NORME ST.97 DE L'OMPI

Version 2.0

RECOMMANDATION RELATIVE AU TRAITEMENT DES DONNEES DE PROPRIETE INTELLECTUELLE AU FORMAT JSON

Révision approuvée par l'Équipe d'experts chargée des API du Comité des normes de l'OMPI (CWS) pour publication le 15 juillet 2026

TABLE DES MATIÈRES

INTRODUCTION 3
DÉFINITIONS ET TERMINOLOGIE 3
NOTATIONS GÉNÉRALES 4
Identificateurs de règle 4
Exemple de structure de données au format JSON 5
PORTÉE DE LA NORME 5
RÈGLES ET CONVENTIONS GÉNÉRALES DE CONCEPTION JSON 6
Aperçu 6
Conventions de nommage JSON 6
RÈGLES DE CONCEPTION DU SCHÉMA JSON 8
Aperçu 8
Modularité 8
Documentation 16
Conventions de nommage des fichiers d'un schéma 18
Structure des propriétés du schéma JSON 18
RÈGLES DE CONCEPTION DES STRUCTURES DU SCHÉMA JSON 19
Aperçu 19
Propriétés 19
Définitions 19
Définitions de types 20
Type primitif JSON 21
Listes de codes 21
Tableaux 21
Objets 22
IDENTIFICATEURS DE SCHÉMA JSON 22
Aperçu 22
RÈGLES DE CONCEPTION DES INSTANCES JSON 22
Ordre des propriétés 23
Validation de l'instance JSON 23

RÉFÉRENCES 23
Normes de l'OMPI 23
Normes et spécifications industrielles 24

ANNEXES

Les Annexes sont disponibles dans : http://www.wipo.int/standards/fr/st97/v2-0/

Annexe I – Règles de conversion des schémas XML de la norme ST.96 en schémas JSON

Annexe II – Schémas JSON

Annexe III – Exemples d'instances JSON

Annexe IV – Liste des sigles et abréviations

Annexe V – Termes de représentation

Annexe VI – Dictionnaire de données de propriété intellectuelle

NORME ST.97 DE L'OMPI

Version 2.0

RECOMMANDATION RELATIVE AU TRAITEMENT DES DONNEES DE PROPRIETE INTELLECTUELLE AU FORMAT JSON

Révision approuvée par l'Équipe d'experts chargée des API du Comité des normes de l'OMPI (CWS) pour publication le 15 juillet 2026

INTRODUCTION

1. La présente norme fournit des recommandations concernant la conception, la création et la mise à jour de ressources JavaScript Object Notation (JSON) qui seront utilisées pour le dépôt, le traitement, l'échange ou la publication de tout type de données de propriété intellectuelle. La présente norme comprend également des règles relatives à la transformation des schémas XML (eXtensible Markup Language) (XSD) de la norme ST.96 de l'OMPI en schémas JSON qui respectent les recommandations susmentionnées, les schémas JSON étant dérivés de la norme ST.96 de l'OMPI.

2. La présente norme a pour but :

- de fournir des schémas et des instances JSON bien structurés;
- de donner des orientations concernant la conception et l'élaboration de pratiques recommandées concernant les données de propriété intellectuelle au format JSON;
- de recommander des principes de conception relatifs à l'extension des schémas JSON élaborés ou à la création de nouveaux schémas JSON conformes; et
- de rationaliser l'échange de données en favorisant la réutilisation des schémas et instances JSON parmi les offices de propriété intellectuelle, ainsi que des données communiquées au public.

DÉFINITIONS ET TERMINOLOGIE

3. Aux fins de la présente norme, les définitions suivantes sont utilisées :

- le terme "ressources JSON" dans la présente norme désigne toute composante utilisée aux fins de la création ou de la mise en œuvre d'une application JSON conforme à la présente norme;
- les termes "JSON", "JSON text", "JSON value" "array", "element", "object", "object type", "property", "member", "property name", "property value", "property type", "keyword", "number", "string", "boolean", "true", "false", "null" et "definition" utilisés dans la présente norme doivent être interprétés tels que définis dans la version du projet draft-2020-12 du JSON Schema Core[1];
- le terme "structure" utilisé dans la présente norme doit être interprété comme un "socle" à partir duquel les schémas JSON sont construits;
- le terme "définition générale" désigne une définition à laquelle d'autres définitions font référence dans le même schéma ou dans d'autres schémas;
- le terme "brevet" désigne un titre de propriété industrielle tel qu'un brevet d'invention, un brevet de plante, un brevet de dessin ou modèle, un certificat d'auteur d'invention, un certificat d'utilité, un modèle d'utilité, un brevet d'addition, un certificat d'auteur d'invention additionnel ou un certificat d'utilité additionnel;
- le terme "marque" désigne une marque de produits ou de services ou un autre type de représentation distinctive répondant à la définition de la marque qui figure dans la législation applicable, englobant mais sans s'y limiter la marque collective, la marque de certification ou la marque de garantie;

[1] La version du schéma JSON est susceptible d'être modifiée, parce qu'elle n'a pas le statut RFC; elle n'a pas été adoptée par un groupe de travail IETF. Cette version de la norme se fonde sur la dernière version, c'est-à-dire la version 2020-12, disponible à l'adresse https://json-schema.org/draft/2020-12/json-schema-core.html.

- le terme "dessins ou modèles industriels" désigne les caractéristiques bidimensionnelles ou tridimensionnelles de la forme ou de la surface des objets, qui correspondent respectivement aux notions de dessin et de modèle lorsqu'une distinction est faite entre l'un et l'autre; le terme "dessins ou modèles industriels" n'englobe pas les brevets de dessin ou modèle;

- le terme "indication géographique" désigne un signe apposé sur des produits ayant une origine géographique particulière, qui possèdent des qualités ou une notoriété due à cette origine; et

- le terme "droit d'auteur" désigne les droits dont jouissent les créateurs sur leurs œuvres littéraires et artistiques et l'expression "œuvre orpheline protégée par le droit d'auteur" désigne une œuvre qui est protégée par le droit d'auteur, mais dont l'auteur ne peut être identifié ou localisé.

4. Les mots clés "DOIT" ou "DOIVENT", "NE DOIT PAS" ou "NE DOIVENT PAS", "DEVRAIT" ou "DEVRAIENT", "NE DEVRAIT PAS" ou "NE DEVRAIENT PAS", "RECOMMANDÉ(S)", "PEUT" ou "PEUVENT" et "FACULTATIF", lorsqu'ils apparaissent en majuscules dans la présente norme, doivent être interprétés comme indiqué ci-dessous uniquement lorsqu'ils sont écrits entièrement en majuscules. Lorsque ces mots ne figurent pas en majuscules, ils doivent être pris dans leur sens courant en français.

- "DOIT" ou "DOIVENT" : ce mot, ou les mots "REQUIS" ou "DEVRA" ou "DEVRONT", signifie que la définition est une condition absolue de la spécification;

- "NE DOIT PAS" ou "NE DOIVENT PAS" : ces mots, ou les mots "NE DEVRA PAS" ou "NE DEVRONT PAS", signifient que la définition est une interdiction absolue de la spécification;

- "DEVRAIT" ou "DEVRAIENT" : ce mot, ou l'adjectif "RECOMMANDÉ(S)", signifie qu'il peut y avoir des raisons valables dans des circonstances particulières pour ignorer un terme particulier mais que toutes les conséquences doivent être comprises et soigneusement évaluées avant de choisir une trajectoire différente;

- "NE DEVRAIT PAS" ou "NE DEVRAIENT PAS" : ces termes, ou les termes "NON RECOMMANDÉ(S)", signifient qu'il peut y avoir des raisons valables dans des circonstances particulières lorsque le comportement particulier est acceptable ou même utile mais que toutes les conséquences doivent être comprises et le cas soigneusement évalué avant d'appliquer un comportement assorti de cette étiquette; et

- "PEUT" ou "PEUVENT" : ce mot, ou l'adjectif "FACULTATIF(S)", signifie qu'un élément est véritablement facultatif. Une application qui ne comporte pas une option particulière DOIT être prête à permettre l'interopérabilité avec une autre application qui comporte l'option encore que, peut-être, avec une fonctionnalité réduite. Dans le même ordre d'idées, une application qui comporte une option particulière DOIT être prête à permettre l'interopérabilité avec une autre application qui ne comporte pas l'option (sauf bien sûr pour la caractéristique que l'option fournit).

NOTATIONS GÉNÉRALES

5. Les notations ci-après sont utilisées d'un bout à l'autre de la présente norme :

- <> : indique un terme utilisé pour décrire un espace qualifié qui, dans l'application, sera remplacé par une valeur d'instance spécifique. Cette convention ne s'applique pas lorsqu'elle apparaît dans le cadre de la syntaxe Schema;

- "" : indique que le texte entre guillemets doit être utilisé in extenso dans l'application. Cette convention ne s'applique pas lorsqu'elle apparaît dans le cadre de la syntaxe Schema;

- { } : indique que l'application est facultative. Cette convention ne s'applique pas lorsqu'elle apparaît dans le cadre de la syntaxe Schema; et

- Caractères `Courier New` : Indique les mots-clés JSON et XML, les noms de composants et les codes

Identificateurs de règle

6. Toutes les règles de conception sont normatives. Les règles de conception sont identifiées au moyen d'un préfixe de [JXX-nn].

