

Extrait du Easter-eggs - Spécialiste GNU/Linux

<http://www.easter-eggs.com>

Optimisation des applications X11 avec XCB

- Études de cas - Développement applicatif -

Date de mise en ligne : lundi 19 octobre 2009



Easter-eggs - Spécialiste GNU/Linux

Sommaire

- [Réduction de la latence](#)
- [Utilisation](#)

[XCB](#) (X protocol C-language Binding) est un remplaçant de Xlib pour l'écriture d'applications ou de toolkits graphiques.

XCB a été conçu pour pallier différents problèmes liés à Xlib, comme :

- ▶ Réduction de la taille : Xlib n'est pas adaptée aux petits systèmes ;
- ▶ Réduction de la latence : Xlib est totalement synchrone ;
- ▶ Accès direct au protocole : Xlib cache beaucoup d'aspects du protocole en ajoutant plusieurs couches d'abstraction ou d'optimisation. Cela rend difficile l'envoi de requêtes X précises ;
- ▶ Support du multi-threading : bien que Xlib puisse le faire, il est bien plus difficile d'y arriver sans se tromper ;
- ▶ Ajout simplifié d'extensions : il est possible d'étendre Xlib, mais il est bien plus facile de le faire via XCB.

La bibliothèque XCB est conçue de manière à générer automatiquement son code directement en C à partir de fichiers de description du protocole X. Ces fichiers décrivent toutes les requêtes et évènements liés au protocole X en XML.

Ainsi il est très facile de rajouter le support d'une extension du protocole avec simplement une description de celui-ci en XML.

Il est également simple de générer du code dans un autre langage, comme xpyb, l'équivalent de XCB pour [Python](#).

De plus, XCB utilise le protocole X de la même façon que Xlib. Certaines applications Xlib fonctionnent de la même façon depuis plus de 10 ans, ce qui garantit une grande pérennité au code écrit avec XCB.

Réduction de la latence

En utilisant l'approche synchrone de Xlib, une application se voit contrainte d'attendre la réponse du serveur X pour effectuer un traitement.

Par exemple, si l'application veut savoir quelle fenêtre a le focus, elle doit envoyer une requête `GetInputFocus` au serveur. Avec Xlib, cela se fait ainsi :

```
Window focused_window;
int revert_to;

/* On fait des choses */
do_many_stuff();

/* On envoie la requête au serveur X, et on attend la réponse */
XGetInputFocus(display, &focused_window, &revert_to);

/* On fait à nouveau des choses */
do_many_stuff_again(focused_window);
```

La fonction `XGetInputFocus` envoie une requête au serveur X (via le réseau, ou une socket locale), qui doit la traiter, puis renvoie la réponse. La fonction `XGetInputFocus` peut alors retourner la valeur au programme appelant.

En cas de latence importante du serveur X, cela peut devenir une source de ralentissement conséquente pour l'application.

En XCB, cela peut se traduire ainsi, en asynchrone :

```
/* On envoie la requête au serveur X */
xcb_get_input_focus_cookie_t focus_req = xcb_get_input_focus_unchecked(connection);

/* En attendant la réponse, on fait des choses */
do_many_stuff();

/* On récupère la réponse du serveur X */
xcb_get_input_focus_reply_t *focus_reply = xcb_get_input_focus_reply(connection, focus_req, NULL);

/* On fait à nouveau des choses */
do_many_stuff_again(focus_reply->focused_window);
```

Bien que le code soit légèrement plus verbeux, XCB permet de cacher la latence du serveur X en effectuant d'autres traitements pendant la réception de la réponse. Il faut donc envoyer la requête le plus tôt possible, et récupérer la réponse le plus tard possible, en faisant tout ce qu'il est possible de faire entre les deux pour minimiser l'effet de la latence sur le programme.

Ainsi, lors de l'envoi de plusieurs dizaines de requêtes en parallèle, il est possible de diminuer la durée d'exécution de certaines parties d'une application d'un facteur très important.

Utilisation

XCB a de plus en plus de cas d'utilisation, et une liste a été dressée [ici](#).

Le plus important étant que les dernières versions de Xlib sont maintenant basées sur XCB pour la couche transport.