2.2 Lecture Summary

2.2 Object-Based Isolation

Lecture Summary: In this lecture, we studied *object-based isolation*, which generalizes the isolated construct and relates to the classical concept of *monitors*. The fundamental idea behind object-based isolation is that an isolated construct can be extended with a set of objects that indicate the scope of isolation, by using the following rules: if two isolated constructs have an empty intersection in their object sets they can execute in parallel, otherwise they must execute in mutual exclusion. We observed that implementing this capability can be very challenging with locks because a correct implementation must enforce the correct levels of mutual exclusion without entering into deadlock or livelock states. The linked-list example showed how the object set for a `delete()` method can be defined as consisting of three objects — the current, previous, and next objects in the list, and that this object set is sufficient to safely enable parallelism across multiple calls to `delete()`. The Java code sketch to achieve this object-based isolation using the PCDP library is as follows:

```
1  isolated(cur, cur.prev, cur.next, () -> {
2      . . . // Body of object-based isolated construct
3  });
```

The relationship between object-based isolation and monitors is that all methods in a monitor object, `M1`, are executed as object-based isolated constructs with a singleton object set, {`M1`}. Similarly, all methods in a monitor object, `M2`, are executed as object-based isolated constructs with a singleton object set, {`M2`} which has an empty intersection with {`M1`}.

Optional Reading:

1. Wikipedia article on Monitors