XProc : un langage pour enchaîner des transformations XML

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 12 juillet 2007. Dernière mise à jour le 7 janvier 2008

https://www.bortzmeyer.org/xproc.html

Il est fréquent qu'un processus de traitement de données XML nécessite plusieurs étapes : inclure d'autres documents (parfois récupérés sur le réseau), transformer avec XSLT, valider, etc. Actuellement, ces étapes sont enchaînées par un programme écrit dans un langage impératif ou déclaratif. Peut-on le faire dans un langage XML ? On pourra bientôt avec XProc, un langage de chaîne ("*pipeline*") de traitement, actuellement à l'état de projet de norme.

Deux dates importantes pour XProc : le 6 juillet, sortie d'une nouvelle version du projet de norme <http://www.w3.org/TR/2007/WD-xproc-20070706/> et, le 10 juillet, sortie de la première version de la première mise en œuvre de XProc, "*XML Pipeline Processor*" <http://norman.walsh.name/2007/07/10/xproc001>.

Que fait le langage XProc ? Prenons une application qui a besoin d'effectuer plusieurs opérations sur des fichiers XML. Par exemple, le site Web www.langtag.net <http://www.langtag.net/> est écrit en XML et le XHTML qui est servi aux clients est produit par une transformation XSLT, suivie d'une validation pour s'assurer de la correction du XHTML. Comment enchaîner ces deux étapes ? Actuellement, c'est réalisé dans un Makefile :

```
%.html: %.xml ${STYLESHEET} language-subtag-registry-version
        xsltproc --stringparam lsr-version `cat language-subtag-registry-version` \
                 --output $@ ${STYLESHEET} $< && xmllint --noout --valid $@
```

Quand on voudra créer `index.html`, make va exécuter xsltproc (le processeur XSLT) puis (`&&` signifie « si la première étape a été un succès, exécuter ensuite la seconde » en shell Unix) xmllint (le validateur). Si une des étapes échoue, make s'arrêtera en erreur et le site Web ne sera pas mis à jour (ce qui garantit que son contenu sera 100 % valide).

Une autre solution, plutôt que le langage **déclaratif** de make aurait été d'utiliser un langage **impératif** comme Python ou Perl. Question de goût. Une autre solution est de sous-traiter cette tâche à un service en ligne comme " *Yahoo Pipes*", avec toutes les limitations qu'il présente (impossibilité de travailler sur des documents privés, par exemple). Mais il n'existait pas de solution tout-XML.

XProc est cette solution : c'est un langage XML qui permet de décrire une chaîne de traitement ("*pipeline*" dans la norme). Cette chaîne est composée d'opérations dont certaines sont pré-définies dans la "*Standard step library*". L'opération ci-dessus (transformation XSLT puis validation) aurait pu s'écrire, en XProc :

```
<proc:pipeline name="langtag.net" xmlns:proc="http://www.w3.org/2007/03/xproc">
  <proc:input port="source" primary="yes"/>
  <proc:input port="schemaDoc" sequence="yes" primary="no"/>
  <proc:input port="stylesheetDoc" sequence="yes" primary="no"/>
  <proc:output port="result"/>

  <proc:xslt>
    <proc:input port="stylesheet">
       <proc:pipe step="langtag.net" port="stylesheetDoc"/>
    </proc:input>
  <proc:xslt>

  <!-- Pas de validation avec DTD dans la bibliothèque standard, donc
  on utilise la version Relax NG de XHTML -->
  <proc:validate-relax-ng>
    <proc:input port="schema">
      <proc:pipe step="langtag.net" port="schemaDoc"/>
    </proc:input>
  </proc:validate-relax-ng>

</proc:pipeline>
```

La façon dont les ports d'entrée-sortie (ici, `source`, `schemaDoc` et `stylesheetDoc`) sont connectés aux vraies ressources dépend de l'implémentation.

Les paramètres des étapes sont nommés, par exemple `stylesheet` pour l'étape `<proc:xslt>` est nommée dans la déclaration de cette étape standard, dans la bibliothèque.

Si une étape de la chaîne (l'exemple ci-dessus est très simple, avec seulement deux étapes) échoue, le traitement s'arrête. XProc a un mécanisme de traitement d'exceptions, qui permet de continuer en cas d'erreur si on le souhaite. Voici l'exemple que donne la norme, pour récupérer les erreurs éventuelles provoquées par un échec dans la récupération d'une ressource en HTTP :

```
<proc:try xmlns:step="http://www.w3.org/2007/03/xproc-step">
  <proc:group>
    <proc:http-request>
      <proc:input port="source">
        <proc:inline>
          <step:http-request method="get" href="http://example.org/something"/>
        </proc:inline>
      </proc:input>
    </proc:http-request>
  </proc:group>
  <proc:catch>
    <proc:identity>
      <proc:input port="source">
        <proc:inline>
          <step:error>HTTP Request Failed</step:error>
        </proc:inline>
      </proc:input>
    </proc:identity>
  </proc:catch>
</proc:try>
```

Bien sûr, on peut trouver que le fichier XML est bien plus long que le court Makefile. Mais la comparaison est partiellement injuste. D'abord, le Makefile dépend de programmes extérieurs comme xsltproc. Si on veut le faire robuste et portable, il faudrait tester d'autres processeurs, le rendre capable d'utiliser plutôt sablotron ou saxon, etc. Au contraire, avec Xproc, si on n'utilise que la bibliothèque standard, on est sûr que cela fonctionne partout. D'autre part, il y a des gens comme moi, qui préfèrent éditer des Makefile et des gens qui préfèrent éditer du XML. Ceux-ci utilisent souvent un éditeur qui leur masque plus ou moins le nombre de lignes sous-jacent.

Je n'ai pas encore testé le programme `xproc` (les exemples XProc ci-dessus sont donc livrés sans garantie) et, de toute façon, la norme n'est pas encore finale. Mais XProc semble susceptible de beaucoup aider les programmeurs qui devront effectuer des opérations sur leurs fichiers XML.

Norman Walsh, un des principaux auteurs de la norme, et le développeur du programme xproc, a écrit plusieurs intéressants articles sur son blog à propos de la mise en œuvre du langage XProc, `<http://norman.walsh.name/2007/04/25/implXProcI.html>` et suivants.

Voici une liste des implémentations actuelles :
— XML Pipeline Processor `<https://xproc.dev.java.net/>`,
— yax `<http://yax.sourceforge.net/>`,
— xproc.xq `<http://code.google.com/p/xprocxq/>`, écrit en XQuery,
— xsp.net `<http://xsp-net.sourceforge.net/>`,
— Cocoatron `<http://www.cocoatron.com/>`.
Sinon, Alain Couthures attire mon attention sur son logiciel tXs `<http://www.agencexml.com/txs/fr>`, qui est largement sur le même créneau que XProc.