The Essential Guide to Custom Software

400 UNIVERSITY AVE, MEZZANINE,TORONTO ONTARIO M5G 1S5

1-866-640-2345 | KONVERGE.COM

Konverge

Table of Contents

Introduction

While many organizations can and do use off-the-shelf software packages, there are occasions when the required functionality just doesn't exist. In these cases, if the need is important, you may decide to have the software built to suit your specific purposes.

In this book, 'The Essential Custom Software Development Guide', we will tell you everything you need to know about developing custom software...

The Case for Custom Software

In the beginning it was all custom. The original computers were bespoke behemoths – every piece of hardware and software were tailored for that particular computer. Then people started to see the value in standardization – if we use the same hardware and operating systems we can mass produce these systems. This gave rise to the mainframe and the mini-computer. Companies like IBM and DEC thrived selling these computers. IBM's AS/400 was so successful that you would be amazed at how many of these systems are still in use around the world.

The 80s saw the rise of the PC, the relational database, the laptop. With all these components in place, a group of German IBMers decided to create a new software category intended to seamlessly run an entire company: Enterprise Resource Plannning, or ERP. That company was SAP, and their original vision was incomplete from a technological standpoint, but the business case was compelling, and it started a groundswell move towards packaged software that has continued to this day.

Packaged software is very good now, and the investment being made by the large players is staggering. Oracle, SAP and Microsoft are all spending over a billion dollars a year on R&D. It is a global war for the hearts and minds of the business software decision maker. The quality of the software and the completeness of vision of the solutions is far superior to what it was even 5 years ago. The rise of the "Cloud" has also dramatically decreased the complexity of implementing packaged software solutions.

Outside of the "Big Three" there are all kinds of players in packaged software, from CRM (Salesforce), to Marketing Automation (Infusionsoft, Hubspot, Act-On, etc.) to Project Management (Jira, Base Camp, Igloo, etc.). A while back I discovered an ERP that specialized solely in companies that manufacture corrugated boxes. There is enterprise software of every color and stripe out in the marketplace.

So given that we could be said to be living in a "Golden Age" of packaged software, is there still a role for custom software? The answer to that question is a resounding "YES". Here are just a few of the many reasons to consider custom software:

1. Competitive Advantage

By definition, packaged software is not going to provide a competitive advantage. Through your particular implementation you may feel that you have a superior solution to your competitors, but ultimately they will have access to the same packaged software and same consultants that you do.

Your "secret sauce", your competitive advantage, can reside in many different places in your organization. It could be your R&D, your manufacturing processes, your logistics, your customer service. It can be many different places. Wherever that competitive advantage resides, you should apply software to leverage this advantage as much as possible. Custom software will best allow you to capture and sustain this advantage, and will also give you the ability to control the intellectual property associated with your competitive advantage.

2. Unique Requirements

Packaged software does a lot more than it used to, but it still doesn't cover the full range of software applications in business. From time to time, you may find you have a business process that cannot be properly addressed through business software. You might have a pricing model that doesn't fit, you may have a logistics system that can't be effectively modeled, your workflow for certain processes may be outside of the available choices within your packaged software.

When you run into these situations you have to ask yourself, do we adapt to the software, or should the software adapt to us? It's a good point at which to assess your processes and see if a new approach might be superior, but in many cases you'll find the appropriate response is to take the custom approach.

3. New Technology

This is a special type of competitive advantage available to so called "first movers"- a distinct and special opportunity to garner the benefits of a new technology when it first enters the marketplace. The time between a technology moving from being revolutionary to commonplace gets shorter and shorter. Today there is probably a window of a couple of years. The packaged software providers are large, monolithic institutions that are going to lag in the adoption of new technology. If you wait for the packaged software to add the new technology, you will miss the window. Your only option is to go custom.

Mobile and the cloud are now seen as commonplace, everyday technologies, but 5 or 6 years ago, the strategic deployment of these technologies could give you a big leg up against your competition. Today technologies like Big Data, and the Internet of Things offer big benefits, but you will have to move fast, and you will have to follow the custom route.

4. Integration

Vendors are going to try to convince you of the seamlessness of their enterprise solutions. “Everything is easily integrated with everything else”, “our family of solutions talk to each other out of the box”, “the data flows from one solution to the other without virtually any intervention”. Don’t believe it – often the vision presented fails to mention that the software is a bunch of packages that were acquired through purchase and have been cobbled together in the most rudimentary of fashions. Different architectures, different data models, and different processes. Data is still often siloed- even packages from the same vendor do not always talk to each other, and integration tools provided by the vendor are often very basic.

