Orbit Bridge

Orbit's Agent for External Interfaces

What is in This Document

This document describes the Orbit Bridge application. The information here will be useful to IT staff who support Treasury Orbit operations within their companies.

Orbit support contact information is provided on page 3

Orbit Bridge trouble shooting notes are provided on page 4

Detailed Orbit Bridge documentation begins on page 10

Orbit Support Contact Info

- **Phone**
- **Support line - 877-ORBIT-TMS (877-672-4886) option 2**
-
- **Dan Farrand**
  - **Direct – 307-367-2276**
  - **Cell – 307-413-5060**
-
- **Chris Matthews**
  - **Direct – 415-419-5296**
  - **Cell – 415-613-1879**
-
- **Email**
- **support@orbit-tms.com**
- **danf@walkingmansoftware.com**
- **chris.matthews@orbit-tms.com**

Orbit Bridge Trouble Shooting (Docs begin on Page 10)

- Stop Orbit Bridge Polling
  - Click the disclosure control (top left corner of “Main” tab, next to the check-box labeled “Run”. This causes the “Run” and “Run on Startup” checkboxes to become enabled
  - Un-check the “Run” checkbox. This will stop Bridge from polling. To restart polling, click the “Run” checkbox so that it is checked
  - Check the “Run On Startup” checkbox to tell Bridge to always begin polling when the Orbit Bridge application first starts up
- Start Orbit Bridge
  - Double-click the orbit_bridge.exe
  - Bridge will start and take a few moments to establish connectivity with the Orbit backend server. Connectivity is controlled by the settings on the “Connection Settings” tab
  - If the “Run on Start” checkbox was checked when Orbit Bridge quit, Bridge will automatically start polling after establishing connectivity
  - ity is established by looking at the status box located just below the task list

  - All active tasks in the tasks list will be painted green. Green means we have a legitimate the pathname to the folder the task is monitoring. Any red lines means the folder the task is supposed to be monitoring is no longer available. If you see red, it usually means a network folder/drive has become disconnected. When you restore the network drive, click the “Reset” button(top-middle area of “Main” tab, next to the “Test” button) and Bridge will re-evaluate all of the pathnames
- SOAP Timeout Error messages
  - Occasionally when Orbit Bridge starts, it fails to make contact with the Orbit backend server. You will notice that Bridge seems to be stuck. After 60 seconds Bridge reaches its timeout limit and pops up an error message that usually says something like “SOAP Timeout”
  - This is often simply a startup issue. Dismiss the dialog and click the “Reset” button (top, middle of Main tab). Orbit will attempt the connection again. This second attempt will be successful unless there is a real problem

Orbit Bridge Trouble Shooting (Docs begin on Page 10)

- Loss of visibility to folders located on mapped network drives
  - If you see red lines in the Task List, Orbit is of telling you that pathnames for the watch folders are no longer valid. This happens if the mapped letter drive containing those folders drops off the network
  - When this happens, Treasury users will also see red message lines appearing on the Orbit Message board
  - Resolution: Remap the letter drive. Be sure and use the same letter that was previously used. Click the "Reset" button on the "Main" tab. Orbit will rescan the watch folder pathnames for each task
- SOAP Error Dialog stops processing (rare)
  - If any task encounters an error as a result of sending a command to the Orbit server a SOAP Error dialog will be presented and processing will stop
  - Resolution: Make note of the error before dismissing the dialog. Make a note of the time so that you can find the point in the log where the problem occurred. These errors are most often "SOAP Timeout" problems. This is usually caused by a temporary network issue. Dismiss the dialog. Uncheck the "Run" checkbox. Click the "Reset" button and re-check the "Run" checkbox
- Need to Manually rerun a task
  - If a task fails due to a data problem or some other issue, follow these steps to rerun it.

    1) Use the disclosure button to enable the "Run" checkbox

    2)Uncheck the "Run" checkbox to stop Bridge cycling

    3)Select the task in the task list

    4)Go to the "Setup" tab and look at the folders involved in the task. The file you want to rerun is most likely in folder identified by the "Archive Path"

    5)Go find the file and move it from the archive back into the watch folder path

    6)Return to the "Main" tab and select the task you want to run

    7)Click the "Test" button. You will be asked if you want to test run all tasks or just the selected task. Respond with "Selected" task only. Orbit will process the selected task. Look at the results logged in the list box to the right of the task list. You should be able to tell from the text logged by the task weather the task completed successfully or not

    8)Be sure to re-check the "Run" checkbox to start automatic polling again

Orbit Bridge Trouble Shooting (Docs begin on Page 10)

- Tasks complain of data problems after you fix them in Orbit
  - Bridge caches values for objects like Bank Accounts (AnAccount), Transaction Codes , Banks and Entities. Sometimes a task will report a problem because one of these objects is not configured properly. After you fix the setup in Orbit, you may find that Bridge still reports the same problem. This could be due to the fact that the old setup is still sitting in Bridge's session cache. Left to it's own, the session cache will only be cleared if Orbit Bridge is closed and restarted.
  - Resolution: do one of these
    - Close Orbit Bridge and restart it
    - Use the "Clear Cache" button to clear one or both of the session caches. You will be prompted for each cache
    - Use the "Reset" button to reset the complete runtime environment – this includes closing and reopening the connection the the Orbit backend server, reloading the LocalDB (TClient_Defaults gets reloaded) and clearing session caches. You will be prompted to confirm if you wish to include cache clearing in the "Reset" process
  - The 2 session caches are the General Cache and the REPORT_RUN cache
    - The General Cache is where object values such as Accounts and Transaction Codes are saved. The only purpose of this cache is to improve performance
    - The Report_RUN cache is where Bridge records the date-time of when a report runs. These entries come into play for reports that are only supposed to run once per day. If a report has run and you clear the REPORT_RUN cache, the report will probably run again
- Orbit Message Board reports payment errors and payments are marked as errors, but no traces can be found in the AutoClient errors or archive folders
  - This can happen when the AutoClient emission folder falls off the network. The Bridge payment process will fail to write the MT101 message file and will mark the payment as an error
  - Resolution
    - Get the emission folder back online
    - Use the Orbit Payments Queue window to reset the payment status from ERROR to APPROVED
    - The next Bridge polling cycle will process the payments

Orbit Bridge Trouble Shooting (Docs begin on Page 10)

- ?

Orbit Bridge Trouble Shooting (Docs begin on Page 10)

- ?

Orbit Bridge Trouble Shooting (Docs begin on Page 10)

- ?

External Interfaces and Orbit Bridge Technical and Operating Documentation

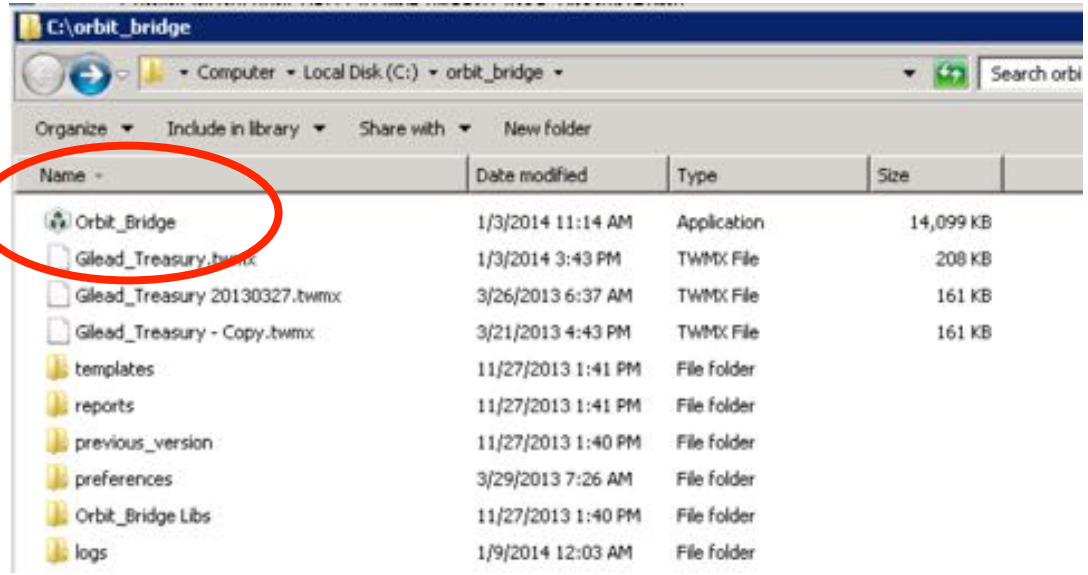
External Interfaces and Orbit Bridge

- The Orbit back-end database runs in our network on hardware OTS controls and maintains. So how do we interface Orbit to services that live behind our customer's firewall ? This is accomplished through an application called Orbit Bridge
- Orbit Bridge is software we provide that runs on a machine behind your firewall. It runs within the confines and security of your network
- Orbit Bridge acts as an agent to enable interfaces between systems inside your network and the Orbit back-end server running outside your network
- Orbit Bridge operates file based interfaces. The basic operating mode that Bridge follows is:
  - Run a list of tasks every 10 minutes (10 minute wait cycle)
  - At the end of of each wait cycle, run each task starting with the first and running in sequence to the last
  - Tasks monitor a designated folder – called the "watch folder" and when a file appears in that folder an action is performed
  - Tasks are highly specialized and perform very specific actions
  - There are different kinds of task. Each task has a specific behavior/capability programed into it. For example, the "FILEACT_RECEPTION_DISPATCH" task looks at files that appear in the AutoClient reception/fileAct folder. It identifies BAI PDR/CDR files that appear there and moves them to separate PDR/CDR folders where they will be processed by a subsequent task
  - The setup of each Task includes one or more arguments that are passed to the task when it runs. Task arguments are used to select optional actions and/or provide variable information to the task (for example: email addresses where notifications should be sent)
  - Typically when a task is done, it moves the file out of the folder the task is watching and into some archive location. In most cases, the folder being watched by the task will be empty when the task is done. If it's watch folder is empty when the task runs, no action will be performed

