What is a Software Defect?

***In Press*, Software QA Magazine**

One discussion that plagues development groups is how serious a bug is. Can we call it a feature? Does it have to be fixed now or can it wait until the next release? What are the consequences if we ship it?

This article looks at this issue from a different angle:

> *How should the law determine whether a bug is serious enough that the customer should be entitled to cancel the contract, return the software, and demand a refund?*

I've been asked to write the first draft of language to include in a new law. More than any other piece of the new statute, this section will affect the day-to-day interactions within product development groups. So, I'm asking you to review this proposal and send me your comments, at kaner@kaner.com.

Background: UCC Draft Article 2B

A committee of the National Conference of Commissioners on Uniform State Laws (NCCUSL) is drafting a new Article (2B) for the Uniform Commercial Code (UCC). UCC Article 2 is the Law of Sales, which currently governs the licensing and sales of most software products. In the future, Article 2B will govern the sale and licensing of all software products (and of most other types of information products, such as cable TV).

For the last year, I've been attending meetings of the Article 2B Committee, arguing that customers need more protection than the drafts of Article 2B have been providing. [1] One issue that keeps coming up is the severity of bugs.

A software defect is a "material breach" of the contract for sale or license of the software if it is so serious that the customer can justifiably demand a fix or can cancel the contract, return the software, and demand a refund. The American Bar Association's Committee on Computer Programs calls these *Material Bugs*.[2] I'll use the same phrase.

If a defect is not "material" then the customer is stuck with the program and is probably entitled to, at most, a partial refund.

How should we decide whether a bug is serious enough to be called "material"?

The July (and previous drafts) of Article 2B

Here's the proposed definition of a material breach of the software contract in July, 1996 (and in several previous drafts).

```
2B-109 (July) (b)  A breach is material if the contract so provides
or, in the absence of express contractual terms, if the circumstances,
intent of the parties, language of the contract, and character of the
breach indicate that the breach caused or may cause substantial harm
to the interests of the aggrieved party, or if it meets the conditions
of subsection (c) or (d).

(c)     A breach is material if it involves:
   (1)   knowing or grossly negligent disclosure or use of confidential
      information of the aggrieved party not justified by the license;
   (2)   knowing infringement of the aggrieved party's intellectual
      property rights not authorized by the terms of the license and
      occurring over more than a brief period; or
```

(3) an uncured, substantial failure to pay a license fee when due which is not justified by an existing, colorable dispute about whether payment is due.

(d) A material breach occurs if the aggregate effect of the nonmaterial breaches by the same party satisfy the standards for materiality.

Look closely at (c), which defines a material breach by listing examples. All of these protect the publisher from breaches of the contract by the customer. For example, the customer is in trouble if s/he (1) discloses the publisher's confidential information, (2) pirates the software, or (3) doesn't pay for the program.

None of these helps the customer argue that a given bug is material. In fact, because the statute lists a series of things that are "material," a common rule of statutory interpretation would suggest to lawyers and judges that nothing that is not in the list can be material. So, maybe no bug can be so important that it materially breaches of the contract.

The September and November Drafts

The September, 1996 draft added a further clause to the list. A breach is also material if it involves:

a failure to perform in a manner consistent with express performance standards;

In September, I asked Ray Nimmer (the Reporter of the Committee and the lead author of the Article 2B drafts) what this means. What is an "express performance standard?" Is it just a precise, verifiable statement about the program that the seller makes to the customer? Can the customer get a refund if the program fails to match the written documentation?

He said, no, that's not what he meant. He was referring to the specific promises that were contained in a negotiated contract, that said that the program *must* do specified things or have specified characteristics.

So I said that this is unfair. What about customers who don't have negotiated contracts that specify all the right things? What about mass-market customers? (Mass-market software is made available to the general public, in a way that is not customized for individual customers. For example, Microsoft Word is mass-market software.) If the program doesn't work, don't these customers have any recourse?

Ray said that this is a genuine problem. But he doesn't know how to solve it. *How should we define material defect when the bug is not a direct nonconformity with an important part of the specification?*

Then he said to me that I was the person who kept raising these issues, so maybe I could take a crack at drafting the definition.

And I promised that I would. But I need your help.

How Should We Define a Serious Defect?

This paper provides a first, working draft of a definition of a material bug.

Failure to conform to specifications is a common theme in the legal books, but many of the software development contracts provide vague, incomplete specifications that will change over time without being updated in the contract itself. Whether the specification or user documentation addresses the following issues, all of them should be considered material bugs, shouldn't they?