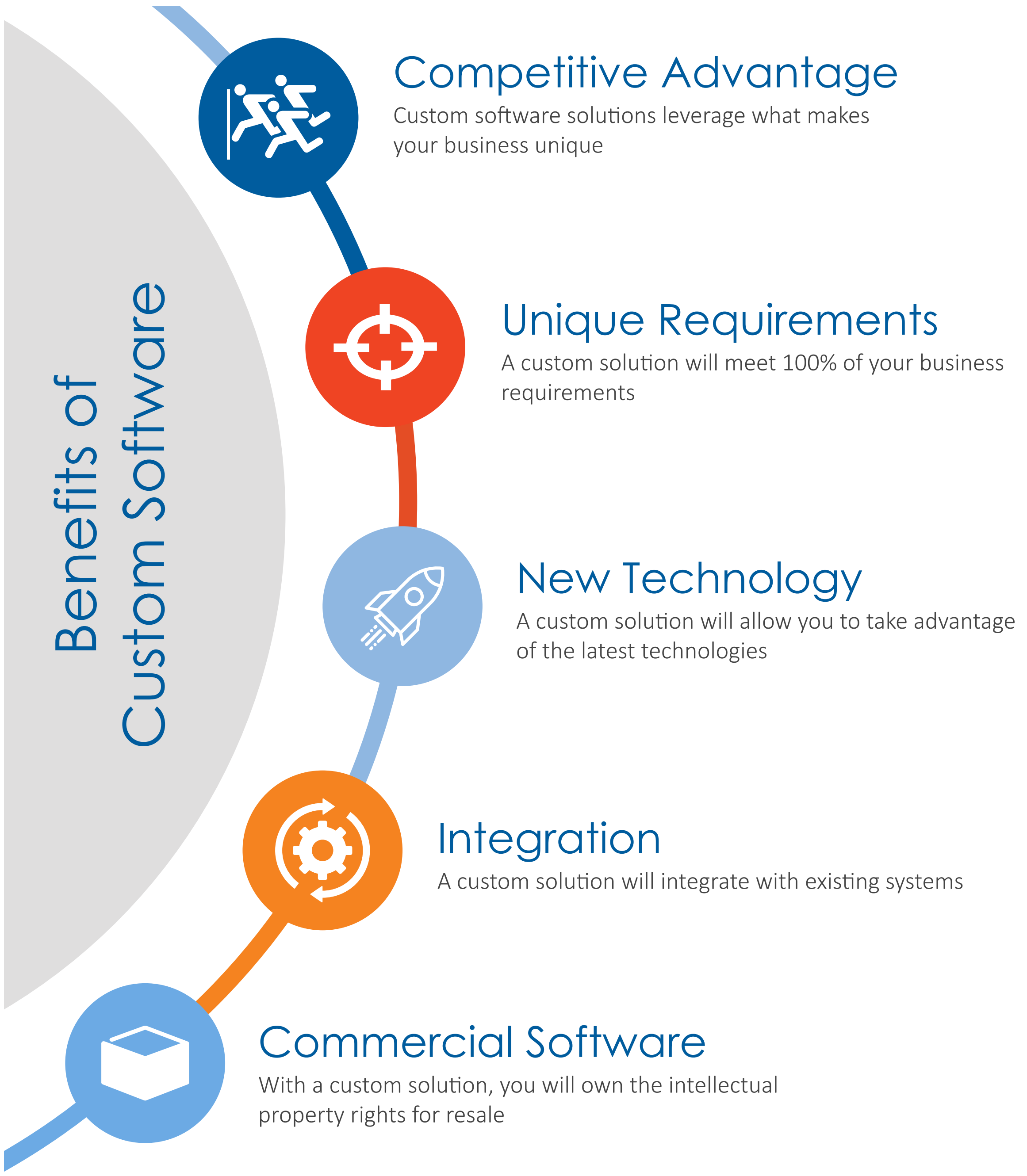
There is no way around it – if you really want to avoid data silos and make sure that everyone in your organization has ready access to the information they need, you almost always have to build custom integrations between your applications. This is an area where the custom approach really shines – ironically, the “data at your fingertips” promise of enterprise software can’t really be delivered without a strong assist from custom software.

5. Commercial Software

If you’re developing software for resale, in the majority of cases you do not want to build it on top of a packaged software package. That would create licensing issues, would give you less flexibility, and would complicate maintenance. Packaged solutions can be very helpful in prototyping and building MVPs (Minimum Viable Products), but when you’re ready to build your real product, it’s best to create it in a fashion where you own the underlying intellectual property. The only way to accomplish this is through custom software.

By the way, don't kid yourself, packaged software still needs a lot of customization. They just call it different things – configuration, implementation, adaptation. Purchase any enterprise ERP, and try to get anything done out of the box – you won't have a lot of success- they all require a team of skilled consultants to get them off the ground.

We are living in a golden age of packaged software, but there's still a strong role for custom software in virtually every organization. Would I recommend building an ERP from scratch? A CRM? Of course not. But any CIO will tell you that custom software can often be your best friend.



Benefits of Custom Software

Competitive Advantage
Custom software solutions leverage what makes your business unique

Unique Requirements
A custom solution will meet 100% of your business requirements

New Technology
A custom solution will allow you to take advantage of the latest technologies

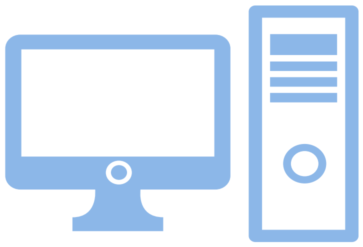
Integration
A custom solution will integrate with existing systems

Commercial Software
With a custom solution, you will own the intellectual property rights for resale

What a Custom Software Solution Looks Like

Computers and software have come a long way since the first digital computers were created in WWII. We regularly use software across multiple devices while we go about our day. With a broad range of devices for software, what a custom software solution looks like can be quite broad as well.

Desktop

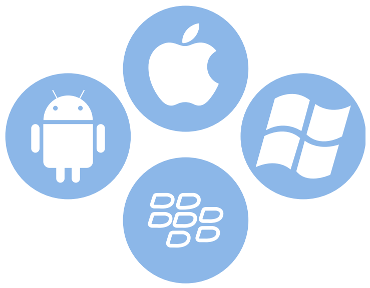


Desktop software runs on your desktop computer and must be installed on a particular operating system (Windows 10, iOS, etc.) They often have strict hardware requirements to function correctly and require regular updates to both the software and computer hardware. Desktop software is quite secure and does not necessarily require an internet connection to work.

Web Applications

Web applications run on a hosted web server and are delivered through the internet. Therefore, they require an internet connection. To function correctly, they rely on resources over the internet, including storage and CPU processing. One major advantage to web applications is that the software can be accessed anywhere no matter what device you are on so long as you have an internet connection.

Native Applications

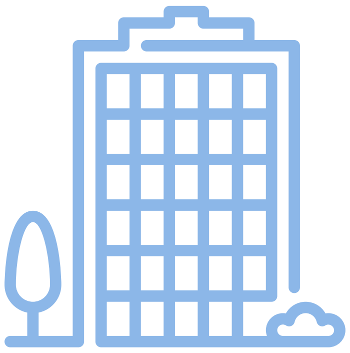


Native applications run on a specific operating system. While this is the same as desktop software, native applications include tablets and mobile phones. Common native applications run on Android, iOS, and Windows. They are installed through an application store such as Google Play, Apple's App Store, or the Windows Store. They are developed specifically for one platform in order to take full advantage of all the device features. With native apps, an internet connection may not be required to use it (aside from downloading the app or updates from the application store).