Orbit Bridge - Runtime Requirements

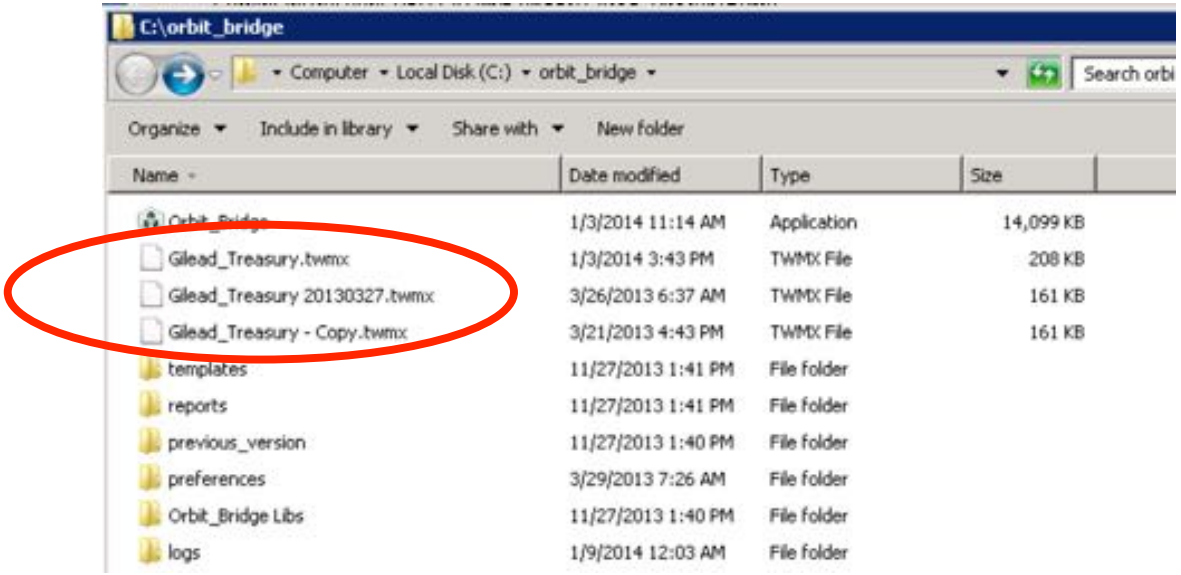
- Orbit Bridge runs as an application on a small, dedicated Windows 7 or Mac OSX machine
- The host computer may be a physical or virtual machine
- The OS may be W7, Windows Server or Mac OSX 10.6+
- The performance required from the machine is minimal. Any i5 class machine with 4BG of memory will suffice
- The bridge machine must be physically secure and must have access to the internet on port 443
- It's useful but not required for there to be email credentials that will allow Orbit Bridge to send SMTP mail out using your Email server
- Orbit Bridge communicates with the Orbit Application Server in our data center through port 443 using HTTPS/SOAP.
- All communications between Orbit Bridge and other services are initiated by Orbit Bridge
- Orbit Bridge does not have the capability to accept or listen for connections initiated outside the application
- Orbit Bridge runs unattended, but does run as a regular user. Bridge DOES NOT run as a faceless service. Bridge does have a UI and it's user must be logged in while Bridge is running.
- Normally Bridge is running on a headless machine and is supported via remote-desktop
- Whomever is supporting Bridge, must be careful not to logout of the user account that is running the Bridge application. If you logout of the Orbit Bridge user, the Bridge application will be closed will no longer be running

Orbit Bridge – Application Folder



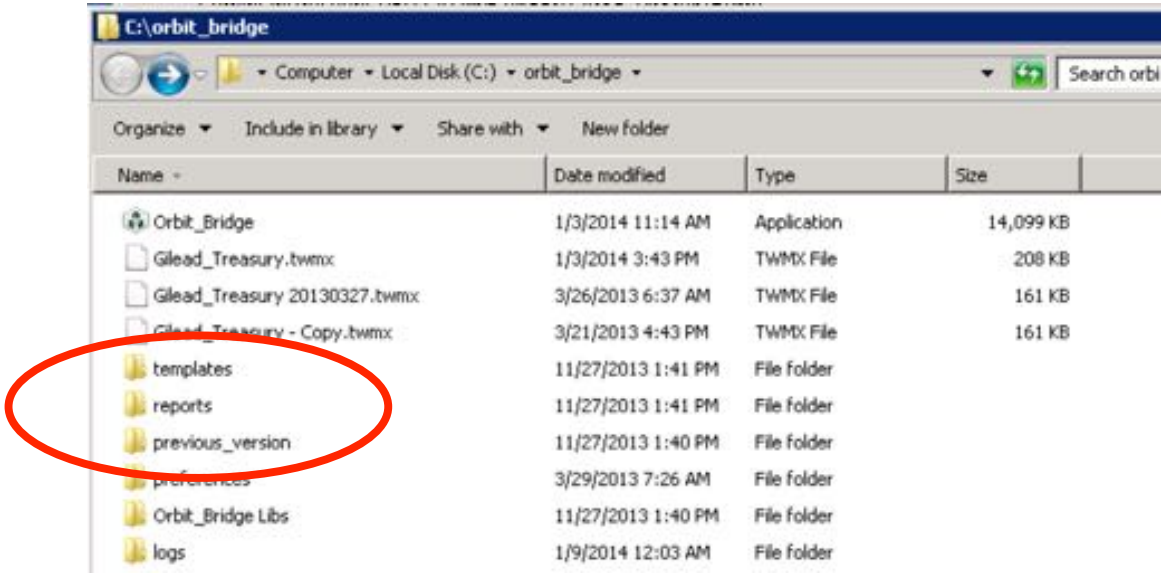
- The Orbit Bridge executable is called Orbit_Bridge.exe (red circle)
- It is located in a folder typically called Orbit_Bridge. This folder is normally located at the root level of your “C” drive, but the Orbit_Bridge folder can be located anywhere that is convenient
- Orbit Bridge is mostly self contained within the Orbit_Bridge folder. To move Bridge, you can simply move the folder
- If any of your Bridge tasks are expected to un-zip files, Bridge will need access to the zlib1.dll which it expects to be located in c:\windows\system
- The Orbit_Bridge.exe must be located in the same folder as the “Orbit_Bridge Libs”, “preferences”, “templates”, “reports” and “logs” folders

Orbit Bridge – Application Folder



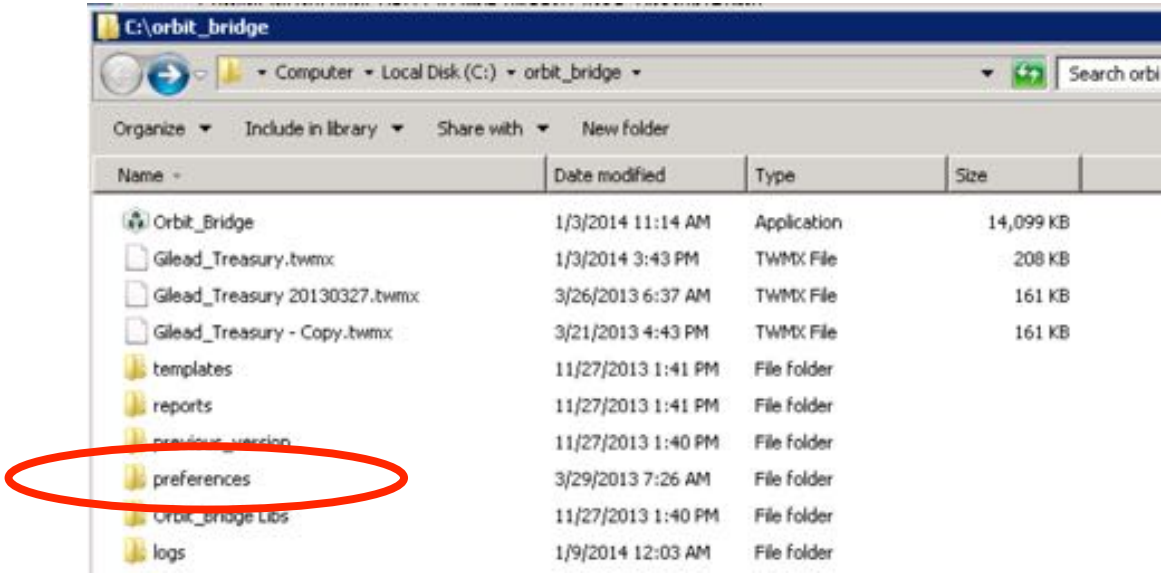
- These files are SQLite database files referred to in this documentation as LocalDB. Bridge only uses one of these files as a time. Other files may be backups or contain different configurations. Bridge will use one of these as the current LocalDB file
- Bridge uses LocalDB to store task/action setup information and to cache frequently used Orbit data. These files have the .twmx extension/file type. If you start Bridge and it does not know where to find the LocalDB, it will prompt you with a file navigation dialog and ask for you to locate it
- In this example, there are several of these files. Only one of them is used at any one time. The others are backup copies or copies with alternative setup/configuration. It's a good idea to keep backup copies of your localDB. SQLite is pretty reliable, but accidents can happen. Anytime you change a task configuration, you should make a backup
- The LocalDB file that is used by Bridge when it starts up is defined in the "preferences_watcher.txt" file. See slides 9 and 10

Orbit Bridge – Application Folder



- The “templates” and “reports” folders are included in the basic installation of Bridge. These folders are used by the “RUN_REPORTS” task. the job of the “RUN_REPORTS” action is to run a report on the Orbit server and email the results to designated users.
  - The report request form may ask for the report output to be XLS. In this case, Bridge uses a file it will find in the “templates” folder as a template for creating a file containing HTML that is readable by Excel as a spreadsheet
  - The “reports” folder used to hold temporary files related to running a report
- “previous_version” folder is not part of the standard Bridge install, but it is a good idea to save the “Orbit_Bridge.exe” and “Orbit_Bridge Libs” folder before updating Bridge with new versions

Orbit Bridge – Application Folder



- When Bridge starts, it looks for a file called “preferences_watcher.txt”. Bridge expects to find this file inside the “preferences” folder which must be located within the same folder as the Orbit_Bridge.exe
- Bridge won’t startup if it can’t find a preferences_watcher.txt file
- The preferences file contains startup information, Orbit connectivity settings and other parameterized defaults
- The format of the preferences file is described on the next slide

Orbit Bridge – Preference File

```
01) YN
02) C:\Orbit_Bridge\Gilead_Treasury.twmx
03)
04) orbit_bridge
05) XXXXX
06) 1250XXXXXX
07) https://orbit1.wms51.com
08) /cash/
09) 443
10) 443
11) 30
12)
13) N
14) 600
15) 10
```

