Software Development Agreement Checklist (Pro-Developer)

A Practical Guidance® Checklist by Sonia Baldia, Kilpatrick Townsend & Stockton LLP

Sonia Baldia
Kilpatrick Townsend & Stockton LLP

A software development agreement is a contract for the design, development, testing, installation, and implementation of custom software, or maintenance or modification of currently existing software. This checklist highlights key issues, from the developer's perspective, in drafting and negotiating a software development agreement. For a more detailed discussion on the key issues in drafting and negotiating a software development agreement, see Software Development Agreement Negotiating and Drafting.

This checklist addresses only software that a customer controls and uses on its systems. This practice note does not address situations when the customer:

- Controls or uses the software on a system hosted by the software developer or any third party
- Is a government entity
- Must comply with industry-specific regulations, laws, or restrictions to use the software
- Uses or accesses the software outside of the United States

Initial Considerations

- **Parties.** Confirm each party's legal status and whether any third parties (such as group affiliates) will benefit from the proposed agreement.
- **Third-party guarantees.** confirm whether any third party will guarantee the customer's performance or financial obligations.
- **Project scope.** Unambiguously describe what the developer will create as a defined term in either the preamble, the definitions section, or in a stand-alone section at the beginning of the document.
- **Requirements analysis.** The parties should document a set of requirements describing the customer's business objectives and the role the software will play in meeting those objectives.
- **Design.** Ensure the developer determines the technical details of how the software will perform and create a set of software design documents on which it will rely to build the software.
- **Build/development.** The developer should determine the programming language it will use for coding the software
- **Testing.** Structure this phase as an iterative process, where the developer tracks and reports any deficiencies, bugs, or defects uncovered during quality assurance testing, and then fixes and re-tests the software until it resolves the problems.
- **Implementation/deployment.** Understand how the parties intend to deploy the software and whether the developer will initially deploy the software in stages by releasing it to only a limited number of the target users at a time.
- **Milestones.** Use milestones to define the crucial project tasks to be completed during project planning, as well as to specify the activities to be performed and completed by the parties at certain key points throughout the life cycle of the project.

Deliverables

Include a list of tangible items that the parties deliver during the software development process. Consider the following issues when determining the deliverables:

- **Reference to milestones.** The parties typically refer to project milestones, previously agreed on, to create this list. A deliverable is a measurable and verifiable item that a party undertakes to help achieve that milestone. The parties may need to produce one or more deliverables to achieve a specific milestone.
- **Timing.** Prepare the list of deliverables after the parties agree on the requirements specifications, and include specific due dates for the completion of each deliverable.
- **Progress payments.** If the parties structure the agreement to include progress payments, ensure that the successful delivery of the items described in the deliverables are determined by objective specifications that the parties have mutually agreed on, not the subjective opinion of the customer.
- **Delivery.** Avoid arbitrary deliverable delivery dates, and instead negotiate dates that realistically reflect the time needed for each party to perform its respective obligations to complete every deliverable.
- **Acceptance testing.** Require the customer to comply with the acceptance testing procedure (discussed below) for reviewing each deliverable.

Pricing

The parties usually negotiate pricing for software development services on either a time and materials or fixed price basis. Time and materials may be the most profitable pricing model for the developer because it is difficult to estimate the amount of time needed to undertake a software development arrangement.

If the customer insists on a cap when the developer charges a time and materials rate, consider negotiating bonus payments for successfully completing deliverables before the applicable due dates to help increase the potential revenue.

Ancillary Expenses

Ensure that the agreement addresses additional fees for any ancillary services that the customer desires. Examples include:

- Out-of-pocket expenses
- Training on how to use the software
- Maintenance and support of the software
- Consulting on projects related to the software development agreement

Payment

The payment section defines when and how the customer pays the developer. When drafting the payment terms, consider:

- How customers can pay their invoices, including the currency in which payment must be made and specific wire or check submission instructions
- When payment is due (e.g., 30 days from the date of invoice)
- If the developer can charge interest on overdue unpaid balances
- Making the customer responsible for collection costs, including reasonable attorneys' fees
- Making the customer responsible for levies or taxes relating to its use of services, other than the developer's income taxes

Because software development projects can be lengthy, waiting for a single lump payment at the end of the project involves significant financial risk for the developer. As a result, you should negotiate periodic payments that the customer makes throughout the term of the agreement (e.g., after the successful completion of each project milestone, or in regular monthly installments).

In addition, the developer should negotiate a larger up-front initial payment to minimize risk to its cash flow and working capital needs while undertaking the project (e.g., 25–50% of the overall price).