Mobile

Mobile applications run on tablets or smart phones. Most mobile devices come with pre-installed software such as a web browser or email client, while specialized apps can be downloaded from an app store. Mobile applications are native applications which are developed for specific operating systems, such as iOS, Android, or Windows.

Enterprise



Enterprise Application Software (EAS) is a custom software solution that is created for the needs of an organization, rather than individual users. Businesses, schools, governments, or charities are users of EAS solutions. The main goal behind enterprise software is to improve enterprise productivity, efficiency, and profitability.

A custom software solution can be created on any of these platforms, and sometimes more than one. For example, our mobile inspection software, Field Eagle runs as a native app on Windows 8.1 tablets and on the internet as a web application. The tablet component is for inspectors when they are mobile out in the field and the web application component is for the administrator back in the office. The tablet and web application components synchronize with one another to exchange data to make a complete custom solution for field inspections that replace clipboards and forms.

Cost-Benefit Analysis for Custom Software

While custom software can bring a lot of benefits to your business, it may not be the right solution in every situation. Understanding the business case for custom software is important.

Benefits of Custom Software

Automate Business Processes

By automating business processes you can improve profitability by lowering the cost of doing business. Automating business processes allows employees to accomplish more each day, which can make a custom software solution pay for itself quickly through saving man hours.

Mistakes are a common source of extra man hours. Mistakes happen in every business and many can be eliminated through automation. A simple example would be form validation. If your business is doing data entry with pen and paper, automating this task with digital forms can eliminate data entry errors.

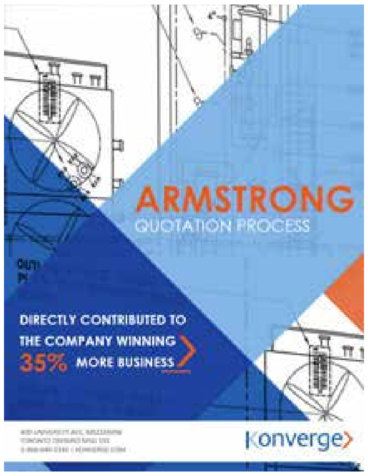
Sometimes mistakes come with greater consequences and you need to take extra care. In situations where you have to comply with industry regulations, such as those set by OSHA or CCOHS, a mistake can make your business liable for fines and possible civil action.

Furthermore, custom software that automates compliance creates a digital paper trail to help you gain and keep certifications like those established by ISO or CSA. This saves your organization time and keeps everything centralized and accessible.

Accelerate Sales Cycles

Custom software can help you decrease the time and cost required to make a sale, which increases cashflow.

For example, if part of your sales process involves sales quotations, long quotation lead times can lead to missed opportunities SA Armstrong, a global commercial HVAC manufacturer, came to us with this problem. One of the services most appreciated by Armstrong's customers is its ability to design custom solutions. These custom solutions make use of both off the shelf parts and components designed specifically for the solution.


ARMSTRONG
QUOTATION PROCESS
DIRECTLY CONTRIBUTED TO THE COMPANY WINNING 35% MORE BUSINESS
Konverge

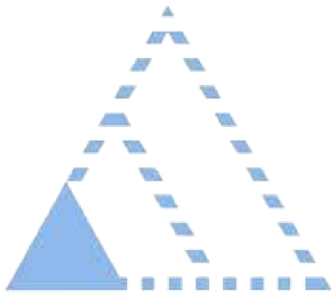
To learn more about our work with SA Armstrong, download the Case Study.

[Download the Case Study]

Preparing quotations for complex systems like these was a long and expensive process for Armstrong. It required both sales and engineering talent. This resulted in the account's sales representative being heavily involved in conveying customer requirements to the engineers, and then presenting the suggested solution to the client. The end result was that engineers spent time on tasks not related to their job function, and salespeople wasting time being the go-between.

Konverge created a custom software solution for SA Armstrong that automated the steps that the engineers used to create a custom solution. This made it possible for salespeople to cut quotation times from weeks to minutes. The result: SA Armstrong won 35% more business, thus increasing its market share and profitability significantly.

Minimum Viable Product Route



The idea of a Minimum Viable Product (MVP) is to develop only the core features that are sufficient to launch the software into the marketplace. Gathering insights from an MVP can save money on developing a product with more features that are unnecessary and are often the result of incorrect assumptions. The more features a product has, the higher the costs and risk if the product fails.

With this route it is possible to recognize revenue more quickly and then use that revenue to add features that will capture more revenue. An MVP works better for most companies than spending money on a fully polished feature-loaded solution and can increase the likelihood of your project's success.

Cut Costs Spent on 3rd Party Software

Another factor when looking at the cost-benefit of a custom software solution is to identify costs that can be cut from money spent on 3rd party service providers that come with monthly, annual, or per usage fees. With 3rd party solutions, your business is at risk of the vendor increasing its rates or closing shop and leaving your company in the lurch. A custom software solution means your application will have more customized functionality and the software itself could serve as valuable intellectual property.

Konverge worked with CentreCorp to build a property viewing application so that sales representatives could show prospective clients property leasing information. Prior to working with Konverge, whenever updates to the application were required, CentreCorp had to hire a third party service provider to conduct changes to the application. This resulted in long wait times for the application to be updated which was time consuming and costly for CentreCorp. Konverge redesigned the application with additional features, updated user interface, and added the functionality for CentreCorp to have full administrative powers of their application so that an expensive 3rd party provider was no longer needed.

CentreCorp
Mobile Tablet Application
Leverages the Sales Team
& Increases Operational Productivity
Konverge

To learn more about our work with CentreCorp, download the Case Study.

Download the Case Study

Capture Market Share

If you can improve the experience for your customers and increase their satisfaction, you will be ahead of the competition by not only retaining customers, but acquiring new ones. Custom software can create revenue by making it easier to retain and attract customers by offering a new way to access your products or services. Consider Starbucks mobile application which offers a convenient way to pay for purchases and collect reward points, or HBO Go which gives unlimited access to HBO's movies and shows online on any device. With a custom software solution, both these companies improved their service to their customers, putting them ahead of their competitors. It is important not to wait for your competitor to come out with a new service before creating something for your customers yourself.