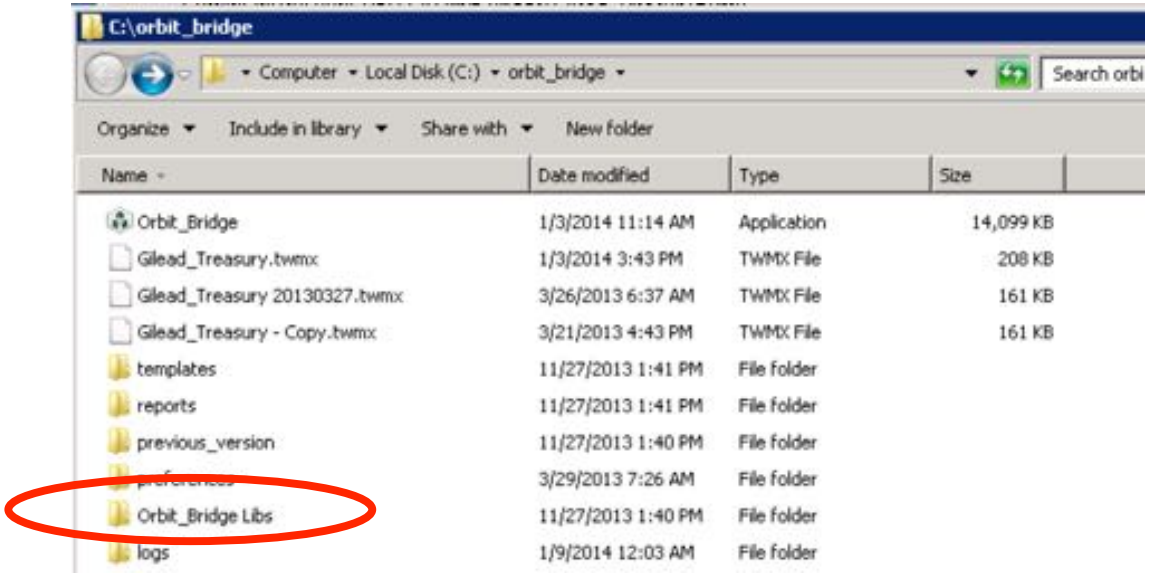
- The content of a preferences file is shown above
- Note that it is stored in plain text – including line XXXXX which represents the the password used by the “orbit_bridge” user to authenticate with the Orbit back-end
- 1) The first line in the file YN = flags used by the Orbit Bridge application
- 2) The second line is the path to the current localDB file that Bridge will use at start-up. Note this is a hard path. If you move Bridge to a new location, this path will no longer be valid and Bridge will prompt you to tell it where the file is located
- 3) This line is normally blank but may be used to store the Clients Name
- 4) The Orbit Bridge UserID used by Bridge to connect to the client. Note here that you must setup up an Orbit userID that will be used by Bridge to login to the Orbit back end server
- 5) The login password used by the Orbit Bridge userID to login to Orbit
- 6) This number is your clientID and is used to identify you as a customer in the Orbit back-end
- 7) https://… is the URL of the Orbit back-end server
- 8) /cash/ is additional path information for the URL
- 9,10) 443, 443 are the port numbers used for communication with the Orbit back-end for HTTP and HTTPS. Note these are both coded 443 which implies that Orbit is never using HTTP

Orbit Bridge – Preference File

```
01) YN
02) C:\Orbit_Bridge\Gilead_Treasury.twmx
03)
04) orbit_bridge
05) XXXXX
06) 1250XXXXXX
07) https://orbit1.wms51.com
08) /cash/
09) 443
10) 443
11) 30
12)
13) N
14) 600
15) 10
```

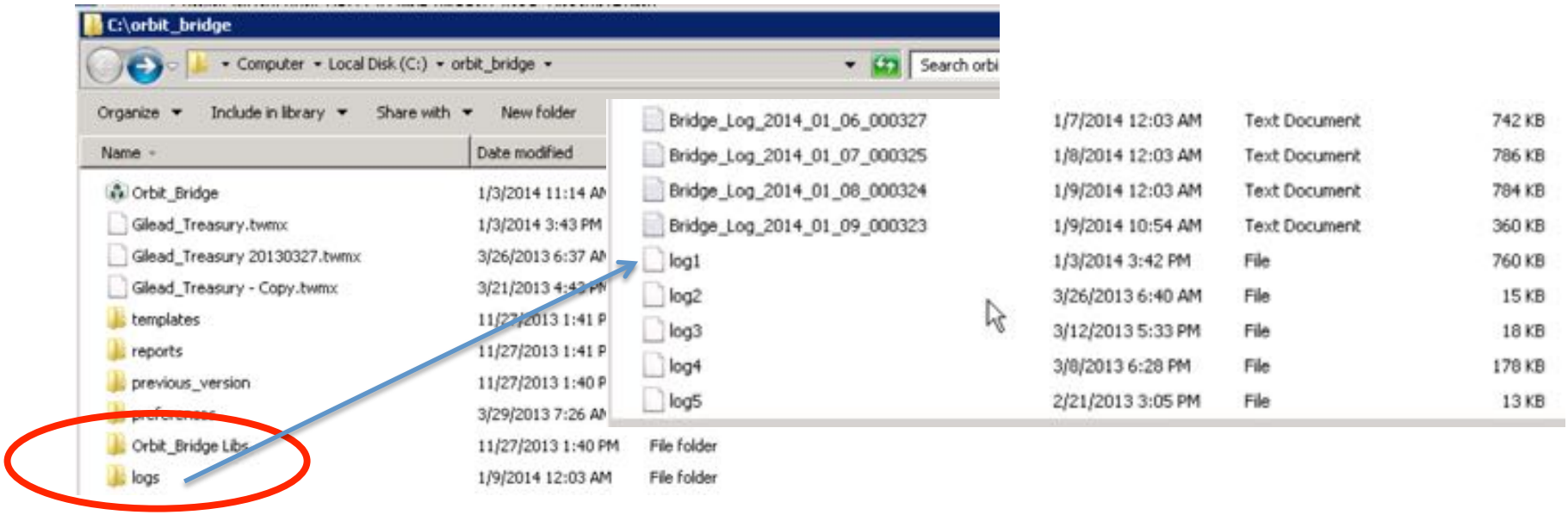
- 11) 30 is the SOAP timeout limit. When Bridge sends an HTTPS SOAP message to the Orbit Server, it will wait 30 seconds for a reply
- 12) This line stores coding that tells Bridge how to display the task list. As of 3/8/2015, the setting in this line has not effect on Bridge. By default the task list will always be displayed listing “Watch Names”
- 13) The flag setting for “Run on Start”
- 14) The polling frequency in seconds (600 means Bridge will run through it’s task list every 10 minutes)
- 15) The number of days Orbit Bridge logs will be retained

Orbit Bridge – Libs



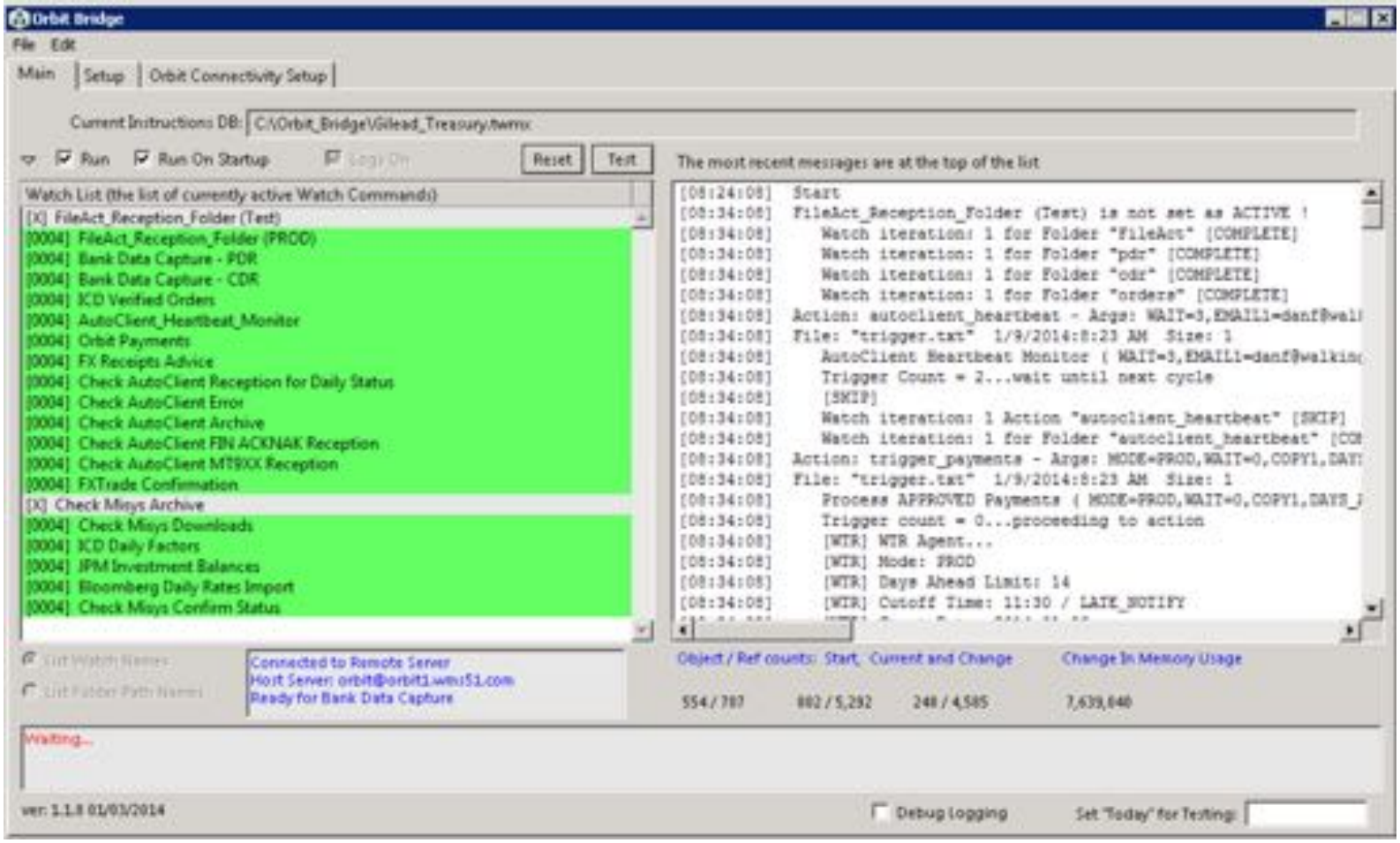
- The “Orbit_Bridge Libs” folder contains DLL’s and other components used by Orbit_Bridge.exe
- When a new version of Bridge is installed, the Orbit_Bridge.exe file and the contents of the Orbit_Bridge Libs folder are replaced

