

EOF-Using EOF-EO Editing Context

localInstanceOfObject

Pierce T. Wetter III

I thought I'd expand on that last sentence because `localInstanceOfObject` seems to confuse multiple people.

`localInstanceOfObject()` actually returns a *fault* from the new editing context for the passed in object.

If you have 2 peer contexts:

```
databaseContext -> ec1
                -> ec2
```

and you call `localInstanceOfObject()`, and the object is in the database, you'll get a copy of the `databaseContext` snapshot of the object when the fault resolves from `ec2`. That is, the editing context goes up one level to the database context and returns whatever the database context has, not whatever `ec1` has (because they're peers). If its an uncommitted object, then the object only exists in `ec1`, so the fault can't resolve into `ec2`. Similarly, any uncommitted changes in general from `ec1` won't show up in `ec2`, because they're peers.

What Mark just said was that if you have an uncommitted object, and you have:

```
databaseContext -> ec1 -> ec2
                -> ec3
```

(i.e. `ec2` and `ec3` are child editing contexts of `ec1`, but peers of each other)

Then when the fault fires in `ec3`, it goes up one level, pulls the objects and changes from `ec1` (committed or not) and returns it to you in `ec3`. But that only works because `ec1` is common to both `ec2` and `ec3`. In other word, `localInstanceOfObject()` returns an object from "one level above" the source editing context which is not necessarily a copy of the object you want, it depends on how the destination context lies with respect to the source context.