ISO/IEC JTC 1 N 9864

2009-11-02

ISO/IEC JTC 1
Information Technology

Document Type:	**Procedural Documentation**
Document Title:	**Letter Ballot on the JTC 1 Standing Document on API**
Document Source:	**JTC 1 Secretary**
Reference:	**This document is circulated to JTC 1 National Bodies for a three-month approval ballot. JTC 1 National Bodies are instructed to submit their vote via the online balloting system by the due date indicated.**
Action ID:	**VOTE**
Due Date:	**2010-02-02**
No. of Pages:	**5**

Secretariat, ISO/IEC JTC 1, American National Standards Institute, 25 West 43rd Street, New York, NY 10036; Telephone: 1 212 642 4932; Facsimile: 1 212 840 2298; Email: lrajchel@ansi.org

Standing Document
Guidelines for API Standardization

1 Characteristics and Definition

1.1 Application Program Interface (API) Related Concepts

An API is a boundary across which application software uses facilities of programming languages to invoke services. These facilities may include procedures or operations, shared data objects and resolution of identifiers. A wide range of services may be required at an API to support applications. Different methods may be appropriate for documenting API specifications for different types of services.

The information flow across the API boundary is defined by the syntax and semantics of a particular programming language, such that the user of that language may access the services provided by the application platform on the other side of the boundary. This implies the specification of a mapping of the functions being made available by the application platform into the syntax and semantics of the programming language.

An API specification documents a service and/or service access method that is available at an interface between the application and an application platform.

An API specification may take the form of one of the following:

a Programming language specification, which is a description of a language defined within the programme of work of SC 22, such as Fortran, Ada and C;

b Language independent API specification, which is a description of a set of functionality in terms of semantics (in an abstract syntax) and abstract data types that can be bound to multiple programming languages;

c Language specific API specification, which is a description of a set of functionality in terms of the syntax and data types of some programming language.

Language-independent API specifications are useful in defining specifications for invoking services at the API. The language independent specification serves primarily as the reference used to assure consistency across different language bindings. However, one or more language bindings to programming languages such as COBOL or C must also exist. Language specific API specifications are used by programmers, writing in a particular programming language, to invoke a service provided by the application platform. They may be used by a program to invoke a supporting service offered by another application software entity.

1.2 Level of Abstraction

The concept of "Level of Abstraction" is complex, with several (possibly non-conflicting) uses. Usage of "Level of Abstraction" implies variation in the amount of functionality offered to the calling program by each invocation.

The same service may be provided by multiple API specifications which differ in level of abstraction. For example, a less abstract API specification for X.400 electronic mail services may provide the application programmer substantial control over details of its interaction with the mail servers. On the other hand, a more abstract API specification may provide a simple, single subroutine call for sending a file as a mail message to a mailbox.

Under this usage, a more abstract API specification is easier to use than a less abstract specification provided that the conventions adopted in implementing the service are appropriate to the application. A less abstract API specification is used where there are application specific requirements relating to details of the interaction or the implementation.

Another usage of "Level of Abstraction" reflects the degree to which the implementation method is visible/invisible to the users of the API specification calls.

An API specification may reflect a pure abstraction driven only by the service requirements (e.g. a protocol independent network API specification), or it may reflect details of the implementation. These details may be associated with one of several alternative methods for service satisfaction (e.g. OSI, TCP/IP, or ISDN communication service API specification). The details could also reflect aspects of alternative platforms' implementations. In this context, use of a more abstract API specification yields greater portability and implementation independence while a less abstract API specification may provide more control or improved performance.

The level of abstraction of API specification varies with the programming language and the abstractions inherent to the specific service. Therefore, a "uniform" level of abstraction across the set all API specifications is not appropriate.

N2 Methods and Components for JTC 1 API Work

2.1 Relation to Other Standards

A standard API specification specifies a mapping between a programming language and the features of a particular service, and thereby provides access to that service from applications written in a particular programming language. Such a mapping is said to create a binding between the service and the programming language.

A standard API specification may be part of the standard that specifies the associated programming language, may be part of the standard that specifies the associated service, or may be a separate standard that refers to other standards that define the associated programming language and service. Thus, programming language standards can be considered as one kind of standard API specification.

The following policies apply:

2.1.1. Standard API specifications shall identify the standards that specify the programming language and the service associated with it, if these are not specified by the standard API specification itself.

2.1.2. Standard API specifications shall be consistent with, and shall avoid duplication of, requirements specified by the associated service and programming language standards.

2.1.3. Where it can be expected that implementations will support bindings to a service for multiple programming languages, any requirements on interoperability between these bindings should be specified, including requirements on exchange of data values.

2.1.4. Where it can be expected that implementations will support bindings to multiple services for a single programming language, any requirements on compatibility between these bindings should be specified, including requirements on coordination of identifier name spaces. There is a need to list the requirement for correct interworking between the services accessed via the language binding.