Orbit Bridge – Logs



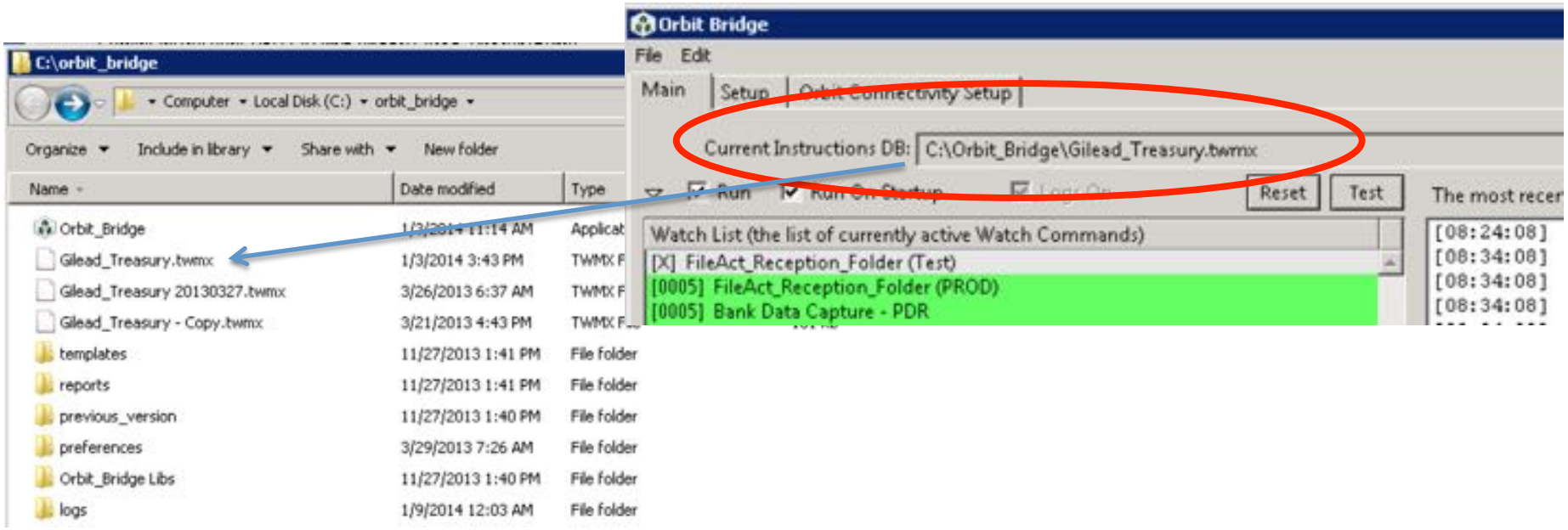
- The “logs” folder contains log files generated by Orbit Bridge while it’s running
- There are 2 categories of log: Bridge_Log and log1, log2…
  - Bridge_Log files contains messages posted by individual tasks as they run
  - log1 type files contain problems encountered by the Bridge application itself
- Bridge creates a new Bridge_Log file for each day and keeps Bridge_Log files for the previous 20 days
- Log1 is always the current session log. If you are looking for messages from Bridge itself, they will be found in Log1

Orbit Bridge – User Interface



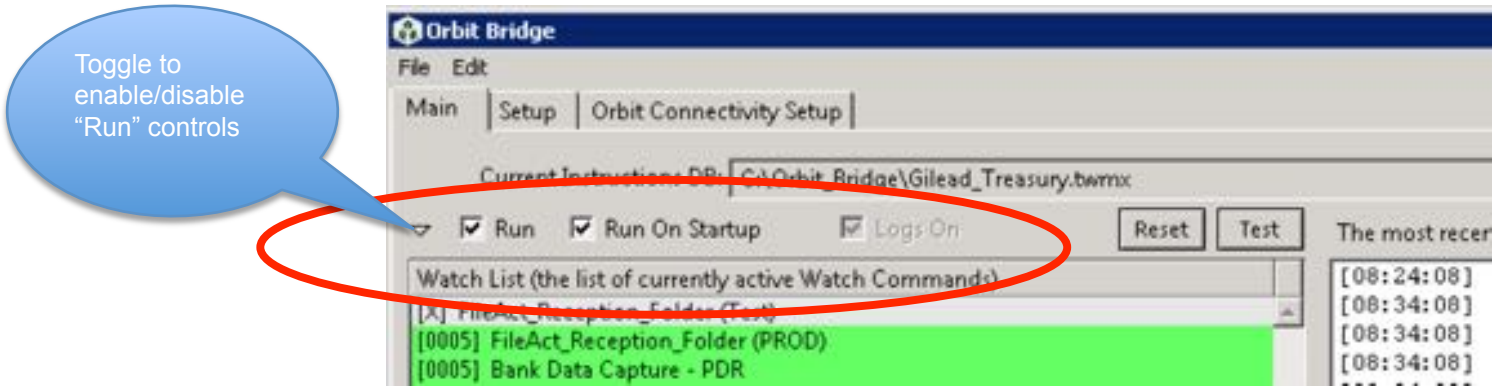
- When you start Orbit_Bridge, this is the main window that appears
- The main window has 3 tabs: “Main”, “Setup” and “Orbit Connectivity”. Newer versions of Bridge will also have a “Crypto” tab that is used to generate and test RSA key pairs
- During operation, the “Main” tab is where everything happens
- UI components are described on the next several slides

Orbit Bridge – User Interface



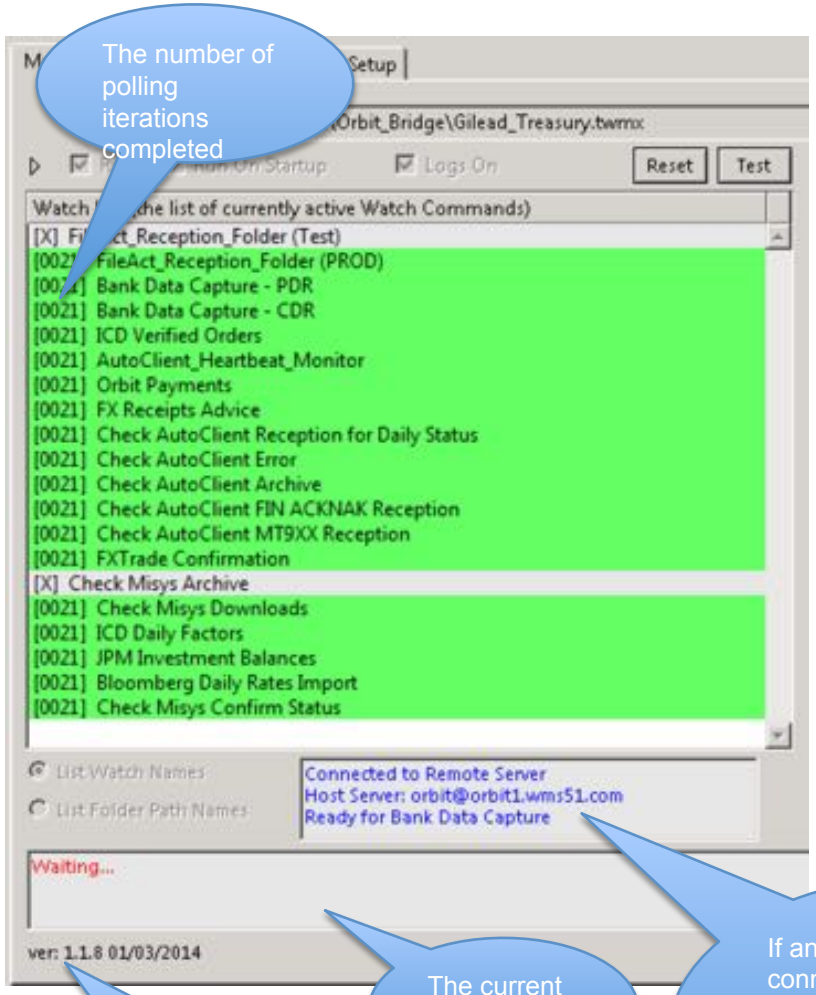
- Bridge uses a local SQLite database (LocalDB) to store task setup information and to cache frequently used Orbit data.
- When Bridge starts, it looks for a file called “preferences_watcher.txt”. Bridge gets a path to the current LocalDB from the preferences file (see slides 10,11)
- The full path name to the current LocalDB file is listed at the top of the main window
- It’s best practice to store your LocalDB document inside the same folder where Orbit_Bridge.exe lives
- If you move LocalDB or change it’s name, the path stored in preferences_watcher.txt will become invalid. The next time you start Bridge it will ask you where to locate the LocalDB file by prompting you with a file navigation dialog

Orbit Bridge – User Interface



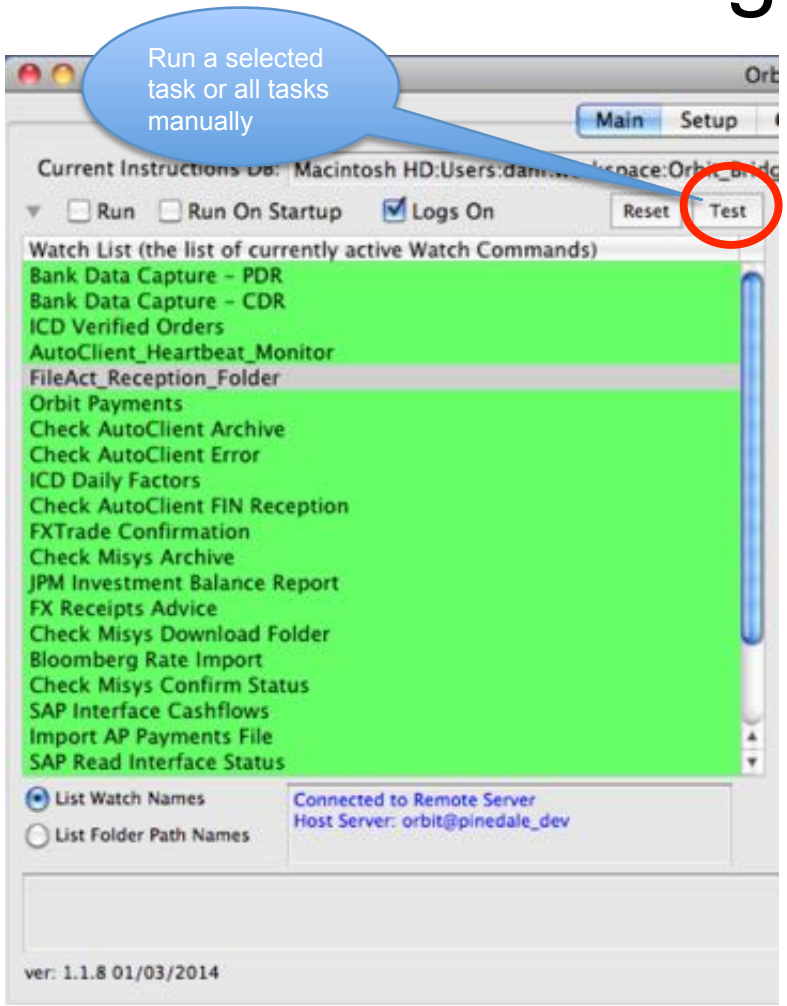
- To tell Bridge to start polling the task list, check the "Run" check box
- If you check the "Run On Startup" check box, Bridge will automatically start polling the task list when Bridge is started
- The "Run" and "Run On Startup" controls can be locked (disabled) to protect against inadvertent change by toggling the triangular "disclosure" control

