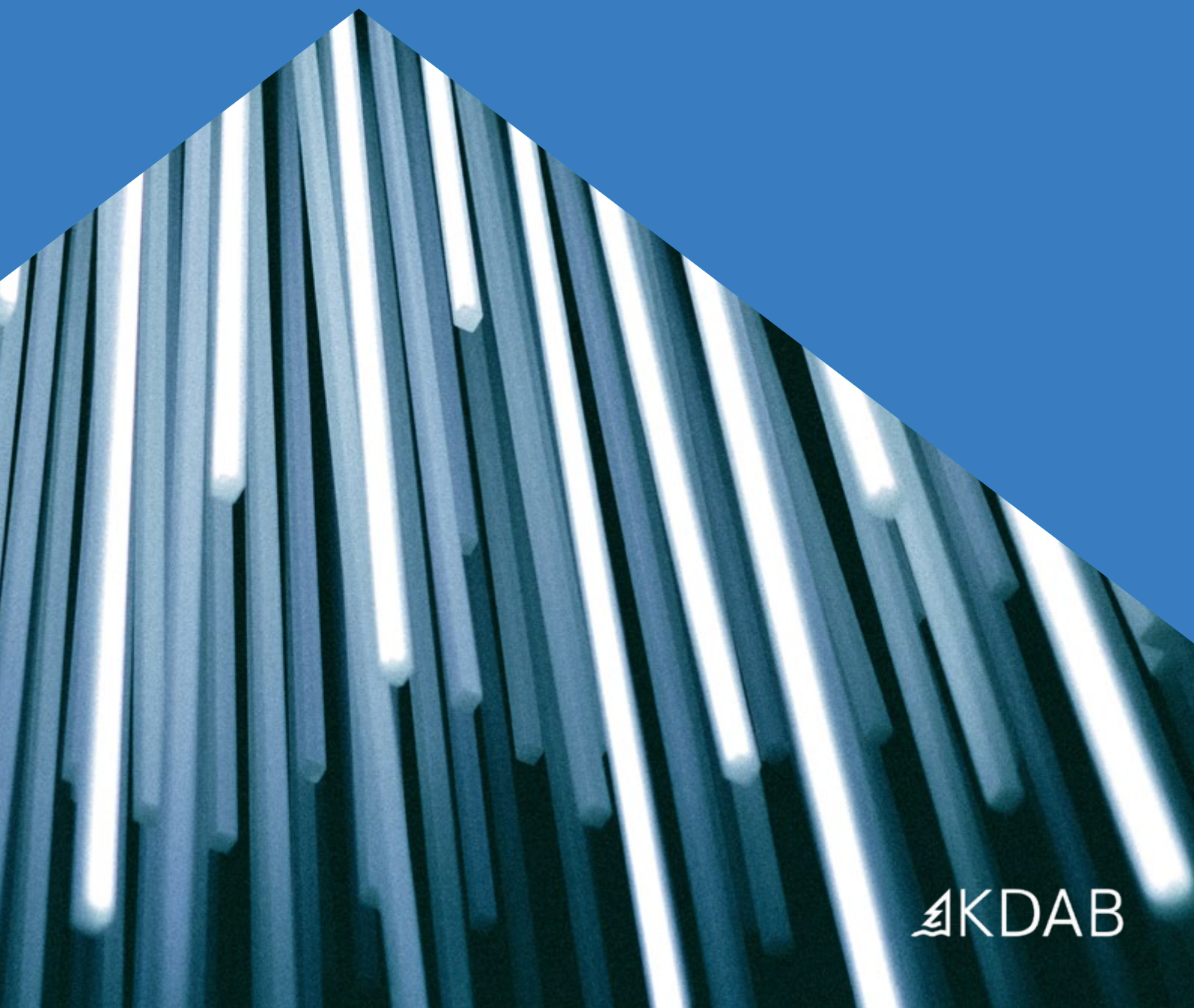
Dodging disruption with software

How to prevent commoditization

Matthias Kalle Dalheimer | President and CEO, KDAB



KDAB

No matter what you sell, software competency is your most important asset.

