Deltas

Deltas are data structures that describe relative change. Deltas describe how a thing should be modified. It does rely on how the thing looked before the change or how it will look after the change. Delta describes only what was changed.

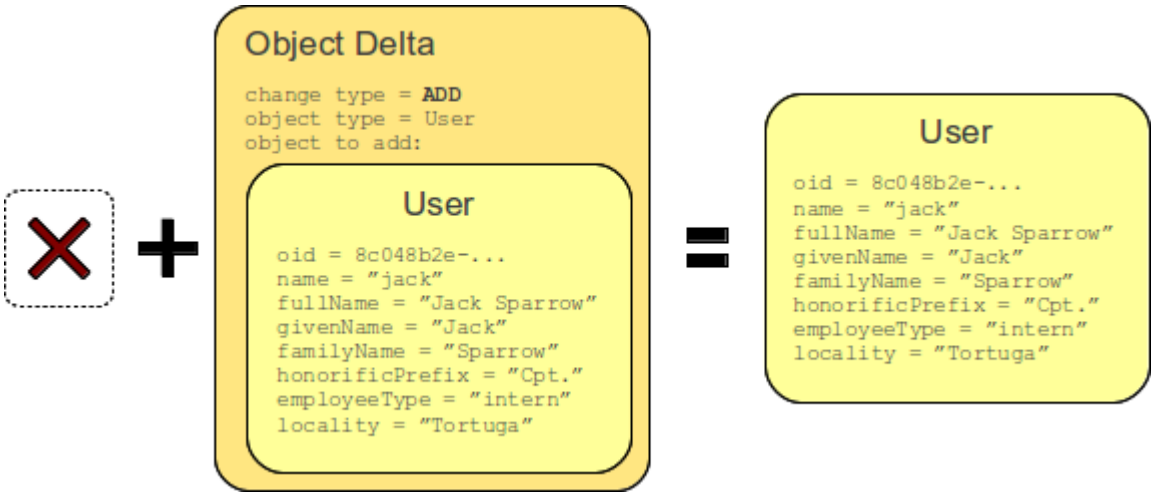
There are several types of deltas. The basic difference is about the *thing* is changing:

- **Object Delta** describes a change of an object. It means that object delta identifies the changed object by OID. Object delta tells about:
  - **ADD** of a new object. Application of such delta simply results in creation of a new object. This delta contains a complete new object to be added.
  - **MODIFICATION** of an existing object. Application of this delta modifies object that already exists. This is the most complex delta because it can describe the change in very fine details. It contains modifications in a form of *item deltas* (see below).
  - **DELETE** of existing object. Application of this delta deletes existing object - with all this properties and containers and references. This delta does not contain almost anything. It just has the OID of the object to delete.
- **Item Delta** describes a change of an item which is a property, container or a reference. It describes only a very small change - a change of a *single item*. Therefore complex changes can only be described by using several item deltas together. A group of item deltas is called *modifications* because they describe how an object is modified (they cannot apply to add or delete object delta). Item delta describes **values** that are being added, removed or replaced with respect to an item. Therefore the item delta may also be of several types:
  - **add** of new values. The values in item delta are added to the existing values. Existing values are left as they are.
  - **delete** of existing values. The values in item delta are removed from the set of existing values. Other existing values are left as they are.
  - **replace** of the values. All existing values are removed and all the values in item delta are added.

A picture is worth a thousand words therefore next sections provide examples of the most common delta types.

Object ADD Delta

Object ADD delta is adding a new object. It contains a complete representation of new object to be added.



The diagram above illustrates the application of ADD delta. The object did not exist before delta application. The delta contains an object that should be added. When the delta is executed a new object is created that is an copy of the object specified in the delta.

The object provided in the delta may or may not have OID. If it does then that OID will be used. If it does not then it will be generated when the delta is executed. The ADD object delta obviously works only if such object does not exist already. Attempt to apply ADD delta to an existing objects results in an error.

