ANSI Guidelines on Software in Standards

Approved by the ANSI Patent Group and Intellectual Property Rights Policy Committee

Printed in the United States of America

About the
American National Standards Institute

ANSI is a nonprofit, privately funded membership organization that coordinates the development of U.S. voluntary national standards and is the U.S. member body to the International Organization for Standardization (ISO) and, via the United States National Committee (USNC), the International Electrotechnical Commission (IEC).

The Institute was founded in 1918, prompted by the need for an "umbrella" organization to coordinate the activities of the U.S. voluntary standards system and eliminate conflict and duplication in the development process. For over seventy years, this system has been successfully administered by the private sector, via ANSI, with the cooperation of federal, state and local governments. The Institute serves a diverse membership of over 1300 companies, 250 professional, technical, trade, labor and consumer organizations and some 30 government agencies. Standards exist in all industries, including safety and health, telecommunications, information processing, petroleum, medical devices, etc.

Some of the Institute's key functions include:

- Coordinating the self-regulating, due process consensus based U.S. voluntary standards system;
- Administering the development of standards and approving them as American National Standards;
- Providing the means for the U.S. to influence development of international and regional standards;
- Promoting awareness of the growing strategic significance of standards technology to U.S. global competitiveness.

Guidelines

I. ANSI Guidelines On Software in Standards

These guidelines apply to the inclusion of normative software[1] in American National Standards which software is intended to be extracted from the standard and used in implementations of the standard.[2] As a general rule, standards should provide a description of features from which competing and interoperable implementations can be developed and not serve as endorsements for a particular solution or implementation. ANSI strongly discourages any use of software in standards which is inconsistent with this set of Guidelines. ANSI recommends that copyrighted software should only be included for informational purposes, or in forms which do not mandate particular implementations of the standard. Object code should never be included in a standard as a normative requirement. While ANSI opposes use of software standards to mandate particular implementations and believes that use of software in standards should be avoided to the extent possible, ANSI recognizes that there may be circumstances in which inclusion of some software, provided it is accompanied by adequate legal permissions, may facilitate development of multiple, competing and interoperable implementations of the standard. Examples of such software could include:

- pseudo code (code that is human readable and similar to programming languages but cannot be directly processed or compiled directly to be processed by hardware that manipulates data according to instructions);
- schema examples;
- data structure definitions;
- ASN.1 structure definitions;
- ABNF grammar specifications;
- example programming instructions that are sufficiently limited in scope that they do not, either singularly or in the aggregate, perform a complete or a substantial part of a function and are illustrative, at most, of limited sections of an independent fully described specification; or
- sample programming instructions provided solely for conformance testing purposes.

Whenever software is included in a standard, ANSI strongly recommends that it be accompanied by legal permissions sufficient to ensure that there will be no legal

---

[1] For purposes of these Guidelines, "software" generally refers to a set of instructions written in any programming language that either directly, or when further compiled, performs a complete function when executed by hardware that manipulates data according to instructions.

[2] This would include software that must be used in implementations of the standard (*i.e.*, copying into the implementation, extraction from the standard and porting, translating, or creating some other form of derivative work of the included software).

impediment to the use, or the accompanying modification or extraction, of the software in any desired manner or implementation consistent with the standard by any implementer.[3]

A standard may include software that is submitted to the consensus body/technical committee with or without asserted restrictions. Restricted software, if incorporated, either expressly or by reference, into the standard, may be either essential or non-essential to its implementation. If the software is essential and restricted, it is not possible to implement the standard without infringing on the copyright associated with that software. Many of the issues regarding essential and non-essential software are the same but the licensing implications may be very different.

Standards often can be written around copyrighted material using performance-based requirements or creating a new expression of the underlying idea within the technical process. Accordingly, a standards developer should carefully consider these types of preferred options before considering the inclusion of copyrighted software source code in a standard. This will help ensure that the resulting standard is more flexible because it is not tied to any particular product or single implementation.

The legal issues relating to copyrighted material are very different than those relating to patented material. Because copyright law does not bestow on the copyright holder intellectual property rights similar to those patent law provides for patent holders, there are compelling reasons to treat copyrighted and patented material differently when they are reflected in standards. There are important differences between these rights. For example, a patent represents a right based on an independent judgment that the holder has contributed an innovation. No such judgment has been made with respect to a copyright.

In addition, copyright only protects one particular expression of an idea, while a patent defines a specific technology or invention in a more abstract sense and grants fairly broad and exclusive rights to the patent holder. As a result, the likelihood of alternative implementations that do not infringe the copyright in the software is much greater with copyright. The limited scope of copyright protection protects against copying and would not preclude independent implementations that perform the same function; thus it is possible for competitors to work around a copyright by developing their own implementation. In contrast, it is possible to have patents that are "essential" to some desired feature or function and the more exclusive rights granted to patent holders make alternative implementations virtually impossible. If a standard requires that all implementers of the standard copy a specific copyrighted work, then by being endorsed as a standard, the copyright right has taken on a significance far beyond that which the original copyright right provided.

Incorporating copyrighted software in a standard raises additional issues that must be addressed. These issues include:

---

[3] Public domain software will always be able to meet this test. Copyrighted software typically cannot meet this test unless it is provided under a suitable license. In order to meet this test, the license must allow extraction, use and modification of the code without legal consequences for any implementation. An example of a suitable license would be a simple permissive open source license like the BSD license. The limitations imposed by most licenses, whether commercial or open source, are inconsistent with this requirement.

- The software has to be maintained, which raises issues as to what is to be done if a glitch is discovered in the software and who is responsible for developing a solution. Although this is an issue in general, it is even more important to clearly define maintenance responsibilities when copyrighted software is used in a standard.
- Similarly, there may be a need to extend the software to address desired enhancements. Again, who is responsible for addressing this issue? The impact on the intellectual property must be properly understood.
- The range and complexity of possible licensing terms is very broad. Often the need to protect trade secrets may complicate the process and restrict access to the copyrighted material until late in the standardization process.

A standard that requires the use of particular software should be an exceptional situation and agreed within the consensus body. Whenever possible, a standard should be based on functional specifications and should be an unencumbered expression of a proposed implementation as opposed to mandating the use of a specific and proprietary copyrighted software/source code.