AIP[1]-018: Consistency & Reusability of Blockchain Services

platfowner@, liayoo@ 2021-03-24

Goals

- List up all services available so far and seek improvement ideas for their consistency and reusability

Requirements

We are looking for solutions for the following issues:

- Consistency in db path definitions
- Reusability of duplicated features (e.g. admin)

Service List

Terminology

We use the following definitions:

- **Service types**: Types of services like ‘escrow’, ‘payments’, ‘staking’, ‘billing’, ‘gas_fee’, ‘sharding’, etc.
- **Service names**: Name of services including apps like ‘escrow’, ‘consensus’, ‘collaborative_ai’, ‘afan’, etc. Notably terms ‘escrow’ and ‘gas_fee’ are used as service names as well as service types.
- **App names**: Names of apps like ‘collaborative_ai’ and ‘escrow’ which is reserved by obtaining the ownership of the states under /apps/<app_name>. It’s notable that some services (e.g. consensus) are defined as apps using _createApp native function for convenience.

DB Paths

Table 1. Database paths of services.

[1] AI Network Improvement Proposal. Visit https://docs.ainetwork.ai for the full list.

Service name	Path name	DB path	Notes
service accounts	(balance)	/**service_accounts**/<service_type>/<service_name>/<account_key>/balance e.g. /service_accounts/payments/collaborative_ai/0x_asdf|0/balance	
	(admin)	/service_accounts/<service_type>/<service_name>/<account_key>/admin e.g. /service_accounts/payments/collaborative_ai/0x_asdf|0/admin	
	_transfer	/**transfer**/<from>/<to>/<key>/value e.g. /transfer/0x_A_User/payments|collaborative_ai|0_xasdf|0/100/value	
escrow	(balance)	/service_accounts/**escrow/escrow**/<account_key>/balance e.g. /service_accounts/escrow/escrow/0x_asdf:0x_qwer:0/balance	
	(admin)	/service_accounts/**escrow/escrow**/<account_key>/admin e.g. / service_accounts / escrow / escrow / payments|collaborative_ai|0x_A_User|0 : 0x_Teachable_NLP : 112233 / admin	
	_openEscrow	/escrow/<source_account>/<target_account>/<escrow_key>/**open_escrow**/ e.g. /escrow / payments|collaborative_ai|0x_A_User|0 / 0x_Teachable_NLP / 112233 / open_escrow	
	_hold	/escrow/<source_account>/<target_account>/<escrow_key>/**hold**/<record_key> e.g.	
	_release	/escrow/<source_account>/<target_account>/<escrow_key>/**release**/<record_key> e.g.	
payments	(balance)	/service_accounts/**payments**/<service_name>/<account_name>/balance e.g. /service_accounts/payments/collaborative_ai/0x_asdf|0/balance	
	(admin)	AS-IS: /payments/<service_name>/config/admin e.g. /payments/collaborative_ai/config/admin TO-BE: /**manage_app**/<**app_name**>/config/admin e.g. /manage_app/collaborative_ai/config/admin	Also, owner restrictions set at /payments/<service_name>/config
	_pay	/payments/<service_name>/<user_address>/<account_key>/**pay**/<record_key> e.g.	
	_claim	/payments/<service_name>/<user_address>/<account_key>/**claim**/<record_key> e.g.	
gas fee	(balance)	/service_accounts/**gas_fee/gas_fee**/<block number>/balance e.g. /service_accounts/gas_fee/gas_fee/10000/balance	
	(admin)	/**manage_app**/**gas_fee**/config/admin	

		e.g. /manage_app/gas_fee/config/admin	
	_collectFee	/gas_fee/**collect**/<from account>/<block number>/<tx_hash>/ e.g. / gas_fee / collect / 0x_A_User / 10000 / 0x_a_tx_hash/	
	_distributeFee	/gas_fee/**distribute**/<block number>/<to account>/ e.g. / gas_fee / distribute / 10000 / 0x_A_User /	
stake & unstake	(balance)	/service_accounts/**staking**/<service_name>/<account_name>/balance, /**staking**/<service_name>**/balance_total** e.g. /service_accounts/staking/collaborative_ai/0x_asdf|0/balance, /staking/collaborative_ai/balance_total	
	(admin)	/**manage_app**/<app_name>/config/admin e.g. /manage_app/collaborative_ai/config/admin	
	_stake	/staking/<service_name>/<address>/<account_key>/**stake**/<record_key> e.g. /staking/collaborative_ai/0x_asdf/0/stake/123	
	_unstake	/staking/<service_name>/<address>/<account_key>/**unstake**/<record_key> e.g. /staking/collaborative_ai/0x_asdf/0/unstake/123	
billing accounts	(balance)	/service_accounts/**billing**/<service_name>/<account_key>/balance e.g. /service_accounts/billing/collaborative_ai/0/balance	
	(admin)	/**manage_app**/<app_name>/config/admin e.g. /manage_app/collaborative_ai/config/admin	
	_createApp	/create_app/<app_name> e.g. /manage_app/collaborative_ai/	

Services and Permission Control

Table 2. Permission control of services.