- La valeur "JXX" est un préfixe qui sert à classer les types de règle comme suit :

  a) JGD pour les règles de conception générales;

  b) JSD pour les règles de conception du schéma JSON;

c) JCD pour les règles de conception de la structure; et

d) JID pour les règles de conception des instances.

– La valeur "nn" définit le prochain numéro disponible dans l'ordre d'un type de règle donné. Il y a lieu de noter que le numéro ne désigne pas la position de la règle, en particulier pour une nouvelle règle. Une nouvelle règle sera placée dans le contexte pertinent. Par exemple, l'identificateur [JGD-10] identifie la dixième règle de conception générale. La règle [JGD-10] peut être placée entre les règles [JGD-05] et [JGD-06] au lieu de suivre [JGD-09] s'il s'agit de la position la plus appropriée pour cette règle.

– L'identificateur de la règle supprimée sera conservé tandis que la règle sera remplacée par le terme "Supprimé".

Exemple de structure de données au format JSON

7. Des exemples de structure des données au format JSON sont présentés dans les encadrés dans une police à largeur fixe. Les exemples de syntaxe de la structure de données au format JSON sont surlignés pour en faciliter la lecture.

PORTÉE DE LA NORME

8. La présente norme vise à apporter des ressources JSON à utiliser pour le dépôt, la publication, le traitement et le partage de données et d'informations en matière de propriété intellectuelle. La présente norme a pour objectif d'apporter des orientations aux offices de propriété intellectuelle et aux organisations concernées qui traitent des données et documents relatifs à des brevets, des marques, des dessins ou modèles industriels, des indications géographiques ou des œuvres orphelines protégées par le droit d'auteur.

9. La présente norme vise à apporter des orientations aux offices de propriété intellectuelle et aux organisations concernées qui créent ou modifient des données de propriété intellectuelle en tant que ressources JSON. Il est nécessaire de suivre cette norme pour assurer l'échange de données entre les offices de propriété intellectuelle qui utilisent des ressources JSON, telles que des schémas, des instances, des messages ou des charges utiles pour les interfaces de programmation d'application (API).

10. Les règles et conventions de conception contenues dans le présent document fournissent une série complète de règles pour la création des schémas et instances JSON utilisés pour tous les types d'information en matière de propriété intellectuelle. Les règles de conversion ont pour objet de guider les offices de propriété intellectuelle afin qu'ils puissent convertir leurs schémas ST.96 personnalisés en schémas JSON équivalents.

11. La présente norme comprend les annexes suivantes :

– Annexe I : Règles de conversion des schémas XML de la norme ST.96 en schémas JSON;

– Annexe II : Schémas JSON[2];

– Annexe III : Exemples d'instances JSON;

– Annexe IV : Liste des sigles et abréviations;

– Annexe V : Termes de représentation; et

– Annexe VI : Dictionnaire de données de propriété intellectuelle.

12. La présente norme exclut ce qui suit :

a) questions relatives à l'architecture logicielle; et

b) langues d'application.

[2] Les schémas JSON convertis présentent généralement les mêmes noms de composants et la même structure de données que ceux définis dans la version correspondante de la norme ST.96, à l'annexe III. Cependant, plusieurs modifications manuelles ont été apportées à certains schémas JSON afin de tenir compte des différences entre les technologies XML et JSON. De plus amples renseignements sur le schéma modifié manuellement sont fournis dans les notes de diffusion de la version correspondante de la norme ST.97.

13. Tous les objets, à l'exception des objets de type de données et des valeurs d'énumération définis par la présente norme, DOIVENT figurer dans le Dictionnaire de données de propriété intellectuelle, c'est-à-dire à l'annexe VI de la norme.

14. Chaque version de l'annexe III DOIT être publiée intégralement, y compris les composantes actualisées et non modifiées du schéma JSON.

RÈGLES ET CONVENTIONS GÉNÉRALES DE CONCEPTION JSON

Aperçu

15. La présente section contient des règles et conventions de conception JSON générales et de haut niveau qui s'appliquent à tous les échanges de données JSON et à toutes les activités de développement JSON, plutôt qu'à un langage de programmation spécifique qui sérialise et désérialise les données vers ou à partir de JSON. Les règles et conventions générales énumérées ci-dessous fournissent l'assise commune du schéma JSON, de l'instance JSON et de l'élaboration de la structure des données JSON pour toutes les données afin d'inclure les données liées ou non à la propriété intellectuelle, telles que les données des contenus mixtes. Les niveaux d'imbrication DEVRAIENT être limités au minimum lors de la création de nouveaux schémas JSON, de nouvelles instances JSON et de l'élaboration de structures de données JSON qui ne sont pas proposées dans la norme ST.97 de l'OMPI ou dont la compatibilité avec la norme ST.96 ne doit pas nécessairement être examinée.

Conventions de nommage JSON

16. Ces conventions sont nécessaires pour assurer la cohérence, l'uniformité et l'exhaustivité du nommage et de la définition de toutes les ressources JSON.

17. Les conventions de nommage JSON figurant dans la présente section reposent sur les lignes directrices et principes décrits dans la cinquième partie du document ISO 11179[3] – Principes de nommage et d'identification. Le nom des objets et les noms de propriété comprennent les termes suivants :

- La *classe objet* désigne une activité ou un objet dans un contexte d'entreprise et représente le regroupement ou l'agrégation logique de données (dans un modèle logique de données) auquel une propriété appartient. La classe objet est désignée par le terme de classe objet.
- Le *terme de propriété* détermine les caractéristiques de la classe objet.
- Le *terme de qualification* est constitué d'un ou plusieurs mots qui aident à définir un élément de donnée et à le différencier d'autres éléments de donnée connexes et qui peuvent être associés à un terme de classe objet ou à un terme de propriété si cela s'avère nécessaire pour rendre un nom unique.
- Le *terme de représentation* classe le format de l'élément de donnée selon de grandes catégories. Les termes de représentation énumérés à l'annexe V devraient être utilisés.

[JGD-01] Le type d'objet et les noms de propriété DOIVENT être composés de mots anglais, écrits selon l'orthographe anglaise figurant dans le dictionnaire Oxford English Dictionary. Les seules exceptions acceptées sont les sigles, les abréviations ou d'autres mots tronqués qui sont énumérés à l'annexe IV.

[JGD-02] Le type d'objet et les noms de propriété DEVRAIENT comprendre uniquement des noms, des adjectifs et des verbes au présent, à l'exception des sigles, des abréviations et d'autres mots tronqués qui sont énumérés à l'annexe IV.

[JGD-03] Les caractères utilisés dans les noms de propriété DOIVENT être limités à la série suivante : 'a-z, A-Z et 0-9'.

[JGD-04] La longueur maximale d'un type d'objet et des noms de propriété NE DEVRAIT PAS être supérieure à 35 caractères. L'utilisation d'abréviations ou de sigles est encouragée pour réduire la longueur d'un nom d'élément. Toute nouvelle abréviation ou tout nouveau sigle DOIT être ajouté(e) à l'annexe IV.

[JGD-05] Le type d'objet et les noms de propriété DEVRAIENT être concis et s'expliquer d'eux-mêmes.

[3] https://www.iso.org/fr/standard/60341.html

[JGD-06] Le type d'objet et les noms de propriété DOIVENT figurer en caractères minuscules de type "camel" (LCC). Par exemple, "`currencyCode`":"EUR".

[JGD-07] Les noms de type d'objet DOIVENT figurer en caractères minuscules de type "camel" (LCC) et avoir un type suffixe. Par exemple, `ApplicantType`.

[JGD-08] Les sigles et les abréviations énumérés à l'annexe IV DOIVENT toujours être utilisés en lieu et place du nom développé complet.

[JGD-09] Les sigles et abréviations DOIVENT figurer comme indiqué à l'annexe IV pour les noms de propriété et de types d'objet.

[JGD-24] Les sigles et abréviations placés au début d'un nom d'élément DOIVENT être constitués de caractères bas de casse, par exemple, "`pre`", "`bioDeposit`". Si un sigle figure en majuscules au début d'un nom d'élément, tous les caractères doivent être en minuscules, par exemple, "`idref`" et "`wipo`" dans le nom de propriété de "`wipoST3Code`". Par ailleurs, toutes les valeurs d'une énumération, les valeurs de sigles et les valeurs d'abréviations DOIVENT figurer en majuscules comme indiqué à l'annexe IV.

[JGD-10] Un terme de classe objet DOIT toujours avoir la même signification sémantique dans un domaine spécifique de la propriété intellectuelle, tel que les brevets, les marques, les dessins et modèles industriels, les indications géographiques ou le droit d'auteur, et PEUT comprendre plus d'un mot. Par exemple, `contactInformation`.