Costs of Custom Software

Now that we have looked at some of the benefits a custom software project can provide, let's look at the costs of a custom software project.

Development Costs

This is the cost to develop your initial custom software. Included in this cost is requirements gathering and documentation (completed by a business analyst), project management costs (done by a project manager or coordinator), software development (which includes planning, writing code, graphics and user interface design), and quality assurance testing.

Maintenance

Any custom application is going to need to be maintained – over time bugs and software conflicts will arise that need to be addressed. Inevitably, new functionality will be needed to keep the solution up to date. For any custom solution, its necessary to budget for ongoing maintenance.

Hosting

Hosting is an ongoing cost for custom software. There are a few options for hosting (cloud or on a secure internal network), however in both cases, you need to consider your bandwidth requirements and your anticipated future growth. Increased traffic to your web application is good for business, but you need to make sure your hosting can deal with increases in traffic.

Security is another important factor to consider with hosting. Depending on the type of data your application will store, there may be security requirements you need to meet. For example, banks and financial institutions need security measures to ensure Payment Card Industry Data Security Standard (PCI DSS) compliance. E-commerce sites will need to implement a Hyper Text Transfer Protocol Secure (HTTPS) protocol, which is the secure version of HTTP, the protocol over which data is sent between your browser and the website that you are connected to. The added security required for some applications may require a more costly hosting solution.

Security Patches and Version Updates

Inevitably, those inconvenient update messages on our devices happen. Sometimes they even force us to reboot our devices so they can be installed. Similarly updates occur within software languages and frameworks. Often these updates have additional features and security measures that are very beneficial for your software. Letting version updates linger can have negative consequences such as having to pay more for updates because your custom software is out of date, an inability to install the latest security patches, and the inability to add features or fix bugs.

Feature Upgrades

As your custom software solution is being used by your customers or employees, their feedback will guide your decisions for feature upgrades. Feature upgrades improve the user experience and functionality of your software. You should continuously set aside a budget for ongoing feature upgrades for the lifespan of your custom software.

Selecting a Vendor

If you've considered the costs and benefits of a custom software solution, your next step may be to begin evaluating a company to develop a solution for you. The right partner is critical to your success. Finding one is not as difficult as it might appear if you use the criteria listed below.

The criteria are not listed in order of importance, as the project's needs will dictate the relative importance of each criteria. In some rare cases, for example, knowledge of the technical environment may be the most important criteria (real time process control systems for example).

Please note we have not addressed price in this list. Price is always important, but it's critical to keep these other criteria in proper perspective, particularly when you take into account the cost of a failed project.

Stability

Acquiring a custom software solution from a software house often means contracting with the company to maintain the solution in the future. Unless your internal IT resources are going to assume this task during a formal handoff, the vendor will be your first resource to fix issues, enhance the system, or modify its behavior as the business needs dictate. For more complex projects spanning several years, the stability of the vendor becomes important even before the handoff.

You want to select a vendor with an established presence, a satisfied list of clients and the financial stability to survive downturns, recessions and ever increasing competition. In short, you want the vendor to have been around a while, giving you some assurance that they will remain in business for the expected life of your custom software solution.

Track Record

This point is closely related to the vendor's stability; in particular we draw your attention to the following points.

1. Business Analysis Success

Based on the reasons for failure we cited above, we suggest that you request references from the vendor's clients and ask them the question directly: "Did the developer get the requirements right the first time?"

Yes is the best answer, of course, but maybes or are not necessarily bad news. There are reasons on the clients' side which contribute to not defining requirements correctly, too: Lack of executive involvement, lack of dedicating the time needed to study the documents and participate in the process, and so on.

2. Budget success

How well has the vendor performed on previous projects with respect to timelines and budgets?

This answer, too, is sometimes not so easy to evaluate. In some cases, especially when the client did not clearly understand their own objectives or requirements, the project begins, runs well on time on budget, and then hits a hiccup when the software is delivered and found to be disappointing. It must then be redesigned and rebuilt and thus the project goes overbudget for reasons which have little to do with the vendor.

Ultimately, it is the vendor leading the client through the development process. It is thus the vendor's responsibility to ensure that the solution will solve the client's requirements. What's important here is the track record – the company's performance over many projects.

3. Cultural Fit/Alignment

The requirements gathering process should be illuminating to the client, leading to insights into an individual's job function, and to improvements in business process design. This is exciting because it shows people how to do their jobs more efficiently, and offers the promise that when the software is delivered they will be more efficient and effective.

4. Their Process

Based on the remarks above, we suggest that you formulate a set of questions for your potential software vendor:

1. You should ask for a full briefing on their software development methodology, and whether they follow any of the industry standards (ITIL, CPI, ISO, etc.).
2. Ask them whether their methodology is based on a traditional approach, an agile approach, or a hybrid.
3. Ask to see the working documents and systems they use during their process as well as the deliverables that are generated from their systems and processes.
4. Ask the references if the vendor followed the process description.

5. Responsiveness

This again is a question for the references. How quickly did the vendor respond to important issues? Depending on the project's importance to your organization, this may be more or less critical.

6. Technical knowledge

Judging technical competence, especially when performed by non-technical people, is challenging. But a track record speaks for itself. Questions to ask the vendor's references include:

1. Budget performance. While being on time and on budget doesn't always mean the company has solid technical expertise, it usually implies that they were not delayed by technical issues too often.
2. Does the supplied solution meet the performance criteria which were established as its objectives?

Software Development Methodologies

The objective of software process specifications is to produce a repeatable, predictable software development process.

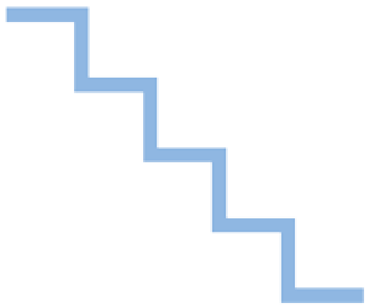
The ISO standard 12207 covers software development processes. The standard defines a software process lifecycle for 23 Processes, 95 Activities, 325 Tasks and 224 Outcomes. The areas covered are:

- Acquisition (initial start-up and requirements specification),
- Supply (develop the project plan)
- Development (design, code and test the software)
- Operation (system commissioning and ongoing operation)
- Maintenance (defect repair and enhancements).