- the product doesn't work at all
- the disks are blank
- the product caused substantial harm to the property or the business of the licensee
- the licensor supplied a product that lack promised or advertised features or capabilities

- the software deprives the customer of a key benefit that she reasonably expected (for example, anyone would expect a word processor to be able to print and it's unreasonable if a particular one can't provide that benefit.)
- the customer spent so much time, effort, and money dealing with a bug that the customer's costs exceeded the cost of the product. (If you think that this isn't a serious enough loss to the customer, what if the customer's losses run at ten times the cost of the software, or one hundred times the cost? At some point, the amount of the losses caused by the software become excessive, doesn't it?)

In the course of writing this article, I reviewed a few dozen books, articles and standards on software quality and software defects, trying to understand how different segments of the software industry deal with errors. I also reviewed many court cases and contract books, trying to understand how attorneys currently deal with software errors. These materials provided lots of data, but not a solid framework. I think that the framework provided here is sensible, useful, fair, and workable for software developers as well as for legal practitioners, but it needs review.

Reflecting the Relationships Between Licensor and Licensee

The licensor is analogous to the seller in a traditional sale. Under Article 2B, what is sold in the typical software transaction is a license to use the software, rather than a copy of the software. The licensee is the customer, who buys the license.

I think there are four common classes of software transaction:

1. ***The customer writes the specifications and requirements and asks the developer to write a program as specified.***

   In my view, the software developer meets its [3] obligations if it writes a program that meets the specifications. If the specs say "2+2=5" then the program does not breach the software contract if it generates the wrong answer (5) whenever it adds 2+2. Let the specifier beware.

2. ***The developer writes the specifications and requirements, in preparation for custom development, but the customer is sophisticated.***

   If the customer is a computer expert, then it is able to review the requirements and specifications just as well as the staff of the developer. The customer is also probably in a better position to understand its own requirements than the developer. Therefore it is reasonable to hold this customer accountable for reviewing the specifications.

   Article 2 of the current Uniform Commercial Code defines a "merchant" as follows:

   ```
   2-104 (1) "Merchant" means a person who deals in goods of the kind or
   otherwise by his occupation holds himself out as having knowledge or
   skill peculiar to the practices or goods involved in the transaction
   or to whom such knowledge or skill may be attributed by his employment
   of an agent or broker or other intermediary who by his occupation
   holds himself out as having such knowledge or skill.
   ```

   ```
   2-104(3) "Between merchants" means in any transaction with respect to
   which both parties are chargeable with the knowledge or skill of
   merchants.
   ```

   Article 2B uses essentially the same definition:

   ```
   2B-102 (26) "Merchant" means a person that deals in information of the
   kind, a person that by occupation purports to have knowledge or skill
   peculiar to the practices or information involved in the transaction,
   or a person to which knowledge or skill may be attributed by the
   person's employment of an agent or broker or other intermediary that
   purports to have the knowledge or skill.
   ```

   When Microsoft buys Apple computers, it is a merchant. When a large, local hospital buys a bunch of Apples, it might be a big business, but it is not a merchant.

3. ***The developer writes the specifications and requirements, in preparation for custom development, but the customer is not sophisticated.***

   When doctors, dentists, insurance brokers, small grocery store owners, and other small business people buy software, they have no clue how to specify the software, no clue how to evaluate a requirements document, no clue how to test the software, and no clue how to cost-effectively find a consultant who has these skills. In common computer parlance, these customers are called *Clueless*.

   In UCC parlance, these customers are *non-merchants*.

   These customers rely on the knowledge and experience of the developer. If the developer makes errors in defining the requirements or the specifications, which result in serious errors in the operation of the program, this is the developer's bug, not the customer's.

4. ***The developer writes a mass-market product. The customer has no input into the design or development of the product.***

   In the mass-market case, design errors belong to the developer, not the customer. Internal specifications that were used during development are largely irrelevant to the customer. The end product works in a reasonable way, as advertised and as documented, or it does not.

The Proposed Statutory Language

This proposal modifies Section 2B-108 of Article 2B. I am a novice at drafting statutes. The language will be cleaned up during the Article 2B review process. The proposal expresses my sense of the fundamental differences between these transactions.

SECTION 2B-108. BREACH OF CONTRACT.

(a) Whether a party is in breach of contract is determined by the terms of the agreement and by this article. Breach occurs if a party fails to perform an obligation timely or exceeds a contractual limitation.