When a software-savvy competitor enters your market, they provide new benefits that win over your customers.

Maintaining leadership

If your company produces physical products, you might think that your most important asset is the in-depth experience acquired in your field, your loyal customer base, or your top-notch manufacturing facilities. Those may be the key pieces that helped you become an established leader, but they're not what will keep you there. No matter what you sell – agricultural conveyers, passenger cars, commercial airplanes, coffee machines, biomedical sorters, chemical tanks, mining equipment, diesel locomotives, or vacuum cleaners – software competency is your most important asset.

This seems extremely counterintuitive, especially for companies with products that don't even contain software!

However, over and over again, industry giants in many fields have been toppled by Silicon Valley start-ups and game-changing innovations. When a software-savvy competitor enters your market, they provide new benefits that win over your customers. That leaves you in a game of catch-up to regain market share and re-establish your brand.

How can you avoid your company being upstaged by an upstart?

While one single blueprint doesn't work for every company; there are certain things that, if in place, will allow you to build your software skills, letting you compete, innovate, and beat your competitors.

Attitude

There are two critical ideas that a company must absorb in order to transform itself into a software capable organization.

Your innovation speed relies entirely on your software capability.

Software is the bedrock of any corporate rejuvenation. Product features can be overshadowed by other innovations, manufacturing gains can be easily replicated in low cost countries, and your

While software may be key to enabling innovation, your company's industry experience still gives you an edge.

The pillars of software strength and industry experience need to be joined at the hip.

decades-long industry knowledge can be bypassed under the right circumstances. Software creation, deployment, and connections are the only thing fast and expansive enough to consistently beat other types of innovation.

Your experience is critical to staying in the lead.

While software may be key to enabling innovation, your company's industry experience still gives you an edge. New software companies trying to enter your industry won't know the customer's pain points and they'll need to re-learn through trial and error the mistakes you've already made.

You cannot rely on a single one of these pillars – you must use both. Without software expertise, your experience can be circumvented by competitor products that may in fact be less well-engineered but are customer pleasing. You can't fall prey to the hubris that "nobody else knows our industry as well as we do". Someone can learn enough to win your customers – it's happened many times before. Blackberry and Apple, Blockbuster and Netflix, Encyclopedia Britannica and Wikipedia are all great examples.

This doesn't mean that your industry experience isn't a factor – it is. It gives you insights into your customers and their problems, knowledge that can be very difficult to accumulate. It can also keep you innovating ahead of everyone else, provided you have a means to capitalize on it.

Marrying software and experience

The pillars of software strength and industry experience need to be joined at the hip. That's far easier said than done, and many companies make costly mistakes in building their software competency, such as acquiring a start-up or building an independent R&D center.

We've seen many big-name car companies trying these tactics to catch up with Tesla but with little success. That's because a startup can't change the momentum of a massive company no matter how

You need to use your knowledge to out-perform the disruptors at their own game.

skilled or motivated they are. Output or insights from those spin-off endeavours are often ignored. Even when the spin-off's innovations or developments aren't ignored, their results are rarely integrated into the mainline organization and don't change the company's approach.

Planning may be one of the most critical capabilities your new software literacy will require.

These culture incompatibilities also occur when an internal startup look-alike division is spawned as a center of innovation. These efforts only serve to give leadership a false sense of security that the company is on their way to becoming software savvy.

A better approach is to build new teams that combine new software people and existing employees who are eager to change and learn. Integrate these teams into the main organization and ensure that people with software fluency are in decision-making roles and aren't being blocked by "we've always done it this way" thinking in HR, legal, purchasing, or other critical departments.

You must know who you are, take your assets, and digitize them. You may be the best in the world at what you do – you just need to use your knowledge to out-perform the disruptors at their own game.

Planning

Software engineers often spend a third or more of their time designing and planning before they write any production code. In fact, planning may be one of the most critical capabilities your new software literacy will require. That's because well-built software has a great many aspects that need proper planning. Here are three examples of KDAB customers that developed software planning into an asset.

Managing customer expectations

Determining solid software requirements is a necessary first step to any software, and it requires discipline around managing customer expectations.