Some of the most popular software development processes are:

- Waterfall
- Rapid Application Development (RAD)
- Spiral
- Agile

Waterfall



The waterfall model (also known as the traditional model) suggests that developers follow the phases in this order:

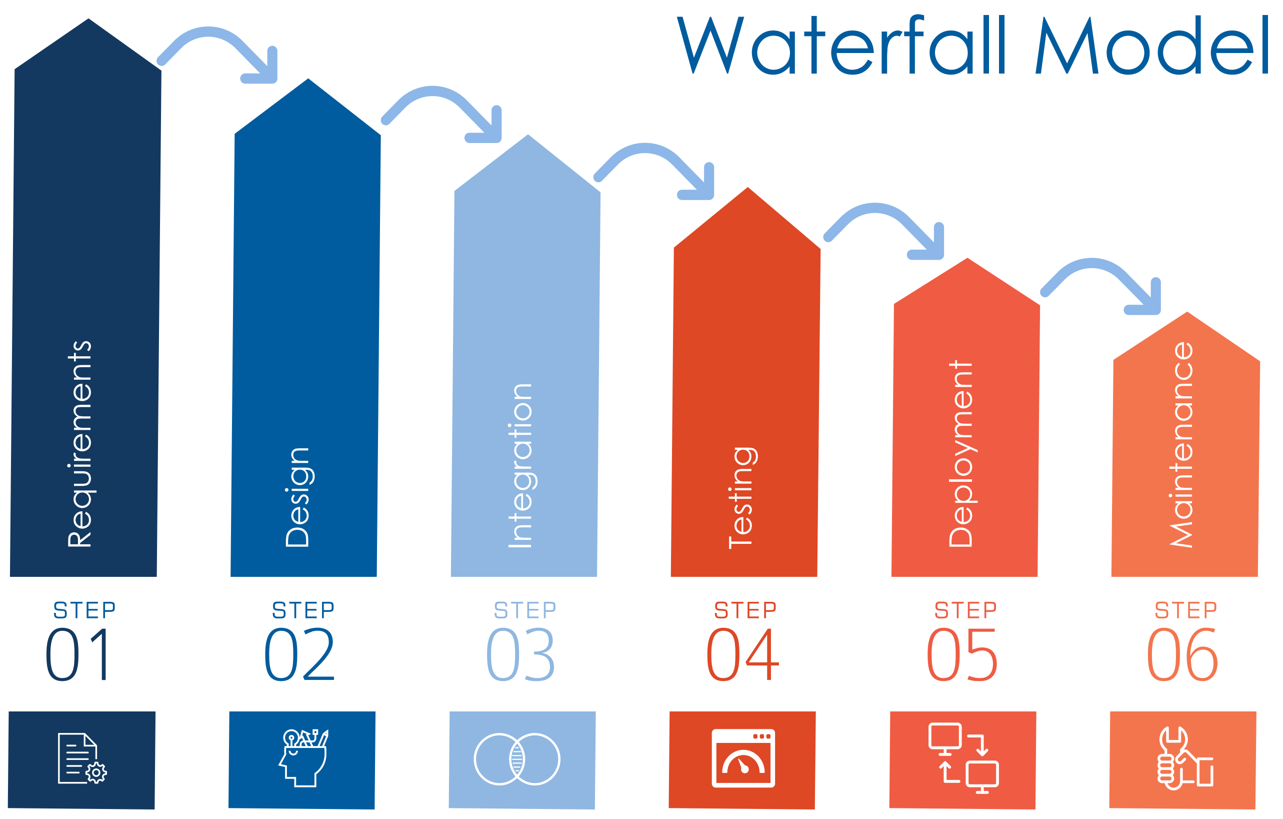
1. Requirements specification
2. Software Design
3. Integration
4. Testing (or Validation)
5. Deployment (or Installation)
6. Maintenance.

On completing one phase, the team proceeds to the next. The steps and activities defined for the end of each phase include quality assurance tasks and any errors are rectified before moving on.

The Waterfall method is one of oldest software development processes and it has many proponents. Its rigor ensures that the phase is “correct” before the team moves on to the next phase. Research shows that correcting a software error at the final Q/A stage versus up front at the requirements stage is 1,000 times more expensive. So making sure one is building the right thing up front is worth its weight in gold. (Barry Boehm’s Software Engineering Economics, 2002).

On the other hand, the inflexibility of a pure Waterfall approach reduces its value in the eyes of developers. Many people believe that the captured specification of the user’s requirements is simply never good enough to use as the final form of the design.

The answer is to use models, mockups and wire frames to gather the requirements. With these as the common language between the developers and users, the shared understanding of the requirements is often very good and can indeed be used as the basis for the next step, the Design.



Rapid Application Development

Rapid Application Development, most often called RAD, relies on building a series of versions of the software. Version 1 is the smallest delivery of functionality, assembled quickly based on initial conversations with its users. Each subsequent delivery adds to this base level of functionality until the complete system has been delivered.

It sounds like a great idea as it gets things moving quickly, progress is steady and observable and the rapid delivery of iterations of software keeps the users involved. RAD can be very effective for smaller and less complex systems. More complex projects require a more extensive requirements gathering phase than is traditionally performed for a RAD project. And for large projects the resulting design phase can extend over a lengthy period.

Agile Development

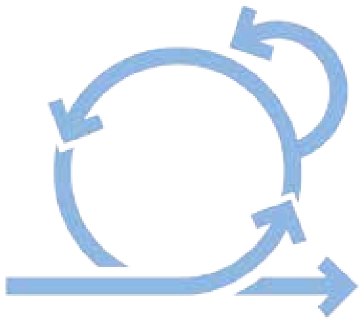
Agile development methods are similar to RAD approaches in that they produce successive iterations of the software until the final delivery constitutes the whole solution. But whereas RAD relies on a structured approach in the planning and execution phases, Agile relies more on soliciting feedback at appropriate times.

