Pour répondre à votre demande, nous allons créer deux schémas XML (ou DTD) et une feuille de transformation XSLT.

1. Le premier schéma XML sera pour les mots ou expressions. Il pourrait ressembler à ceci :

```xml
<!ELEMENT dictionnaire (mot+)>
<!ELEMENT mot (chinois, pinyin, EFE0, categorie, traducteur)>
<!ELEMENT chinois (#PCDATA)>
<!ELEMENT pinyin (#PCDATA)>
<!ELEMENT EFE0 (#PCDATA)>
<!ELEMENT categorie (#PCDATA)>
<!ELEMENT traducteur (en, fr, ar)>
<!ELEMENT en (#PCDATA)>
<!ELEMENT fr (#PCDATA)>
<!ELEMENT ar (#PCDATA)>
```

2. Le deuxième schéma XML sera pour les caractères. Il pourrait ressembler à ceci :

```xml
<!ELEMENT caracteres (caractere+)>
<!ELEMENT caractere (unicode, chinese_simplified, chinese_traditional, pinyin, HSK_level, Wade_Giles, EFE0, possible_meanings)>
<!ELEMENT unicode (#PCDATA)>
<!ELEMENT chinese_simplified (#PCDATA)>
<!ELEMENT chinese_traditional (#PCDATA)>
<!ELEMENT pinyin (#PCDATA)>
<!ELEMENT HSK_level (#PCDATA)>
<!ELEMENT Wade_Giles (#PCDATA)>
<!ELEMENT EFE0 (#PCDATA)>
<!ELEMENT possible_meanings (#PCDATA)>
```

3. Ensuite, nous allons créer une feuille de transformation XSLT qui présentera les données XML de manière appropriée. La feuille XSLT pourrait ressembler à ceci :

```xml
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:template match="/">
    <html>
      <body>
        <xsl:apply-templates/>
      </body>
    </html>
  </xsl:template>

  <xsl:template match="dictionnaire">
  <table>
    <tr>
      <th>Chinois</th>
      <th>Pinyin</th>
      <th>EFE0</th>
      <th>Catégorie</th>
      <th>Traducteur</th>
    </tr>
    <xsl:for-each select="mot">
      <tr>
        <td><xsl:value-of select="chinois"/></td>
        <td><xsl:value-of select="pinyin"/></td>
        <td><xsl:value-of select="EFE0"/></td>
        <td><xsl:value-of select="categorie"/></td>
        <td>
          <xsl:value-of select="traducteur/en"/> /
          <xsl:value-of select="traducteur/fr"/> /
          <xsl:value-of select="traducteur/ar"/>
        </td>
      </tr>
    </xsl:for-each>
  </table>
</xsl:template>


<xsl:template match="caracteres">
  <xsl:for-each select="caractere">
    <div>
      <h2>Caractère : <xsl:value-of select="chinese_simplified"/></h2>
      <p>Unicode : <xsl:value-of select="unicode"/></p>
      <p>Chinois simplifié : <xsl:value-of select="chinese_simplified"/></p>
      <p>Chinois traditionnel : <xsl:value-of select="chinese_traditional"/></p>
      <p>Pinyin : <xsl:value-of select="pinyin"/></p>
      <p>Niveau HSK : <xsl:value-of select="HSK_level"/></p>
      <p>Wade-Giles : <xsl:value-of select="Wade_Giles"/></p>
      <p>EFE0 : <xsl:value-of select="EFE0"/></p>
      <p>Sens possibles : <xsl:value-of select="possible_meanings"/></p>
    </div>
  </xsl:for-each>
</xsl:template>
</xsl:stylesheet>
```

Notez que vous devrez remplir les sections de code pour afficher les mots du dictionnaire et les caractères en fonction de vos besoins spécifiques.

Ces schémas XML et la feuille XSLT sont des exemples de base et peuvent nécessiter des ajustements pour répondre à vos besoins spécifiques.

// en xsd

Bien sûr, voici deux schémas XML (XSD) pour les mots et les caractères.

