

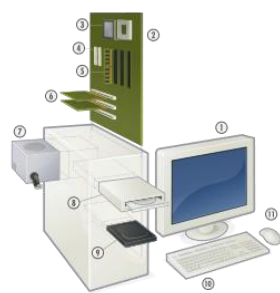
1) Notion de composant (component) et d'interface :

En UML, un **composant** est un **élément logiciel remplaçable et réutilisable** qui fournit ou reçoit un service bien précis. Il peut être vu comme une pièce détachée du logiciel (Les plug-ins, les drivers, les codecs, les bibliothèques sont des composants).

Les composants fournissent des services via des interfaces. Un composant peut être remplacé par n'importe quel autre composant compatible c'est-à-dire ayant les mêmes interfaces.

Il existe deux types d'interface :

- ✓ **Les interfaces requises** : Ce sont des interfaces qui fournissent un service au composant et dont il a besoin pour fonctionner.
- ✓ **Les interfaces fournies** : Ce sont des interfaces par lesquels le composant fournit lui-même un service.



Ex : parallèle avec les composants d'un ordinateur.

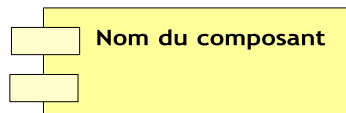
Un ordinateur est un ensemble de composants modulaires qui fournissent et reçoivent des services (carte mère, carte graphique, disque dur, clavier, écran...). Chacun de ces composants est remplaçable par un autre composant (pas forcément identique) à condition qu'il ait des interfaces compatibles (nous ne pouvons pas mettre un écran avec une connexion VGA à la place d'un écran avec une connexion HDMI).

NB : ceci n'est qu'un parallèle pour faire comprendre la notion de composant. **En UML les composants ne sont pas des éléments matériels mais des éléments logiciels.** Éléments logiciels qui par contre seront installés sur des éléments matériels (ce que nous verrons lorsque nous aborderons le diagramme de déploiement).

2) Représentation graphique :

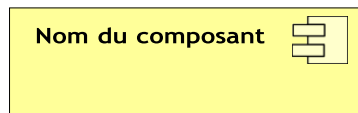
2-1) Les composants :

Il existe plusieurs possibilités pour représenter un composant, à vous de choisir :



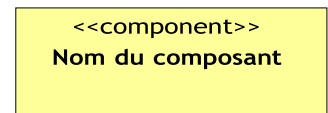
Un rectangle dans lequel figure :

- Le nom du composant.
- 2 petits rectangles l'un au-dessus
- de l'autre à cheval du côté gauche.



Un rectangle dans lequel figure :

- Le nom du composant.
- Le symbole  en haut à droite.



Un rectangle dans lequel figure :

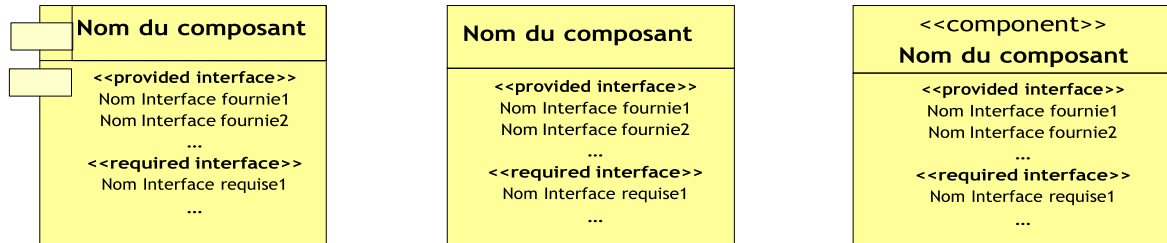
- Le stéréotype <<component>>.
- Le nom du composant.

2-2) Les interfaces :

La aussi, il existe plusieurs possibilités pour représenter les interfaces :

- Intégrées dans la représentation du composant :

Nous reprenons l'une des trois représentations du composant que nous venons juste de voir et nous ajoutons un compartiment dans lequel nous listons les interfaces requises et fournies (grâce aux stéréotypes `<<required interface>>` et `<<provided interface>>`).