(b) A breach of contract is material if the contract so provides. In the absence of express contractual terms, a breach is material if the circumstances, including the language of the agreement, expectations of the parties, and character of the breach, indicate that the breach caused or may cause substantial harm to the interests of the aggrieved party, that the injured party will be substantially deprived of the benefit it reasonably expected under the contract, or that the breach meets the conditions of subsection (c), (d), (e), (f) or (g).

(c) If the licensee provides the specification documents that are incorporated in the contract, then a breach is material if:

   (i) the software fails to perform in conformance with and in the time required by express performance standards or specifications;

   (ii) the software fails to perform in conformance with the specifications and this failure either deprives the licensee of a significant benefit of the product or results in costs to the licensee that exceed the price paid for the software;

   (iii) where the specifications are silent, the software's performance is unreasonable and it results in costs to the licensee that exceed the price paid for the software. The licensee has the burden of demonstrating that a reasonable licensor would consider the software’s performance to be unreasonable.

(d) If the contract is between merchants, and it contains specification documents, then a breach is material if:

  (i) the software fails to perform in conformance with and in the time required by express performance standards or specifications;

  (ii) the software fails to perform in conformance with the specifications and this failure either deprives the licensee of a significant benefit of the product or results in costs to the licensee that exceed the price paid for the software;

  (iii) where the specifications are silent, the software’s performance is unreasonable and it results in costs to the licensee that exceed the price paid for the software. The licensee has the burden of demonstrating that a reasonable licensor would consider the software’s performance to be unreasonable.

(e) If the contract is not between merchants, and the licensor provides the specification documents that are incorporated in the contract, then a breach is material if:

  (i) the software fails to perform in conformance with and in the time required by express performance standards or specifications;

  (ii) the software fails to perform in conformance with the specifications and this failure either deprives the licensee of a significant benefit of the product or results in costs to the licensee that exceed the price paid for the software;

  (iii) the software fails to perform in conformance with the end user documentation or other documentation delivered to the licensee and this failure either deprives the licensee of a significant benefit of the product or results in costs to the licensee that exceed the price paid for the software;

  (iv) where the specifications and other documentation are silent, the software’s performance is unreasonable and as a result, it either deprives the licensee of a significant benefit of the product or it results in costs to the customer that exceed the price paid for the software. The licensee has the burden of demonstrating that a reasonable person would consider the software’s performance to be unreasonable.

(f) If the contract is for a mass-market license, then a breach is material if:

  (i) the software fails to perform in conformance with the end user documentation or other documentation delivered to the licensee and this failure either deprives the licensee of a significant benefit of the product or results in costs to the customer that exceed the price paid for the software;

  (ii) where the documentation is silent, the software’s performance is unreasonable and as a result, it either deprives the licensee of a significant benefit of the product or it results in costs to the licensee that exceed the price paid for the software. The licensee has the burden of demonstrating that a reasonable person would consider the software's performance to be unreasonable.

(g) A material breach of contract occurs if the cumulative effect of nonmaterial breaches by the same party satisfies the standards for materiality.

(h) If there is a breach of contract, whether or not material, the aggrieved party is entitled to the remedies provided for in this article and the agreement.

What Happens from Here?

By the time you read this proposal, I will have circulated it to the Article 2B Drafting Committee. They'll probably consider it at the January 10-12 Drafting Committee meeting at the Sofitel Hotel in Redword City, California. The next meeting of the Committee will be in Atlanta from February 21 to 23, 1997. I will compile comments that people send me, and will summarize them for this meeting. You can also attend either meeting yourself. Few of the attendees are non-lawyers, but you are welcome to speak if you have something informative to say.

This process will continue for a few more months (four meetings are scheduled in 1997), probably resulting in legislation that is introduced in the state legislatures in 1998. Whether you or I participate in this process or not, the result will include rules that govern software quality, laying out the ground rules under which we decide whether bugs are features and whether they need to be fixed. We can influence the process.

To read the latest draft of Article 2B, and to send comments directly to Ray Nimmer, the Drafting Committee's Reporter, visit the Article 2B home page at `www.law.uh.edu/ucc2b`.

---

[1] Some readers of this magazine might have read my previous paper on Article 2B: Kaner, C. "Uniform Commercial Code Article 2B: A New Law of Software Quality," *Software QA*, Volume 3, #2, 1996, p. 10.

[2] American Bar Association, Section on Patent, Trademark, & Copyright Law, Committee on Computer Programs *Model Software Licensing Agreement*, 1992.

[3] In legal writing, it is common to use the word "it" instead of "he" or "she" whenever the being (here, the software developer) is likely to be a corporation rather than an individual human.