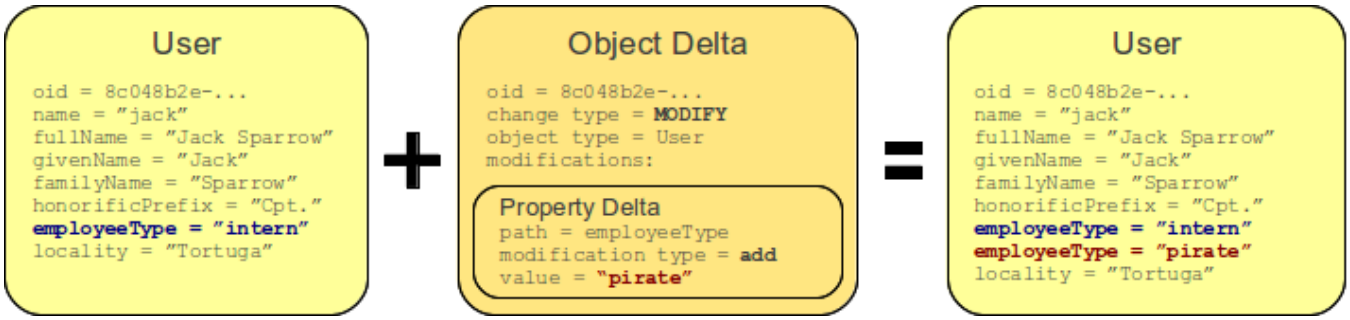
Object ADD delta

```
<objectDelta>
    <changeType>add</changeType>
    <objectType>c:UserType</objectType>
    <objectToAdd>
        <c:user>
            <c:name>jack</c:name>
            ...
        </c:user>
    </objectToAdd>
</objectDelta>
```

Object MODIFY Delta

TODO

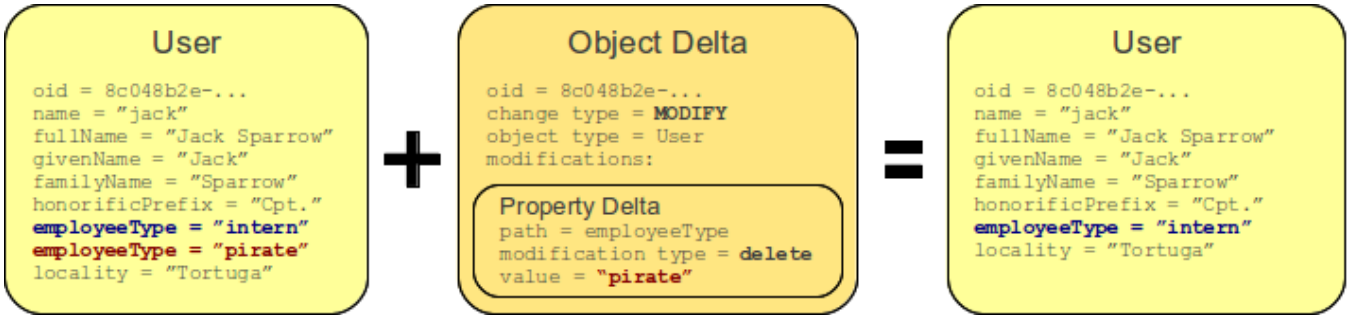
Object MODIFY:add Delta



Object MODIFY:add delta

```
<objectDelta>
    <changeType>modify</changeType>
    <objectType>c:UserType</objectType>
    <oid>e3ba0a70-6ef3-11e2-8c1f-001e8c717e5b</oid>
    <modification>
        <modificationType>add</modificationType>
        <value>
            <c:employeeType>pirate</c:employeeType>
        </value>
    </modification>
</objectDelta>
```