Orbit Bridge – User Interface



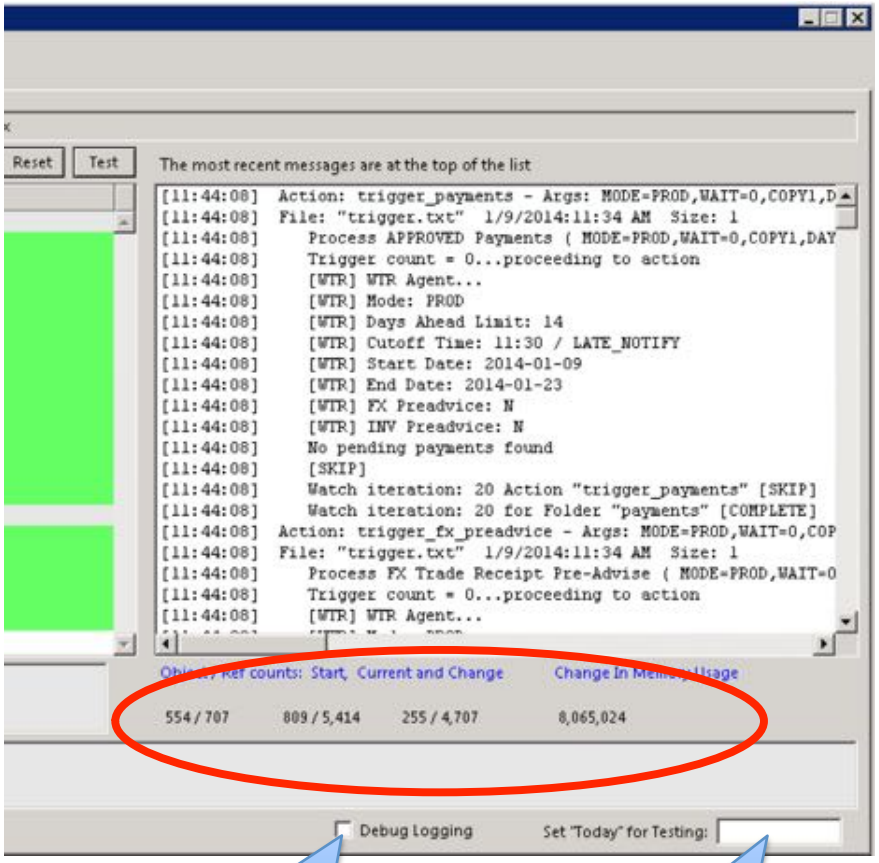
- The list box on the left side of the Main tab is the "Watch List"
- The "Watch List" is a list of all of the tasks currently configured in Bridge
- At the end of each "wait" cycle (usually 10 minutes), Bridge will execute every listed task that is painted green. Each task is run in the order that it appears in the list
- Lines painted grey are "in-active"
- Green lines are "active" and Bridge has confirmed that the path to the watch folder the task is monitoring is valid
- If a task line is red, the path to the task watch folder is not valid. If a the path to a tasks watch folder is invalid, the task will fail when it runs and a message board notice will be posted to the OrbitCM client application
- When a wait period ends, Bridge runs all green tasks present in the task list. When all of the tasks have run, nters another wait period. A Wait period and f task list executions is called 1 cycle

Orbit Bridge – User Interface



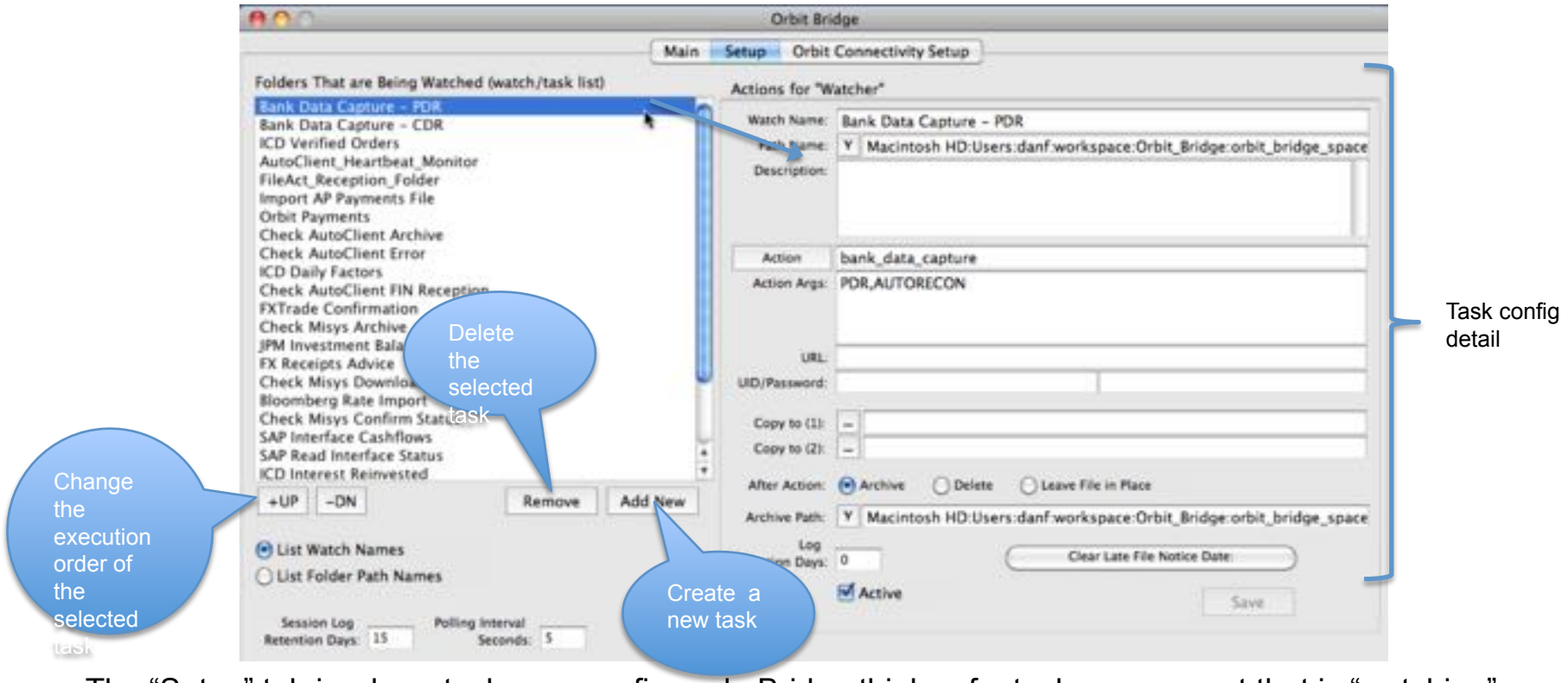
- You can run tasks manually
- Us the toggle to enable the “Run” check-box control. Uncheck the control to stop polling
- Click on the task you want to run to select it
- Click the “Test” button. The button will ask if you want to run ALL tasks or just the selected one
- The task or tasks will run
- WARNING – the button says “Test”, but tasks will be running against whatever Orbit back-end server Bridge is connected to. “Test” does not imply anything about the back-end environment (test or production) that Bridge is currently connected to

Orbit Bridge – User Interface



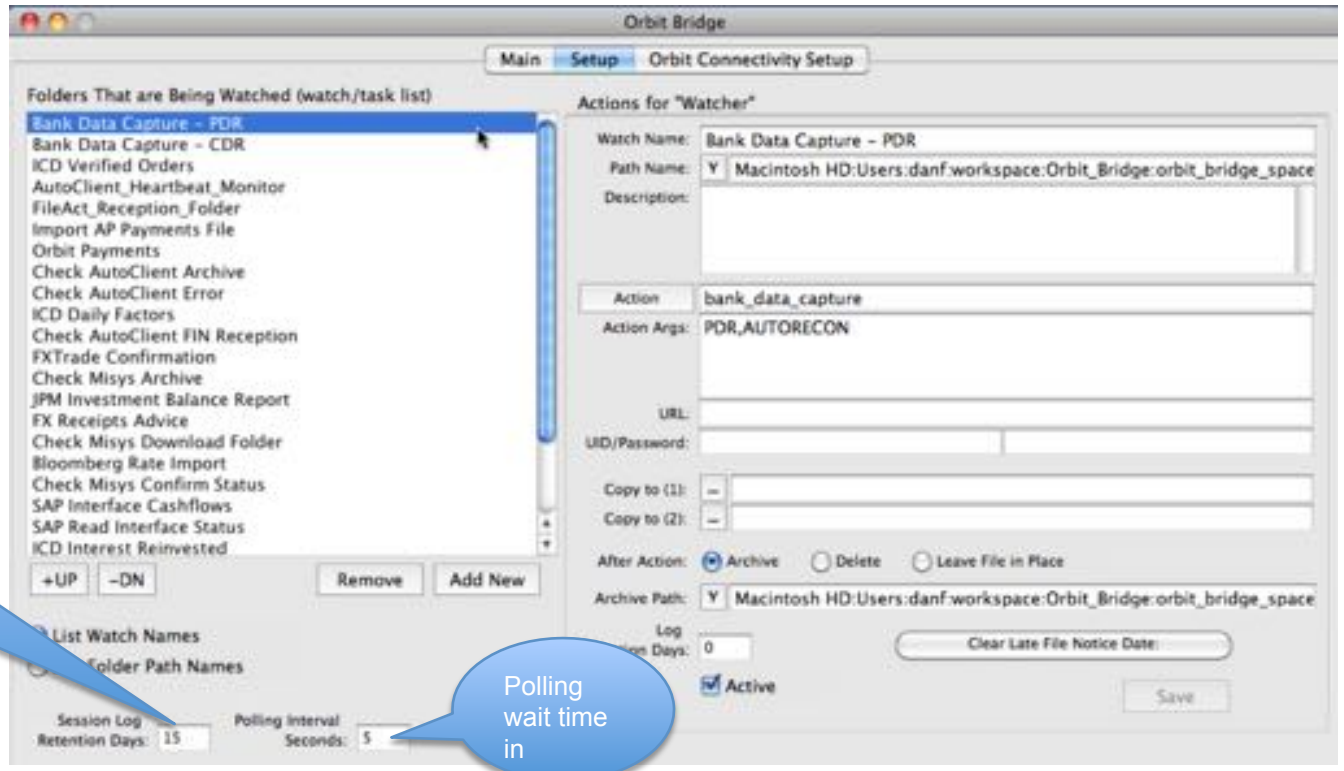
- The listbox on the left side of the "Main" tab is a view into the Orbit Bridge log. What you see here is also written to Bridge_Log… files in the "log" folder
- When a wait cycle is done, Bridge runs all active, green tasks. As each task runs, it logs it's progress, problems and success
- When all green tasks in the task list have run, Bridge begins another wait cycle
- Recent log messages appear at the top. Older messages appear further down the list
- The log listbox holds 500 lines. As new log lines are written to the top of the list, old lines are purged off the bottom
- The Bridge_Log… files contain all log message lines. Orbit begins a new Bridge_Log file every day just after midnight. Bridge Log files are kept for the number of days set in the "Session Log Retention Days" field on the "Setup" tab
- The values circled in red are used to track Bridge memory usage and help identify any memory leaks that may be occurring. As far as we know, Bridge does not have any memory leaks and has successfully run unattended for weeks at a time without being restarted

