MARQUETTE
UNIVERSITY
LAW SCHOOL

Contract Drafting Sample Answer

Eric Goldman

May 2005

1. QUESTION #1.

This was a hard question for students. I'm not entirely sure why. As you'll see, from my perspective, this question was deceptively simple. There are just a few issues with the contract, but they are pretty significant. However, very few student answers to Q1 even remotely resembled mine. A lot of answers contained cosmetic or "fluff" responses, presumably because more substantive responses simply weren't apparent.

Acceptance

The contract lacks an acceptance provision, so BC's deliverables will be governed by the UCC default rule of perfect tender. Because software is inherently buggy, it cannot satisfy the perfect tender rule. As a consequence, customers will have a virtually unlimited right to reject the software, terminate the transaction and sue for damages attributable to an imperfect tender. As a further consequence, BC's revenue recognition will be pushed back until after the customer gives its acceptance under the UCC default provisions.

This is a mission-critical omission from the contract, and BC must fix it. BC should include a contractual acceptance provision that overrides the UCC defaults. This acceptance provision will have three principal components:

- a mechanical/administrative process for evaluating/inspecting deliverables
- a substantive standard for evaluating the acceptability of the deliverable
- a statement of sole and exclusive remedies for a non-conforming deliverable

From BC's standpoint, the best outcome would be to skip the acceptance process altogether and move the customer into the performance warranty (or, better yet, skip the performance warranty and move the customer into maintenance). One approach would be to state that the customer accepts all deliverables upon delivery. A more customer-palatable approach would be to give a short inspection period, a generous substantive standard (i.e., material conformance to specifications), and vendor-controlled remedies (such as repair/replace or, failing that, BC has option to terminate).

License Grant

The license grant (and some ancillary provisions, like the assignment clause) are internally contradictory. I correct some inconsistencies in Question 2. At minimum, there is an inherent tension between the restrictions on the customer's right to "transfer" the software and the fairly

expansive permission to assign the contract to acquirers. The latter may be problematic if it costs BC future sales (or if their pricing does not adequately respond to changes in the customer's identity). As a result, BC may want to restrict contract assignment more tightly.

The license grant "use" is confusing for a copyright license. As a practical matter, the current drafting simply is too imprecise to delimit the customer's behavior.

Finally, the license grant is incorrect because customers can use contractors to modify the software. In theory the inaccurate restrictions should be favorable to BC (i.e., BC can use the restrictions to go after customers whenever it wants). However, I don't think that's the most likely outcome. Instead, I expect high transaction costs negotiating with customers who spot the inaccuracy, and I further fear that the imprecision creates opportunities for judges to interpret the contract liberally given that it does not reflect the parties' understandings. I fix this in Q2.

Confidentiality

We might question why BC thinks its software is confidential (see below). Assuming, however, that BC views its software as a trade secret, then the 5 year restriction (Section 8(C)) guarantees that BC will lose those trade secrets (i.e., the trade secrets will drop into the public domain) at the end of that 5 year window. Instead, if BC wants to treat its software as a trade secret, it should not time-limit the confidentiality obligations.

Furthermore, the drafting of the provision isn't very explicit as a trade secret license. If the goal is to retain the software as a trade secret, the provision should explicitly state that the customer can't use or disclose the software.

Risk Allocations

Warranty Disclaimer. The warranty disclaimer is missing magic words—either "as is" or express reference to the implied warranty of merchantability. As a result, this language may not disclaim the default UCC implied warranties. BC needs to use the appropriate magic words to ensure the disclaimer works properly.

Dollar Cap. The dollar cap should be stated in all caps lettering. Otherwise, there is a risk that a court will find that it was not sufficiently conspicuous to give it effect.

Consequential Damages Waiver. The waiver of consequential damages should expressly reference negligence claims. Otherwise, the waiver may not apply if the customer is able to structure a claim as a tort/negligence claim.

Grantback License

Because the customer can modify the software to some degree, there is a potential risk that the customer will develop and then patent modifications that would limit BC's future development path. The importance of this point depends on how extensively customers can modify the software. The tools given to customers to make modifications may be so limited that customers

can never really develop anything of consequence. Otherwise, BC should consider getting customers to license back the IP rights to their improvements to prevent the blocking patents. Note that getting such a grantback license may be a tough sell.

