# C.14 Concevoir des objets actifs

## Objectif

Appréhender la notion de threads.

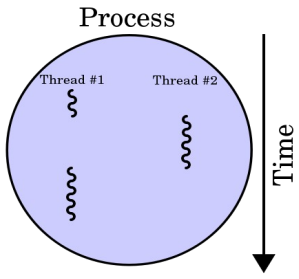
## Préambule

La **vue des processus** précise les threads et les processus qui forme les mécanismes de concurrence et de synchronisation du système.

Rappel : On aura besoin de thread(s) dans une application lorsqu'on devra parallèliser des traitements.

## Thread

Un thread ou fil (d'exécution) ou encore processus léger, est similaire à un processus car tous deux représentent l'exécution d'un ensemble d'instructions du langage machine d'un processeur. Du point de vue de l'utilisateur, ces exécutions semblent se dérouler en parallèle. Toutefois, là où chaque processus possède sa propre mémoire virtuelle (son espace mémoire adressable), les threads d'un même processus se partagent sa mémoire virtuelle.

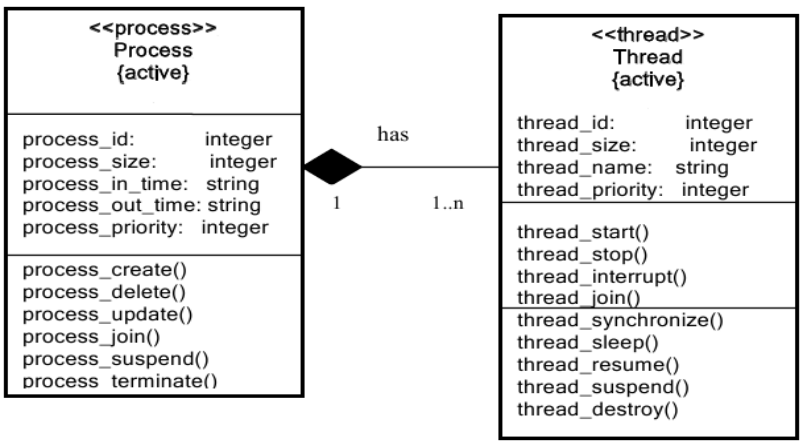



## Avantages

Multi-tâche moins coûteux
Communication entre threads plus rapide et plus efficace

## Inconvénients

Programmation plus difficile car elle nécessite des mécanismes de synchronisation
Risque d'interblocage

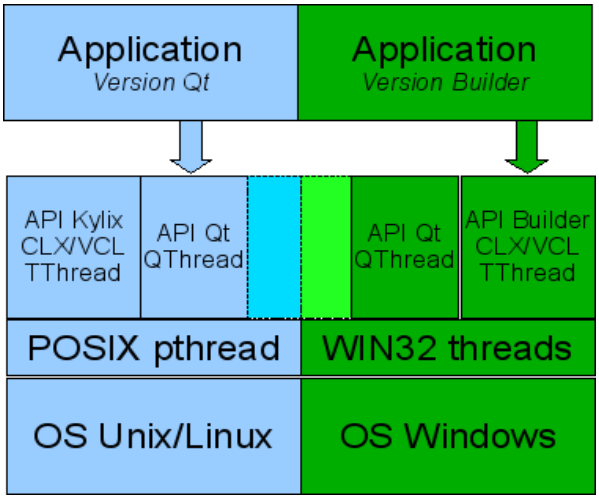


*Support*

Les systèmes d'exploitation mettent en œuvre généralement les threads. Le standard des processus légers POSIX est connu sous le nom de pthread. Le standard POSIX est largement mis en oeuvre sur les systèmes UNIX. Microsoft fournit aussi une API pour les processus légers : WIN32 threads (Microsoft Win32 API threads).

Les EDI fournissent généralement un framework pour les Threads. Il existe aussi des API comme la GNU Common C++.

Dans le cadre d'un développement C++, l'utilisation d'une API simplifie la programmation.




*Exemple*

La mise en place des threads revient à **dériver une classe de base fournie par l'API** choisie et à écrire le code du thread dans une méthode spécifique. Sous Qt, on dérive une classe **QThread** et on écrit le code du thread dans la méthode **run**() :

```
#include <QThread>
class TAcquisitionMesures : public QThread
{
public:
    TAcquisitionMesures();
    void run();
};
```

L'utilisation des threads se fera de la manière suivante sous Qt :

```
// Crée un thread
TAcquisitionMesures *acquisitionMesures = new TAcquisitionMesures();

// Démarre un thread (appelle la méthode run())
acquisitionMesures->start();

// Arrête un thread
acquisitionMesures->stop();
```