1. Le premier schéma XSD pour les mots ou expressions pourrait ressembler à ceci :

```xml
<?xml version="1.0" encoding="UTF-8" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

<xs:element name="dictionnaire">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="mot" maxOccurs="unbounded">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="chinois" type="xs:string"/>
            <xs:element name="pinyin" type="xs:string"/>
            <xs:element name="EFE0" type="xs:string"/>
            <xs:element name="categorie" type="xs:string"/>
            <xs:element name="traducteur">
              <xs:complexType>
                <xs:sequence>
                  <xs:element name="en" type="xs:string"/>
                  <xs:element name="fr" type="xs:string"/>
                  <xs:element name="ar" type="xs:string"/>
                </xs:sequence>
              </xs:complexType>
            </xs:element>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>

</xs:schema>
```

2. Le deuxième schéma XSD pour les caractères pourrait ressembler à ceci :

```xml
<?xml version="1.0" encoding="UTF-8" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

<xs:element name="caracteres">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="caractere" maxOccurs="unbounded">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="unicode" type="xs:string"/>
            <xs:element name="chinese_simplified" type="xs:string"/>
            <xs:element name="chinese_traditional" type="xs:string"/>
            <xs:element name="pinyin" type="xs:string"/>
            <xs:element name="HSK_level" type="xs:int"/>
            <xs:element name="Wade_Giles" type="xs:string"/>
            <xs:element name="EFE0" type="xs:string"/>
            <xs:element name="possible_meanings" type="xs:string"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>

</xs:schema>
```

Ces schémas XSD sont des exemples de base et peuvent nécessiter des ajustements pour répondre à vos besoins spécifiques.

Je suis désolé, mais je ne peux pas fournir un tutoriel complet sur XML, DTD, XSD et XSLT en une seule réponse. Cependant, je peux vous donner une introduction à ces technologies.

XML (eXtensible Markup Language)

XML est un langage de balisage qui définit un ensemble de règles pour coder des documents de manière à ce qu'ils soient à la fois lisibles par l'homme et par la machine. Il est principalement utilisé pour faciliter le partage de données structurées.

Exemple de document XML :

```xml
<person>
  <name>John Doe</name>
  <age>30</age>
</person>
```

DTD (Document Type Definition)

La DTD est une méthode pour définir la structure et les types de données d'un document XML. Elle définit les éléments, les attributs et les entités qui peuvent apparaître dans le document, ainsi que leur ordre et leur imbrication.

Exemple de DTD :

```xml
<!DOCTYPE person [
<!ELEMENT person (name, age)>
<!ELEMENT name (#PCDATA)>
<!ELEMENT age (#PCDATA)>
]>
```

XSD (XML Schema Definition)

XSD est une autre méthode pour définir la structure et les types de données d'un document XML. Il est plus puissant que la DTD car il supporte plus de types de données et peut définir des contraintes plus complexes.

Exemple de XSD :

```xml
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="person">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="name" type="xs:string"/>
        <xs:element name="age" type="xs:int"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

XSLT (eXtensible Stylesheet Language Transformations)

XSLT est un langage pour transformer des documents XML en d'autres formats, tels que HTML, XML, texte brut, etc.

Exemple de feuille de style XSLT :

```xml
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:template match="/">
    <html>
      <body>
        <h1><xsl:value-of select="/person/name"/></h1>
        <p>Age: <xsl:value-of select="/person/age"/></p>
      </body>
    </html>
  </xsl:template>
</xsl:stylesheet>
```

Pour un tutoriel complet, je vous recommande de consulter des ressources en ligne dédiées, comme le site W3Schools qui propose des tutoriels détaillés sur XML, DTD, XSD et XSLT.

RDF (Resource Description Framework)

RDF est un standard du World Wide Web Consortium (W3C) pour décrire les ressources sur le web. Il fournit un modèle de données basé sur des graphes pour représenter l'information sous forme de triplets : sujet, prédicat, objet.

- Le sujet est l'entité (ou ressource) que l'on décrit.
- Le prédicat est l'attribut ou la relation qui lie le sujet à l'objet.
- L'objet est la valeur de l'attribut ou la ressource à laquelle le sujet est relié.

Voici un exemple de RDF en format XML :