Orbit Bridge – User Interface



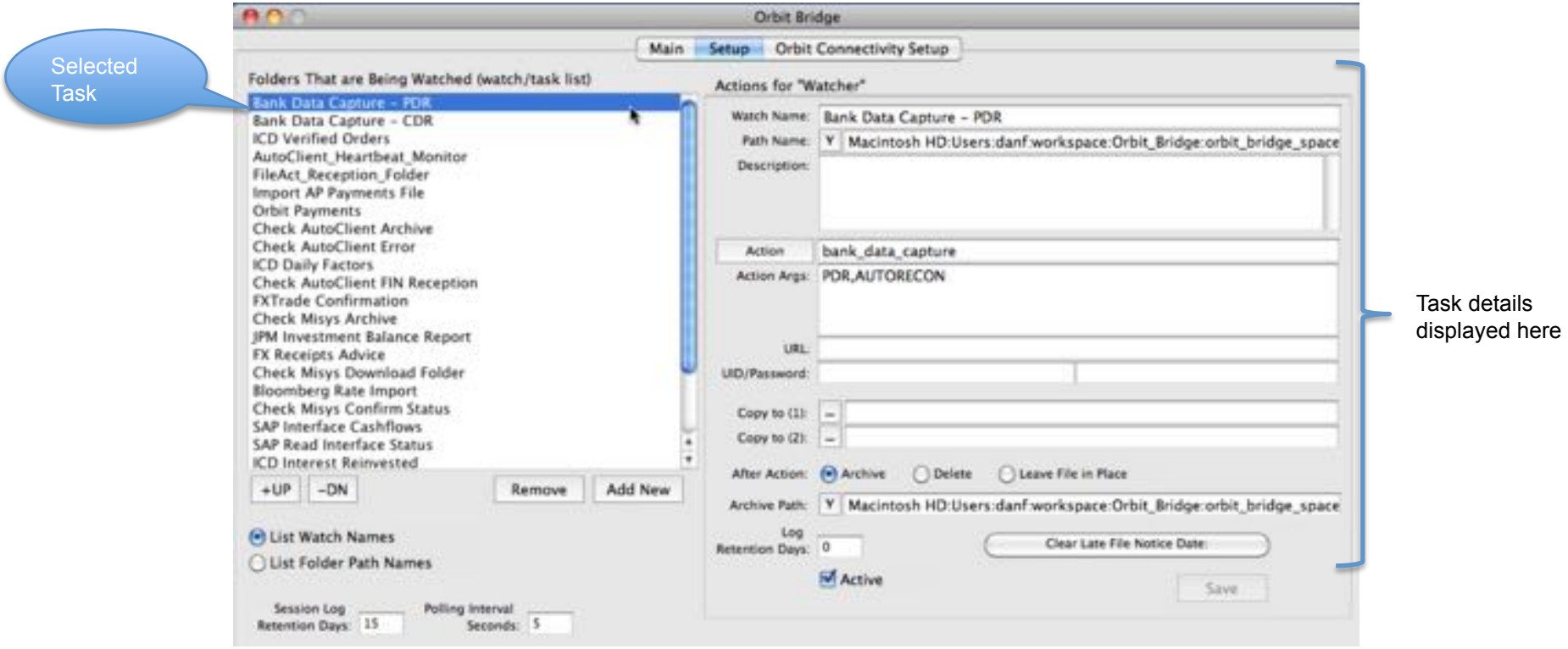
- The “Setup” tab is where tasks are configured. Bridge thinks of a task as an agent that is “watching” a folder so the UI refers to a task as a “Watcher”. We will use task in this documentation
- The list of the left side is the “task list” or “Watch List”
- To add a new task/watcher, click the “Add New” button
- To remove a task, select it and click the “Remove” button
- To change the execution order of a task in the list, select it and press the “+UP” or “-DN” buttons
- To view and edit the configuration for a task, select it in the task list. The configuration details of the task appear in the data entry fields on the right

Orbit Bridge – User Interface



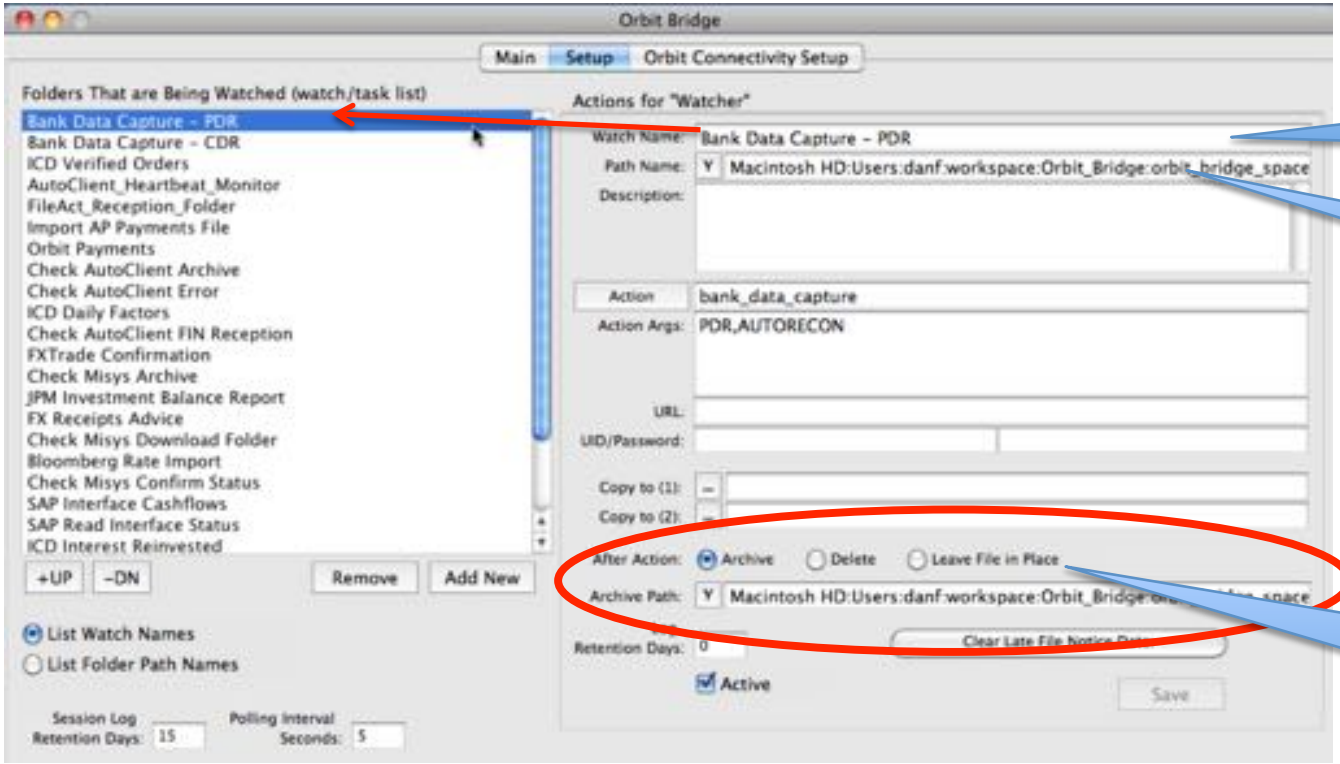
- Set the number of days worth of Bridge_Logs you want retained in the “Session Log Retention Days” field
- Set the wait time between each task execution by enter a number in the “Polling Interval Seconds” field
  - Orbit will sleep for the number of seconds entered
  - At the end of the wait period, Orbit will give each active task in the task list a chance to execute
  - You should set the “Polling Interval” to a large enough value that allows all tasks to complete execution before the end of the next wait cycle
  - To wait 10 minutes, enter 600 seconds. 10 minutes is a good minimum if you have more than a few tasks
  - Orbit will work ok if tasks execution time goes beyond the polling interval. The timer that controls the polling does not begin a new wait period until task list execution is entirely complete. If task execution runs beyond the polling interval, it just means that task execution will not run every 10 minutes on the hour. The start of each execution cycle will drift forward in time with each cycle

Orbit Bridge – User Interface



- Before you can edit a task, polling must be off. To turn polling off, uncheck the “Run” checkbox on the “Main” tab
- To edit a task, select it from the task list
- The configuration details of the task are displayed on the right

Orbit Bridge – Task Setup



1) Task Name as it appears in the task list

2) Watch folder path name

3) After Action the task will perform one of these actions on each file it finds in it's watch folder

