Priority Inversion in C++

In real-time systems, if you have threads with different priorities, it is possible to encounter a priority inversion. A priority inversion occurs when a low-priority thread prevents a higher-priority thread from running. This situation arises when a low-priority thread acquires a mutex and is pre-empted by one or more medium-priority threads that do not relinquish the CPU. If a high-priority thread then attempts to acquire the mutex locked by the low-priority thread, it will wait (potentially forever) for the medium-priority threads to complete.

One way to deal with this problem is to use a priority inheritance protocol. If a low-priority thread holds a mutex and a high-priority thread attempts to acquire the mutex, the priority of the thread holding the mutex is temporarily raised to the level of the thread waiting for the mutex. This allows the low-priority thread to keep running until it releases the mutex; as soon as it does, its priority is reduced back to its previous level and the high-priority thread acquires the mutex.

Ice supports the priority inheritance protocol on many POSIX platforms. (Windows does not provide such a protocol.)

For POSIX platforms that support the priority inheritance protocol, mutexes by default do not use it. You can use the `getDefaultMutexProtocol` function to retrieve the current default for your platform:

C++

```
namespace IceUtil {
    enum MutexProtocol { PrioInherit, PrioNone };

    MutexProtocol getDefaultMutexProtocol();
}
```

On Posix systems that do not support priority inheritance and on Windows, this function always returns `PrioNone`.

The return value of `getDefaultMutexProtocol` determines whether a default-constructed `Mutex` or `RecMutex` uses priority inheritance. By default, this function returns `PrioNone`. You can override this default by explicitly specifying a different protocol when you construct a `Mutex` or `RecMutex`. On Windows, if you specify `PrioInherit` when you construct a mutex, the setting is ignored and the mutex is constructed as if you had specified `PrioNone`.

To change the value returned by `getDefaultMutexProtocol`, you can edit `cpp/config/Make.rules`, modify the value of the `DEFAULT_MUTEX_PROTOCOL` macro, and then rebuild the `IceUtil` library.

See Also

- The C++ Mutex Class
- The C++ RecMutex Class