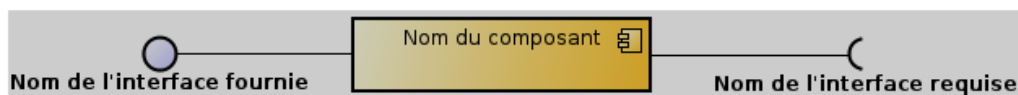
- Dans un classeur séparé du composant dans lequel sont listés les différents services :

- ✓ Les interfaces requises sont reliées au composant par une flèche en pointillés sur laquelle figure le stéréotype `<<use>>`.
- ✓ Les interfaces fournies sont reliées au composant par une flèche en pointillés sur laquelle figure le stéréotype `<<realize>>` (le bout de la flèche est un triangle vide).

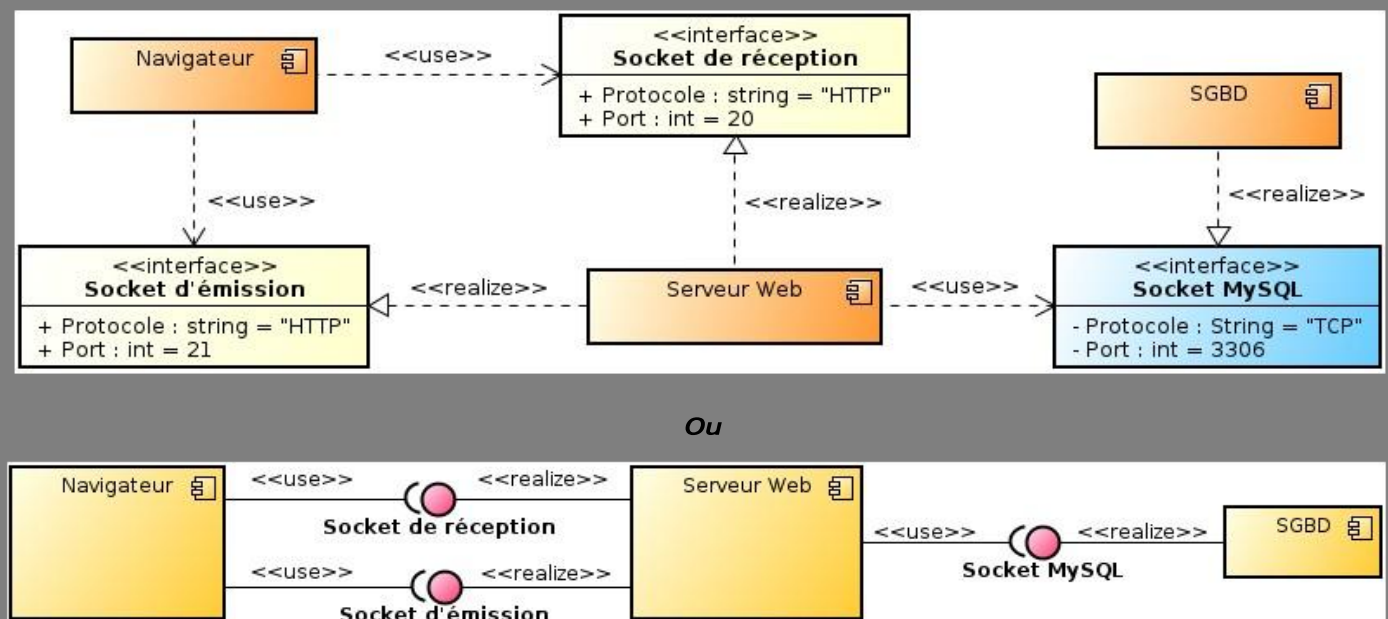


- Avec des connecteurs d'assemblage :

- ✓ Les interfaces requises (représentées par un demi-cercle) et les interfaces fournies (représentées par un cercle) sont raccordées au composant par un trait.

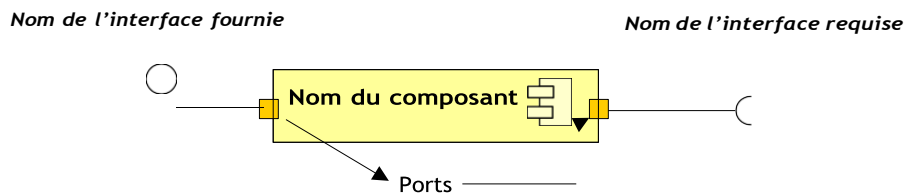


Ex : Transfert de données par Internet.