[JGD-11] Le terme de propriété dans un nom DOIT être unique dans le contexte d'une classe objet mais PEUT être réutilisé dans différentes classes objets.

[JGD-12] Un terme de qualification PEUT être attaché à un terme de classe objet ou à un terme de propriété si cela s'avère nécessaire pour rendre un nom unique.

[JGD-13] Lorsqu'un nom comprend un terme de classe objet, un terme de propriété et un terme de représentation, le terme de classe objet DOIT précéder le terme de propriété et le terme de propriété DOIT précéder le terme de représentation. Un terme de qualification DEVRAIT précéder le terme associé de classe objet ou le terme de propriété. Par exemple, `claimTotalQuantity`.

[JGD-14] Si le terme de propriété finit avec le même mot que le terme de représentation (ou un mot équivalent), le terme de représentation DOIT être supprimé.

[JGD-15] Lorsqu'un terme de représentation est requis, les termes de représentation de l'annexe V DOIVENT être utilisés pour les termes de représentation dans les noms de composantes basiques.

[JGD-16] Dans un domaine de la propriété intellectuelle, tous les types d'objet et les noms de propriété DOIVENT être uniques.

[JGD-17] Le ou les mots dans un nom DEVRAIENT être utilisés au singulier sauf si le concept est lui-même au pluriel. Par exemple, `TotalMarkSeries`.

[JGD-18] Le nom d'une propriété ou d'un type d'objet qui comprend une collection de composantes contextuellement reliées DEVRAIT avoir le suffixe "Bag". Par exemple, `emailAddressBag` représente une collection d'éléments `emailAddress`.

[JGD-19] Des mots qui relient comme "and", "of" et "the" NE DEVRAIENT PAS être utilisés dans les types d'objets et les noms de propriété à moins qu'ils ne fassent partie de la terminologie commerciale.

[JGD-20] Le type d'objet et les noms de propriété NE DOIVENT PAS être traduits, modifiés ou remplacés dans un but, quel qu'il soit.

[JGD-21] Le type d'objet et les noms de propriété NE DOIVENT PAS faire référence à des numéros d'article et de règle. Par exemple, `PCTRule702C` pour le PCT.

[JGD-22] Les niveaux d'imbrication DEVRAIENT être limités au minimum lors de la création de nouveaux schémas JSON, d'instances JSON et de structures/fragments de données JSON.

[JGD-23] *Supprimé*

RÈGLES DE CONCEPTION DU SCHÉMA JSON

Aperçu

18. Le schéma JSON décrit la structure d'une instance JSON, qui exprime les contraintes sur la structure et le contenu du document. La présente norme devrait être conforme à la spécification du schéma JSON du secteur. La dernière version disponible au moment de la publication de la présente norme est le projet 2020-12[4] et la présente version de la norme se réfère à ce projet de spécification.

[JSD-01] Les schémas JSON DOIVENT se conformer aux spécifications du schéma JSON : Version 2020-12 de JSON Schema Core et la version 2020-12 de JSON Schema Validation

[JSD-02] Les schémas JSON DOIVENT indiquer qu'ils se conforment à la version 2020-12 du schéma JSON à l'aide du mot clé $schema et de la valeur "https://json-schema.org/draft/2020-12/schema".

Exemple : Indiquer la version du schéma JSON

```
"$schema": "https://json-schema.org/draft/2020-12/schema"
```

19. Le schéma JSON devrait être codé en UTF-8 pour garantir une interopérabilité maximale.

[JSD-03] Les schémas JSON DOIVENT utiliser la norme ISO/IEC 10646 – UCS – ensemble de caractères Unicode. Le format UTF-8 DOIT être utilisé pour coder les caractères Unicode.

Modularité

20. La modularité permet la création de composantes de schéma pour favoriser la flexibilité dans la conception et la réutilisabilité. Dans la conception, il est recommandé d'éviter la définition de toutes les propriétés et composantes logiques sous la forme d'un seul schéma JSON monolithique, qui empêche de partager et de réutiliser des propriétés individuelles ou des composantes logiques définies comme étant un groupe dans un schéma.

21. Le schéma présenté ci-après ne respecte **pas** le principe de modularité. Il N'EST PAS recommandé par la présente norme.

`applicationNumber.json` (Exemple de document de schéma composé incorrect)

```
{
  "$id" : "applicationNumber.json",
  "$schema" : "https://json-schema.org/draft/2020-12/schema",
  "type" : "object",
  "additionalProperties" : false,
  "properties" : {
    "applicationNumber" : {
      "$ref" : "#/$defs/applicationNumber"
    }
  },
  "required" : [ "applicationNumber" ],
  "$defs" : {
    "applicationNumber" : {
      "description" : "Description: Numbers used by IPOs in order to identify each
application received; Version: V2_0",
      "type" : "object",
      "additionalProperties" : false,
      "properties" : {
        "ipOfficeCode" : {
          "anyOf" : [ {
           "type" : "string",
```