2. QUESTION #2.

I believe Sections 1 and 8 and Attachment A should all be combined in a single section. Rather than trying to work with the language, which is cumbersome and (in some cases) ill-conceived, I have written this language from scratch.

A threshold question: is the software really meant to be kept as a trade secret? I question if BC really can pull this off; vendors frequently disclose software in ways that would drop it out of trade secret protection. However, I've drafted this assuming that it should be characterized as a trade secret. In that case, as discussed above, there should be no time limit on the disclosure restrictions; because a trade secret could last perpetually, so should the restrictions protecting the trade secret. In the termination section, I would add a provision stating the effects of termination, and I would survive Section 1.2(b) accordingly. [As discussed below, in all likelihood, the licenses survive termination anyway, so we might very well survive all of Section 1 post-termination.]

I have also omitted any idea that BC can change license terms during the duration of the contract. As discussed below, I don't think the contract means what it says.

My proposed Section 1:[1]

> **1.1 License.** Subject to the Agreement's terms and conditions, BC hereby grants to Customer a non-exclusive license to:
>
> **(a)** permit designated individuals (for whom Customer has paid the applicable license fee) to use, copy and create derivative works of the Development Software to customize Customer's Software deployment;
>
> **(b)** use and copy any such derivative works in conjunction with permitted uses of the Deployment Software;
>
> **(c)** use and copy the Deployment Software in connection with the operation of Customer's website to track up to the number of profiled users for which Customer has paid the applicable license fee;
>
> **(d)** use and copy the DCCs up to the number of copies for which Customer has paid the applicable license fee; and

[1] A more pedantic approach would say, before each instance of "use," "under BC's patents and trade secrets" and in front of other verbs "under BC's copyrights." I don't think this extra layer of detail is necessary but it would more explicitly segregate "use" from being interpreted under copyright law.

(e) use, copy and internally distribute the Documentation as reasonably necessary to exercise the foregoing licenses.

1.2 Restrictions.

(a) Contractors. Customer may permit contractors to exercise the license rights in Section 1.1(a) so long as each contractor enters into a written non-disclosure agreement with Customer (and names BC as a third party beneficiary of that agreement) prohibiting the contractor from using or disclosing the Development Software for any reason other than to perform development services for Customer.

(b) Non-Disclosure. Except as provided in Section 1.2(a), Customer may not disclose the Software or Documentation to any third parties, except that Customer may publicly display Software-generated web pages in accordance with the ordinary operation of the Software and Customer's website.

(c) Other Restrictions. Customer may not reverse engineer the Software, allow third party outsourcers to operate the Software on the Customer's (or any third party's) behalf, or operate the Software for any third party's benefit.

Comments:

- I did not say "non-transferable" or "non-assignable," deferring both topics to Section 10(B) (whichever direction BC wants to go). I could include the limitation "non-sublicenseable" although that's not literally true, as permitting contractors to make derivative works is, in fact, a sublicense.
- The terms "Development Software," "Deployment Software," "DCC," "Software" and "Documentation" all should be defined in a definitions section (although we could include the definitions in each subpart).
- I separated out the license grant for the Documentation.
- I took out the reference to "internal purposes." I addressed the issue in Section 1.2(c).
- I don't say what IP rights I'm withholding; if I've drafted my affirmative IP licenses properly, I don't need to state the converse. Nevertheless, many drafters would state both the negative of the license and disclaim any implied licenses out of an abundance of caution. I can't say not to do this, but I don't know that it's really necessary if you've drafted the licenses properly.
- It would be easy to add another subsection stating that BC and its licensors retain title to the Software and Documentation. We could also move the requirement that copies retain copyright/proprietary notices into this section.[2]
- Because the customer can copy the software, I saw no reason to limit the number of copies made for archival/backup purposes.

[2] We didn't discuss this in class, but these restrictions may no longer be necessary. They were essential prior to the Berne Convention, because publishing a copyrighted work without a copyright notice dedicated the work to the public domain. Today, deleting a copyright notice violates both 17 U.S.C. §506(d) (if there is fraudulent intent) and 17 U.S.C. §1202.

My word count for Section 1: 272. Total word count for prior Sections 1 (180 words) and 8 (196 words) and Attachment A (100 words): 476. The comparison isn't totally fair because I do offload some words to other sections. On the other hand, I could save a few more words if I kept at it.