1. Every task has a name that appears in the task list

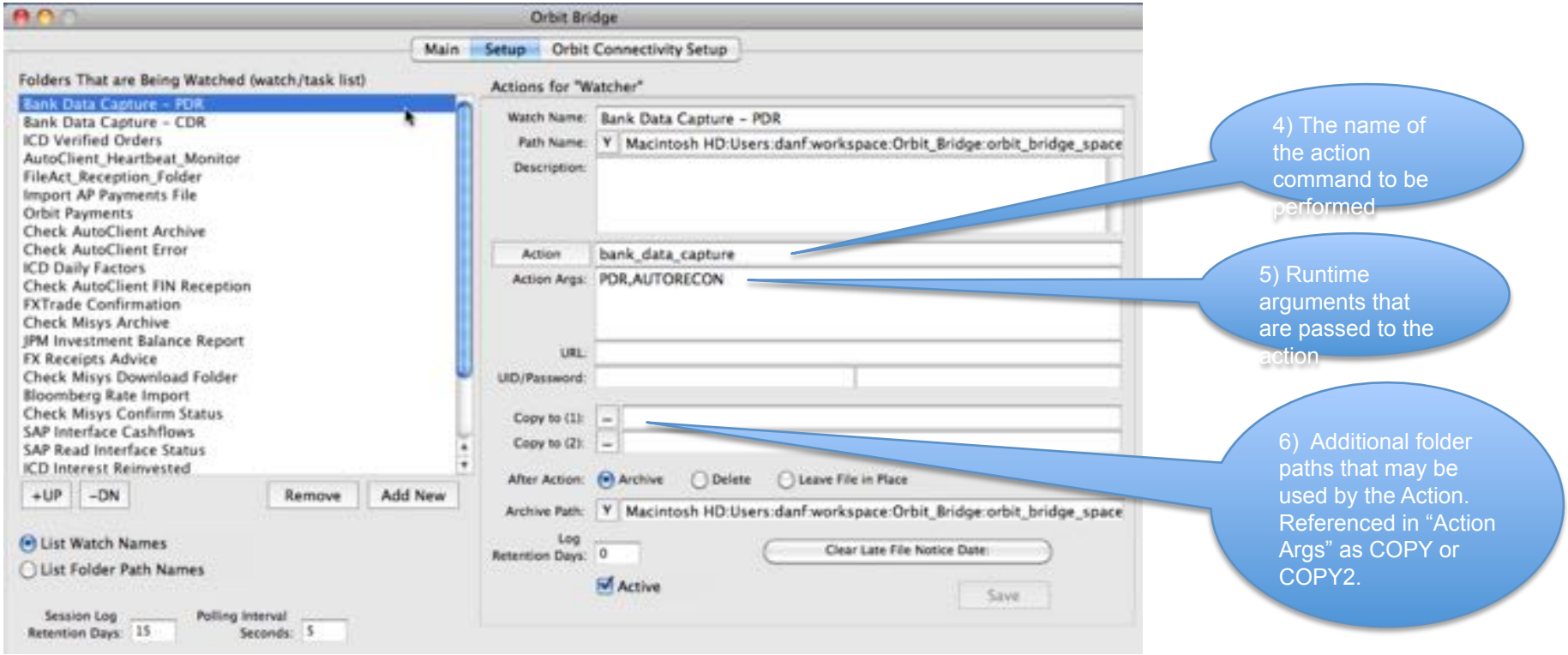
2. Bridge thinks of a task as an agent that watches a folder. Field 2 is the path name to the folder the task is watching

At the end of a wait period, every task is given a chance to execute. When a task executes it looks to see if a file is present in it's watch folder. If a file is present, it applies its processing logic to the file. If multiple files are present,

the task will perform its specific processing on each file.

3. When a task is finished applying its processing to a file, it performs the "After Action" specified (red circle). Possible after actions are: A) move the file to the Archive folder specified in the "Archive Path" field. B) Delete the file. C) Do nothing and leave the file in place

Orbit Bridge – Task Setup



4. Every task has an "Action" that is performed on every file that appears in it's watch folder. An action is the name of a command that Bridge is programed to perform
5. Actions may have zero, one or more runtime arguments that are coded in the "Action Args" field
6. The "Copy To" fields provide 2 additional pathnames to folders that my be used by the Action. If either of these are used, they will be referenced in "Action Args" as COPY1 or COPY2

The following slides describe all of the current Actions available in Orbit

Orbit Bridge - Actions

- FILEACT_RECEPTION_DISPATCH

- What it Does
  - Watch the AutoClient FileAct reception folder(s) looking for files delivered by the SWIFT FileAct service. In FileAct, each data file is delivered with a companion file that we call a Header file. Header files usually have one of the following file extensions: .FA, .PAR, .OK, .DLV. Other extensions are possible. Header files may be formatted as XML or in key=value format. To learn what the data file is all about, Bridge examines the contents of the Header file looking at the <RequestType></RequestType> element
  - The FILEACT_RECEPTION_DISPATCH task reads the Header files and based on the Request Type, moves the data file (also identified in the Header file) to a different folder for processing by a follow-on task
  - Common uses for the task are to stage PDR/CDR BAI/MT940 data files, Bank Analysis 822 Statement files

- Command Arguments
  - This task operates on FileAct companion files which we call Header files. The first thing the task does is attempt to identify if the file being processed is a Header file. If no other arguments are provided, the task looks at the file extension. If the extension is FA, PAR or OK the file is identified as a Header file.
  - TARGET_FILETYPE – This optional argument is used to specify a file extension that identifies Header files. Use this argument if the file extension is not one of the standard ones (FA,PAR,OK). This may also be used in combination with TARGET_FILENAME (see examples)
  - PRIORITY_FILETYPE – This optional argument is used when duplicate copies of Header files appear that have different extensions. For example we have seen instances where PAR and OK files both appear where the OK file is a duplicate of the PAR file. Use PRIORITY_FILETYPE to tell BRIDGE which type to give priority to
  - TARGET_FILENAME – Use this optional argument to match to the front of a file name (see examples)

Orbit Bridge - Actions

- FILEACT_RECEPTION_DISPATCH (continued)
  - RT=SUBSTR|CODE – A number of different RequestType patterns are recognized (see source code for AFileActReception.readTranslatedContent). However, you may add patterns using one or more RT= arguments. The RT= value is composed of 2 fields delimited by a |. The first field will be used in a substring test (if fileRequestType.substr(field1Value) > 0 then). If the substring is > 0 then field2 is assigned as the file type (RT=bai2.xxx|PDR = would identify a RequestType like camt.xxx.bai2.xxx as a identifying a PDR file
  - LOG_FILE_ACTIONS=Y. When this option is set, Orbit will log file actions in the TText table. UI exists in the OrbitCM application to view these messages
  - ARCHIVE_UNKNOWN_RT=XXXX – Use this optional argument to direct Orbit to move both FA and data files for unrecognized request types. XXXX = the name of a folder within the reception folder into which both FA and data files will be moved
  - Recognized Type Arguments: PDR, CDR, 822,LBX_IMAGE, BOA_STMT_REPT,PAIN_CONFIRM

- Examples

  - PAIN_CONFIRM,COPY1,TARGET_FILETYPE=dlv,TARGET_FILENAME=BOA_PAYROLL_

    - The first argument identifies the data Type: PAIN_CONFIRM. When a file matches the Type, it will be copied to the folder identified by the second argument: COPY1
    - TARGET_FILETYPE=dlv – If the file extension = .dlv, the file is treated as a Header File so processing will continue
    - TARGET_FILENAME=BOA_PAYROLL_ - Any file with an extension of dlv beginning with this string will be identified as being a PAIN_CONFIRM data Type (the file will be whatever Type is coded in argument 1). In this case argument 1 does not need to be a Recognized Type. We are using the combination of TARGET_FILETYPE and TARGET_FILENAME to match a file and tell the Task to perform the COPY1 action

Orbit Bridge - Actions

- FILEACT_RECEPTION_DISPATCH (continued)
  - PDR,COPY1,COPY2
    - Copy PDR file to both folders COPY1 and COPY2
    - Header file is .FA or .PAR or .OK and only 1 header file appears
    - Header file is handled by archive
  - PDR,COPY1,CDR,COPY2
    - copy PDR to to folder COPY1
    - Copy CDR to to folder COPY2
    - Header file is .FA or .PAR or .OK and only 1 header file appears
    - Header files are handled by archive
  - LBX_IMAGE,COPY1,LBX_DATA,IGNORE
    - Copy LBX_IMAGE files to folder COPY1
    - Leave LBX_DATA files untouched. LBX_DATA Header is ignored also. LBX_IMAGE Header is handled by archive
    - Header file is .FA or .PAR or .OK and only 1 header file appears
  - LBX_IMAGE,COPY1,LBX_DATA,IGNORE,PRIORITY_FILETYPE=PAR
    - Copy LBX_IMAGE files to folder COPY1
    - Leave LBX_DATA files untouched (IGNORE). LBX_DATA Header files are ignored also. LBX_IMAGE Header is handled by archive
    - Header file is .FA or .PAR or .OK. More than one Header file will appear. We handle the PAR file and ignore any others (PRIORITY_FILETYPE=PAR)

Orbit Bridge - Actions

- FILEACT_RECEPTION_DISPATCH (continued)
  - PDR,COPY1,CDR,COPY2,PRIORITY_FILETYPE=FA,LOG_FILE_ACTIONS=Y
    - Copy PDR files to folder COPY1
    - Copy CDR files to folder COPY2
    - Header file is .FA or .PAR or .OK. More than one Header file will appear. We handle the FA file and ignore any others (PRIORITY_FILETYPE=FA)
    - Actions on this file are logged
    - Header files are handled by archive
  - PDR,COPY1,COPY2,CDR,IGNORE,PRIORITY_FILETYPE=PAR
    - Copy PDR to COPY1 and COPY2
    - Ignore CDR, leave in place for another task to handle. CDR Header will be ignored also. The PDR header will be handled by archive
    - Header file is .FA or .PAR or .OK. More than one Header file will appear. We handle the PAR file and ignore any others (PRIORITY_FILETYPE=PAR)
  - BOA_STMT_REPT,COPY1
    - Copy BOA_STMT_REPT file to the COPY1 folder
    - Header file is .FA or .PAR or .OK and only 1 header file appears
    - Header files will be handled by archive
  - PRIORITY_FILETYPE=PAR,LBX_IMAGE,COPY1,LBX_DATA,IGNORE,UNKNOWN,IGNORE
    - Copy LBX_IMAGE files to folder COPY1
    - Leave LBX_DATA files untouched (IGNORE). LBX_DATA Header will be ignored also. LBX_IMAGE Header will be handled by archive
    - UNKNOWN,IGNORE = Ignore request types that are not recognized
    - Header file is .FA or .PAR or .OK. More than one Header file will appear. We handle the PAR file and ignore any others (PRIORITY_FILETYPE=PAR)

Orbit Bridge - Actions

- FILEACT_RECEPTION_DISPATCH (continued)
  - 822,COPY1
    - Copy 822 files to the COPY1 folder
    - Header file is .FA or .PAR or .OK and only 1 header file appears
    - Header files will be handled by archive

Orbit Bridge - Actions

- BANK_DATA_CAPTURE

- What it Does
  - The watch folder will contain PDR (previous day) or CDR (current day) bank data files in BAI2 format. A parameter is provided that tells the action if the watch folder contains PDR or CDR data. Bank data files are uploaded into Orbit
  - PDR/CDR data is placed in the watch folder by by either an automated SFTP download script or by FILEACT_RECEPTION_DISPATCH (if you are using Alliance Lite II and FileAct for bank data communications)

- Command Arguments
  - CDR – the data file is a CDR file
  - PDR – the data file is a PDR file
  - AUTORECON – if present, directs Bridge to run Auto-Reconciliation after importing bank data

  - Example: PDR,AUTORECON (upload a PDR bank data BAI2 file and run Auto-Reconciliation)
  - Optional argument: LOG_FILE_ACTIONS=Y. When this option is set, Orbit will log file actions in the TText table. UI exists in the OrbitCM application to view these messages.