4 https://json-schema.org/draft/2020-12/schema

```
          "enum" : [ "AD", "AE", "AF", "AG", "AI", "AL", "AM", "AO", "AP", "AR",
"AT", "AU", "AW", "AZ", "BA", "BB", "BD", "BE", "BF", "BG", "BH", "BI", "BJ", "BM",
"BN", "BO", "BQ", "BR", "BS", "BT", "BV", "BW", "BX", "BY", "BZ", "CA", "CD", "CF",
"CG", "CH", "CI", "CK", "CL", "CM", "CN", "CO", "CR", "CU", "CV", "CW", "CY", "CZ",
"DE", "DJ", "DK", "DM", "DO", "DZ", "EA", "EC", "EE", "EG", "EH", "EM", "EP", "ER",
"ES", "ET", "EU", "FI", "FJ", "FK", "FO", "FR", "GA", "GB", "GC", "GD", "GE", "GG",
"GH", "GI", "GL", "GM", "GN", "GQ", "GR", "GS", "GT", "GW", "GY", "HK", "HN", "HR",
"HT", "HU", "IB", "ID", "IE", "IL", "IM", "IN", "IQ", "IR", "IS", "IT", "JE", "JM",
"JO", "JP", "KE", "KG", "KH", "KI", "KM", "KN", "KP", "KR", "KW", "KY", "KZ", "LA",
"LB", "LC", "LI", "LK", "LR", "LS", "LT", "LU", "LV", "LY", "MA", "MC", "MD", "ME",
"MG", "MK", "ML", "MM", "MN", "MO", "MP", "MR", "MS", "MT", "MU", "MV", "MW", "MX",
"MY", "MZ", "NA", "NE", "NG", "NI", "NL", "NO", "NP", "NR", "NZ", "OA", "OM", "PA",
"PE", "PG", "PH", "PK", "PL", "PT", "PW", "PY", "QA", "QZ", "RO", "RS", "RU", "RW",
"SA", "SB", "SC", "SD", "SE", "SG", "SH", "SI", "SK", "SL", "SM", "SN", "SO", "SR",
"SS", "ST", "SV", "SX", "SY", "SZ", "TC", "TD", "TG", "TH", "TJ", "TL", "TM", "TN",
"TO", "TR", "TT", "TV", "TW", "TZ", "UA", "UG", "US", "UY", "UZ", "VA", "VC", "VE",
"VG", "VN", "VU", "WO", "WS", "XN", "XU", "XV", "XX", "YE", "ZA", "ZM", "ZW" ],
          "description" :     Andorra: AD; United Arab Emirates: AE; Afghanistan:
AF; Antigua And Barbuda: AG; Anguilla: AI; Albania: AL; Armenia: AM; Angola: AO;
African Regional Intellectual Property Organization (ARIPO): AP; Argentina: AR;
Austria: AT; Australia: AU; Aruba: AW; Azerbaijan: AZ; Bosnia and Herzegovina: BA;
Barbados: BB; Bangladesh: BD; Belgium: BE; Burkina Faso: BF; Bulgaria: BG; Bahrain:
BH; Burundi: BI; Benin: BJ; Bermuda: BM; Brunei Darussalam: BN; Bolivia
(Plurinational State of): BO; Bonaire, Sint Eustatius and Saba: BQ; Brazil: BR;
Bahamas: BS; Bhutan: BT; Bouvet Island: BV; Botswana: BW; Benelux Office for
Intellectual Property (BOIP): BX; Belarus: BY; Belize: BZ; Canada: CA; Democratic
Republic of the Congo: CD; Central African Republic: CF; Congo: CG; Switzerland:
CH; Côte d'Ivoire: CI; Cook Islands: CK; Chile: CL; Cameroon: CM; China: CN;
Colombia: CO; Costa Rica: CR; Cuba: CU; Cabo Verde: CV; Curaçao: CW; Cyprus: CY;
Czechia: CZ; Germany: DE; Djibouti: DJ; Denmark: DK; Dominica: DM; Dominican
Republic: DO; Algeria: DZ; Eurasian Patent Organization (EAPO): EA; Ecuador: EC;
Estonia: EE; Egypt: EG; Western Sahara: EH; European Union Intellectual Property
Office (EUIPO): EM; European Patent Office (EPO): EP; Eritrea: ER; Spain: ES;
Ethiopia: ET; European Union: EU; Finland: FI; Fiji: FJ; Falkland Islands
(Malvinas): FK; Micronesia (Federated states of): FM; Faroe Islands: FO; France:
FR; Gabon: GA; United Kingdom: GB; Patent Office of the Cooperation Council for the
Arab States of the Gulf (GCC Patent Office): GC; Grenada: GD; Georgia: GE;
Guernsey: GG; Ghana: GH; Gibraltar: GI; Greenland: GL; Gambia: GM; Guinea: GN;
Equatorial Guinea: GQ; Greece: GR; South Georgia and South Sandwich Islands: GS;
Guatemala: GT; Guinea-Bissau: GW; Guyana: GY; Hong Kong, China: HK; Honduras: HN;
Croatia: HR; Haiti: HT; Hungary: HU; International Bureau of the World Intellectual
Property Organization (WIPO): IB; Indonesia: ID; Ireland: IE; Israel: IL; Isle of
Man: IM; India: IN; Iraq: IQ; Iran (Islamic Republic of): IR; Iceland: IS; Italy:
IT; Jersey: JE; Jamaica: JM; Jordan: JO; Japan: JP; Kenya: KE; Kyrgyzstan: KG;
Cambodia: KH; Kiribati: KI; Comoros: KM; Saint Kitts and Nevis: KN; Democratic
People's Republic of Korea: KP; Republic of Korea: KR; Kuwait: KW; Cayman Islands:
KY; Kazakhstan: KZ; Lao People's Democratic Republic: LA; Lebanon: LB; Saint Lucia:
LC; Liechtenstein: LI; Sri Lanka: LK; Liberia: LR; Lesotho: LS; Lithuania: LT;
Luxembourg: LU; Latvia: LV; Libya: LY; Morocco: MA; Monaco: MC; Republic of
Moldova: MD; Montenegro: ME; Madagascar: MG; Marshall Islands: MH; North Macedonia:
MK; Mali: ML; Myanmar: MM; Mongolia: MN; Macao, China: MO; Northern Mariana
Islands: MP; Mauritania: MR; Montserrat: MS; Malta: MT; Mauritius: MU; Maldives:
MV; Malawi: MW; Mexico: MX; Malaysia: MY; Mozambique: MZ; Namibia: NA; Niger: NE;
Nigeria: NG; Nicaragua: NI; Netherlands (Kingdom of the): NL; Norway: NO; Nepal:
NP; Nauru: NR; Niue: NU; New Zealand: NZ; African Intellectual Property
Organization (OAPI): OA; Oman: OM; Panama: PA; Peru: PE; Papua New Guinea: PG;
Philippines: PH; Pakistan: PK; Poland: PL; Portugal: PT; Palau: PW; Paraguay: PY;
Qatar: QA; Community Plant Variety Office (European Union) (CPVO): QZ; Romania: RO;
Serbia: RS; Russian Federation: RU; Rwanda: RW; Saudi Arabia: SA; Solomon Islands:
```

```
SB; Seychelles: SC; Sudan: SD; Sweden: SE; Singapore: SG; Saint Helena, Ascension
and Tristan da Cunha: SH; Slovenia: SI; Slovakia: SK; Sierra Leone: SL; San Marino:
SM; Senegal: SN; Somalia: SO; Suriname: SR; South Sudan: SS; Sao Tome and Principe:
ST; El Salvador: SV; Sint Maarten (Dutch part): SX; Syrian Arab Republic: SY;
Eswatini: SZ; Turks and Caicos Islands: TC; Chad: TD; Togo: TG; Thailand: TH;
Tajikistan: TJ; Timor-Leste: TL; Turkmenistan: TM; Tunisia: TN; Tonga: TO; Türkiye:
TR; Trinidad and Tobago: TT; Tuvalu: TV; Taiwan, Province of China: TW; United
Republic of Tanzania: TZ; Ukraine: UA; Uganda: UG; United States of America: US;
Uruguay: UY; Uzbekistan: UZ; Holy See: VA; Saint Vincent and the Grenadines: VC;
Venezuela (Bolivian Republic of): VE; British Virgin Islands: VG; Viet Nam: VN;
Vanuatu: VU; World Intellectual Property Organization (WIPO) (International Bureau
of): WO; Samoa: WS; Nordic Patent Institute (NPI): XN; International Union for the
Protection of New Varieties of Plants (UPOV): XU; Visegrad Patent Institute (VPI):
XV; Unknown states, other entities or organizations: XX; Yemen: YE; South Africa:
ZA; Zambia: ZM; Zimbabwe: ZW; Version: V2_0"        }, {
          "type" : "string",
           "enum" : [ "AN", "CS", "DL", "DD", "DT", "RH", "SU", "YD", "YU" ],
           "description" : "Version: V2_0; AN: Netherlands Antilles; CS:
Czechoslovakia; DL: German Democratic Republic; DD: German Democratic Republic; DT:
Federal Republic of Germany; RH: Southern Rhodesia; SU: Soviet Union; YD:
Democratic Yemen; YU: Yugoslavia/ Serbia and Montenegro"
         } ]
        },
        "st13ApplicationNumber" : {
          "type" : "string",
         "pattern" : "\\d{2}\\d{4}\\d{9}",
         "description" : "Description: Application number format recommended in
WIPO Standard ST.13. The sequence of indispensable elements in the application
number format is IP type (2 digits), year designation (4 digits) and serial number
(9 digits).; Version: V2_0"
        },
        "applicationNumberText" : {
          "type" : "string",
         "description" : "Description: Free format of application number; Version:
V2_0"
        }
      },
      "oneOf" : [ {
        "required" : [ "st13ApplicationNumber" ]
      }, {
        "required" : [ "applicationNumberText" ]
      } ]
    }
  }
}
```

22. L'approche de conception préférée consiste à séparer les données en groupe de petites composantes représentées par les modules de schéma, présentés dans le nouveau schéma de numéro de demande ci-après. Ce schéma JSON repose sur de plus petits modules de schéma JSON définis individuellement dans leurs propres schémas.

`applicationNumber.json` (Exemple de schéma modulaire)

```
{
  "$id" : "applicationNumber.json",
  "$schema" : "https://json-schema.org/draft/2020-12/schema",
  "type" : "object",
  "additionalProperties" : false,
  "properties" : {
    "applicationNumber" : {
      "$ref" : "#/$defs/applicationNumber"
    }
  },
  "required" : [ "applicationNumber" ],
  "$defs" : {
    "applicationNumber" : {
      "description" : "Description: Numbers used by IPOs in order to identify
each application received; Version: V2_0",
      "$ref" : "applicationNumberType.json#/$defs/applicationNumberType"

    }
  }
}
```

`applicationNumberType.json` (Exemple de schéma modulaire – suite)

```
{{
  "$id" : "applicationNumberType.json",
  "$schema" : "https://json-schema.org/draft/2020-12/schema",
  "type" : "object",
  "additionalProperties" : false,
  "$defs" : {
    "applicationNumberType" : {
      "type" : "object",
      "unevaluatedProperties" : false,
      "description" : "Version: V2_0",
      "properties" : {
        "ipOfficeCode" : {
          "$ref" : "ipOfficeCode.json#/$defs/ipOfficeCode"
        }
      },
      "allOf" : [ {
        "oneOf" : [ {
          "allOf" : [ {
            "properties" : {
              "st13ApplicationNumber" : {
                "$ref" :
"st13ApplicationNumber.json#/$defs/st13ApplicationNumber"
              }
            },
            "required" : [ "st13ApplicationNumber" ]
          }, {
            "not" : {
              "required" : [ "applicationNumberText" ]
            }
          } ]
        }, {
          "allOf" : [ {
            "properties" : {
              "applicationNumberText" : {
                "$ref" :
"applicationNumberText.json#/$defs/applicationNumberText"
              }
            },
            "required" : [ "applicationNumberText" ]
```

```
          }, {
            "not" : {
              "required" : [ "st13ApplicationNumber" ]
            }
          } ]
        } ]
      } ]
    }
  }
}
```

`ipOfficeCode.json` (Exemple de schéma modulaire – suite)

```
{
  "$id" : "ipOfficeCode.json",
  "$schema" : "https://json-schema.org/draft/2020-12/schema",
  "type" : "object",
  "additionalProperties" : false,
  "properties" : {
    "ipOfficeCode" : {
      "$ref" : "#/$defs/ipOfficeCode"
    }
  },
  "required" : [ "ipOfficeCode" ],
  "$defs" : {
    "ipOfficeCode" : {

      "description" : " Description: Two-letter alphabetic codes which represent
the names of states, other entities and intergovernmental organizations the
legislation of which provides for the protection of IP rights or which
organizations are acting in the framework of a treaty in the field of IP;
Version: V2_0",
"$ref" : "extendedWIPOST3CodeType.json#/$defs/extendedWIPOST3CodeType",

    }
  }
}
```

