Feed Change Proposal

Introduction

Feeds in their current state are highly flexible and their V1 code stands to benefit from some V2 cleanup and refactoring in order to reduce potential bugs and deadlock issues and ease code maintenance going forward. This document proposes a slightly less flexible user model that will enable this cleanup and lead to improved reliability of feeds (as well as addressing a feed start issue that a user raised).

The remaining of this document consists of "**Current User Experience**", "**Proposed User Experience**", and "**Design Implications**".

Current User Experience

1. Creating a primary feed
   ```
   create feed <feed-name> using <adapter>(args[]) [apply function
   <function-name>]
   ```
2. Creating a secondary feed
   ```
   create secondary feed <secondary-feed-name> from feed
   <primary-feed-name> [apply function <function-name>]
   ```
3. Connecting a feed
   ```
   connect feed <feed-name> to dataset <dataset-name> [using
   policy <policy-name>]
   ```
4. Disconnecting a feed
   ```
   disconnect feed <feed-name> from dataset <dataset-name>
   ```

- A feed starts running as soon as it is connected to the first dataset.
- A user can connect a secondary feed to a primary feed while the primary feed is running without needing to stop the primary feed.
- A user can disconnect a primary feed from a dataset and the feed keeps running if a secondary feed is connected to it.
- Each feed can only apply a single function. This function however, can call other functions inside it.
- A user can't set up a feed network then start the feed since the feed starts immediately with the first connection.

Proposed User Experience

1. Creating a feed
   `create feed <feed-name> using <adapter>(args[])`
2. Connecting a feed to a dataset
   `connect feed <feed-name> to dataset <dataset-name> [apply functions <function1-name> <function2-name> <function3-name> .....] [using policy <policy-name>(args[])]`
3. Starting a feed
   `start feed <feed-name>`
4. Stopping a feed
   `stop feed <feed-name>`
5. Disconnecting a feed
   `disconnect feed <feed-name> from dataset <dataset-name>`

To be added later:

1. Suspending a feed. This operation stops the hyracks job feeding different dataset but leaves the adapter buffering data.
   `suspend feed <feed-name>`
2. Resuming a feed. This operation is called on a suspended feed in order to continue ingestion into datasets.
   `resume feed <feed-name>`

- There is no such thing as a primary and secondary feed
- The feed doesn't start immediately, and a feed can be connected to multiple datasets
- Connections can't be created for a running feed
- Connections can't be removed from a running feed
- A user can set up feed network before starting the feed ensuring all datasets get the same set of records.

I also propose the addition of a new type of feed “***change feed***”. This feed will be a feed that carries with it in addition to inserts.

1. deletes
2. upserts
3. rollback

such a feed should not allow for application of functions and should be connected to a single dataset only.

Since the new model enable the user to achieve the features of secondary feeds, I don’t see a harm in removing them.

For example, instead of using the following:

```
create feed TwitterFeed if not exists using
"push_twitter"(("type-name"="Tweet"),("consumer.key"="************"),
("consumer.secret"="**************"),("access.token"="**********"),
("access.token.secret"="*************"));

create secondary feed ProcessedTwitterFeed from feed TwitterFeed
apply function testlib#addHashTags;
connect feed ProcessedTwitterFeed to dataset ProcessedTweets;
```

A user can write the following:

```
create feed TwitterFeed if not exists using
"push_twitter"(("type-name"="Tweet"),("consumer.key"="************"),
("consumer.secret"="**************"),("access.token"="**********"),
("access.token.secret"="*************"));

connect feed TwitterFeed to dataset ProcessedTweets apply functions
testlib#addHashTags;

start feed TwitterFeed;
```

And the final outcome would be the exact behavior.

Design Implications

In the existing design:

- Each source in a feed network is a separate Hyracks job and each dataset that is being fed requires a separate Hyracks job to receive the data.
- The central feed manager needs to keep track of all running jobs and their state and connections.

These two points are direct implications of the flexibility which allows connecting and disconnecting feeds arbitrarily.

In the new design:

- A user can't create connections on an already running feed.
- A user can't disconnect a dataset from a running feed.
- Providing different QoS might be a bit challenging.

Conclusion

In my opinion, the benefits of the proposed change outweighs the cost and the ability to attach a feed to a running feed is a reasonable thing to give up considering the advantages we will gain:

1. More control for users (when to start the flow of data)
2. More control for developers (less concurrency issues)
3. Less resource consumptions (single hyracks job per feed no matter the number of connected datasets)

4. Less things to keep track of (no need to keep track of all the plugging points where a feed can attach itself to)
5. Simple design that is easier to implement correctly (lots of things are simplified making it easier to implement a generic and clear framework for feeds)

Appendix - Secondary feeds in the proposed change

Example1:

Existing

```
create feed TwitterFeed using TwitterAdaptor (”query”=”Obama”,
”interval”=60);
create secondary feed ProcessedTwitterFeed from
feed TwitterFeed apply function addFeatures;
// immediately start flow of data
connect feed ProcessedTwitterFeed to dataset ProcessedTweets;
```

Proposed

```
create feed TwitterFeed using TwitterAdaptor (”query”=”Obama”,
”interval”=60);
connect feed TwitterFeed to dataset ProcessedTweets apply function
addFeatures;
start feed TwitterFeed;
```

Example2:

Existing

```
create feed CNNFeed using CNNAdaptor (” topics ”=” politics , sports
”) ;
create secondary feed ProcessedCNNFeed from
feed CNNFeed apply function addInfoFromCNNWebsite;
// immediately start flow of data
connect feed ProcessedCNNFeed to dataset ProcessedCNN;
```

Proposed

```
create feed CNNFeed using CNNAdaptor (” topics ”=” politics , sports
”) ;
connect feed CNNFeed to dataset ProcessedCNN apply function
addInfoFromCNNWebsite;
start feed CNNFeed;
```

Example3:

Existing

```
create feed Feed A using TweetGen (( ” server ”=” 10.1.0.1:9000 ” ))
apply function tweetlib#f1;
create secondary feed Feed B from feed Feed A apply function
tweetlib#f2;
connect feed Feed A to dataset D1 using policy Discard ;
connect feed Feed B to dataset D2 using policy Discard ;
```

Proposed

```
create feed Feed A using TweetGen (( ” server ”=” 10.1.0.1:9000 ” ));
connect feed Feed A to dataset D1 apply function tweetlib#f1 using
policy Discard;
connect feed Feed A to dataset D2 apply functions tweetlib#f1
tweetlib#f2 using policy Discard ;
start feed A;
//Note: the compiler can detect that in both connections, f1() is
applied first and so can utilize share computation nodes.
```