You want your company to be agile and responsive, but not at the mercy of every customer's whim.

Funnel customer inputs into a product roadmap that allocates limited engineering resources appropriately.

We work with a biotech company that builds high-volume magnetic cell sorters, used for many research and applied science purposes. Common to many companies without a pre-established and structured software practice, they tried to be responsive to their customers' every request and they frequently added new features to their software whenever customers asked for them. Unfortunately, although this approach initially appears as incredibly responsive, shipping every customer a custom version of software is completely unmaintainable in the long run. It makes it impossible to fix bugs or make other changes without introducing new problems. You want your company to be agile and responsive, but not at the mercy of every customer's whim.

We helped our customer understand how to appropriately manage their customer requests and the critical role product management plays as a feature gatekeeper. By installing discipline around customer feature requests, they were able to funnel inputs into a product roadmap that allocated limited engineering resources appropriately and across all customers, not just one at a time.

Making hardware part of the solution

The silicon and boards you select for your project can be a major planning exercise. That's because it dictates the software solutions you can use, the capabilities your products can implement, and your product growth strategy. Your hardware supplier should be part of the solution, not part of the problem.

For our customer who builds industrial touch screen assemblies, software was never really part of their strategy. While they provided sample code and device drivers to their customers, they never built their customer-facing applications. As a result, they didn't realize how much work their customers needed to do in order to turn their hardware into working products, and how a lack of supporting software was steering customers to inexpensive overseas imitations.

KDAB helped this supplier completely revamp their strategy by offering not just the touchscreen assemblies but screens with boards

Nothing impacts the end consumer more than the application design and user interface.

If your application design is awkward, the user will accumulate negative impressions each time they use it.

running a pre-configured Linux OS and modular software that could readily support the majority of use cases. This architecture now lets their customers move readily between products in their lineup with minimal software changes.

The company also decided to provide turnkey solutions for larger segments of their market, such as vending machines. This allowed products to be configured without coding, making it possible for customers to design and roll out products extremely efficiently.

By understanding and embracing their customer's software dependencies when it came to hardware planning, this electronics-parts business was able to create a fundamentally market-differentiating feature for their touch screens.

Rethinking application design

Nothing impacts the end consumer more than the application design and user interface: it's a massive part of the user experience. If your user interface is non-intuitive or your application design is awkward, the user will accumulate negative impressions each time they interact with your product.

A company that made extremely high-end audio systems had this problem. While their audio equipment could vibrate your very soul with every subtle nuance of an orchestral performance, their media player looked and acted like it was designed in the 1980s. That's not the impression you want to give your customer who might be dropping more than a year's salary on speakers and subwoofers.

KDAB helped this audio engineering company rethink their approach to their media player, building an intuitive smart-phone-like UI that could control and configure the equipment. Because people still want to play music from non-audiophile sources, we also worked with our customer to integrate mp3 playback, Spotify, digital TV, and YouTube streaming capabilities. By helping this company develop their software skillset, they were able to expand their product's capabilities to integrate other online music services without our help, ensuring that their media player was as world-class as their hi-fi.

The skill of properly managing and building software is an essential capability for a software organization.

Building

The job of the software build manager seems so prosaic. While normal people barely even understand that it's a necessary role, even developers who should know better sometimes don't give build engineers the credit they deserve. Yet the skill of properly managing and building software is an essential capability for a software organization as soon as you've got multiple engineers, multiple locations, and multiple product lines – which is basically any company using software.

The zip, copy, and paste style of version management should have been dead decades ago.

Managing versions

The zip, copy, and paste style of version management should have been dead decades ago, yet still lives on in organizations without any computer science training or software best practices. That's exactly the state Speidel found itself in when starting the development of their Braumeister micro-brew equipment.

Speidel is a generations old company experienced in building metal tanks, but not in building software. They needed our help in establishing the foundation of their new software discipline, including such things as proper version management and release processes.