`extendedWIPOST3CodeType.json` (Exemple de schéma modulaire – suite)

```
{
  "$id" : "extendedWIPOST3CodeType.json",
  "$schema" : "https://json-schema.org/draft/2020-12/schema",
  "type" : "object",
  "additionalProperties" : false,
  "$defs" : {
    "extendedWIPOST3CodeType" : {
      "description" : "Version: V2_0",
      "anyOf" : [ {
        "$ref" : "wipoST3CodeType.json#/$defs/wipoST3CodeType"
      }, {
        "$ref" : "wipoFormerST3CodeType.json#/$defs/wipoFormerST3CodeType"
      }, {
        "$ref" : "igoCodeType.json#/$defs/igoCodeType"
      } ]
    }
  }
}
```

`wipoST3CodeType.json` (Exemple de schéma modulaire – suite)

```
{
  "$id" : "wipoST3CodeType.json",
  "$schema" : "https://json-schema.org/draft/2020-12/schema",
  "type" : "object",
  "additionalProperties" : false,
  "$defs" : {
    "wipoST3CodeType" : {
      "type" : "string",
      "description" : "Description: Andorra: AD; United Arab Emirates: AE;
Afghanistan: AF; Antigua And Barbuda: AG; Anguilla: AI; Albania: AL; Armenia: AM;
Angola: AO; African Regional Intellectual Property Organization (ARIPO): AP;
Argentina: AR; Austria: AT; Australia: AU; Aruba: AW; Azerbaijan: AZ; Bosnia and
Herzegovina: BA; Barbados: BB; Bangladesh: BD; Belgium: BE; Burkina Faso: BF;
Bulgaria: BG; Bahrain: BH; Burundi: BI; Benin: BJ; Bermuda: BM; Brunei
Darussalam: BN; Bolivia (Plurinational State of): BO; Bonaire, Sint Eustatius and
Saba: BQ; Brazil: BR; Bahamas: BS; Bhutan: BT; Bouvet Island: BV; Botswana: BW;
Benelux Office for Intellectual Property (BOIP): BX; Belarus: BY; Belize: BZ;
Canada: CA; Democratic Republic of the Congo: CD; Central African Republic: CF;
Congo: CG; Switzerland: CH; Côte d'Ivoire: CI; Cook Islands: CK; Chile: CL;
Cameroon: CM; China: CN; Colombia: CO; Costa Rica: CR; Cuba: CU; Cabo Verde: CV;
Curaçao: CW; Cyprus: CY; Czechia: CZ; Germany: DE; Djibouti: DJ; Denmark: DK;
Dominica: DM; Dominican Republic: DO; Algeria: DZ; Eurasian Patent Organization
(EAPO): EA; Ecuador: EC; Estonia: EE; Egypt: EG; Western Sahara: EH; European
Union Intellectual Property Office (EUIPO): EM; European Patent Office (EPO): EP;
Eritrea: ER; Spain: ES; Ethiopia: ET; European Union: EU; Finland: FI; Fiji: FJ;
Falkland Islands (Malvinas): FK; Micronesia (Federated states of): FM; Faroe
Islands: FO; France: FR; Gabon: GA; United Kingdom: GB; Patent Office of the
Cooperation Council for the Arab States of the Gulf (GCC Patent Office): GC;
Grenada: GD; Georgia: GE; Guernsey: GG; Ghana: GH; Gibraltar: GI; Greenland: GL;
Gambia: GM; Guinea: GN; Equatorial Guinea: GQ; Greece: GR; South Georgia and
South Sandwich Islands: GS; Guatemala: GT; Guinea-Bissau: GW; Guyana: GY; Hong
Kong, China: HK; Honduras: HN; Croatia: HR; Haiti: HT; Hungary: HU; International
Bureau of the World Intellectual Property Organization (WIPO): IB; Indonesia: ID;
Ireland: IE; Israel: IL; Isle of Man: IM; India: IN; Iraq: IQ; Iran (Islamic
Republic of): IR; Iceland: IS; Italy: IT; Jersey: JE; Jamaica: JM; Jordan: JO;
Japan: JP; Kenya: KE; Kyrgyzstan: KG; Cambodia: KH; Kiribati: KI; Comoros: KM;
Saint Kitts and Nevis: KN; Democratic People's Republic of Korea: KP; Republic of
Korea: KR; Kuwait: KW; Cayman Islands: KY; Kazakhstan: KZ; Lao People's
Democratic Republic: LA; Lebanon: LB; Saint Lucia: LC; Liechtenstein: LI; Sri
Lanka: LK; Liberia: LR; Lesotho: LS; Lithuania: LT; Luxembourg: LU; Latvia: LV;
Libya: LY; Morocco: MA; Monaco: MC; Republic of Moldova: MD; Montenegro: ME;
Madagascar: MG; Marshall Islands: MH; North Macedonia: MK; Mali: ML; Myanmar: MM;
Mongolia: MN; Macao, China: MO; Northern Mariana Islands: MP; Mauritania: MR;
Montserrat: MS; Malta: MT; Mauritius: MU; Maldives: MV; Malawi: MW; Mexico: MX;
Malaysia: MY; Mozambique: MZ; Namibia: NA; Niger: NE; Nigeria: NG; Nicaragua: NI;
Netherlands (Kingdom of the): NL; Norway: NO; Nepal: NP; Nauru: NR; Niue: NU; New
Zealand: NZ; African Intellectual Property Organization (OAPI): OA; Oman: OM;
Panama: PA; Peru: PE; Papua New Guinea: PG; Philippines: PH; Pakistan: PK;
Poland: PL; Portugal: PT; Palau: PW; Paraguay: PY; Qatar: QA; Community Plant
Variety Office (European Union) (CPVO): QZ; Romania: RO; Serbia: RS; Russian
Federation: RU; Rwanda: RW; Saudi Arabia: SA; Solomon Islands: SB; Seychelles:
SC; Sudan: SD; Sweden: SE; Singapore: SG; Saint Helena, Ascension and Tristan da
Cunha: SH; Slovenia: SI; Slovakia: SK; Sierra Leone: SL; San Marino: SM; Senegal:
SN; Somalia: SO; Suriname: SR; South Sudan: SS; Sao Tome and Principe: ST; El
Salvador: SV; Sint Maarten (Dutch part): SX; Syrian Arab Republic: SY; Eswatini:
SZ; Turks and Caicos Islands: TC; Chad: TD; Togo: TG; Thailand: TH; Tajikistan:
TJ; Timor-Leste: TL; Turkmenistan: TM; Tunisia: TN; Tonga: TO; Türkiye: TR;
Trinidad and Tobago: TT; Tuvalu: TV;
```

```
Taiwan, Province of China: TW; United Republic of Tanzania: TZ; Ukraine: UA;
Uganda: UG; United States of America: US; Uruguay: UY; Uzbekistan: UZ; Holy See:
VA; Saint Vincent and the Grenadines: VC; Venezuela (Bolivian Republic of): VE;
British Virgin Islands: VG; Viet Nam: VN; Vanuatu: VU; World Intellectual
Property Organization (WIPO) (International Bureau of): WO; Samoa: WS; Nordic
Patent Institute (NPI): XN; International Union for the Protection of New
Varieties of Plants (UPOV): XU; Visegrad Patent Institute (VPI): XV; Unknown
states, other entities or organizations: XX; Yemen: YE; South Africa: ZA; Zambia:
ZM; Zimbabwe: ZW; Version: V2_0",
      "enum" : [ "AD", "AE", "AF", "AG", "AI", "AL", "AM", "AO", "AP", "AR",
"AT", "AU", "AW", "AZ", "BA", "BB", "BD", "BE", "BF", "BG", "BH", "BI", "BJ",
"BM", "BN", "BO", "BQ", "BR", "BS", "BT", "BV", "BW", "BX", "BY", "BZ", "CA",
"CD", "CF", "CG", "CH", "CI", "CK", "CL", "CM", "CN", "CO", "CR", "CU", "CV",
"CW", "CY", "CZ", "DE", "DJ", "DK", "DM", "DO", "DZ", "EA", "EC", "EE", "EG",
"EH", "EM", "EP", "ER", "ES", "ET", "EU", "FI", "FJ", "FK", "FM", "FO", "FR",
"GA", "GB", "GC", "GD", "GE", "GG", "GH", "GI", "GL", "GM", "GN", "GQ", "GR",
"GS", "GT", "GW", "GY", "HK", "HN", "HR", "HT", "HU", "IB", "ID", "IE", "IL",
"IM", "IN", "IQ", "IR", "IS", "IT", "JE", "JM", "JO", "JP", "KE", "KG", "KH",
"KI", "KM", "KN", "KP", "KR", "KW", "KY", "KZ", "LA", "LB", "LC", "LI", "LK",
"LR", "LS", "LT", "LU", "LV", "LY", "MA", "MC", "MD", "ME", "MG", "MH", "MK",
"ML", "MM", "MN", "MO", "MP", "MR", "MS", "MT", "MU", "MV", "MW", "MX", "MY",
"MZ", "NA", "NE", "NG", "NI", "NL", "NO", "NP", "NR", "NU", "NZ", "OA", "OM",
"PA", "PE", "PG", "PH", "PK", "PL", "PT", "PW", "PY", "QA", "QZ", "RO", "RS",
"RU", "RW", "SA", "SB", "SC", "SD", "SE", "SG", "SH", "SI", "SK", "SL", "SM",
"SN", "SO", "SR", "SS", "ST", "SV", "SX", "SY", "SZ", "TC", "TD", "TG", "TH",
"TJ", "TL", "TM", "TN", "TO", "TR", "TT", "TV", "TW", "TZ", "UA", "UG", "US",
"UY", "UZ", "VA", "VC", "VE", "VG", "VN", "VU", "WO", "WS", "XN", "XU", "XV",
"XX", "YE", "ZA", "ZM", "ZW" ]
    }
  }
}
```