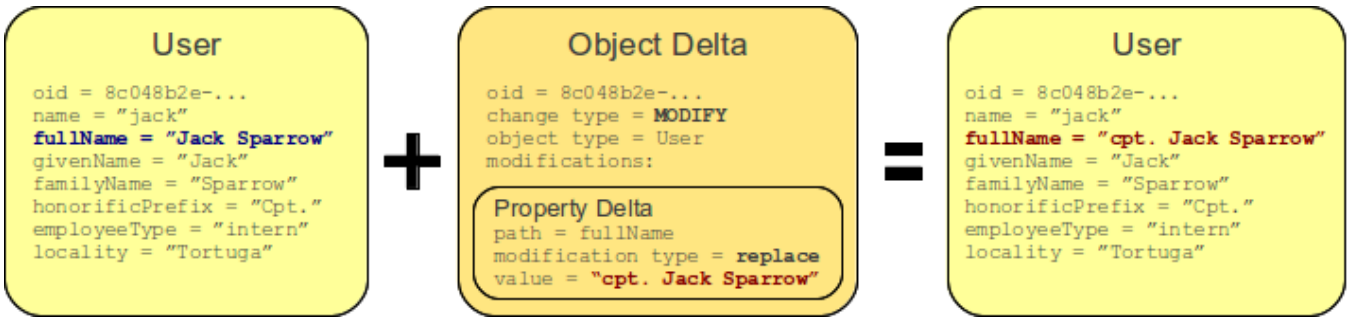
Object MODIFY:delete Delta



Object MODIFY:delete delta

```
<objectDelta>
    <changeType>modify</changeType>
    <objectType>c:UserType</objectType>
    <oid>e3ba0a70-6ef3-11e2-8c1f-001e8c717e5b</oid>
    <modification>
        <modificationType>delete</modificationType>
        <value>
            <c:employeeType>pirate</c:employeeType>
        </value>
    </modification>
</objectDelta>
```

Object MODIFY:replace Delta



Object MODIFY:replace delta

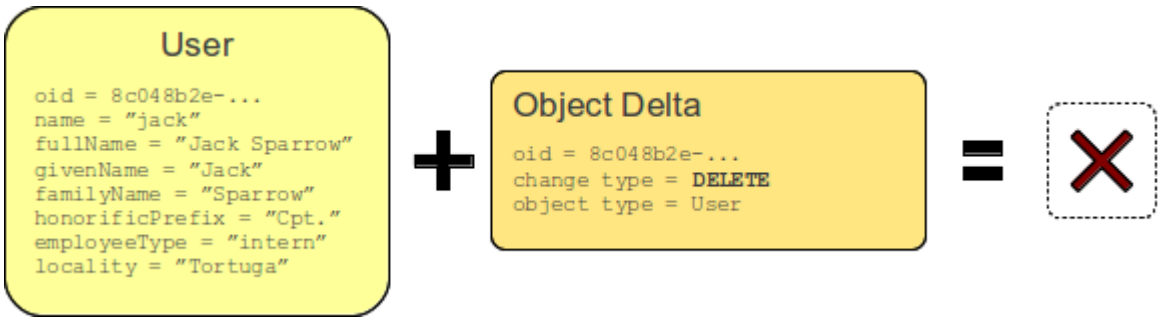
```
<objectDelta>
    <changeType>modify</changeType>
    <objectType>c:UserType</objectType>
    <oid>e3ba0a70-6ef3-11e2-8c1f-001e8c717e5b</oid>
    <modification>
        <modificationType>replace</modificationType>
        <value>
            <c:fullName>cpt. Jack Sparrow</c:fullName>
        </value>
    </modification>
</objectDelta>
```

Complex Object MODIFY Delta

TODO: show how to use several item deltas in a single object delta

Object DELETE Delta

Object DELETE delta is deleting an existing object. It contains just an OID and a type of an object to delete.



The diagram above illustrates the application of DELETE delta. An object did exist before delta application. The delta contains OID and type of the object. When the delta is executed the object is deleted and it no longer exists.

The DELETE object delta obviously works only if such object does exist. Attempt to apply DELETE delta to an non-existing objects results in an error.

Object DELETE delta

```
<objectDelta>
    <changeType>delete</changeType>
    <objectType>c:UserType</objectType>
    <oid>e3ba0a70-6ef3-11e2-8c1f-001e8c717e5b</oid>
</objectDelta>
```