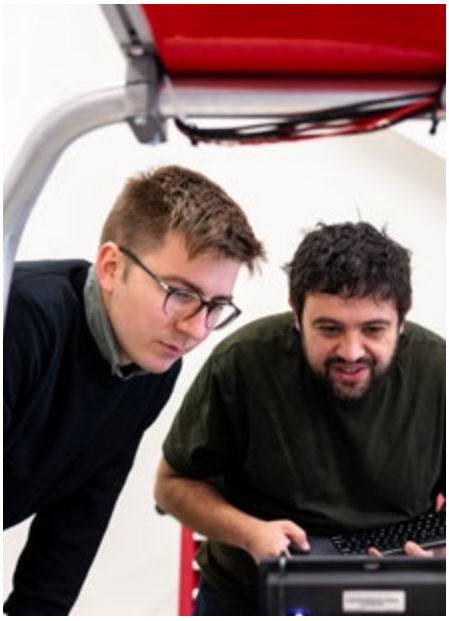
We showed them how to use version control, tags, forks, branches, and merging to get a handle on their software contributions from their team and outside help. And we showed them how to cleanly manage their software releases, enabling them to keep their developers, testers, and support staff in sync, track and fix bugs related to each release, and safely update their customers' products in the field.

The result of this is that Speidel has been able to reinvent themselves from a tank producer to a microbrewery innovator, spawning a whole new line of business, an active online store, and a dynamic user community that loves their products.

Using a shared code base

Most companies don't build one product, they build many. If they're smart, they use a single shared code base for each product and manage customizations carefully.

Creating a proper test infrastructure is a critical building block to making a rock-solid product.



Hosting tests on a developer's machine does not guarantee that the tests will be run properly.

We work with an airplane interior manufacturer that buys empty shells from Boeing, Airbus, or Bombardier, and outfits them for commercial airlines, government planes, and corporate jets. Previously they outsourced their software needs for plane IT systems, climate control, and entertainment systems to outside suppliers. Because they had so many variants of product and most of their buyers expecting very custom and specific features, software management was a nightmare for them and nearly every plane had different software. They realized that these two factors – not owning their software, and not controlling their versions – gave away their ability to differentiate, was very expensive, and was extremely error prone. They decided to spin up a software capability with KDAB's help.

We helped them create a customizable application that could alleviate much – but not all – of the changes that were tailored to each customer. Then to help them manage individual product lines and other special cases, we showed them how to construct a build system, software architecture, and version control discipline that could easily support multiple product lines from the same code base.

Today, the company is able to fulfill custom orders much more simply, speeding the time to delivery and greatly reducing their expenses. They are controlling their own software destiny and actively growing their capability to create more exciting product variations for their customers.

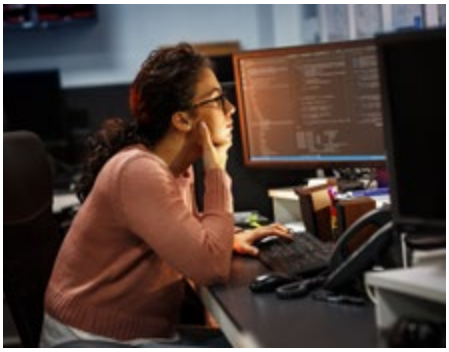
Testing

Untested software is buggy software. Automated testing is something that can help fulfill a software product's testing requirements. But what is appropriate to automate, how do you automate tests, and how do you use the results?

Testing infrastructure

Hosting tests on a developer's machine does not guarantee the tests will always be run or that they'll be run properly. As we helped our airline customer understand, creating a proper test infrastructure

When continuous integration works well, it helps significantly reduce problems in distributed development.



Software folks don't always respond like everyone else. Maybe that's because they natively speak with machines all day.

allows tests to be run consistently for every build and is a critical building block to making a rock-solid product. That includes making a server room with independent test servers, and the ability to integrate testing into their build system. Not only were they able to improve their testing reliability, but the consistent reporting that was generated helped the organization build confidence in their new software engineering capabilities.

Understanding the need for continuous integration

The process of developers frequently integrating their changes back into the mainline code is called continuous integration (CI). When it works well, it helps significantly reduce problems in distributed development. However, CI mandates a build server, automated builds, and automated unit tests that aren't easy to get set up initially.

