Operations manage asynchronous interactions.

For example, to create an operation which performs no retries, times out after the standard timeout, uses the timer “timer”, sends events to the “request” event handler and passes the results to the “continuation” event handler:

```java
operation = new Operation(0, Constants.OPERATION_TIMEOUT,
                         timer, request, continuation);
```

Checkpointing captures and restores an application’s execution state.

For example, to checkpoint a user’s environment “env”:

```java
operation.handle(new
                 EnvironmentEvent(null, this, EnvironmentEvent.CHECK_POINT, env.getId()));
```

And, to restore the user’s environment from the latest checkpoint:

```java
operation.handle(new
                 RestoreRequest(null, this, env.getId(), -1));
```

Migration moves or copies an application and its data.

For example, to migrate a user’s environment “env” to the machine named “loc”:

```java
operation.handle(new
                 MoveRequest(null, user, user.env.getId(),
                             "sio://"+loc+"/User",
                             false));
```

Remote event passing sends events to remote services, and discovery routes events to services with unknown locations.

For example, to send an event “msg” for user “fetchUser”, whose location “fetchLocation” may or may not be known:

```java
SymbolicHandler destination;
if (null == fetchLocation) {
    destination = new DiscoveredResource(new
        Query(new Query("", Query.COMPARE_HAS_SUBTYPE, UserDescriptor.class),
             Query.BINARY_AND,
             new Query("user", Query.COMPARE_EQUAL, fetchUser)));
} else {
    destination = new NamedResource(fetchLocation, "/User/ + fetchUser);
}
operation.handle(new RemoteEvent(this, closure, destination, msg));
```

All examples are taken from the source code of the Emcee application, which manages users and their applications. Emcee runs in the “/User” environment and uses a direct child environment for each user.