2-3) Les ports :

Le port est le point de connexion entre le composant et son environnement, il est la matérialisation de l'interface. Nous le représentons par un petit carré à la périphérie du composant.

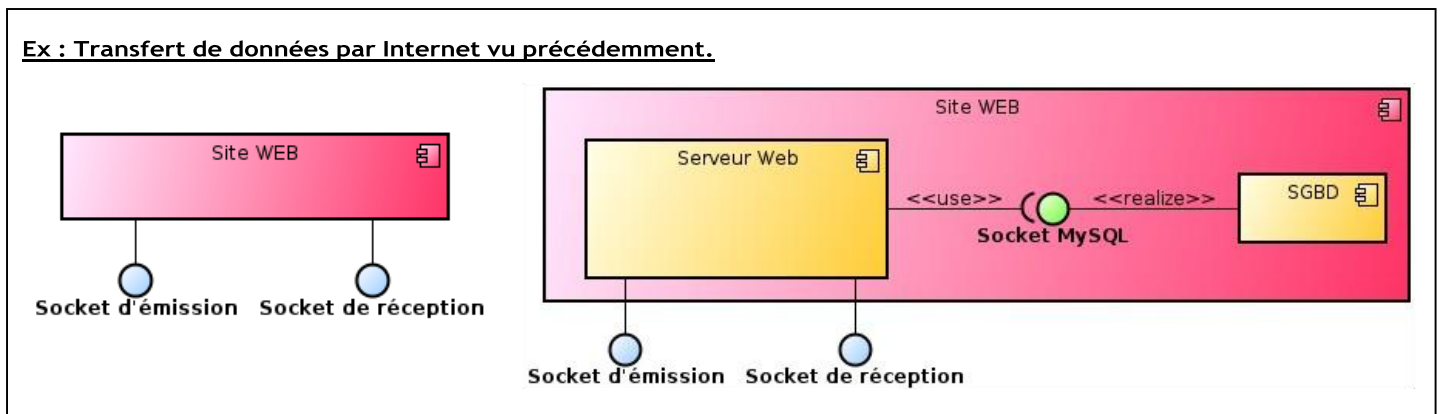


2-4) Boîte noire-boîtes blanche.

Un composant peut être vu de 2 manières :

- comme une boîte noire dont nous ne connaissons pas le contenu et auquel nous accédons via les interfaces qui sont la seule partie visible.
- comme une boîte blanche en spécifiant les objets qui constituent le composant et en indiquant leurs relations.

Ex : Transfert de données par Internet vu précédemment.



3) Rôle du diagramme de composants :

Le diagramme de composants fait partie des diagrammes structuraux (*statiques*) d'UML. Il permet de représenter les différents éléments logiciels (*composants*) du système et leurs dépendances (relations qui les lient). Les diagrammes de composants sont donc utilisés pour définir les dépendances et relations entre objets à un niveau supérieur à celui du diagramme de classes.

Comme nous venons de le voir dans la section précédente lorsque nous définissons la notion de composant, ces dépendances peuvent être :

- des relations de compositions (*boîte blanche*)
- des relations d'assemblages et de connexion (*via des interfaces*)

Mais elles peuvent aussi être d'autre type tel que :

- des contraintes de compilation, des éditions de liens (les composant sont alors des fichiers de code source, des fichiers en binaire, des bibliothèques...). Dans ce cas, un stéréotype peut préciser la nature de la dépendance.

Dans UML, les diagrammes de déploiement modélisent l'architecture physique d'un système. Les diagrammes de déploiement affichent les relations entre les composants logiciels et matériels du système, d'une part, et les relations entre eux d'autre part. Les éléments utilisés par un **diagramme de déploiement** sont principalement les **nœuds**, les **composants**, les **associations** et les **artefacts**. Les caractéristiques des ressources matérielles physiques et des supports de communication peuvent être précisées par stéréotype