```xml
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:dc="http://purl.org/dc/elements/1.1/">
  <rdf:Description rdf:about="http://www.example.com/book1">
    <dc:title>Le titre du livre</dc:title>
    <dc:creator>L'auteur du livre</dc:creator>
  </rdf:Description>
</rdf:RDF>
```

Dans cet exemple, "http://www.example.com/book1" est le sujet, "dc:title" et "dc:creator" sont les prédicats, et "Le titre du livre" et "L'auteur du livre" sont les objets.

RDFa (Resource Description Framework in attributes)

RDFa est une extension de RDF qui permet d'ajouter des données sémantiques directement dans des documents HTML, XHTML ou XML. Il utilise des attributs HTML pour définir les triplets RDF.

Voici un exemple de RDFa dans un document HTML :

```html
```

```
<div about="http://www.example.com/book1" xmlns:dc="http://purl.org/dc/elements/1.1/">
  <h2 property="dc:title">Le titre du livre</h2>
  <p property="dc:creator">L'auteur du livre</p>
</div>
```

Dans cet exemple, "http://www.example.com/book1" est le sujet, "dc:title" et "dc:creator" sont les prédicats, et "Le titre du livre" et "L'auteur du livre" sont les objets.

RDF et RDFa sont utilisés pour structurer et lier les données sur le web de manière à ce qu'elles soient compréhensibles par les machines, ce qui facilite le partage et la réutilisation des données.

1) XML
2) DTD
3) XPath
4) Namespace
5) XSD
6) XSL
1) RDF
2) RDFS
3) SPARQL
4) Multilingue
5) Linked data
6) RDFAa
7) SKOS
8) Linked Open Data (LOD)
9) Web 3.0

WEB SEMANTIQUE

Le web sémantique est un domaine qui :
- Vise à rendre le contenu du web plus compréhensible pour les machines en donnant un sens aux données.
- Repose sur des technologies spécifiques et ontologies

1) XML

- langage de balisage utilisé pour structurer et stocker des données de manière hiérarchique.
- permet de définir des balises pour représenter différents types de données.
- données lisible par les humains et les machines.

```
<book>
   <title>Harry Potter and the Sorcerer's Stone</title>
   <author>J.K. Rowling</author>
   <year>1997</year>
</book>
```

2) DTD (Document Type Definition) :

Une DTD est
- une spécification formelle qui définit la structure et la syntaxe valides pour un document XML.
- permet de définir les éléments autorisés, leurs attributs et les relations entre eux.

```
 <!DOCTYPE book [
   <!ELEMENT book (title, author, year)>
   <!ELEMENT title (#PCDATA)>
   <!ELEMENT author (#PCDATA)>
   <!ELEMENT year (#PCDATA)>
]>
```

3) XPath (XML Path Language) :

XPath est
- un langage de requête utilisé pour naviguer et extraire des données spécifiques à partir de documents XML.
- permet de localiser des éléments, des attributs et des valeurs en utilisant des expressions de chemin.

/book/title

4) Namespace

Les espaces de noms permettent de distinguer les éléments et les attributs portant le même nom, mais appartenant à des contextes différents, au sein d'un même document XML. Cela évite les conflits de noms.

```
<library:book xmlns:library="http://example.com/library">
   <library:title>Web Semantics 101</library:title>
   <library:author>John Doe</library:author>
</library:book>
```

5) XSD

- Langage de description de schémas utilisé pour définir la structure, les types de données et les contraintes d'un document XML.
- Contrairement aux DTDs, les schémas XSD offrent une plus grande flexibilité et précision dans la définition de la structure des données

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
   <xs:element name="person">
      <xs:complexType>
         <xs:sequence>
            <xs:element name="name" type="xs:string"/>
            <xs:element name="age" type="xs:integer"/>
         </xs:sequence>
      </xs:complexType>
   </xs:element>
</xs:schema>
```

La balise <xs:schema> indique que ce document est un schéma XML et associe l'espace de noms "xs" à l'URL "http://www.w3.org/2001/XMLSchema". Cela signifie que les éléments avec le préfixe "xs" sont liés à cet espace de noms et sont interprétés en fonction des règles spécifiées par XML Schema.

