Integration of Canadian Mutual Funds data (CANNEX) with Mutual Fund Database & Analytics platform.

Goal - To parse and load the fund data to CANNEX

1. To download the file (name configured in the properties file) from FTP *(*File Transfer Protocol) using FTP downloadable libraries.
2. Schedule process to download the file on a daily basis. (using QUARTZ scheduler)
3. Parse the file and insert into database.

Technologies

1. Java 5.0
2. MySql database 5.1
3. Quartz open source java scheduler API
4. Commons FTP library for FTP downloads and uploads.

Scheduler

1. The scheduler starts the standard quartz scheduler by reading the quartz. Properties file and load the job from the XML file specified and starts the scheduler.

2. The scheduler fires the job at given time on a daily basis using the Quartz open source Java scheduler and connects to the FTP and downloads the Fund file specified and parses the file and import it to the database.

FTP session Handling

1. Connect to FTP using FTP commons library and if the connection fails it will logs a message in the log file stating "*failed to connect*". The script makes repeated attempts to connect to the FTP till it reaches the maximum no. of trails set with certain span of intervals and closes for the day. Once the connection is successful it downloads the file.

2. Checks for the file specified whether it exists or not. If the specified file is found it downloads the file and the session will be closed. If the file does not exist logs the message in the log file stating "*file does not exist*". Reconnects to the FTP after a specific span of interval set in the properties file and checks the existence of the file the process will be repeated with the maximum no. of trails set in the properties file. As the maximum no. of trails exceeds the FTP session will be closed for the day.

Parser

1. Once the file is downloaded from FTP, the downloaded file is parsed using the Parser (written in JAVA).

2. Scans the header and the fund information along with the record type and stores in a map.

3. Calls the procedure and populates the data in the database.

4. Loop through all the funds and populates the data.

5. Reads the trailer and matches the count.