One of our customers that builds electronic test equipment needed help in building software and testing it, since their specialty was in electronics – not software. The concept of testing software after making a trivial change seemed strange to them, and as a result sometimes a new software update turned customers' highly expensive test tools into multi-colored plastic bricks.

KDAB worked with this test-equipment manufacturer to develop a software practice that included a CI system to help them build and test continuously, ensuring that if software fixes broke anything it would be detected long before it reached the customer's hands. And most importantly, we showed them how to make CI work for the organization, so it's not just dropped after a few months like many fledgling CI processes are.

Culture

Software folks don't always respond like everyone else. Maybe that's because they natively speak with machines all day. But don't expect that a fancy office with pinball machines, espresso machines and daily gourmet donuts is the way to a contributing software organization.

If you have no in-house software competency, it can be difficult to hire those first few developers.

Embracing software must embrace cultural changes too, not the trappings and trinkets.

Hiring good developers and keeping them

If you have no in-house software competency, it can be difficult to hire those first few developers. What do you look for, what skills matter, and how do you know people's skill level? Does your company have the right things to attract software people?

Our electronics testing-equipment client had a problem building up their software capability. Over five years, we helped them create a competent software staff by helping them find, interview, and hire their team. While the team was being fleshed out, we were their "software department as a service", slowly transitioning our staff out and training new hires to take our place. To develop their hiring process, we helped them with software-appropriate hiring criteria and how to interview new candidates.

We also helped their HR department understand the types of perks that would resonate with software engineers, like remote working facilities instead of cafeteria food quality.

Another of our clients had a two-week enforced plant shutdown that made sense for the factory supply chain but didn't for the globally located and remote engineering team. We recommended that they implement a flexible time-off system instead so the engineers could work when they were productive instead of forcing them to take time off. By better understanding the needs of software developers, establishing some guidelines and putting some best practices in place, the company was able to continue to grow their software team on their own.

A huge challenge for companies that are moving from physical goods to software-enabled goods is understanding the value of software.

Understanding the value of software

A huge challenge for companies that are moving from physical goods to software-enabled goods is understanding the value of software.

“Digital transformation” may be an industry buzzword, but it is meaningless to your company without a strategy.

You must be brutally honest with regard to your own software competency and how you stack up against your competition.

Companies with hard product knowledge often try to associate the cost of software with the cost of physical parts in pricing and legal discussions. A price-per-part model isn’t at all appropriate since software can be replicated infinitely at no cost but takes an incredible amount of human capital to produce the first instance.

Several of our big manufacturing clients get bogged down in this, and their purchasing and licensing departments fought to contractually negotiate a software price per part although the cost of producing the software is the same if one or one million products are sold.

Legal departments can be tricky about understanding software too, since they tend to want to assign liability to everyone else and indemnify themselves, even when they’re the ultimate software supplier. We help these customers understand the financial and legal implications of transitioning a company into a software-capable one, including new ways to purchase, negotiate, and license.

Get started

“Digital transformation” may be an industry buzzword, but it is meaningless to your company without a strategy. Creating a mobile app or adding a touchscreen to your product does not magically transform your company into an innovation leader – you have to reboot your culture.

You must be brutally honest with regard to your own software competency and how you stack up against your competition. You need to develop a software strategy that is fully integrated into both your product development process and your product roadmap. You must develop the competency to execute on your software strategy in a consistently repeatable way, and you need to continually maintain and evolve your code base and your team’s skills.

About the KDAB Group

The KDAB Group is the world's leading software consultancy for architecture, development and design of Qt, C++ and OpenGL applications across desktop, embedded and mobile platforms. KDAB is the biggest independent contributor to Qt and is the world's first ISO 9001 certified Qt consulting and development company. Our experts build runtimes, mix native and web technologies, solve hardware stack performance issues and porting problems for hundreds of customers, many among the Fortune 500. KDAB's tools and extensive experience in creating, debugging, profiling and porting complex applications help developers worldwide to deliver successful projects. KDAB's trainers, all full-time developers, provide market leading, hands-on, training for Qt, OpenGL and modern C++ in multiple languages.

www.kdab.com