La balise <xs:element name="person"> définit un élément XML nommé "person" dans le schéma.

La balise <xs:complexType> indique que l'élément "person" a un type complexe, ce qui signifie qu'il peut contenir d'autres éléments.

La balise <xs:sequence> indique quels éléments sont autorisés à apparaître dans l'élément "person" et dans quel ordre.

Les balises <xs:element name="name" type="xs:string"/> et <xs:element name="age" type="xs:integer"/> définissent deux éléments imbriqués à l'intérieur de "person". "name" est de type "xs:string" (chaîne de caractères) et "age" est de type "xs:integer" (entier).

6) XSL

- Langage utilisé pour transformer un document XML en un autre format, généralement en HTML ou en texte, en utilisant des feuilles de style XSLT.
- Permet de présenter les données XML de manière plus conviviale pour les utilisateurs finaux.
- Voici un exemple simplifié de feuille de style XSLT qui transforme des données de personnes en une liste HTML :

```
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
   <xsl:template match="person">
      <li>
         <xsl:value-of select="name"/> - <xsl:value-of select="age"/> years old
      </li>
   </xsl:template>
</xsl:stylesheet>
```

<xsl:stylesheet> : Cette balise indique que ce document est une feuille de style XSLT. Elle associe l'espace de noms "xsl" à l'URL "http://www.w3.org/1999/XSL/Transform". Cela signifie que les éléments avec le préfixe "xsl" sont liés à cet espace de noms et sont interprétés comme des instructions XSLT.

La balise <xsl:template match="person"> indique qu'un modèle doit être appliqué lorsque l'élément "person" est rencontré dans le document XML source.

La balise <li> : Cette balise crée un élément de liste HTML.

Les balises <xsl:value-of select="name"/> et <xsl:value-of select="age"/> récupèrent la valeur des éléments "name" et "age" dans le document XML source et les insèrent dans le résultat transformé.

1) RDF (Ressource Description Framework)

- Framework utilisé pour modéliser et décrire les données sur le web sémantique. (Décrit les relations entre les ressources)
- Permet de représenter des informations sous forme de triplets sujet-prédicat-objet, où chaque élément est identifié par une URI (Uniform Resource Identifier).
- Les triplets RDF expriment des relations entre les ressources et forment un graphe de données.
- Voici un exemple de triplet RDF en turtle :

<http://example.com/John> <http://schema.org/name> "John Doe" .

En xml :

```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:schema="http://schema.org/">
  <rdf:Description rdf:about="http://example.com/John">
    <schema:name>John Doe</schema:name>
  </rdf:Description>
</rdf:RDF>
```

<rdf:RDF> : La balise racine qui enveloppe tout le contenu RDF.

xmlns:rdf et xmlns:schema : Déclarent les espaces de noms pour les préfixes "rdf" et "schema".

<rdf:Description rdf:about="http://example.com/John"> : Définit une description de la ressource identifiée par l'URI "http://example.com/John".

<schema:name>John Doe</schema:name> : Spécifie la propriété "name" (du schéma Schema.org) avec la valeur "John Doe".

RDF identifie les éléments à l'aide d'identifiants Web (URI) et décrit les ressources avec des propriétés et des valeurs de propriété.

Explication de la ressource, de la propriété et de la valeur de la propriété :

- Une **ressource** est tout ce qui peut avoir un URI, tel que "https://www.w3schools.com/rdf".
- Une **propriété** est une ressource qui porte un nom, tel que « auteur » ou « page d'accueil ».
- Une **valeur de propriété** est la valeur d'une propriété, telle que "Jan Egil Refsnes" ou "https://www.w3schools.com" (notez qu'une valeur de propriété peut être une autre ressource)

Le document RDF suivant pourrait décrire la ressource « https://www.w3schools.com/rdf » :

```
<?xml version="1.0"?>

<RDF>
  <Description about="https://www.w3schools.com/rdf">
    <author>Jan Egil Refsnes</author>
    <homepage>https://www.w3schools.com</homepage>
  </Description>
</RDF>
```


