External Balance Daemon

Introduction

The Sippy Softswitch version 5.0 now offers a Remote Procedure Call (RPC) API to track the balance of accounts, customers and vendors. The new RPC API allows the customers to implement their own software platforms to keep track of balance. The softswitch will have it's own implementation of the balance subsystem called balanced.

The RPC specification is a schema written in a Thrift Interface Description Language and provided by Sippy Software. It includes number of remote methods and data structures. These will be described further in this document.

The Balance Entity

The balance subsystem works with abstract balance entities and linked to any Sippy entity such as Account, Customer or Vendor. The balance entity is referred to as 'account' throughout this document.

An account has three billing attributes: balance, credit limit and commodity. The balance and credit limit are self-explanatory and the commodity is a read-only attribute which is created when the account is created and is used solely for grouping when totals are requested. Balances for Accounts, Customers and Vendors have a three letter currency code as a commodity.

An account also has a unique identifier called `i_balance` and reference counter which must be increased when it is assigned to some entity such as Vendor, Account or Customer. When the reference counter reaches zero that means the balance entity is no longer used and it is safe to delete it.

Balance Tracking API

Balance API methods used by the call processing engine

The following methods are used in real time by call processing engine so they must be as fast as possible.

sales@sippysoft.com
support@sippysoft.com
http://www.sippysoft.com/

- `get_balance(i_balance)` - this method received an identifier of an account and returns a `BalanceInfo` structure.
- `make_debit(i_balance, amount, i_balance_update, unblock_ids[])` - the method decreases the `balance` value of the account specified by the `i_balance` identifier. The `i_balance_update` is a unique value obtained from the `next_i_balance_update()` call. The `unblock_ids[]` is a list of blocked amounts acquired with the `block_amount()` call and which will be released after the `make_debit()` call is completed. The `amount` must be greater than zero or an error must be returned. The `make_debit()` call is allowed to make the balance value negative. The returned value is a freshly updated `BalanceInfo` structure.
- `add_credit(i_balance, amount, i_balance_update)` - the method increases the `balance` value of the Balance entity specified by the `i_balance` identifier. The `i_balance_update` is a unique value obtained using the `next_i_balance_update()`. The `amount` must be greater than zero or an error must be returned. The returned value is a freshly updated `BalanceInfo` structure.
- `block_amount(i_balance, amount, i_balance_update, service_id, expires, unblock_ids[])` - blocks the specified amount and returns the block ID (`i_balance_update`) as well as the updated balance info. Returns an error if an attempt to make the balance less than zero has been made or account specified with `i_balance` does not exist. To ensure a block is not stuck forever a blocked amount is released automatically as specified by the `expires` parameter. The default value for the `expires` is 10 minutes.
    - **Important note: this call fails if no `register_service()` call has been made at the session start.**
- `unblock_amount(i_blocked_amount)` - releases a blocked amount. Returns no value.
- `clear_blocked_amounts()` - releases all the blocked amounts which have been blocked by the current service ID which is specified by the `register_service()` call. Returns nothing. This call is intended to be used once at a start-up time of a balance daemon client in order to clear the possibly stuck blocked amounts in case when the client has crashed and has been restarted.
    - **Important note: this call fails if no `register_service()` call has been made at the session start.**
- `register_service(service_id)` - assign a service ID to the current communication session. After this method has been called all the subsequent calls will be bound to the specified service ID. The `service_id` is an arbitrary constant string which identifies the balance API client.
- `next_i_balance_update()` - this method returns a unique identifier which is used in transactional method calls such as `make_debit()`, `add_credit()`, `block_amount()`, etc to ensure that that calls are not applied more than once causing a discrepancy in billing or account housekeeping.

- `clear_blocked_amounts()` - releases all the blocked amounts which have been blocked by the current service ID which is specified by the `register_service()` call. Returns nothing. This call is intended to be used once at a start-up time of a balance subsystem client in order to clear the possibly stuck blocked amounts in case when the client has crashed and has been restarted.
    - **Important note: this call fails if no `register_service()` call has been made before.**

The API methods used by Sippy Softswitch XML-RPC API and Web UI

- `create_balance(balance, credit_limit, commodity, ref_count)` - creates new balance entity with the specified parameters and returns the `i_balance` value of the newly created account. The `commodity` becomes a read-only attribute of the balance entity. The reference counter `ref_count` must be at least 1.
- `inc_ref_count(i_balance, i_balance_update)` - increments the reference counter on an account.
- `dec_ref_count(i_balance, i_balance_update)` - decrements the reference counter. When the reference counter reaches zero the account is safe to be deleted. Please note however that the API does not have an explicit method to delete an account.
- `get_balances(i_balances[], filter)` - given a list of balance IDs returns a list of balances whose balance, credit limit or available balance match the filter expression.
- `get_totals(i_balances[])` - given a list of balance IDs returns sum of credit_limit and balance grouped by commodity.
- `set_credit_limit(i_balance, new_credit_limit)` - sets new value for credit limit.

Contact

For more information about the External Balance Daemon please sales@sippysoft.com or support@sippysoft.com.