Change Requests

During the course of the project, the customer may want to make changes to the requirements or specifications in a way that impacts the project's scope, schedule, or cost. Ensure this provision requires that:

- A requested change must be described in writing
- The request can only be agreed to by an authorized representative of the other party
- The other party must accept or reject the requested change within a certain period of time

You should also address the following issues when drafting this clause:

- **Refusal right.** Include a provision that gives the developer the right to refuse a change request that preemptively limits the amount the developer can charge for the request.
- **Termination restrictions.** To avoid the risk that this provision becomes a backdoor termination right for the customer, consider limiting the situations in which the

customer can terminate the agreement if the parties cannot agree on a change request.

- **Pro rata payment.** If the parties cannot reach an agreement on the change request, the customer should pay the developer for any work product that has been completed and in progress.
- **IP ownership.** Consider retaining ownership and all IP rights in the work product to disincentivize the customer from early termination.

Acceptance Testing

Include an acceptance procedure for the customer to review and approve each deliverable before the developer proceeds to the next deliverable.

General Terms

To prevent delays and to encourage prompt responses, the developer should:

- Provide a specific time period for the customer to review and respond in writing to a submission
- Address whether a submission should be deemed approved if the customer does not respond within the specified time frame, or when the customer makes the software available for end users
- Ensure that the customer cannot reject any deliverable that failed due to any third-party software or hardware unless the developer recommended or required that the customer use that software or hardware

Testing Standards

Include only objective standards for acceptance testing. Disputes often arise when the customer rejects a deliverable for a subjective reason, such as a perception of how the deliverable should operate or the customer's internal expectation of functionality. Instead, require that the customer perform acceptance tests using the requirements specifications or detailed design documents.

Customer Obligations

You should also require that the customer:

- Begin acceptance testing immediately upon the developer's submission of the deliverable
- Provide notice of its determination immediately to the developer
- Include a detailed explanation of its rejection in sufficient detail to enable the developer to recreate and to verify the noncompliance

Remedies

In the event a deliverable has failed acceptance testing, limit the remedies available to the customer by:

- Allowing the developer to fix and to re-submit a deliverable (usually three attempts)
- Allow the customer to terminate the contract if the developer cannot successfully produce a deliverable and
- Prohibit the customer from recovering monetary damages aside from a partial refund of any fees paid up to the date of termination

Intellectual Property Rights

The agreement must address which party owns the software and all intellectual property (IP) rights, including:

- Right, title and interest in the software
- Software source and object code, all related documentation, and manuals for the software (including all IP rights in these items)
- Developer's work product created during the project (e.g., scripts, product concept, product backlog, internal software, and data used for testing)
- Developer's own pre-existing work product used in the development of the software
- All related copyrights, patents, trademarks, and trade secrets

Developers usually rely on their own pre-existing, proprietary techniques, know-how, methodologies, utilities, processes, algorithms, and tools to develop software for multiple customers. As a result, you should:

- Ensure that the developer retains ownership of all such pre-existing work product, and all IP rights in the work product
- Clarify that the developer can use its pre-existing work product in the development of other software for future clients, free and clear any ownership claims, liens, or approval rights of the customer
- Grant a limited license to the customer to use that work product solely to use the software, if any pre-existing work product is incorporated in the software

You also should include a clause stating that the developer is not obligated to transfer ownership of the software (or any related IP rights) until the customer has fully paid for the work performed under the agreement.

Representations and Warranties

Representations and warranties in a software development agreement should be tailored to directly address issues that may impair a party's ability to perform its obligations or imperil the overall project.

Standard Representations and Warranties

Examples of general representations and warranties in a software development agreement include:

- The developer has or will have and maintain sufficient resources, facilities, capacity, and personnel to assure that all work will be provided in a timely and workmanlike manner.
- There are no commitments, obligations or agreements with any third party that would conflict with either party's obligations under the software development agreement, or otherwise restrict a party from entering into the software development agreement.
- During the term of the software development agreement, neither party will enter into any commitment, obligation or agreement that conflicts with its duties under the software development agreement.
- Each party has obtained all licenses and permits required to perform its obligations under the software development agreement.
- Each party will comply with all applicable laws, rules or regulations during the term of the software development agreement.

Performance Warranty

To the extent that you offer a warranty:

- Use only objective standards to measure performance of the services (e.g., specifications)
- Do not use subjective standards to measure the operational effectiveness of the services (e.g., the services operate to the reasonable satisfaction of the customer)
- Limit the warranty to a "material" conformance with the specifications so that you are responsible only for fixing errors/bugs that significantly impair the customer's ability to use the services

Disclaimed Warranties

Any performance warranty offered by the developer should not cover:

- Unauthorized modifications of the services by the customer or any third party
- Use of the services with third-party software or hardware, or in an environment or manner not originally contemplated by the parties
- Errors or misuse of the services by the customer, its employees, or agents

Include the standard disclaimers to all other warranties, express or implied, and insert language specifying that the developer makes no warranties that:

- The services will operate uninterrupted
- Be free of defects or viruses
- The customer's data will be secure from loss, damage or theft

Sole and Exclusive Remedy

Limit customer's remedy for a breach of warranty to commercially reasonable efforts to cure the breach. Include a disclaimer that this is the "sole and exclusive" liability of the developer.

