What is the lock order for Mib objects to safely access them in a multi-threaded agent?

By default AGENT++ is a multi-threaded agent. Therefore access to each MibEntry in the a Mib has to be synchronized.

In order to reduce blocking of concurrent requests, AGENT++ uses a two level lockin:

1. Mib instance level - locks the whole MIB
2. MibEntry level - locks a scalar, sub-tree, or table

As tables and complex (sub-tree) entries may contain also MibEntry objects, for example scalars, additional levels can be implemented by the user.

In any case, the locking procedure boundary must be implemented according to the following schema:

```
Mib* mib;
...

mib->lock_mib();

// code to lookup a MibEntry (replace "my context" with "" or your context and the OID by the table entry OID,
for example):
MibTable* table = (MibTable*) mib->get("my context", "1.3.6.1.4.1.????.1");

// enter protected region:
table->start_synch();

// now you can drop the Mib lock
mib->unlock_mib();

// do the real work on table
...

table->end_synch();
```

If you need to add/remove objects from the Mib, the locking schema looks sligtly different:

```
Mib* mib;
...

mib->lock_mib();

// code to lookup a MibEntry:
MibTable table = ...;

// enter protected region:
table->start_synch();

// remove table from Mib:
MibContext* ctx = mib->get_context(DEFAULT_CONTEXT);
ctx->remove(*table->key());
table->end_synch();
delete table;
// now you can drop the Mib lock
mib->unlock_mib();
```