`wipoFormerST3CodeType.json` (Exemple de schéma modulaire – suite)

```
{
  "$id" : "wipoFormerST3CodeType.json",
  "$schema" : "https://json-schema.org/draft/2020-12/schema",
  "type" : "object",
  "additionalProperties" : false,
  "$defs" : {
    "wipoFormerST3CodeType" : {
      "type" : "string",
      "description" : "Description: Netherlands Antilles: AN; Czechoslovakia:
CS; German Democratic Republic: DD; Federal Republic of Germany: DT; Southern
Rhodesia: RH; Soviet Union: SU; Democratic Yemen: YD; Yugoslavia/ Serbia and
Montenegro: YU; Version: V2_0",
      "enum" : [ "AN", "CS", "DD", "DT", "RH", "SU", "YD", "YU" ]
    }
  }
}
```

`st13ApplicationNumber.json` (Exemple de schéma modulaire – suite)

```
{
  "$id" : "st13ApplicationNumber.json",
  "$schema" : "https://json-schema.org/draft/2020-12/schema",
  "type" : "object",
  "additionalProperties" : false,
  "properties" : {
    "st13ApplicationNumber" : {
      "$ref" : "#/$defs/st13ApplicationNumber"
    }
  },
  "required" : [ "st13ApplicationNumber" ],
  "$defs" : {
    "st13ApplicationNumber" : {
      "$ref" :
"st13ApplicationNumberType.json#/$defs/st13ApplicationNumberType",
      "description" : "Description: Application number format recommended in
WIPO Standard ST.13. The sequence of indispensable elements in the application
number format is IP type (2 digits), year designation (4 digits) and serial
number (9 digits).; Version: V2_0"
    }
  }
}
```

`st13ApplicationNumberType.json` (Exemple de schéma modulaire – suite)

```
{
  "$id" : "st13ApplicationNumberType.json",
  "$schema" : "https://json-schema.org/draft/2020-12/schema",
  "type" : "object",
  "additionalProperties" : false,
  "$defs" : {
    "st13ApplicationNumberType" : {
      "type" : "string",      "description" : "Version: V2_0",

      "pattern" : "\\d{2}\\d{4}\\\\w{2}d{7}"
    }
  }
}
```

`applicationNumberText.json` (Exemple de schéma modulaire – suite)

```
{
  "$id" : "applicationNumberText.json",
  "$schema" : "https://json-schema.org/draft/2020-12/schema",
  "type" : "object",
  "additionalProperties" : false,
  "properties" : {
    "applicationNumberText" : {
      "$ref" : "#/$defs/applicationNumberText"
    }
  },
  "required" : [ "applicationNumberText" ],
  "$defs" : {
    "applicationNumberText" : {
      "type" : "string",
      "description" : "Description: Free format of application number; Version:
V2_0"
    }
  }
}
```

23. Les schémas JSON devraient utiliser le mot clé "`$defs`" pour créer des définitions générales pour les propriétés et leurs contenus qui peuvent être réutilisés, comme l'illustre l'exemple ci-dessus. Cela correspond à peu près à la création de déclarations générales d'éléments et de types nommés dans un schéma XML.

[JSD-04] Les schémas JSON DEVRAIENT utiliser le mot clé "`$defs`" qui inclut une définition réutilisable pour chaque propriété ou type de propriété.

[JSD-05] Les développeurs DOIVENT utiliser les schémas JSON existants qui sont définis à l'annexe II de la présente norme, le cas échéant, avant d'en créer de nouveaux.

[JSD-06] Les développeurs DEVRAIENT créer de nouveaux schémas JSON seulement après avoir déterminé qu'aucun schéma JSON existant ne décrivait de manière adéquate la structure en question.

Documentation

24. Les schémas JSON devraient être autodescriptifs. Les développeurs devraient viser à concevoir des noms de structure JSON qui ont un sens. En outre, le schéma JSON devrait avoir une documentation décrivant le schéma et les structures JSON. Pour favoriser la réutilisabilité en restant général, le schéma JSON ne devrait pas fournir de documentation sur des détails de mise en œuvre relatifs à un système spécifique.

[JSD-07] La documentation NE DEVRAIT PAS décrire des détails ou d'autres informations qui ne sont pas directement liés à la signification de la structure.

25. Une description de schéma JSON permet à un développeur de schémas de discerner facilement le but, l'utilisation et le contenu d'un schéma. Cette information est très utile lorsqu'un développeur doit choisir un schéma à utiliser comme canevas pour la création d'un autre schéma.

[JSD-08] Les schémas JSON DEVRAIENT inclure la documentation relative au schéma JSON en utilisant le mot clé "`description`".

[JSD-09] Les éléments énumérés dans le tableau 1 ci-après DEVRAIENT être inclus dans la documentation de tous les schémas JSON.

Tableau 1. Éléments de la description du schéma JSON

Nom de l'élément de la description	Valeur	Requis/Facultatif
`Description`	Description textuelle claire de l'information décrite par le schéma.	Requis, sauf pour les schémas JSON provenant de types simples de définitions de schémas XML de la norme ST.96 pour lesquels une description n'est pas fournie, tels que `DateType`.
`Version`	Nombre de versions majeures et mineures du schéma	Requis
`SchemaCreatedDate`	Date de création du schéma	Facultatif
`SchemaLastModifiedDate`	Date de la dernière modification du schéma	Facultatif
`SchemaContactPoint`	Nom ou coordonnées de l'organisation à contacter pour toute question relative au schéma	Facultatif
`SchemaReleaseNoteURL`	Emplacement où les notes de diffusion du schéma sont publiées	Facultatif

[JSD-10] Les éléments de la description tels que la date de création du schéma et le numéro de version ci-dessus DEVRAIENT être séparés par des points-virgules, avec des espaces possibles après le point-virgule, correspondre à la valeur associée au mot clé "`description`" comme il ressort de l'exemple suivant :

Exemple de description pour `applicationBody` (Schéma au niveau du document)

```
"description" : "Description: Body of a patent application; SchemaCreatedDate:
2012-07-13; SchemaLastModifiedDate: 2026-06-09; SchemaContactPoint:
standards@wipo.int; SchemaReleaseNoteURL:
https://www.wipo.int/standards/en/json_material/st97/v2-0/release-notes.html;
Version: V2_0 "
```

Exemple de description pour `ipOfficeCode` (Schéma pas au niveau du document)

```
"description" : " Description: Two-letter alphabetic codes which represent the
names of states, other entities and intergovernmental organizations the legislation
of which provides for the protection of IP rights or which organizations are acting
in the framework of a treaty in the field of IP; Version: V2_0"
```

Exemple de description pour `appellateBodyCategoryType.json` (Schéma de définitions de type énumération)

```
"description" : " Office appeal board: Appeal board within the IP office; Court:
Court; Appeal Court: Second instance court; Supreme Court: Highest appellate court,
Version: V2_0;"
```

Exemple de description de schéma de définition de type

```
"description" : "Version: V2_0"
```

Conventions de nommage des fichiers d'un schéma

26. Le nom du fichier JSON Schema doit respecter la convention relative aux caractères minuscules de type "camel".

27. Les noms de fichier d'un schéma et les noms d'un schéma sont souvent associés. Les noms de fichier d'un schéma reposent sur les noms de schéma correspondants. C'est ainsi, par exemple, que le nom de fichier `postalAddressType.json` vient du nom de schéma `postalAddressType`. Par conséquent, les conventions de nommage des fichiers de schéma sont liées aux règles des conventions de nommage JSON dans la présente norme.

28. Un fichier de schéma peut contenir les informations de la version, qui consistent soit en des indicateurs de version majeure et mineure, soit en des indicateurs de version majeure, mineure et corrective. Les informations de la version s'appliquent uniquement aux schémas au niveau du document. Un schéma qui en est au stade de projet peut être révisé. Les projets de schéma doivent être indiqués comme tels dans le nom de fichier du schéma, au moyen de la lettre "D" et du numéro de la révision. Des versions d'essai peuvent être fournies à titre exceptionnel uniquement en cas de besoin commercial particulier et elles ne seront pas considérées comme une version officielle et ne seront donc pas mises à la disposition du public. Une version d'essai sera basée sur une version officielle, mais pourra comprendre des composantes de schéma de différentes versions officielles comme moyen de transition vers une version officielle. Les versions d'essai DOIVENT être désignées comme telles dans le nom de fichier du schéma, en ajoutant la lettre "T" au numéro de révision à côté du numéro de la version de base.