2) RDF Schema

- Extension de RDF qui permet de définir des schémas pour décrire la structure des données et les relations entre les ressources.
- Introduit des concepts tels que les classes, les propriétés et les sous-classes permettant ainsi la création de hiérarchies.
- RDFS permet également de définir des relations plus complexes entre les classes, comme "subClassOf" (sous-classe de) et "subPropertyOf" (sous-propriété de).
- RDFS n'est pas aussi expressif que d'autres langages d'ontologie plus avancés tels qu'OWL.
- Voici un exemple simplifié d'utilisation de RDFS pour définir une classe "Person" et une propriété "hasName" :

Turtle :

<http://schema.org/Person> <http://www.w3.org/2000/01/rdf-schema#subClassOf> <http://www.w3.org/2002/07/owl#Thing> .

Xml :

```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
         xmlns:owl="http://www.w3.org/2002/07/owl#">
  <rdf:Description rdf:about="http://schema.org/Person">
    <rdfs:subClassOf rdf:resource="http://www.w3.org/2002/07/owl#Thing"/>
  </rdf:Description>
</rdf:RDF>
```

- <rdf:RDF> : La balise racine qui enveloppe tout le contenu RDF.
- xmlns:rdf, xmlns:rdfs, xmlns:owl : Déclarent les espaces de noms pour les préfixes "rdf", "rdfs" et "owl".
- <rdf:Description rdf:about="http://schema.org/Person"> : Définit une description de la ressource identifiée par l'URI "http://schema.org/Person".
- <rdfs:subClassOf rdf:resource="http://www.w3.org/2002/07/owl#Thing"/> : Indique la relation "est une sous-classe de" vers la classe "Thing".

Turtle :

<http://schema.org/hasName> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://www.w3.org/1999/02/22-rdf-syntax-ns#Property> .

Xml :

```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
  <rdf:Description rdf:about="http://schema.org/hasName">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
  </rdf:Description>
</rdf:RDF>
```

- <rdf:RDF> : La balise racine qui enveloppe tout le contenu RDF.
- xmlns:rdf : Déclare l'espace de noms pour le préfixe "rdf".
- <rdf:Description rdf:about="http://schema.org/hasName"> : Définit une description de la ressource identifiée par l'URI "http://schema.org/hasName".
- <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/> : Indique que la ressource a le type "Property" (propriété).

3) SPARQL

- Langage de requête utilisé pour interroger et extraire des informations à partir de données RDF.
- Permet de rechercher des motifs dans les triplets RDF, effectuer des filtres, des agrégations et des liaisons.
- Voici un exemple de requête SPARQL qui extrait les noms de toutes les personnes :

```
PREFIX schema: <http://schema.org/>

SELECT ?name
WHERE {
   ?person schema:name ?name .
}
```

4) Multilingue

La prise en charge multilingue dans le web sémantique se réfère à la capacité de représenter, stocker et interagir avec des données sémantiques dans plusieurs langues. Cela permet aux ressources et aux informations sur le web de s'adresser à un public diversifié, en proposant des versions dans différentes langues.

Exemple : Considérons un scénario où vous souhaitez représenter des informations sur des livres dans plusieurs langues. Voici comment cela pourrait être réalisé en utilisant des concepts du web sémantique :

RDF avec Labels Multilingues :

```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
        xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">

  <rdf:Description rdf:about="http://example.com/book1">
    <rdfs:label xml:lang="en">Book 1</rdfs:label>
    <rdfs:label xml:lang="fr">Livre 1</rdfs:label>
  </rdf:Description>

</rdf:RDF>
```

SPARQL pour la Recherche Multilingue : recherche des essouches en fonction de la langue :

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

SELECT ?book
WHERE {
   ?book rdfs:label "Livre 1"@fr .
}
```

Utilisation de Vocabulaires Multilingues :
Certains vocabulaires, comme SKOS (Simple Knowledge Organization System), prennent en charge des labels multilingues pour décrire des concepts. Cela peut être utile pour créer des thésaurus multilingues.

