

# POST Processing enhancements

Thoughts on ways to enhance the Post Processing Model

Current Model

- When a POST occurs the Sling Main Servlet hands off the processing of the POST method to the POST processor
- The post processor checks to see the type of operation that has been defined on the POST
- If the POST processor has a handler for the operation it hands off the processing to the `org.apache.sling.servlets.post.PostOperation` implementation that handles that operation
  - If No Operation has been defined it the POST Processor called the `ModifyOperation` Which is the default operation
  - Each `PostOperation` implements the method `run(SlingHttpServletRequest request, PostResponse response, SlingPostProcessor[] processors)`
    - Each `SlingPostProcessor` implements `process(SlingHttpServletRequest request, List<Modification> changes)`

## High level thoughts

The post processors are defined so that custom operations can be implemented by an implementer.

- `SlingPostProcessor` are the main interface that most developers utilize
- `PostOperations` receive an array of potential Processors which may, or may not be applied.
- Each `SlingPostProcessor` is given a List of previous modifications.

The List of modifications, may or may not reflect any changes that have taken place as part of a series of `SlingPostProcessor` interactions. Any changes that occur in a Processor SHOULD be reflected in the list, but the list itself is utilized as an abstraction for reporting.

## Concerns

### No guarantee of accuracy

`PostResponse` is populated with what the Operation believes should be sent as a response. If the Operation is accurately trying to reflect what was modified it would use the List of modifications which it receives after calling all of the `SlingPostProcessors` which in themselves do not guarantee an accurate reporting of everything that was modified.

Why is this a problem? Let's say I want to have add a `SlingPostProcessor` that does one of the following

1. Record any changes that are made to the underlying repository
2. Prevent a certain type value or property from being created
3. Expect the `List<Modification>` to correctly reflect changes up until that moment

## Replication of Work

Each `SlingPostProcessor` and each `PostOperation` is responsible for modifying the repository. \*\* expound on this \*\*

## Possible Solution

Change the `List<Modification>` from a list of what occurred to a list of what needs to be done. This would have the Post Operation iterate through the Post Processors and then, as a final step, take the list of modifications and execute those steps to the `ResourceProvider`.

Benefits

- modification of the repository occurs after all processes have had their say
- Simplifies `SlingPostProcessor` as it just needs to focus on logic
- Allow for guard rails that can occur as the list would be a true reflection of will be changed
- Improved separation of `ResourceProvider` and JCR

## Issues

A modification may not want to be transmitted back to the end requestor. If I have a service which is tracking changes, I may not want the data that I am adding be reported back. Therefore the `Modification` object should be able to track whether the modification should be reported

