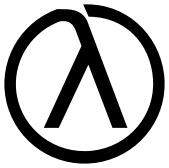

Les technologies web, à courant et à contre courant

Au cœur du web se trouve le protocole HTTP. Celui-ci a ses caractéristiques, qui sont les suivantes :

- Il est **déconnecté**.

  Quand deux nœuds du réseau communiquent l'un avec l'autre, ils n'ont pas à maintenir une connexion directe, à la différence de ce qui se passe pour deux téléphones durant une conversation.

- Il est **asynchrone**.

Diverses technologies « web » se sont succédées au cours de l'Histoire de ce dernier. Toutes n'ont pas eu le même respect des caractéristiques fondamentales du web, qui découlent de celles du protocole HTTP.

Pour faire simple, on peut considérer que la philosophie des technologies qui vont « dans le sens du courant » du web se résume à un terme : REST (pour « REpresentational State Transfer »). Ces technologies accèdent, via des URL, à des ressources en embrassant le caractère déconnecté et asynchrone du protocole HTTP (plutôt que d'essayer de le contourner) et en utilisant les différents verbes HTTP pour représenter leurs opérations.

1. Technologies qui vont dans le sens du courant

Voici un catalogue non exhaustif de technologies que l'on peut considérer comme fidèles à l'esprit dans lequel a été conçu le web.

Sites web HTML

Il s'agit là de la fonction première, historiquement, du web ; alors il eu été étonnant que cette technologie ne soit pas dans l'esprit du web.

Services web RESTful

Les services web RESTful sont la généralisation datant de 2000, concomitante avec la version 1.1 du protocole HTTP, de la notion de ressource web qui existait depuis 1996 avec le protocole HTTP 1.0. En d'autres mots, il s'agit de la version aboutie de 2000 de l'ébauche de 1996.

Rien ne peut être plus dans l'esprit du web qu'un service web RESTful, puisque c'est littéralement le modèle d'architecture qui défini l'esprit du web. Cette conformité à l'esprit du web tient pour l'essentiel en deux aspects :

- Une approche orientée **ressources** identifiées par des **URI**.

- Une interaction avec ces ressources utilisant directement les **verbes** définis par le protocole **HTTP** que sont `GET`, `HEAD`, `POST`, `OPTIONS`, `CONNECT`, `TRACE`, `PUT`, `PATCH` et `DELETE`.

2. Technologies à contre courant

Et voici maintenant un catalogue non exhaustif de technologies que l'on peut considérer comme à contre courant de l'esprit du web, au point pour une partie de ce qui les constitue.

Services web SOAP

Alors que les services web RESTful adoptent une approche orientée ressources utilisant directement les verbes HTTP, SOAP est une autre forme de services web qui s'écarte de l'esprit du web sur chacun de ces deux points essentiels :

- SOAP adopte une approche **RPC** (Remote Procedure Call) plutôt qu'une approche orientée ressources. Dans cette approche, il ne s'agit donc pas de définir des ressources accessibles en ligne, mais des opérations appelables à distance, qui prennent des paramètres et retournent des résultats. Dans le cas de SOAP, paramètres et résultats ont des structures définies dans un document XML qui permet au client et au serveur de s'entendre.

  La différence entre l'approche orientée ressources et l'approche RPC peut être comparée, dans l'esprit, à celle qui existe dans le domaine des langages de programmation entre, respectivement, les langages déclaratifs et les langages impératifs. En effet, la notion de ressource, comme celle d'expression dans les langages de programmation déclaratifs, définissent plutôt des réponses à la question « quoi ? ». A l'inverse, les opérations RPC s'apparentent à des instructions impératives, dans une approche qui répond plutôt à la question « comment ? ».

- Les **opérations** définies par un services SOAP sont **spécifiques et arbitraires** et non pas basées sur une utilisation directe des verbes HTTP.

Applets, Flash, Silverlight

Dans les années 1990 et jusqu'au milieu des années 2000, les application web « classiques » sont extrêmement limitées en comparaison de ce que permettent déjà les applications natives sur ordinateurs. En gros, ces applications web se limitent à de simples formulaires HTML qui peuvent être soumis pour afficher une réponse sous la forme d'une nouvelle page HTML. Du côté des ordinateurs de bureau, les interfaces graphiques sont de plus en plus sophistiquée et les jeux vidéo conquièrent la troisième dimension avec les premières cartes accélératrices.

Naissent alors tout un tas de technologies qui visent à permettre l'affichage au sein du navigateur d'interfaces graphiques qui empruntent à l'univers des application de bureau.

Ainsi, Sun Microsystems, créateurs du langage Java, proposait d'intégrer à des pages HTML des applets Java, exécutant du code Java capable d'utiser l'API graphique AWT, initialement prévue pour réaliser des applications de bureau.

Microsoft propose la même chose avec ses composants ActiveX, prenant la forme de DLL compilées, réalisables en Visual Basic ou en Visual C++, les langage phares que proposait alors la firme.

D'autres approches, plus sophistiquées graphiquement, étaient les applets Flash d'Adobe ou Sliverlight, à nouveau de Microsoft (les applets ActiveX n'ayant pas trouvé leur public).

Toutes ces approches n'ont jamais véritablement séduit des légions de développeurs, qui sentaient bien qu'elles constituaient des solutions bancales, conceptuellement inélégantes et non conformes à la philosophie du web du fait qu'elles trahissaient sa vocation universaliste en incorporant des technologies pouvant être spécifiques au système d'exploitation du poste client.

JSF (Java Server Faces)

Dans le monde Java, plusieurs technologies ont été lancées successivement pour tenter coller aux tendances du moment en matière de développement d'applications web. Dans le milieu des années 2000, l'industrie informatique se trouve à la croisée de plusieurs chemins.

- D'une part, la technologie des applications se cherche encore : après plusieurs revirements entre Dynamic HTML, scripts serveurs de pages et emploi d'AJAX, on s'y perd un peu entre ce qui doit tourner du côté du client (en Javascript dans le navigateur) et ce qui doit tourner du côté du serveur (en Java, donc, pour ce qui concerne JSF).

- D'autre part, les applications locales de bureau sont encore largement les plus utilisées, mais l'adoption des applications web progresse à un rythme qui laisse penser que cette tendance pourrait s'inverser à moyen terme. Les grandes entreprises se trouvent alors avec des hordes de développeurs « classiques » qu'elles sentent devoir progressivement reconvertir en développeurs web.

JSF est un framework qui tente d'apporter des réponses à ces deux préoccupations :

- Sur le plan technologique, JSF tente de masquer la question de l'opposition entre un client dynamique, comme à l'époque du Dynamic HTML, et un serveur omnipotent, comme à celle des scripts JSP, et proposant de faire générer un client dynamique par le serveur.

- La communication entre ce client généré et les couches plus profondes du serveur adoptent une approche RPC, simulant (plus ou moins bien) une approche synchrone, avec laquelle les développeurs « classiques » sont plus familiés qu'avec l'approche plus authentiquement web, toute en asynchronicité. Cela permet à tous ces développeurs non (encore ?) formés aux approches web de néanmoins créer des applications web.

Evidemment, tout cela est très à rebourd de la philosophie profonde du web. Tout le propos de cette technologie semble au contraire être de contourner cette philosophie pour ne pas brusquer des développeurs qui n'y sont pas formés. Quant à l'approche RPC, elle n'est pas plus conforme à la philosophie du web ici qu'elle ne l'est concernant SOAP.