As requested by @mariari.

Note that context discussed in this and subsequent posts will be assumed here.

The objective of a local nameservice application, as I understand it, is to manage a mapping of human-readable names to external identities (and other metadata) which is local to a specific controller (consensus provider). This controller wishes to allocate names under its namespace, perhaps in return for tokens or other forms of payment, and clients wish to purchase these names and associate them with external identities or other metadata that others might want to look up.

A simple version of this application would consist of two resource logics:

- A name service

resource logic, which allows a new name record

to be created iff.: * The human-readable name has not yet been registered.

- Appropriate payment is provided (if applicable).
- The human-readable name has not yet been registered.
- Appropriate payment is provided (if applicable).
- A name record

resource logic, which contains the human-readable name (in the label) and the external identity and other metadata (in the value).

A possible layout for the name record

value with generic metadata is:

type NameRecordValue = (ExternalIdentity, Metadata)

type Metadata = JSONObject

The name record

resource logic would allow the value to be changed iff. a valid commitment is provided by the external identity in question (if desired, this could be separated into two external identities, one to authorize changes, and one which will actually be looked up).

Does this satisfy the basic requirements you have in mind@mariari?