Indemnification

Software developers often provide indemnification for:

- Personal injury or property damage caused by the developer's personnel or agents, in situations where the developer is performing on-site work
- Any claim that the software infringes on a third party's IP rights

Limit indemnification to claims based on infringement of a third party's U.S. intellectual property rights (if the contract is U.S. based) in existence on or before the date of the contract.

In addition, customers sometimes seek indemnification from developers for breaching their confidentiality and data security obligations. Try to limit these indemnities to breaches caused by a "material failure" to comply with the obligations.

Indemnification Procedure

Ensure that the indemnification provision also addresses the mechanics of providing indemnity to the other party, including:

- The indemnified party's obligation to notify the indemnifying party promptly of the pending or threatened action
- Requiring the indemnified party to provide cooperation and technical assistance to the indemnifying party
- Selection of counsel and control of defense of the case
- Whether the indemnifying party may settle the case unilaterally, or only with the indemnified party's approval
- Monetary caps on indemnity (including attorneys' fees)

Exclusions to Developer Indemnification

Developers should exclude claims made due (in whole or in part) to the customer's own unauthorized acts. Examples include:

- Modifications to the software without the developer's prior written approval
- Failure to install updates or upgrades to the software that would have avoided the infringement

- Suggested changes to the specifications or software that cause the infringement
- Use of the software with any third-party hardware and software not authorized by the developer
- Use of the software for any reason other than intended purpose
- Gross negligence or willful misconduct of the customer's employees or agents

Remedies

In lieu of paying damages for IP infringement, software developers typically:

- Replace or modify the services with substantially equivalent services so that the services are no longer infringing
- Obtain for the customer the right to continue using the services
- Terminate the applicable services and reimburse customer for any prepaid but unused services as of the date of termination

Sole and Exclusive Remedy

- Include a disclaimer that this the "sole and exclusive" remedy for indemnity claims against the developer.

Limitation on Liability

A limitation of liability clause generally limits the:

- Types of damages recoverable by a party (e.g., special damages, consequential damages or lost profits)
- Amount of damages recoverable by a party (e.g., a fixed amount, a multiple of fees paid, or an amount recoverable only over a specific period of time)

The developer faces more exposure to risk through its own conduct and less exposure to risk through the conduct of the customer. As a result, the developer should consider a unilateral limitation of liability provision that:

- Prohibits the customer from recovering any indirect damages from the developer (e.g., speculative, incidental, punitive, and consequential damages)
- Caps the developer's potential liability to a fixed amount, or a certain amount of revenue generated over a fixed period of time (e.g., fees paid over the past 12 months, or fees paid under the SOW giving rise to the claim)
- Sets a statute of limitations on recoverable damages (e.g., prohibit any action brought more than 12 months after the initial event giving rise to the alleged liability)
- Ties any indemnity and warranty remedies to the limitation of liability clause to narrow the scope and amount recoverable
- Caps the aggregate amount of attorneys' fees that the customer may recover in a dispute
- Includes an aggregate cap that limits total amount recoverable from the customer over the life of the contract

Confidentiality

If the parties have already entered into a separate confidentiality agreement, you should include a brief provision that references the executed confidentiality agreement and incorporates its terms as part of the software development agreement.

Otherwise, you should include a confidentiality clause that prohibits either party from using or disclosing the other party's proprietary and confidential information to third parties. For a more detailed discussion on confidentiality agreements and tips on negotiating confidentiality provisions, see Confidentiality Agreements.

Staffing

Because software developers often rely on key employees who have a specialized skillset or expertise, customers typically require developers to commit to assigning specific individuals to the project. Developers should:

- Refrain from committing specific personnel exclusively to a project
- Consider limiting the circumstances under which the customer can require the removal and replacement of a particular project member (e.g., repeated offenses or constant poor work performance)
- Avoid providing a specific number of employees that will be staffed on the project to maintain flexibility in undertaking multiple projects at the same time

Draft supervisory mechanisms to facilitate effective coordination and communication between the parties. Common mechanisms include:

- Management committee comprised of senior employees and executives from each party, which oversees the overall development process
- A decision tree for managing and escalating day-to-day questions, disputes, and unforeseen issues
- A project manager designated by each party to manage the decision-making process

For complex projects, consider including:

- A roster of directly responsible individuals (DRI) (i.e., a list of individuals directly responsible for managing the completion of specific tasks) as an exhibit
- The names and contact information of individuals who will act as the official points of contact for each party
- The names and contact information of individuals who are responsible for responding to specific types of inquiries