3. QUESTION #3.

Customer Control of its Own Fate/Vendor Hold-Up Games

Let's focus on the customer's big-picture objectives. The customer wants to control its own fate and avoid opportunities for vendor hold-up games. We can identify some problems with the contract based on these two core objectives in mind.

Source Code and Maintenance

The contract does not provide the customer with the opportunity to obtain a complete copy of the source code. Ideally, the customer would like to get that copy on day 1 so that the customer can withstand any bad vendor behavior. However, at minimum, the customer would like some mechanism to get the source code if the vendor engages in bad behavior. The typical solution would be a source code escrow. From the customer's standpoint, this is a poor substitute for getting its hands on the code directly, but it does offer something more than the customer is getting under the current contract.

The contract says that the vendor can simply stop providing maintenance using a 90 day termination for convenience clause. This means that the vendor could stop maintenance but not provide tools to the customer (i.e., source code) to take over the maintenance itself—in other words, leaving the customer completely stuck. Why does the vendor need the right to stop maintenance at its convenience? The customer needs the vendor to stay in the game, so this clause should be struck. Further, the customer also should say that if the vendor stops maintenance for any reason (convenience or otherwise), the customer gets the source code so that it can take over maintenance itself.

Termination of License Rights

Currently, the contract says that the customer loses its license if the vendor terminates the agreement. From the customer's standpoint, losing the license is a very harsh remedy—it means the end of the customer's project, and the loss of all ancillary investments in addition to the amounts paid to the vendor.

Let's consider an example to see how this plays out. Assume that the customer pays the entire $600,000 initially contemplated. Then, in a subsequent decision, the customer procures an "add-on" option priced at $10,000. If the customer fails to pay that $10,000, the vendor's license termination remedy means that the customer loses the entire $600,000 investment. A more acceptable approach would say that if the customer fails to pay the $10,000 for the add-on, the customer loses the license to the add-on, but this failed purchase does not disturb previously completed transactions.

As a result, the customer wants to limit the "loss of license" as a remedy to a small number of circumstances. Better yet, the customer should negotiate that the licenses are perpetual and irrevocable to the extent that the customer has paid for those licenses.

Particularly disconcerting are situations where the vendor can unilaterally decide to terminate the contract and, accordingly, terminate the customer's license rights (see, e.g., Sections 5 and 7). This is objectionable to the customer. The customer wants the vendor to do everything in its power to preserve the customer's rights to use the software, and in no case does the customer want the vendor to have an "easy out" by terminating the contract. Loss of license should be a remedy of last resort, and every place where the vendor can yank the license needs to be carefully vetted.

Right to Use Contractors

The facts state that the customer contemplates hiring independent contractors to modify the software in the future. This is important because it provides a check on the vendor's pricing of professional services; if the customer has to go back to the vendor for any future changes, the vendor can charge monopoly prices to the customer. The contract inconsistently addresses the customer giving contractors access to the software. Section 8(B) references "contractors" in a backhanded way that contemplates that contractors can access the software, but there are multiple redundant restrictions both on modifying and disseminating the software that may limit customer's ability to do so. The customer will want to expressly permit modifications (see below) and the right to use contractors to do so.

Pricing of Custom Development

The structure of Attachment C, Sec. 2 is not customer-favorable. Like any "time-and-materials" contract (which this is), the customer is at the vendor's mercy. If the vendor is slow/inefficient, the customer has no recourse.

The dollar cap doesn't help. Optically, it suggests that the vendor should be motivated to complete the work at the stated dollar cap. However, in practice, if the vendor promises to complete the work 100% for the dollar cap but is only 75% complete when the vendor hits the dollar cap, what can the customer do? The customer didn't get its desired result (100% completion for the stated cap). At that point, the customer can (a) pay the vendor more money to complete the work, or (b) take over the project and complete the work itself (or use a contractor to do so). All of these solutions cost the customer more money. As a result, the vendor can play a hold-up game (or simply sandbag), using the customer's residual obligation to pay to complete the project to extract more cash from the customer.

Therefore, the customer would prefer to pay a fixed fee for completed projects, with a timetable for deliverables, a standard for acceptance of those deliverables and onerous remedies for non-conforming or late deliveries.