Two approaches are common in Agile, although there are many variants:

Extreme Programming (XP)

In this approach the phases are performed by a small team who remain together for the duration of the project and who do all of the tasks themselves. Communication between team members is thus highly efficient and cycle times for each iteration are often measured in days or weeks versus the more traditional approach of months or years.

But small teams, no matter how good they are, can only do so much in a day and XP is thus best suited to smaller projects.



Scrum

A new approach to Extreme Programming was proposed in 1986, by which its authors hoped to improve the speed and flexibility of commercial product development projects.

The authors, Hirotaka Takeuchi and Ikujiro Nonaka, called it Scrum because the word is a rugby formation with all the players closely bunched together. They likened the goal of a rugby match to the goal of a software project, in that the entire team is dedicated to reaching the other side of the field.

The team consists of the cross-functional experts needed to analyze requirements, design, code, test, install and make operational the software system. Each iteration is called a sprint and the various team members are responsible for their individual components, as well as the overall system itself.

Scrum teams are flexible enough to allow the users of the system to change their minds frequently about what is being built. They do this on the assumption that complex systems cannot easily be understood in advance of their delivery (an excellent assumption!), and thus expecting users to get the requirements right is not realistic.

But for complex systems with many interacting facets, there really is no alternative to an exhaustive requirements gathering stage followed by a carefully designed solution. Changing your mind about what you want is okay if it only impacts the latest sprint or iteration. But it’s not good if the change means starting from the beginning.

Software Development Methodologies Pros & Cons

Waterfall

Pros

- The Waterfall method is well known amongst the software developers therefore it is easy to use
- Works well for smaller projects where requirements are very well understood
- Cost effectiveness: Time spent early in the software production cycle can lead to greater economy at later stages

Cons

- Testing understood as a ‘one time’ action at the end of the project
- High amounts of risk and uncertainty
- Inflexible
- Poor model for complex and object-oriented projects

RAD

Pros

- Reduced development time
- Flexible and adaptable to changes
- Good for large projects
- Overall reduction in project risk
- Larger emphasis on simplicity and usability of UI design

Cons

- Cannot use for small projects
- Requires more resources and money to implement
- Only systems that can be modularized can be built using RAD
- Requires highly skilled developers
- Needs both customer and developer commitments to complete a project

Agile

Pros

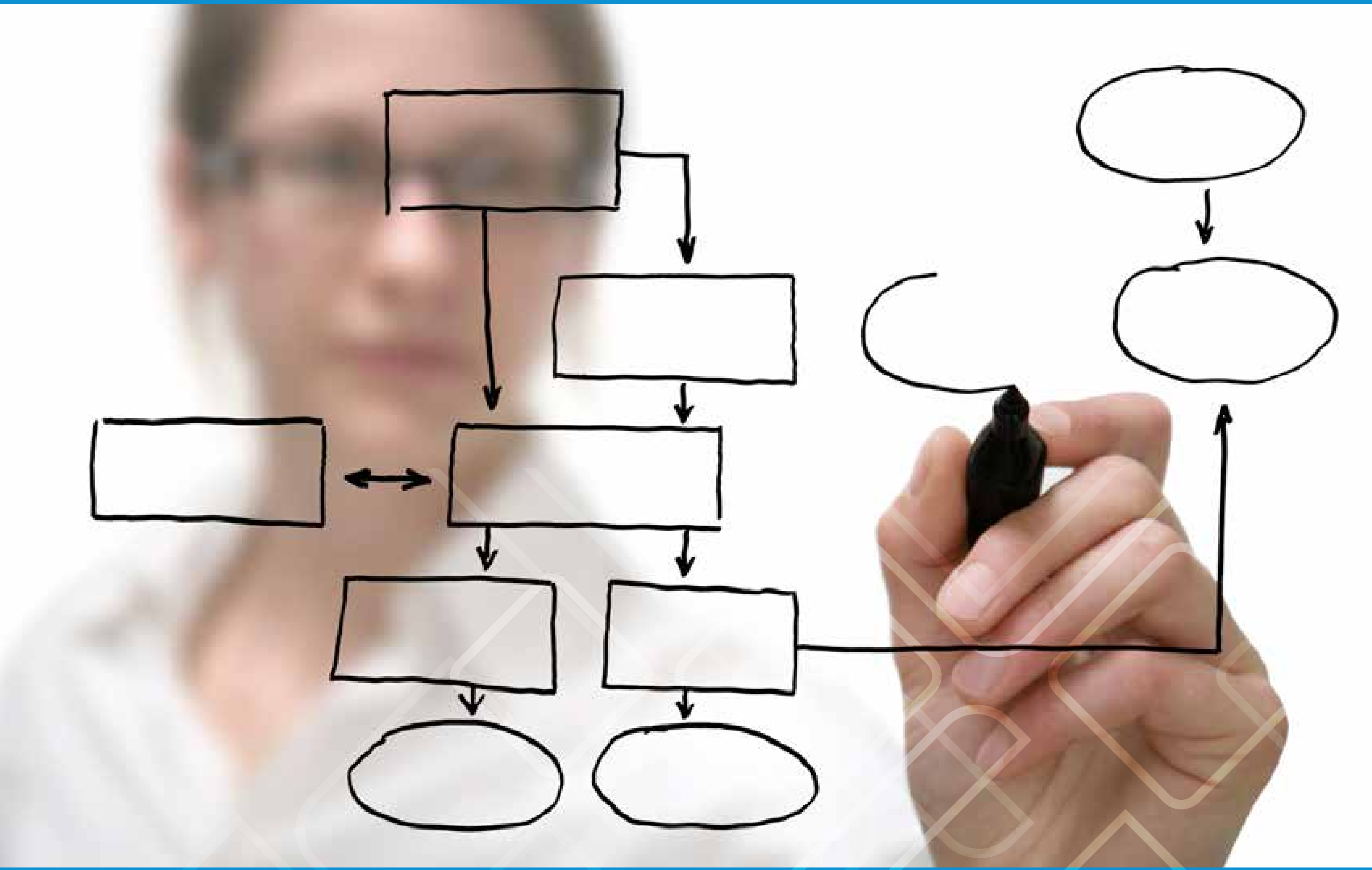
- Focus more on the application rather than documenting things
- Daily meetings and discussions for the project following Agile model can help to determine the issues well in advance and work on it accordingly
- Requirements changing even in late stage of development
- The end result is the high quality software in least possible time duration