[JSD-11] Les caractères utilisés dans les noms de fichier de schéma DOIVENT appartenir à la série suivante : 'a-z, A-Z, 0-9, caractère de soulignement "_", et point "."'.

[JSD-12] Un nom de fichier de schéma DOIT comporter deux parties obligatoires avec un délimiteur et les informations facultatives de la version avec deux délimiteurs, c'est-à-dire : `<component name>{"_"V"<major version number>"_"<minor version number>"}."<file extension>;`. Par exemple, `emailAddressType.json`, `languageCode.json`, `applicationBody_V2_0.json`.

[JSD-13] Un nom de fichier du projet de schéma DOIT comporter quatre parties obligatoires avec deux délimiteurs et les informations facultatives de la version avec deux délimiteurs supplémentaires, c'est-à-dire : `<component name>{"_"V"<major version number>"_"<minor version number>"}_"D"<revision number>"."<file extension>`, par exemple, `trademarkApplication_V2_0_D1.json`. Si un projet de schéma est fondé sur un schéma existant et que son nom de fichier inclut les informations de la version, les numéros des versions majeure et mineure dans le nom de fichier du projet de schéma DEVRAIENT être ceux qui sont indiqués dans le fichier de schéma sur lequel est fondé le projet de schéma. Si le projet de schéma est nouveau, le numéro de la version majeure dans le nom de fichier du projet de schéma DEVRAIT être celui qui est indiqué dans le domaine de la propriété intellectuelle correspondant et un numéro de version mineure dans le fichier du projet de schéma DEVRAIT être zéro "0".

Structure des propriétés du schéma JSON

29. Les schémas JSON devraient avoir la propriété `"type" : "object"` afin de garantir que le format JSON soit utilisé uniquement pour les structures imbriquées et non pour les valeurs individuelles. À partir de l'exemple `applicationNumber.json` ci-après :

```
{
  "$id" : "applicationNumber.json",
  "$schema" : "https://json-schema.org/draft/2020-12/schema",
  "type" : "object",
  "additionalProperties" : false,
  "properties" : {
    "applicationNumber" : {
      "$ref" : "#/$defs/applicationNumber"
    }
  },
  "required" : [ "applicationNumber" ],
  "$defs" : {
    "applicationNumber" : {
      "description" : "Description: Numbers used by IPOs in order to identify each
application received; Version: V2_0",
      "$ref" : "applicationNumberType.json#/$defs/applicationNumberType"
    }
  }
}
```

[JSD-14] L'objet du schéma le plus extérieur DOIT comporter un mot clé "`type`" dont la valeur est "`object`".

[JSD-15] L'objet du schéma le plus extérieur DOIT comporter un mot clé "`$defs`" dont la valeur est la propriété de l'objet du schéma le plus extérieur.

[JSD-16] L'objet du schéma le plus extérieur DOIT comporter un mot clé "`required`" dont la valeur est un tableau qui contient un seul élément, par exemple, la propriété de l'objet du schéma le plus extérieur.

30. Les extensions de schéma JSON (personnalisations) peuvent être utilisées. Si le type étendu est un type objet, il convient de le référencer au lieu de dupliquer ses propriétés pour favoriser la réutilisabilité.

[JSD-17] Les extensions de schéma JSON (personnalisations) pour les types d'objet DOIVENT être appliquées en renvoyant au schéma JSON du type étendu.

RÈGLES DE CONCEPTION DES STRUCTURES DU SCHÉMA JSON

Aperçu

31. La présente section établit les règles relatives aux structures du schéma JSON, en particulier les tableaux, les objets et les valeurs primitives. La normalisation des noms relatifs aux structures du schéma est essentielle à l'élaboration d'une architecture de données solide.

Propriétés

32. Les propriétés, aussi appelées membres, sont le socle d'une structure JSON.

[JSC-01] Les définitions DEVRAIENT utiliser dans toute la mesure du possible des schémas existants.

[JSC-02] Les propriétés multiples qui peuvent logiquement être groupées PEUVENT être déclarées dans un seul fichier de schéma dans la définition générale.

Définitions

33. Chaque propriété devrait avoir une définition générale qui est définie dans son schéma JSON. Cela permettra de réutiliser le nom de propriété dans de nombreux parents et d'avoir une définition cohérente pour l'ensemble d'entre eux. Veuillez consulter l'exemple de la propriété "`applicationNumber`" ci-après.

[JSC-03] Chaque propriété énumérée dans un mot clé de propriété DEVRAIT renvoyer à une définition générale qui est définie dans le mot clé "`$defs`". La définition générale DEVRAIT avoir le même nom que la propriété.

Exemple de propriété renvoyant à une définition générale

```
{
  "$id" : "applicationNumber.json",
  "$schema" : "https://json-schema.org/draft/2020-12/schema",
  "type" : "object",
  "additionalProperties" : false,
  "properties" : {
    "applicationNumber" : {
      "$ref" : "#/$defs/applicationNumber"
    }
  },
  "required" : [ "applicationNumber" ],
  "$defs" : {
    "applicationNumber" : {

      "description" : "Description: Numbers used by IPOs in order to identify each
application received; Version: V2_0"
      "$ref" : "applicationNumberType.json#/$defs/applicationNumberType",

    }
  }
}
```

34. La définition générale d'une propriété devrait comprendre le nom de fichier et la description de la propriété.

[JSC-04] La définition générale d'une propriété DEVRAIT comprendre le nom de fichier et la "`description`" de la propriété.

35. Les propriétés doivent avoir des types. Ils peuvent être définis directement s'il s'agit de types primitifs (à l'exception des objets) ou renvoyer à une définition générale de propriété dans un autre schéma JSON.

[JSC-05] Une propriété DOIT avoir un type qui est spécifié en utilisant le mot clé "`type`", soit comme propriété directe soit par renvoi à une définition générale.

Définitions de types

36. Les schémas JSON peuvent définir des définitions de type réutilisables auxquelles renvoient les définitions générales de propriété. Ces définitions réutilisables de types devraient comprendre un mot clé "`type`", un mot clé "`properties`" (si le type est "`object`") et toute autre contrainte en matière de valeur.

[JSC-06] Un schéma PEUT définir des définitions générales de type afin de réutiliser les modèles de contenu pour de nombreuses propriétés.

Définition de type réutilisable

```
{
  "$id" : "applicationNumber.json",
  "$schema" : "https://json-schema.org/draft/2020-12/schema",
  "type" : "object",
  "additionalProperties" : false,
  "properties" : {
    "applicationNumber" : {
      "$ref" : "#/$defs/applicationNumber"
    }
  },
  "required" : [ "applicationNumber" ],
  "$defs" : {
    "applicationNumber" : {
      "description" : "Description: Numbers used by IPOs in order to identify each
application received; Version: V2_0",
      "$ref" : "applicationNumberType.json#/$defs/applicationNumberType"
    }
  }
}
```

[JSC-07] Les définitions qui représentent des types DOIVENT avoir des noms qui figurent sous la forme LCC + suffixe "type".

Type primitif JSON

[JSC-08] Le type primitif JSON pertinent le plus précis DEVRAIT être utilisé pour une propriété. Les types primitifs comprennent : "`string`", "`number`", "`integer`", "`object`", "`array`", "`boolean`" et "`null`".

37. Par exemple, si une valeur de propriété est un nombre entier, le type "`integer`" devrait être préféré au type "`number`", qui est plus générique, ou au type "`string`", qui est plus permissif. Pour le type "`string`", les formats incorporés devraient être utilisés s'il y a lieu, par exemple "`date-time`" ou "`duration`".

Listes de codes

38. Dans certains cas, il est intéressant de limiter une valeur à une liste de codes normalisés et acceptables à des fins d'échange de données. Les listes de codes sont un moyen de créer un vocabulaire contrôlé de valeurs autorisées pour un élément de donnée (par exemple, une liste de codes normalisés pour les codes de noms de pays, les codes de langue, les codes d'office de propriété intellectuelle, etc.). Les listes de codes qui existent déjà dans le domaine public et dont la gestion est assurée par des comités des normes compétents tels que l'ISO devraient être utilisées.

[JSC-09] La norme ST.3 de l'OMPI DOIT être utilisée pour la représentation des offices de propriété intellectuelle, des États, d'autres entités, d'organisations, ainsi que pour les pays/organisations désignés et prioritaires.

[JSC-10] La norme ISO 3166-1-Alpha-2 Éléments de code (codes de pays à deux lettres) DOIT être utilisée pour la représentation des noms de pays dans les informations relatives à l'adresse ou à la nationalité.

[JSC-11] La norme ISO 639-1 (codes de langue à deux lettres) DOIT être utilisée pour les codes de langue.