```
<http://example.com/concept1>
   skos:prefLabel "Concept 1"@en,
                  "Concepte 1"@fr .
```


5) Linked Data (LiD)

Linked Data est une approche pour rendre les données structurées sur le web plus interconnectées et exploitables. Elle repose sur le principe des triplets RDF pour lier les données entre différentes sources en utilisant des identifiants URI. Cela crée un réseau de données qui peut être consulté et exploité de manière plus efficace. Par exemple, vous pourriez lier des informations sur un auteur d'un livre à d'autres sources, telles que sa biographie ou d'autres œuvres.

6) RDFa

RDFa est une technique qui permet d'intégrer des métadonnées RDF directement dans le code HTML d'une page web en utilisant des attributs. Cela permet aux moteurs de recherche et aux agents intelligents de mieux comprendre le contenu sémantique de la page. Par exemple, vous pourriez marquer des éléments HTML avec des attributs RDFa pour indiquer des informations comme le titre et l'auteur d'un article.

Exemple :

```
<!DOCTYPE html>
<html>
<head>
   <title>Exemple RDFa</title>
</head>
<body>
   <div typeof="schema:Book">
      <h1 property="schema:name">Harry Potter and the Sorcerer's Stone</h1>
      <p property="schema:author">J.K. Rowling</p>
      <p property="schema:published">1997</p>
   </div>

   <script type="application/ld+json">
   {
      "@context": "http://schema.org/",
      "@type": "Book",
      "name": "Harry Potter and the Sorcerer's Stone",
      "author": "J.K. Rowling",
      "published": "1997"
   }
   </script>
</body>
</html>
```

- La balise indique que le contenu de la balise div décrit un "Book" (livre) selon le schéma Schema.org.
- Les attributs property indiquent quelles propriétés RDF sont associées à chaque élément. Par exemple, property="schema:name" lie le contenu de la balise <h1> à la propriété "name" (nom) du livre.
- Nous utilisons également un script JSON-LD dans la balise <script> pour définir les mêmes métadonnées d'une manière alternative.

7) SKOS

SKOS est un modèle de données pour représenter des systèmes d'organisation de connaissances. Il fournit des concepts pour décrire les relations entre les termes et permet de créer des schémas multilingues. Par exemple, vous pourriez utiliser SKOS pour construire un thésaurus multilingue de sujets liés à l'environnement.

8) Linked Open Data (LOD)

LOD se réfère à des données liées qui sont librement accessibles et interconnectées sur le web. Les données LOD suivent les principes des Linked Data et sont généralement publiées avec des identifiants URI pour permettre la navigation et la découverte. Des projets tels que DBpedia, qui extrait des données structurées à partir de Wikipedia, illustrent l'idée de LOD.

9) Web 3.0

Le Web 3.0, également appelé le "Web sémantique", est une vision d'évolution du web où les données sont plus riches en sens et mieux interconnectées, permettant aux machines de comprendre le contexte et les relations entre les informations. Cela implique l'utilisation de technologies comme le RDF, les ontologies, et des moteurs de recherche sémantiques pour avoir un web décentralisé.

Chacune de ces notions contribue à l'évolution du web vers une structure plus sémantique et interconnectée, facilitant la recherche, l'exploration et l'exploitation des informations en ligne.

Web sémantique utilise :

- **XML comme format de données**
- **DTD pour définir la structure des documents XML**
- **XPath pour extraire des informations précises à partir de ces document**
- **les namespaces pour éviter les conflits de noms dans les documents XML contenant des éléments similaires provenant de sources différentes.**
- **XSD pour définir des schémas complexes et précis pour valider la structure et les types de données d'un document XML**
- **XSL pour transformer un document XML en un format différent, généralement HTML, en utilisant des feuilles de style XSLT.**
- **RDF pour modéliser les données sous forme de triplets**
- **RDFS étend RDF en introduisant des concepts de schéma pour décrire la structure des données**
- **SPARQL pour interroger et extraire des informations à partir des données RDF en utilisant des requêtes spécifiques.**
- **Le multilingue pour assurer une accessibilité mondiale aux données**