I am trying to understand how difficult it is to program things across shards, where there are only messages sent between smart contracts, and now direct calls.

I am using a simple example of Alice that wants to buy a CryptoKitty using a token that lives on a different shard.

1. There are two shards, A

and B

.

1. ERC token ERCT

lives on shard A

.

1. CryptoKitty smart contract CK

lives on shard B

.

1. Alice wants to buy a Kitty from CK

. The price of a Kitty is 1T

It is assumed that contracts can send authenticated messages to each other across shards with guaranteed delivery. For instance, shard A can sign messages using a BLS threshold signature that is trusted by shard B.

Here is an algorithm that uses two messages and works as follows (I wonder if one can come up with a better algorithm):

a) Alice calls ERCT

telling it to buy 1 Kitty for 1T

.

ERCT

stage:

b) ERCT

verifies that Alice posesses 1T

c) ERCT

transfers 1T

to CK

in the internal table of pending payments

d) ERCT

sends an authenticated message to CK

requesting transfer of Kitty to Alice

.

CK

Stage:

e) CK

verifies that Kitty is still available (it may have been bought by someone else)

f) If Kitty is not available, then CK

sends an error message to ERCT

, including the hash of the original request

g) If the Kitty is available, then CK

transfers Kitty to Alice

, and sends a success message to ERCT

, including the hash of the original request

ERCT Stage:

h) If ERCT

receives an error message, it will revert the pending payment, and credit money back to Alice

i) If ERCT

receives success message, it will commit the pending payment.

Would be interesting to see other examples of synchronous algorithms turned into asynchronous messaging