Cons

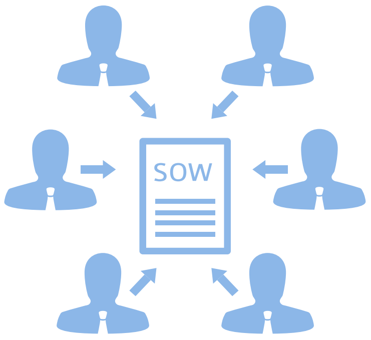
- The project can easily go off track if the client is not clear what final outcome that they want
- There is lack of emphasis on necessary designing and documentation
- Lack of client involvement is often a potential problem especially in big and complex projects
- Agile processes are really only applicable for products where reliability is not very critical

The Software Development Process

With over 20 years of software development experience, we have learned that the best approach is to adopt a collaborative methodology which involves the client in every step of the project. The following is our software development process.



Proposal



The proposal phase culminates in the client receiving a Proposal or Statement of Work (SOW). The SOW outlines the functional requirements, project plan, cost, risks and assumptions, and the rules of engagement for the client and Konverge. An Account Executive leads this phase and organizes the sales team and the other resources required to produce the proposal. The Requirements Outline itemizes the requirements of the Solution Concept and is appended as a schedule to future documents. Requirements are typically organized into feature sets (groups of services), or areas of functionality.

Upon acceptance of the proposal by the client, the project moves into the planning phase.

Planning

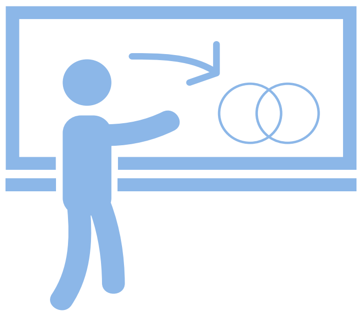
The objective of the Planning Phase is to assign the Project Manager and set up the management system for the project.

A stakeholder analysis is then conducted to define the roles and responsibilities of each member of the project team.

The Planning Stage ends with the creation of the Project Plan showing for each resource, tasks/activities, timeframes, dependencies, and overall resource usage statistics to enable resource risk analysis.

The Project kickoff meeting is then held to review the project plan with the client and introduce the Konverge team.

Analysis



When the Project Sponsor accepts the Statement of Work (SOW), the project moves into the Analysis phase. This phase expands on the project sponsor's requirements and produces a Systems Requirements Specification (SRS). The analysis phase consists of the following steps:

1. System Analysis Team Formed

2. System Requirements Specification is produced showing:
    a. Vision/Scope.
    b. Use Cases
    c. Wire Frames
    d. Solution Architecture

3. A detailed Test Plan.

The SRS is reviewed by the entire project team. Feedback is incorporated into the document until everyone signs off, signifying that they believe it is a precise definition of the intended Software Solution. The project then moves into the Design Stage

Design

The Design Phase determines the architecture and physical design of the software components needed to fulfill the solution concept's requirements. This phase is led by the System's Analyst and orchestrated by the Project Manager. The phase consists of the following stages:

1. Design Team Setup

2. Requirements Presentation Handoff

3. System Design Document Development, including:
    a. Design Guidelines
    b. Domain Models
    c. Component Inventory
    d. Class Diagrams
    e. Design Review

When all the design artifacts are completed the Systems Analyst submits the work to Konverge's Software Development Manager (SDM) for review. The SDM provides feedback on areas of the design that may need further elaboration or planning before engaging the development team. The areas of the project design scrutinized are:

- Mission-critical components to determine whether they have been designed with sufficient detail for a development team to work on
- Whether the design guidelines and prescribed patterns conform with Konverge's software design guidelines
- Whether or not all technical risks have been identified and communicated in the Technical Risk Management document.

This process of developing the design documents, and redesigning them, continues until they are formally accepted. The acceptance of the design documents by the SDM results in the "Core Requirements Complete" milestone.

Development

The Development Phase creates the physical software components to satisfy the design requirements. It consists of the following stages:

1. Development Team Formed
2. Development Environment Network Diagram
3. Development Environment Setup Complete
4. Development Plan
5. Requirements Review
6. Technical Risk Assessmen
7. Component Development
   a. Persistence Tier Development (data repository and the code)
   b. Application Tier Development (the component's application logic)
8. Unit Testing
9. Code Review

10. Component Optimization which includes:
    a. Performance Testing
    b. Peer Review
    c. Component Complete Milestone

12. Update Project Documentation- test cases and the Development Plan

13. Staging- deploy into the final testing environment.

QA

The QA team Passes or Fails features based on conformance to the SRS and determines if they meet the success criteria. The QA team tests to ensure that the application runs on the browsers and platforms that have been established in the Statement of Work.

During this time the QA team and the Developers are involved in a circular process of testing the application, recording bugs, fixing the bugs and then retesting. At the end of this cycle the application is deployed to your environment or remains on the mirror environment (depending on the terms of your SOW), and User Acceptance Testing can begin.

User Acceptance Testing validates issues specific to your business and your workflows. No matter how well we captured your requirements, the true test of the application comes when its end users test it. Sometimes a situation occurs in these tests that was not anticipated by the Project team. In these cases the issues are addressed before the testing continues or is restarted.

On completion of this phase the milestone “Development Complete” is reached and the project moves to the Stabilization phase.

Stabilization

The Stabilization phase validates components against the requirements, identifies defects and reports them to the development team.

The Stabilization phase typically progresses through the following stages:

1. Designing the Quality Control Environment
2. Executing the Test Cases for each component
3. Resolution of all issues arising from the tests
4. “Declaring components “Stable”
5. The performance, reliability and load capacity of the system as a whole is then tested
6. The results of the Performance/Load test are reviewed
7. If satisfactory, the system is declared Stable and is pronounced ready for release to the client.