2.2 Language-Independent API Specifications and Language Bindings

Standard API specifications can specify a direct mapping between a programming language and a service, or an indirect mapping that makes use of an intermediate, abstract interface model and syntax that is separately mapped to the programming language and to the service. Where an indirect mapping is used and the same abstract interface is mapped to multiple programming languages, the specification of the mapping from the service to the abstract interface model and syntax is called a language-independent API specification. A specification of a mapping to a programming language, whether directly from a service or from a language-independent API for a service, is called a language binding for that service (see TR 10182:1993).

Where there are multiple language bindings to a service, some language bindings may depend on a language-independent API specification, while others map directly to the service, and different groups of language bindings may depend on separate language-independent API specifications, for example where the bindings for different programming languages have incompatible requirements. The following policies apply:

2.2.1 Where a standardisation project for an API specification includes multiple language bindings with common interface characteristics, the use of language-independent API specifications should be strongly encouraged.

2.2.2 Where a standardisation project for an API specification includes a language-independent API specification, the language-independent API specification shall be progressed together with at least one language binding that depends on the language-independent API specification.

2.2.3 The use of common, standardised methods where available for the specification of language-independent API specification should be encouraged.

2.3 Conformance and Testability

Clear definitions of conformance and testability are essential for standard API specifications. Not all required functions can be effectively tested. However, where possible, test methods should be readily derivable from the standard. (See ISO/IEC 9646:1994 and ISO/IEC 13210:1994.)

The following policies apply:

2.3.1 If either a programming language or the service specification to which it is interfaced has multiple levels of conformance, then the API standard should have corresponding conformance levels.

2.3.2 The "conformance clauses" and conformance requirements specified in standard API specifications shall distinguish between the requirements on a platform's conforming service implementations and those on conforming applications.

2.3.3 API conformance requirements should include sufficient level of specification that verification test methods can be readily derived.

2.3.4 The use of API specification methods that support the use of automated test procedures should be encouraged.

2.4 Relationship to Models

All API specifications, in mapping between a programming language and a service, must take into account the underlying semantic models of the programming language and the service, whether these are explicit or implicit.

The following policies apply:

2.4.1 The specification of explicit semantic models should be encouraged in the development of standards for programming languages and services, in order to facilitate the development of API specifications which bind them together.

2.4.2 When a difference is encountered between the semantic models of the programming language and the service, the API specification should document the approach taken to either harmonise or address this difference.

2.4.3 Where a JTC 1 standard exists for a model or framework that addresses the scope of proposed work, the relationship of that work to the model shall be documented.

3 Considerations in Proposing API Standardization

3.1 Placement within JTC 1 for Standardization of API Specifications

There are three types of expertise that need to be involved in the development of API specifications. The developers of the service standards must be involved to provide detailed knowledge on the use of service standards. The users of the base standards and the API must be involved to facilitate determining the appropriate levels of abstraction and "common usage". Language experts need to be involved to ensure that the API is properly integrated into the various languages. One of the more important aspects of the language dependent API specifications is that they fit well into the style and model of the language in which they are to be used without conflicting with existing syntax and semantics. Those participating in the development of an API specification would not be expected to limit their participation to a single type of expertise. However, these categories do serve to characterise the kinds of knowledge that needs to be involved.

Level of Abstraction and Language-independence are two characteristics of API specifications that could be used to help determine the placement of work on API specifications. Much like the types of expertise just described, these are not an absolute characterisation, but more an indication of degrees or shadings. For API specifications that were very language independent and were at a lower level of abstraction, the work may rely more heavily on base standards participation. By moving to higher levels of abstraction, more user participation and a rise in the need for language expertise is expected. By moving from more language independence to more language dependence, greater participation by language experts and less participation from base standards is expected. The following policy applies:

3.1.1 To expedite placement of future work, an NP or fast-track submission that includes an API component shall be accompanied by a statement that addresses the questions:

a) Which SC is responsible for the underlying service?

b) Which SC is responsible for the programming language(s)'?

c) Will the API specification require extension(s) to an existing programming language or service'?

d) What is the kind of expertise required in the development of the API specification?

e) What resources of the SC are available to perform the new API specification work?

f) What is the relationship of the API specification work to other work in the SC?

g) Is the appropriate expertise available for review and consideration of the draft API specifications, especially during the CD ballot Stage?

3.2 Coordination

Standardization of API specifications is dependent upon related standards which apply to one or both sides of the interface involved. Therefore, it is important that during the development of standards for API specifications:

a) related work is progressed, and

b) liaison bodies involved with related work are active,

in order to ensure that technically sound and complete standards are developed in a timely fashion. The following policy applies:

3.2.1 Standardization of API specifications should require specific explicit review by specifically identified liaisons at specific stages of development (e.g. CD, at time of registration, should be sent to these identified liaison SCs for required review and comment to the developing SC.)