Other License Issues

Right to Modify

The facts state that the customer can modify the software to some degree using the development software. However, the contract says that the customer cannot modify the software, and if it does, the customer is in breach of the contract and loses the maintenance and indemnity (see below for further discussion of that issue). So the agreement is internally inconsistent. As discussed above, the agreement should be modified to permit the customer to modify the software in accordance with the parties' expectations. Further, maintenance and indemnity should not be affected by permitted modifications.

License Term Changes

Attachment A says that the vendor can unilaterally change license terms with 30 days notice. This cannot be literally true; if a customer pays for software based on the right to make 15 DCC copies, I don't think the vendor can amend its licensing practices to say that those 15 copies are now no longer properly licensed. Instead, what I *think* this means is that the vendor can change the license terms applicable to *future orders* with 30 days prior notice—i.e., if customer orders 15 DCCs on a per-copy basis, then customer can keep those copies, but with a 30 day notice, BC can change the terms so that future DCC orders are paid based on number of offers or number or users or some other parameter.

Customer may still find the revised expression objectionable, as the customer is locked in once the customer makes the initial order, so future changes to the license terms could act as a hold-up game. For example, after customer orders the first 15 DCCs, BC could change the licensing practices so that all future DCCs cost $10M per licensed user—in effect, making it impossible for the customer to order new DCCs.

Therefore, the 30 day right to change licensing practices should be deleted. Further, in the initial order, customer might limit future price changes.

Internal Use Restriction

I think this restriction is inherently ambiguous; even more so in the context of software intended to be used on the web. Whatever it means, I don't think it is consistent with the customer's objectives and, on that basis, should be stricken.

Title to Customizations

Under Attachment C, the vendor retains title to the customizations (this is the standard operation of default copyright and patent rules). This may be OK with the customer. For example, the customer may have to "pay to play" within the vendor's system—it will get the benefits of other customizations the vendor has made for other customers, so it has to contribute to the vendor's storehouse to get those benefits. Further, if the vendor is going to incorporate those customizations into the vendor's core offering and provide maintenance on those customizations,

the customer may end up better off. Finally, the customer may have to pay the vendor more to own the customizations, and the customer simply may not value ownership that highly.

However, the customer needs to consider the implications of the vendor having title. This means two things:

- The vendor, and not the customer, will get any economic upside from the customizations, even though the customer putatively paid for their creation
- The vendor can freely provide the customizations to the customer's competitors, so they will get the benefits of the customer's work at a potentially cheaper price

If these are problematic, then the customer will have to negotiate a different ownership/license structure to the customizations, or the customer will have to prepare the customizations itself (or hire a more pliable contractor to make them).

Confidentiality

The one-way confidentiality restrictions (protecting vendor but not customer) are not inherently objectionable. If we can't think of ways that the customer would be disclosing confidential information to the vendor, then we don't need a mutual clause.

However, we have some facts that suggest that the customer *will* be disclosing confidential information to vendor. At minimum, note that in Attachment B, Sec. 1(E), the vendor will have the right to troll around the customer's computer network. Further, Attachment C, Sec. 7 contemplates that the vendor's employees will be housed in the customer's facilities, working on the customer's computer network, for some period of time. In each of those cases, the customer may be legitimately concerned that the vendor will have access to customer's confidential information. Therefore, I think the contract should contain provisions to protect the customer's confidential information. I would use a mutual NDA clause like the one contained in the reader.

The vendor's confidentiality clause raises some issues. From the customer's standpoint, it's not clear what it means for the "software" to be confidential. But assuming that some parts of the software should be deemed confidential, then 2 changes are necessary. First, the customer needs the right to permit contractors to access the software (see above). Second, the confidentiality obligations should stop when the software is no longer a trade secret, such as if the software is publicly disclosed by the vendor without confidentiality restrictions. Otherwise, the contract can cause the customer to end up in a worse position than the public at large.

Risk Allocation Issues

Consequential Damages Waiver/Dollar Cap

In general, the consequential damages waiver and dollar cap have the possible consequence of leaving the customer with incomplete remedies (and don't give the vendor full incentives to do the right thing). For example, if the website is offline and the vendor should bear the resulting losses, the consequential damages waiver or dollar cap may unnecessarily limit the remedies. This requires some discussion with the customer to understand where the customer is relying on the vendor. In those cases, I would want to spell out the precise remedies for those situations,

which may involve removing the associated obligations from the consequential damages waiver or dollar. In some cases, it may be appropriate to liquidate the damages.

