



► Contract Acceptance: What Is It?

Richard J. Cichelli, SCS President

Contract acceptance is a process which begins when a customer is using or is capable of using an application in production. Its completion signals final payment. Here is how the process should work.

Once an application goes into production, customers begin making punch lists detailing the issues they have with the application. Vendors wish to close the contract and receive final payment. If there is a misunderstanding about contract acceptance, conflicts, often severe ones, can occur.

Customers strive to have every bug fixed and every wish fulfilled. The corresponding vendor position is that "use constitutes acceptance." Neither party is likely to ever be happy if these are their respective positions.

Once an application is in use for its intended purpose, the customer and vendor begin the process of contract acceptance. Contract acceptance is not a state of mind and should not be confused with "satisfaction."

The basis for contract acceptance is the acceptance criteria listed in the contract. Each numbered acceptance criteria item describes some capability of the application. An application meets contract acceptance for an item if the vendor can exhibit the required capability in the application. Exhibiting all the capabilities listed in the acceptance criteria completes contract acceptance.

Note: Agreements can explicitly include acceptance criteria, or they can be defined by the feature list provided in the product's literature. The manuals for the application can also implicitly serve as acceptance criteria. Most vendors will agree to provide a system which is in "substantial conformance to its documentation." During the acceptance process, a number of issues may arise. Explicit acceptance criteria are unlikely to list every

necessary capability of the application. They are also not likely to include items on the punch list.

Acceptance criteria need to be precise and unambiguous. For example, you might have a series of acceptance criteria that say "the system can hyphenate and justify classified ad copy" and "the system can hyphenate and justify a one column by four inch classified ad in under two seconds during the hour of peak system usage at least 95% of the time."

For a customer to say an application is not accepted requires producing a list of one or more acceptance criteria items for which the application is incapable of performing the required function. Working with the acceptance criteria list, the customer and the vendor can usually filter the list to a much smaller subset of open items. The resolution process may involve having the vendor add the capability to the application (if not present) or provide instruction detailing how to invoke the capability if otherwise. Ambiguous or conflicting requirements should be resolved by compromise. For example, high security and transparent access are often mutually incompatible.

Customers should relinquish open acceptance criteria for which they have no current or expected future need.

Since contract acceptance is about the acceptance criteria, it is best if unresolved acceptance items are identified specifically by number. It helps if they are listed in the order of the acceptance criteria.

As vendor and customer review the list, it is the responsibility of the vendor to demonstrate the capability in the system. The customer can, of course, acknowledge that they know and use specified features, thus saving review time. The vendor might also show that the capability is irrelevant or its specification is ambiguous or insufficiently precise. The customer can counter by showing the requested capability is relevant and isn't present in a usable form.



Should the customer expect explicit acceptance criteria to include every necessary and sufficient capability? Highly generalized criteria are subject to multiple interpretations. Alternatively, listing every desired capability in clear, unambiguous detail is a task of the magnitude of developing the application itself. Well-written acceptance criteria detail a sampling of generally important features. They should also include those of particular importance to the customer. The most critical of these are related to the strategic objectives of the project, e.g., those designed to yield return on investment (ROI). It is usually better if those acceptance criteria capabilities not currently in an application are acknowledged by the vendor prior to contract signing.

A punch list is the set of those items which are needed for satisfactory use of a system. Customers make them as they become familiar with an application during use. Bugs, performance issues, localizations, enhancement requests all end up on punch lists. Few applications in daily use are free from punch list items.

Only punch list items which can be related to specific acceptance criteria are relevant to acceptance.

Suppose the vendor implements a new capability not envisioned in the acceptance criteria. Say the new functionality is delivered after contract signing, during installation but before acceptance. Is the customer entitled to have this work according to its desires? Only so far as it relates to the acceptance criteria in the agreement. What about bugs? It is an unfortunate truth that nearly all software has faults. Some faults cause failures. Neither vendors nor customers have sufficient

resources to remove all faults from major applications. Any fault for which there is no work-around and whose resultant failures significantly impact productive use of the application deserves careful attention, even if not part of acceptance.

It is the customer's responsibility to detail the steps necessary to create a failure. It is much easier to identify and fix bugs if they are presented in a focused context. Stating that a problem occurs is insufficient. Every aspect of the failure needs to be documented. Which user witnessed it occur? When does it occur? How can one make it happen again? Reproducibility is essential for resolution. Nonetheless, bugs are typically support issues, not contract acceptance items. The same holds for enhancements requests, wish list items, etc.

How does a customer know when he/she should be paying support? Let's say the application is in daily production and the customer is dependent on the vendor for support, paying for the support provided is appropriate. Surely when the customer uses the application without recourse to alternative methods of doing business it has accepted the application. While going on support is often tied to final acceptance, this can lead to situations where customers are getting the benefits of the application while insisting on free support. Having different criteria for warranty initiation, paid support initiation and final acceptance seems desirable.

When brought before an arbitration committee, the issue of contract acceptance is usually quickly resolved: The vendor is held accountable for providing services in a professional manner and the customer is required to pay for what he or she is using.

Richard J. Cichelli is president of Software Consulting Services, LLC (SCS), a leading supplier of newspaper pre-press and post-press systems. SCS is best known for Layout-8000™, the industry standard advertising dummyming system. For over ten years, Mr. Cichelli worked at the American Newspaper Publishers Association/Research Institute (ANPA/RI), now the Newspaper Association of America (NAA), where he was Research Manager for Computer Applications. During his tenure at the ANPA/RI, he collaborated on the ANPA manual Purchase Contract Negotiations and Language Guidelines and helped many newspapers and vendors resolve their differences amicably.