Service	Depends on	Permission			Notes
		Permission control type (group / individual)	Admin specification	Transfer-from native function	
service accounts	-	individual	(no admin config)	_transfer	
escrow	service account	individual	per account admin config under /escrow	_release	
payments	service account, escrow	group	per app (**<app_name>**) admin config	_claim	
consensus	service account	individual	account owner on the	_unstake	

(staking)			account path		
gas fee	service account	individual	block proposer	_distributeFee	
staking	service account	individual	account owner on the account path	_unstake	
billing	service account	group	per app (**<app_name>**) admin config	_collectFee	

Permission control of service accounts can be summarized as follows:

- Service account itself doesn't have any specific permission control except the write rule "auth.fid === '_transfer'"
- Permissions on service accounts are controlled by other services depending on them
    - Individual permission control is done by the write rule of the corresponding transfer-from native functions using their function-specific admin configs
    - Group permission control is done by the write rule of the corresponding transfer-from native functions using the app admin config

Proposed Design

Key Ideas

- By _createApp()
    - App config (e.g. /manage_app/<app_name>/config) is set
    - Billing accounts are created
    - Service config is set
- App config consists of:
    - Admin config (e.g. /manage_app/collaborative_ai/config/admin)
        - Is reused as admin config for app-dependent services (service accounts, payments, staking, billing)
        - e.g. /manage_app/collaborative_ai/config/admin/{
0x_aaa: true,
0x_bbb: true
}
        - Consists of two types:
            - Group: Applied to all service accounts and service features
            - Individual: Applied to only mapped service accounts and service features
            - Standalone:
            - Native function: Only specific native functions have admin permissions
    - Billing config (e.g. /manage_app/collaborative_ai/config/billing)

        - Billing configs
    - Service config (e.g. /manage_app/collaborative_ai/config/service)
        - Service specific configs
- Replace deposits service with staking service
- For escrow service type, admin is set for each escrow account, and for consistency.

Design Details

- _createApp()
    - path
        - /**manage_app**/<app_name>/**create**/<record_key>/ { admin, billing, service }
        - e.g. /manage_app/afan/create/12345678
    - value details
        - admin
            - The admin setting of the app. The value is copied to `/manage_app/<app_name>/config/admin` and is used for some of the services as well (e.g. payments, staking)
        - billing
            - The optional billing accounts settings of the app. The value is copied to `/manage_app/<app_name>/config/billing`.
            - As of writing, only configuration supported is 'users', which specifies who can spend the AIN in the billing account.
        - service
            - The configuration for the native services for the app. As of writing, the 'staking' and 'payments' services are open or occupied automatically when an app is created. The value is copied to `/manage_app/<app_name>/config/service`
            - For a staking service, a config object is needed with **lockup_duration**.
            - For a payments service, no config object is needed.
    - Upon successful execution of _createApp(), the namespace <app_name> for the app will be reserved in billing, staking and payments services.
- Service/app name restrictions
    - We allow only **alphanumeric characters** and **_ (underscores)** for a service or an app name. Also, native service types are not available to be used as an app name (only exceptions are escrow and gas_fee apps generated by native functions. e.g. /service_accounts/escrow/**escrow** and /service_accounts/gas_fee/**gas_fee**).

Table X. Example of _createApp() value

| {
|---|

```
  “admin”: {
    “0xADMIN_ADDR”: true
  },
  “billing”: {
    “users”: {
      “0xADMIN_ADDR”: true
    }
  },
  “service”: {
    “staking”: {
      “lockup_duration”: 2592000000
    }
  },
  “result”: {
    code: “SUCCESS”
  }
}
```

Further Extensions

- Free tier apps
    - We could provide free tier services for apps with no staking and billing accounts set up. Instead of giving them “credits”, we could set aside 10% of the bandwidth and state gas budgets to the free tier apps and not charge them any fees. Paid apps will still have their shares of the budget, proportional to their stakes, while free apps will have 1/n of the 10% where n = number of free tier apps.
- _deleteApp()
    - We could provide a native function to delete apps and help free up unused space. It could be triggered by setting a value at a path `/manage_app/<app_name>/delete/<request_id>` with a value such as { multiSig }. The multiSig could be a multiSig of the app’s admins, to signify that the admins all agreed to deleting the app. We would also need to have a way to settle the remaining balances in service accounts (payments, staking, billing, etc.).

Conclusion

- We listed up and analyzed all money-related use cases of blockchain services
- For consistency and reusability of blockchain services,
    - We presented how they can be seamlessly integrated with each other by
        - Making them share configs using _createApp() native function and app config structure
        - Migrating consensus deposit/withdraw service to staking service

Links

- AIP-005: Deposit & Withdraw Service (link)
- AIP-010: Simple Payment Service (link)
- AIP-011: Service Account & Transfer (link)
- AIP-012: Escrow Service (link)
- AIP-015: Gas Fee Charging (link)
- AIP-017: Billing Accounts (link)
- AIP-019: Staking Service (link)

Document History

Date	Who	Change	Notes
2021-03-24	platfowner	Initial drafting	
2021-03-31	liayoo, platfowner	Initial version	
2021-05-10	liayoo	Updated for consensus staking's migration from deposit to staking	
2021-05-12	platfowner	Github IDs Link to full list	
2021-05-12	platfowner	Published	