**Use Case: Composite Objects**

**Goal:**
Browse composite / hierarchical objects through a designated tool.

**Actors:**
Data provider (human or system), composite tool (system)
Data consumer (human or system)

**Summary:**
No consensus exists regarding the level of granularity at which PIDs are assigned to data objects. Different usage scenarios require different granularities, and thus PIDs must become hierarchically structurable. If both individual objects and the larger composite receive PIDs, then these implicit relations should be discoverable for humans but also for machine agents that for example copy or analyze objects.

**Preconditions:**
Each element object already bears a PID. There is no super object with a PID yet.

**Main Flow (create composite):**
1. The data provider submits a list of element object PIDs to the composite tool.
2. The composite tool analyzes each element.
   1. If there is information on an element that signals it as being in a composition already and that multiple membership is not allowed, the use case terminates with a corresponding failure message to the data provider.
   3. The composite tool creates a PID for the composite and returns it to the data provider.

**Main Flow (discover composite):**
1. The data consumer submits a PID to the composite tool.
2. The composite tool analyzes the PID information.
   1. If the PID refers to a composite, the tool returns a list of all element PIDs to the data consumer.
   2. If the PID refers to an element, the tool returns a list of all super objects to the data consumer (if any). This list may contain only one entry if no multi-membership is allowed.

**Special cases:**
This use case can be extended to many increasingly complex scenarios. It is now kept intentionally simple.

The current “create” flow of events assumes a case where there is no super object with an assigned PID and where the PID created for the composite does not point to a distinct super object itself. This may not be the case for some communities.

The question of multiple membership may be answered differently for particular communities, thus the event flows take care of both possible scenarios.

Some communities may also have composite objects with a huge number of elements, which may render the discoverability of super objects impractical due to the number of required links and PID operations. Things also get more complicated if an object is simultaneously an element and a super object (for hierarchies with multiple layers).