Insurance

Developers often agree to maintain the following customary business liability insurance for at least the duration of the software development agreement:

- Commercial general liability insurance covering personal injury and property damage caused by the developer during any on-site work
- Automobile insurance for vehicles owned or operated by the developer
- Worker's compensation in an amount required under the laws of the states where the developer provides the services
- Errors and omission insurance
- Excess liability insurance (also known as Umbrella Insurance)
- Cyber liability insurance if the software interacts with or stores any of customer's Personally Identifiable Information (PII), or otherwise be used in mission-critical applications where third-party intrusions or outages could cause significant economic loss

If you are providing any services at the customer's premises, you should make this provision mutual to cover any damage sustained to your property or injury to your employees.

Non-solicitation

This clause generally prohibits a party from hiring any of the other party's employees working on the project without that party's written consent during the term of the agreement and for a specific period of time after termination (usually 12 months). Larger companies usually seek to include the following exceptions to help make compliance practical:

- When an employee responds to a public advertisement or job posting not specifically targeted to the other party's employees
- When an employee was involuntarily terminated by the original employer prior to the solicitation
- When an employee voluntarily terminated their employment not less than a certain number of days prior to the solicitation (usually 90 days, but for specialized employees or key management you should set a period of 180 days)

Termination

The scope of termination rights and consequences of termination are critical for each party. Developers should take great care in drafting and negotiating the customer's termination rights narrowly, with particular focus on the following:

- Limiting the customer's termination triggers to material breach and insolvency (and agreeing to termination for convenience only if the customer agrees to pay a termination or exit fee)
- Defining the scope of licenses (if any), termination assistance obligations and either terms that survive the termination

Customer Responsibilities

To help perform the services in a complete and timely manner, developers usually include a provision detailing the customer's obligations to cooperate with and to assist the developer during the project. Examples include:

- Designating a team project coordinator who is knowledgeable about the project and authorized to make binding decisions for the customer
- Access to the customer's staff, facilities, working space, and equipment
- Access to computers, software, data, and customer information (even if operated by a third party for the benefit of the customer)
- Securing any necessary third-party authorizations to undertake the services
- Backing up all data, files and information prior to the commencement of any work, and assuming sole responsibility for this content
- Preparing the customer's systems for the implementation and deployment of the software

In situations where the customer uses or allows access to the software outside of the U.S., the developer should include a provision requiring the customer to comply with all applicable U.S. export laws and regulations. In addition, consider including a requirement for the customer to comply with all import/export laws of any foreign jurisdictions when using or allowing access to the software.

Source Code Escrow

Parties to a software development agreement usually execute a source code escrow agreement to protect the customer in the event the developer is unable to continue to develop, or later to support, the software. Under this agreement, the developer provides copies of the software source code to a third-party escrow agent as it is being developed. For a more detailed discussion on source code escrow arrangements, see Software Source Code Escrow Agreements: Drafting and Negotiating the Agreement.

Sonia Baldia, Partner, Kilpatrick Townsend & Stockton LLP

Sonia Baldia brings business and technology savvy to her global practice, which encompasses U.S. and international commercial, transactional, and intellectual property (IP) expertise across multiple industries including life sciences, banking and finance, healthcare, energy, information technology (IT), manufacturing, and software. She advises on a wide array of sourcing, technology, and other commercial transactions and helps companies navigate legal issues raised by data, emerging technologies, and digital transformation, both on the buyer and provider side. Sonia routinely advises clients on IP strategy, management, and monetization arrangements, leveraging her technology background and registered patent attorney credentials.

In addition to her U.S. bar admissions, Sonia is also qualified to practice law in India and she leverages that combined experience on behalf of clients in India-related matters.

Prior to rejoining the firm, Sonia was a partner in the Washington, D.C. office of an international law firm where she was part of its technology, IP, international commercial, and India practices. Sonia has also served as a consultant to the U.S. Agency for International Development (USAID) and the U.S. Department of Commerce in Washington, D.C. where she advised foreign governments on IT, telecom and IP-related development projects. She has also served as associate professor of law, teaching courses in IP, technology transfer, and corporate law.

Sonia was ranked in 2022 and prior years by *Chambers USA: America's Leading Lawyers for Business* in Technology & Outsourcing and in 2020, she was recognized for her expertise in outsourcing deals involving India and her broader technology expertise. She has also been consistently recognized by *Legal 500* amongst the leading practitioners in its technology, media, and telecom outsourcing category (2009-2022; and as a "New Generation Partner" in 2020 and a "Leading Lawyer" in 2021-2022). Sonia was recognized in *The Best Lawyers in America*® for Information Technology Law in 2022 and 2023. She is a frequent speaker and writer on digital transformation, global sourcing, IP, and technology topics and she has authored many articles and book chapters.