A number of you made the foregoing point and then simply suggested to eliminate the consequential damages waiver or the vendor's dollar cap. This solution did not impress me! Removing the language reverts the contract back to defaults, which may or may not cover the situation adequately. Plus, the vendor will fight much harder on removing the blanket restriction than a targeted removal. If you *really* care about the remedies, don't rely on the defaults blindly—build the remedies that you want.

Also, I didn't like the duplicative but inconsistent dollar caps between the main agreement and the two attachments. These should be harmonized/combined into a single dollar cap provision. Otherwise, as currently written, the dollar cap could be much smaller than contemplated by Sec. 6 (note that the dollar caps in the attachments "overwrite" that dollar cap).

IP Indemnity

The indemnity offered by the vendor is pretty narrow. While I didn't like the geographic restrictions, I really didn't like that the indemnity does not apply if the software is "modified." We contemplate that the software will be "modified" (both by the vendor, the customer and the customer's contractors), so does this mean that the vendor is, in fact, offering no indemnity? At minimum, the wording should be tightened up to clarify when modifications do not affect the indemnity.

Several of you commented that the IP indemnity should clearly state that it covers only third-party claims. As a practical matter, that's how the clause works as written. The customer can't sue itself for violating IP rights it owns, so the only viable claim of IP infringement would come from third parties.

<u>Other Provisions</u>

Attachment B, Sec. 1(H)

Read literally, if the customer is experiencing a "bug," the vendor can simply send new documentation showing that "bug" as a "feature." The customer wants the vendor to fix the bug, not just issue new documentation. This provision should be struck.

Section 4(B)

Why would the customer agree to display this logo on a splash screen? This is free advertising for the vendor. Furthermore, in a web context, it's not always clear which screen would qualify for containing the logo. The customer should strike this provision.

4. COMMENTS ON STUDENT ANSWERS.

Termination

Many of you commented that the customer had no termination rights. This was generally not correct—some of you simply seemed to miss the termination for convenience in Attachment B, Section 2—but let's assume that was literally true. What termination rights does the customer *need*? Those of you who raised this issue rarely answered this more fundamental question. The vendor has offered very few covenants that it could breach, and the customer has very few ongoing obligations that it would like to cease other than maintenance, which the customer can cancel for convenience.

Many of you commented that there was no fixed end date for this contract. Some of you tried to characterize this contract as an "evergreen" contract; that's not literally true, although the maintenance provision is structured that way. Instead, this is a perpetual contract, and there's nothing inherently wrong with having the contract extend perpetually so long as you look at each obligation to see if it makes sense to last in perpetuity.

License

A number of you commented on the "internal operations" language. In this context, it's completely vague. Some of you proposed to replace it with "internal business operations"—this isn't an improvement!

A number of you complained about the "may not" language in Section 1(C), preferring to change it to "shall not." While "shall not" works, I think "may not" works as well—"may" is permissive language, but "may not" isn't.

Maintenance

Some of you complained about the fact that customer would have to "catch up" maintenance payments to restart maintenance (if it ever gets off the program). I do think this is an odd concept. The rationale is that the vendor has had to invest lots of money maintaining and enhancing the software, so if the customer ever wants the latest version of the software that is the product of those investments, the customer should have to pay the appropriate license fee to get that version. I'm not sure I ever believed that rationale, but this provision is fairly common in software maintenance agreements.

As a practical matter, the contract can say whatever it says. If the customer isn't on the maintenance program and wants to get on it, the parties can negotiate financial terms then. I would suspect that many customers will be able to bypass this catch-up payment simply by refusing to pay it. Many vendors just want as many customers as possible on maintenance (for the predictable cash flow streams), so the vendor isn't likely to be too aggressive demanding the catch-up payment. As a result, I treat this clause as sales-y language. It tries to "scare" the customer into thinking there's big financial negatives to skipping maintenance, even if that would not be literally true.

Acceptance/Warranty

Some of you confused the warranty with an acceptance process. This was just wrong—*warranty and acceptance are completely different concepts with non-overlapping rules*. If you're the customer and the contract is silent about the acceptance process, this is generally *good news*—you are going to get the perfect tender rule, hooray! It's hard to imagine a more customer-favorable rule. If you're a software vendor and the contract is silent about acceptance, *you're screwed*. Warranty disclaimers or a limited performance warranty will not help you. You need to address acceptance and get the customer out of that process and into the warranty/maintenance period.