Article 1 :

1. Non défini
2. La seule chose que recommande ce texte est de bien documenter tous les changements effectués lors de l’évolution de l’évolution de l’API.

Article 2 :

1. Ils ne sont pas définis explicitement mais d’après le texte le ressenti de l’utilisateur par rapport à une API et combien de temps ils mettent ils mettent à faire une tâche. La version de l’API permettant de coder une tâche en moins de temps est considérée comme plus utilisable.
2. En écoutant ce que l’utilisateur en pense et en essayant d’avoir l’abstraction le plus haut niveau possible.

Article 5 :

1. L’organisation internationale de standardisation définit l’utilisabilité ainsi « the extent to which product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in specified context of use. » Cependant il n’y pas de critères précis donnés.
2. Rien de clair à en tirer.

Article 6 :

1. Rien
2. En étudiant le fait que celle-ci est utilisée.

Article 7 :

1. Une bonne utilisabilité peut avoir un rapport avec la maintenance. Lorsque l’API est mise à jour le client ne doit pas souffrir de cette situation
2. En prenant bien compte l’utilisation que les clients font de l’API.

Article 8 :

1. Pas explicitement précisé.
2. En menant en parallèle une « API Peer Review » et une évaluation d’utilisabilité. Cela permet de révéler tous les types de bugs.

Article 10 :

1. Pas explicitement précisé.
2. En menant en parallèle une « API Peer Review » et une évaluation d’utilisabilité. Cela permet de révéler tous les types de bugs. D’après la littérature HCI la méthode la plus courante et documentée pour évaluer l’utilisabilité est le test empirique. Menez de bons tests afin de détecter tous les problèmes et les corriger.

Article 11 :

1. Un des critères pourrait être la fiabilité du code produit par les utilisateurs de l’API. Un autre facteur pourrait être la rapidité à exécuter une tâche. On pourrait ajouter le fait de traiter les erreurs rapidement. Le nombre de changements de fichiers, de recherches, de code complétion et de temps complétion peuvent être de bons facteurs à analyser.
2. En faisant diminuer tous les critères ci-dessus. Un IDE peut aider aussi.

Article 12 :

1. Ce qui peut être étudié c’est est-ce qu’une API est facile à utiliser ou non. Critères étudiés : compréhensibilité, niveau d’abstraction, de réutilisabilité et d’apprentissage.
2. En faisant une étude de dimension cognitive et en comparant par exemple deux versions de l’API ou recueillant les retours de l’expérience utilisateur afin de mieux comprendre les points sur lesquels il y a des zones d’ombre.

Article 13 :

Article 14 :

1. Ici l’article permet de rendre compte à quel point les exemples sont importants pour les utilisateurs d’une API. Avoir de meilleurs exemples pour améliorer l’utilisabilité.
2. Comme pour la réponse précédente il faut avoir des exemples variés, bien décris et ne pas hésiter à répondre sur les sites de discussion qui sont le deuxième lieu où les gens vont chercher l’information après les sites officiels et dédiés.

Article 15 :

1. Non renseigné
2. Avoir une documentation plus solide et continuer les efforts dans la redocumentation des APIs existantes.

Article 16 :

1. Les critères d’utilisabilité sont :

* Avantage du langage
* Avantage de la terminologie
* Avantage de la métaphore
* Avantage des entrées
* Fonctionnalité
* Simplicité
* Consistance
* Minimum de mémoire chargée
* Navigabilité
* Minimum d’entrainement

1. En suivant un processus clair et détaillé comme décrit dans l’article.

Différents critères définissent une bonne API :

* Une API doit fournir suffisamment de fonctionnalités pour que l’appelant puisse exécuter sa tâche.
* Une API doit être minimale, elle ne doit imposer des inconvénients excessifs à l’appelant.
* Les APIs ne peuvent être conçues sans comprendre leur contexte.
* L’objectif des APIs générales devraient d’être sans règles, alors que les spécifiques devraient être dotées de beaucoup de règles.
* Les APIs devraient être conçues dans la perspective de l’appelant.
* Les bonnes APIs ne causent pas de problème
* Les APIs devraient être documentées avant d’être implémentées.
* Les bonnes APIs sont ergonomiques.

Article 18 :

1. Un des critères mis en avant est le contexte structural afin de fournir les meilleurs exemples.
2. En fournissant des exemples plus pertinents.