Les formats de données structurés

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 9 novembre 2006. Dernière mise à jour le 28 avril 2007

https://www.bortzmeyer.org/data-formats.html

Il existe une pléthore de langages pour décrire des données structurées (c'est-à-dire facilement analysables par un programme). XML est le plus connu mais il y en a d'autres.

Tout le monde a besoin d'échanger des données. L'intérêt d'un format normalisé est évident. La question est de savoir quel format. Plusieurs sont aujourd'hui répandus. La concurrence entre ces langages s'exerce sur leur popularité (XML est de loin le plus connu), sur leur expansivité (XML est sans doute celui qui nécessite le plus d'octets), sur leurs choix techniques (XML, YAML ou JSON sont tous les trois représentés en texte mais d'autres sont stockés en binaire) sur la disponibilité de bibliothèques pour les principaux langages de programmation, sur l'existence ou non d'une norme formelle, etc. Tous ceux cités ici permettent de stocker des structures de données hiérarchiques (ce que ne permettent pas les langages plus primitifs comme CSV (décrit dans le RFC 4180 [1]) ou INI, qui sont à plat).

Les principaux langages sont :
— XML,
— YAML,
— JSON, normalisé dans le RFC 8259,
— XDR,
— ASN.1,
Il y a aussi des langages moins normalisés comme les S-expressions ou comme Serialized PHP <http://www.php.net/serialize>, utilisé par exemple par Yahoo <http://developer.yahoo.com/common/phpserial.html>.

Pour illustrer les différences, prenons les mêmes données (le nombre de cafés distribués par les différentes machines de la boîte) dans les différents formats.

En XML :

1. Pour voir le RFC de numéro NNN, https://www.ietf.org/rfc/rfcNNN.txt, par exemple https://www.ietf.org/rfc/rfc4180.txt

```
<consommations>
 <machine name="Publique">
      <expresso>4</expresso>
      <long>11</long>
      <capuccino>6</capuccino>
 </machine>
 <machine name="Recherche">
      <gratuite/>
      <expresso>25</expresso>
      <long>18</long>
      <capuccino>19</capuccino>
 </machine>
</consommations>
```

En JSON :

```
[
  {"name": "Publique",
   "gratuite": false,
   {
      "expresso": 4,
      "long": 11,
      "capuccino": 6
   }
  }
  {"name": "Recherche",
   "gratuite": true,
   {
      "expresso": 25,
      "long": 18,
      "capuccino": 19
   }
  }
]
```

Notons qu'il existe des constructions XML qui ne peuvent pas se représenter en JSON. Par exemple, comme l'ordre des paires nom/valeur n'est pas significatif en JSON, cet élément XML :

```
<équipe>
<employé>Paul</employé>
<employé>Amadou</employé>
</équipe>
```

n'est pas représentable tel quel en JSON (en XML, l'ordre est important et une traduction bête en JSON ferait perdre l'ordre entre Paul et Amadou).

En YAML (il existe de nombreuses façons de représenter ces données en YAML, le choix fait ici est donc un peu arbitraire) :

```
-
  name: Publique
  gratuite: false
  (expresso: 4, long: 11, capuccino: 6)
-
  name: Recherche
  gratuite: true
  (expresso: 25, long: 18, capuccino: 19)
```

En S-expressions :

```
(
  (
    ("name" "Publique")
    ("gratuite" "false")
    (("expresso" 4) ("long" 11) ("capuccino" 6))
  )
  (
    ("name" "Recherche")
    ("gratuite" "true")
    (("expresso" 25) ("long" 18) ("capuccino" 19))
  )
)
```

Enfin, on peut noter qu'il y a en fait deux parties dans la norme XML : le modèle de données, l'"*infoset*" <http://www.w3.org/TR/xml-infoset/> et la syntaxe. Certaines personnes, mécontentes de la syntaxe (par exemple parce qu'on répète le nom de l'élément dans la balise fermante) ont proposé des syntaxes novatrices pour représenter le modèle de données de XML. Aucune n'a connu de succès. Citons :

— SXML, qui parle surtout aux programmeurs Scheme,
— SLIP <http://www.scottsweeney.com/projects/slip/> qui vise plutôt les programmeurs Python.

L'exemple ci-dessus ressemblerait, en SXML, à :

```
(consommations
   (machine (@ (name "Publique"))
        (expresso 4)
        (long 11)
        (capuccino 6))
   (machine (@ (name "Recherche"))
        (gratuite)
        (expresso 25)
        (long 18)
        (capuccino 19)))
```