Deployment

The Release Phase deploys the software into the production environment and ensures a smooth transition into the hands of the end-users. During this phase the following takes place:

1. The Release Team is formed
2. A Delta Analysis is conducted to measure differences between this release and its predecessor.
3. Release Notes are Developed
4. The Deployment Package is finalized
5. The Deployment Package is installed

Once the system is declared operational it moves into Maintenance mode.

Maintenance

During Maintenance, the system:

1. Any defects are corrected in a timely and comprehensive manner.
2. New features and enhancements are discussed and implemented as needed, according to the wishes of the client.

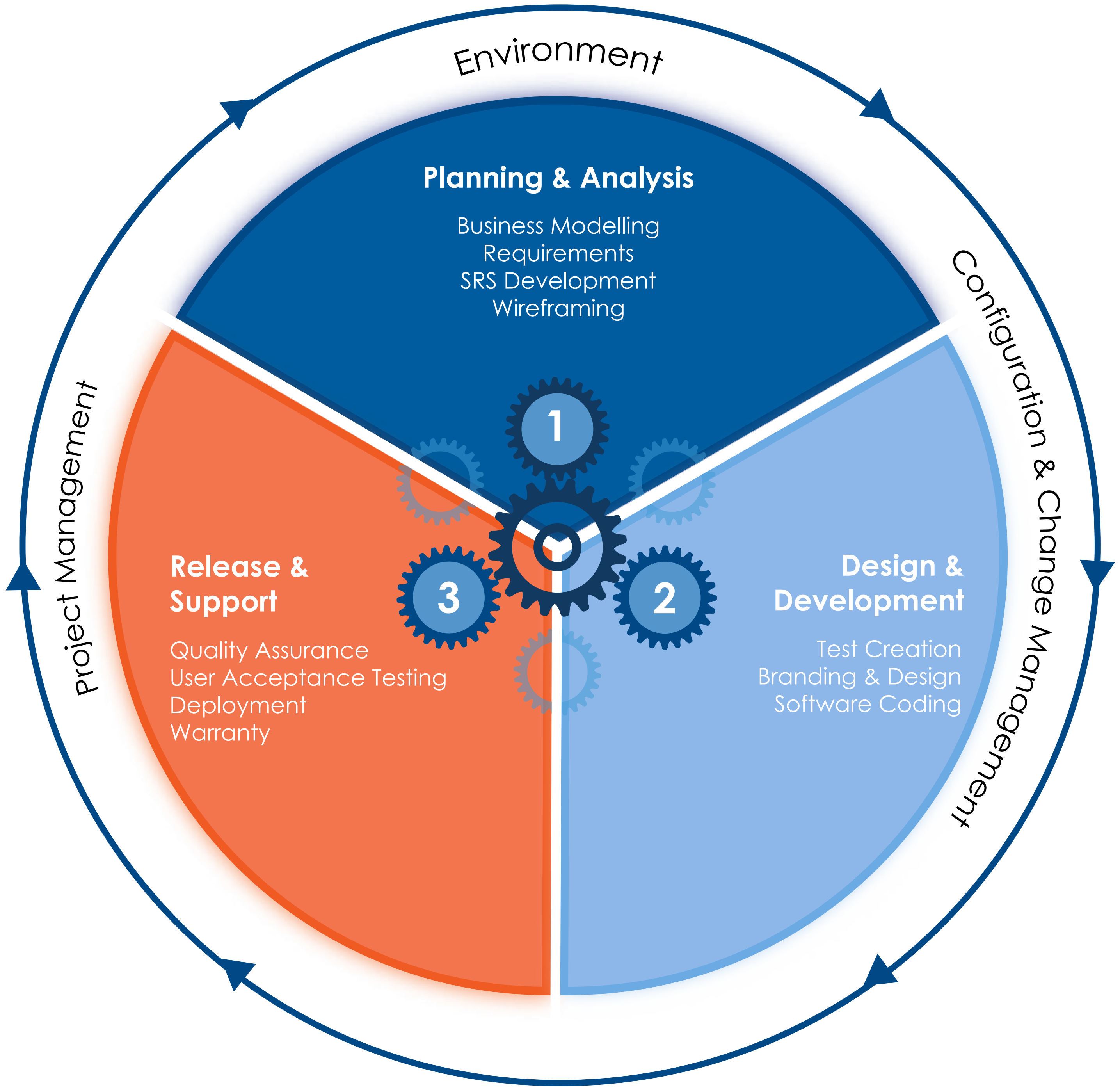
Lessons Learned

On acceptance of the completed project, the team revisits the project from the beginning. The Project Manager compiles a list of all the the successes and issues that occurred during the life of the project.

This is an open discussion designed to improve the process and benefit the clients.

Konverge Collaborative Process

The software development process described is what makes up Konverge's Collaborative Software Development Process. Our process covers the entire software development life cycle from upfront planning, to process and risk management, and then testing and application deployment. Its flexibility and scalability makes it effective on projects of any size.



Conclusion

Buying a custom software solution is a lot like buying a tailored suit. While it costs more, it fits you like no off-the-rack-suit can. The right fit can be the difference between increasing productivity and turning a profit, or being inefficient and returning a loss.

Different organizations have different needs. When deciding whether a custom software solution or an off-the-shelf one is better, consider the benefits weighted against the costs for the two different approaches. There are many advantages to a custom solution and with the right software partner, the risks associated with building custom applications can be mitigated.

We hope our ‘Essential Custom Software Development Guide’ will help youmake the right decision the next time you are looking to leverage technology within your business.

About Konverge

Customers have been coming to Konverge since 1994 for custom software solutions. We specialize in developing breakthrough custom software applications. Our development methodology involves you as an integral part of the process, turning your business ideas into a competitive advantage.

Konverge is headquartered in Toronto, Ontario and works with organizations of all sizes across Canada and around the world.

To learn more about our custom software solutions, visit www.konverge.com or contact us directly:

1-866-640-2345

info@konverge.com

Microsoft Partner
Gold Application Development

Konverge