[JSC-12] La norme ISO 4217-Alpha (codes pour la représentation des monnaies à trois lettres) DOIT être utilisée pour les codes des monnaies.

[JSC-13] Le mot clé JSON `enum` DEVRAIT être utilisé pour définir les listes de codes.

[JSC-14] Les caractères utilisés dans les valeurs d'énumération DOIVENT se limiter aux caractères suivants : {`a-z, A-Z, 0-9`, point (.), virgule (,), espaces, tiret (-) et caractère de soulignement (_)}.

Tableaux

39. Le terme cardinalité est défini comme étant le nombre d'éléments dans un tableau. La cardinalité est indiquée dans un schéma à l'aide des mots clés `minItems` et `maxItems`. Il est recommandé que les développeurs de schémas

ne précisent pas de valeurs implicites pour les indicateurs d'occurrence (c'est-à-dire "`minItems`" : 0) parce que cela pourrait encombrer inutilement un schéma.

[JSC-15] Les schémas JSON DEVRAIENT utiliser les mots clés `minItems` et `maxItems` pour les tableaux, sauf pour la valeur implicite de `minItems` (0).

40. Le type d'élément dans un tableau doit être défini en utilisant le mot clé "`items`". Par souci de simplicité, tous les éléments d'un tableau doivent avoir le même type. Si on souhaite avoir une séquence d'objets de différents types, ces types doivent être définis en tant que propriétés distinctes d'un objet.

[JSC-16] Pour chaque objet de type tableau, il DOIT y avoir un mot clé "`items`" et sa valeur DOIT être un seul objet de schéma et non un tableau. Tous les éléments d'un tableau DOIVENT avoir le même type.

41. Le mot clé "`additionalItems`" ne doit pas être utilisé pour des tableaux, car il n'est pas pertinent lorsque la valeur des "`items`" est un seul objet de schéma.

[JSC -17] Le mot clé "`additionalItems`" ne DEVRAIT pas être utilisé quand "`items`" est un seul objet de schéma.

Objets

"Caractères génériques" en matière de propriété

42. Les schémas JSON ne devraient pas permettre à des propriétés arbitraires d'être incluses dans l'instance JSON tout en restant valables, car cela pourrait nuire à l'intégrité de l'échange de données.

Le mot clé "`additionalProperties`" doit être utilisé avec la valeur "`false`". À défaut, des propriétés non définies seront autorisées dans les instances.

[JSC-18] Un schéma JSON DOIT utiliser "`additionalProperties`" avec la valeur "`false`" pour chaque objet.

43. L'utilisation du mot-clé "`patternProperties`" n'est pas autorisée, sauf si le schéma est de type "contenu mixte". Ce mot clé permet de mettre en correspondance des expressions régulières et des schémas. Par exemple, elle autorise des définitions de schéma implicites reposant sur un nom de propriété. À partir d'un type de nom de propriété particulier, un schéma particulier est appliqué.

[JSC-19] Un schéma NE DOIT PAS utiliser le mot clé "`patternProperties`" sauf si le schéma est de type "contenu mixte".

Ordre des propriétés

44. Le schéma JSON n'applique pas d'ordre particulier pour les propriétés d'un objet. Cependant, si le schéma JSON a un schéma XML correspondant, il est recommandé d'énumérer les propriétés dans le même ordre que dans le schéma XML, aussi bien dans le schéma que dans l'instance JSON.

[JSC-20] Un schéma DEVRAIT utiliser le même ordre de propriété que dans le schéma XML correspondant, s'il en existe un.

IDENTIFICATEURS DE SCHÉMA JSON

Aperçu

45. Un ID dans un schéma JSON fournit un URI qui identifie une catégorie d'information reposant sur un domaine commercial (par exemple, entreprise, brevets, et marques). La présente proposition de norme a choisi d'utiliser des relations plusieurs-à-un entre les quelques ID et les centaines de structures JSON. Un groupe de structures JSON connexes avec des noms uniques sera associé à un ID particulier. Un uniform resource identifier (URI) devrait être utilisé pour l'identification.

[JID-01] Les ID DOIVENT être utilisés dans les schémas à l'aide du mot clé "`$id`".

RÈGLES DE CONCEPTION DES INSTANCES JSON

46. Le schéma JSON définit la structure et les contraintes pour l'instance JSON. Pour accroître les échanges de données (intra et inter offices) et en assurer la qualité, les instances JSON devraient être associées au schéma JSON pour en garantir la validité et la conformité des instances.

Ordre des propriétés

47. Le schéma JSON n'applique pas d'ordre particulier pour les propriétés d'un objet. Cependant, il est recommandé que les propriétés figurent dans le même ordre dans l'instance et dans le schéma JSON.

[JIN-01] Un document d'instance JSON DEVRAIT utiliser le même ordre de propriétés que dans le schéma JSON correspondant, s'il en existe un.

Validation de l'instance JSON

48. La validation d'instances JSON garantit que leur contenu remplisse toutes les exigences définies dans les schémas correspondants.

[JIN-02] Les documents d'instance JSON PEUVENT être validés au regard d'un schéma correspondant durant le traitement.

[JIN-03] Un schéma d'exécution PEUT être créé pour remplir les exigences en matière de performance de l'application dans l'environnement d'exécution. Par exemple, le schéma composé pour le numéro de demande peut être utilisé comme schéma d'exécution.

[JIN-04] Toute modification, mise à jour, révision et nouvelle diffusion DOIT d'abord être soumise à l'Équipe d'experts chargée des API pour approbation avant que les changements puissent être incorporés au schéma d'exécution.

Il est souhaitable qu'une instance JSON soit conforme à un schéma.

[JIN-05] Une instance JSON DEVRAIT être conforme à un schéma JSON particulier, lui-même conforme aux règles décrites dans la présente norme.

RÉFÉRENCES

49. Les normes et ressources ci-après présentent un intérêt aux fins de la présente norme :

Normes de l'OMPI

Norme ST.3 de l'OMPI	Codes à deux lettres pour la représentation des États, autres entités et organisations;
Norme ST.9 de l'OMPI	Données bibliographiques figurant sur les brevets ou se rapportant aux brevets ou aux CCP;
Norme ST.16 de l'OMPI	Identification de différents types de documents de brevet;
Norme ST.26 de l'OMPI	Présentation des listages des séquences de nucléotides et d'acides aminés;
Norme ST.27 de l'OMPI	Échange de données sur la situation juridique des brevets;
Norme ST.60 de l'OMPI	Données bibliographiques concernant les marques;
Norme ST.61 de l'OMPI	Échange de données sur la situation juridique des marques;
Norme ST.67 de l'OMPI	Gestion électronique des éléments figuratifs des marques;
Norme ST.68 de l'OMPI	Gestion électronique des marques sonores;
Norme ST.69 de l'OMPI	Gestion électronique des marques de mouvement et des marques multimédias;
Norme ST.80 de l'OMPI	Données bibliographiques relatives aux dessins et modèles industriels;
Norme ST.87 de l'OMPI	Échange de données sur la situation juridique des dessins et modèles industriels;
Norme ST.88 de l'OMPI	Représentations sous forme électronique des dessins et modèles industriels;
Norme ST.90 de l'OMPI	Traitement et communication des données de propriété intellectuelle aux API Web;
Norme ST.91 de l'OMPI	Modèles et images tridimensionnels (3D) numériques;
Norme ST.96 de l'OMPI	Utilisation du langage XML dans le traitement de l'information en matière de propriété intellectuelle.

Normes et spécifications industrielles

Spécification JSON :	http://www.ecma-international.org/publications/files/ECMA-ST/ECMA-404.pdf
JSON Schema Core, projet 2020-12 :	https://json-schema.org/draft/2020-12/json-schema-core
JSON Schema Validation, 2020-12 draft :	https://json-schema.org/draft/2020-12/json-schema-validation
Spécification Open API v3.2.0 :	https://spec.openapis.org/oas/v3.2.0.html

Norme internationale ISO/IEC 11179-5 "Technologies de l'information – Registres de métadonnées (RM) – Partie 5 : Principes de dénomination et d'identification";

Norme ISO 3166-1 "Codes pour la représentation des noms de pays et de leurs subdivisions – Codes pays";

Norme ISO 639-1 "Codes pour la représentation des noms de langue – Partie 1 : Alpha2-code"; Norme ISO 4217 "Codes pour la représentation des monnaies et des types de fonds";

Norme ISO/IEC 10646 "Technologies de l'information – Jeu universel de caractères codés sur plusieurs octets (JUC)";

Norme internationale ISO/IEC 21778:2017 "Technologies de l'information – Syntaxe d'échange de données JSON"

ECMA -404– Syntaxe d'échange de données JSON

UBL-2.1 Langage d'affaires universel – Langage d'affaires universel Version 2.1

Norme ISO/IEC 10646 "Technologies de l'information – Jeu universel de caractères codés sur plusieurs octets (JUC)";

"Appel à observations (RFC 2119) de l'Internet Engineering Task Force (IETF) – Mots clés à utiliser dans les RFC pour indiquer les niveaux de prescription".

[L'annexe I suit]