- Notes

Orbit Bridge - Actions

- BANK_STATEMENT_ANALYSIS
- What it does
  - The watch folder will contain Bank Fee Statements in either 822 or TWIST formats. Statement files are parsed and uploaded into Orbit. At this time only the 822 format is supported
  - Files arrive here for processing via an automated SFTP download request script or as a result of FILEACT_RECEPTION_DISPATCH having moved the file here from an AutoClient reception folder
  - (note: automated SFTP downloads are not part of Orbit Bridge)

- Command Arguments
  - TYPE=822 or TYPE=TWIST - The Type argument identifies the data file format. TWIST not currently supported
  - EOS=[CHAR] – Identifies an End of Segment character used in the file
  - LF=Y, CR=Y, CRLF=Y – if the file includes line endings
  - DUMP=COPY1 or DUMP=COPY2 – Tell Bridge if you want it to save a parsed dump file of the 822 file

  - Example: TYPE=822,EOS=~,LF=Y,DUMP=COPY1

    (import an 822 file. The file includes Line Feeds as it's line endings and uses ~ as the end-of-segment character. Bridge will delete all line endings and convert end of segment characters to "*"

- Notes

Orbit Bridge - Actions

- BLOOMBERG_RATES_IMPORT
- What it Does
  - The watch folder will contain a Bloomberg data rates file with currency exchange rates, volatility and interest rates for a single day. Rates are uploaded into the Orbit daily rates tables
  - Bloomberg rate data is placed in the watch folder by Bloomberg's Data Library application

- Command Arguments
  - NONE

- Notes
  - Rates included in the file are determined by values coded in a Bloomberg Rate Request file. The Bloomberg Data Library services uses the Rate Request File to select the specific information to be included in the file
  - The BB data file looks like this:

```
START-OF-DATA
AED Curncy|0|1|3.673000|
AMD Curncy|0|1|412.050000|
ARS Curncy|0|1|8.131600|
AUD Curncy|0|1|.939200|
BRL Curncy|0|1|2.234200|
CAD Curncy|0|1|1.086600|
CHF Curncy|0|1|.900200|
CLP Curncy|0|1|554.830000|
CNY Curncy|0|1|6.210800|
COP Curncy|0|1|1874.380000|
CZK Curncy|0|1|20.273100|
```

Orbit Bridge - Actions

- Notes (BLOOMBERG_RATES_IMPORT continued)
  - The strings in the BB file that look like this: "AED Curncy" are called symbols
  - Orbit Bridge needs to be told how to recognize symbols and map them into rates
  - Orbit stores mapping information in the TBloomberg_Rates_Map table
  - The map table is populated by Orbit support by running commands like this directly in the database

    - select rates_import_bloomberg_request(12388080,'danf','EUR Curncy::Euro:Spot:');

    - select rates_import_bloomberg_request(12388080,'danf','AU00S/N Index::Australia:Libor:Overnight');
    - select rates_import_bloomberg_request(12388080,'danf','AU0001W Index::Australia:Libor:1 Week');
    - select rates_import_bloomberg_request(12388080,'danf','AU0001M Index::Australia:Libor:1 Month');
    - select rates_import_bloomberg_request(12388080,'danf','AU0002M Index::Australia:Libor:2 Months');
    - select rates_import_bloomberg_request(12388080,'danf','AU0003M Index::Australia:Libor:3 Months');
    - select rates_import_bloomberg_request(12388080,'danf','AU0006M Index::Australia:Libor:6 Months');
    - select rates_import_bloomberg_request(12388080,'danf','AU0009M Index::Australia:Libor:9 Months');
    - select rates_import_bloomberg_request(12388080,'danf','AU0012M Index::Australia:Libor:12 Months');

  - If Treasury adds a new currency/symbol to the BB Rate Request File, they must notify Orbit Support to add a corresponding entry to the TBloomberg_Rates_Map table. If this is not done, Orbit Bridge will not import the new rate data when it begins appearing in the BB data file

Orbit Bridge - Actions

- JPM_ACCOUNT_VALUES
- What it Does
  - The watch folder will contain Net Asset Value data for external money managers at JP Morgan Chase
  - Files are dropped into the watch folder by an automated SFTP download script (note: automated SFTP downloads are not part of Orbit Bridge)
  - Data files are text in the CSV format shown below. The key columns are
    - Column 1 : Investment Account
    - Column 2 : LineType
    - Column 8 : Net Asset Value
    - Column 9 : As Of Date

	Account Number	Ledger Super Category	Account Name	Account Status	Base Currency Code	Ledger Account Long Name	Cost	Market	RUN_ID	Report Date
196	P 75788	LIABILITIES	GBIC - WELLS CAPITAL OFFSHORE	FINAL	USD	SHORT DERIVATIVES	0.0000	0.0000	77,739,205	19-Oct-2012
197	P 75788	LIABILITIES	GBIC - WELLS CAPITAL OFFSHORE	FINAL	USD	FOREIGN EXCHANGE CONTRACTS PAYABLE	0.0000	0.0000	77,739,205	19-Oct-2012
198	P 75788	LIABILITIES	GBIC - WELLS CAPITAL OFFSHORE	FINAL	USD	CAPITAL PAYABLE	(3,750,261.6300)	(3,750,261.6300)	77,739,205	19-Oct-2012
199	P 75788	LIABILITIES	GBIC - WELLS CAPITAL OFFSHORE	FINAL	USD	INCOME PAYABLE	0.0000	0.0000	77,739,205	19-Oct-2012
200	P 75788	NET ASSETS	GBIC - WELLS CAPITAL OFFSHORE	FINAL	USD	NET ASSETS	0.0000	551,765,858.7200	77,739,205	19-Oct-2012
201	P 75789	ASSETS	GILEAD SCI CO - WELLS CAP	FINAL	USD	CASH EQUIVALENTS	0.0000	0.0000	77,739,205	19-Oct-2012
202	P 75789	ASSETS	GILEAD SCI CO - WELLS CAP	FINAL	USD	CASH	0.0000	0.0000	77,739,205	19-Oct-2012
203	P 75789	ASSETS	GILEAD SCI CO - WELLS CAP	FINAL	USD	FIXED INCOME LONG TERM	0.0000	0.0000	77,739,205	19-Oct-2012

Orbit Bridge - Actions

- JPM_ACCOUNT_VALUES (continued)

- Command Arguments
  - DATEFORM=YYYY-MM-DD
    - This tells Orbit the format of date field (column 9)
  - LINETYPE=NET ASSETS
    - Bridge looks for this string in column 2 of the report to identify report lines where net-asset-value amounts are found
  - EXPECTED=WD:07:00
    - Tells Bridge that this report is expected on Week Days by 0700. If the file does not arrive by that time, Bridge will post a notice to the Orbit Message Board

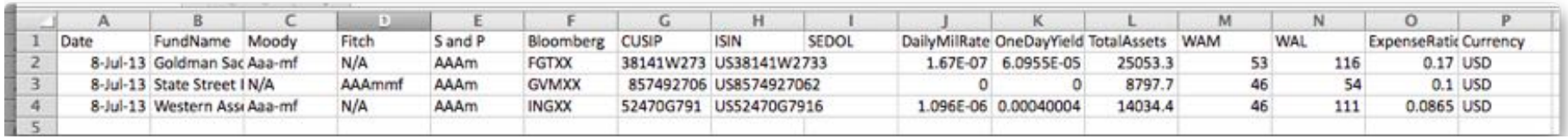
- Notes

Orbit Bridge - Actions

- ICD_DAILY_FACTORS
- What it Does
  - The watch folder will contain CSV formatted earned interest factors for a single day for MMF investment funds at ICD. Orbit uploads factors to MMF tables for use in computing accrued interest
  - Files are dropped into the watch folder by an automated SFTP download script (note: automated SFTP downloads are not part of Orbit Bridge)

- Command Arguments
  - NONE

- Notes
  - An example daily factors file is shown below
  - Bridge uses the Bloomberg ticker symbol in column F to identify the fund
  - Bridge assumes that MMF fund account acctCodes setup in Orbit include the BB ticker symbol as the last part of the code. For example is an entities called 001 and 002 both invested in fund FGTXX, there would be Orbit account codes MMF_001_FGTXX and MMF_002_FGTXX
  - Bridge will decide which rate form to use (columns J or K) based on the setup of the MMF account in Orbit



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1	Date	FundName	Moody	Fitch	S and P	Bloomberg	CUSIP	ISIN	SEDOL	DailyMilRate	OneDayYield	TotalAssets	WAM	WAL	ExpenseRatio	Currency
2	8-Jul-13	Goldman Sac	Aaa-mf	N/A	AAAm	FGTXX	38141W273	US38141W2733		1.67E-07	6.0955E-05	25053.3	53	116	0.17	USD
3	8-Jul-13	State Street I	N/A	AAAmmf	AAAm	GVMXX	857492706	US8574927062		0	0	8797.7	46	54	0.1	USD
4	8-Jul-13	Western Ass	Aaa-mf	N/A	AAAm	INGXX	52470G791	US52470G7916		1.096E-06	0.00040004	14034.4	46	111	0.0865	USD
5																

Orbit Bridge - Actions

- ICD_ORDER_VERIFIED
- What it Does
  - The watch folder will contain CSV formatted “verified order transactions” from ICD. These are MMF purchase or redemption orders Treasury staff have entered on the ICD web-portal. Orbit will upload these and create purchase/redemption transaction.
  - This task may also import messages from ICD reporting monthly interest payments made back into MMF fund accounts
  - Files are dropped into the watch folder by an automated SFTP download script (note: automated SFTP downloads are not part of Orbit Bridge)

- Command Arguments
  - TYPE=I
    - Use this argument to tell Bridge that the transaction file being processed is reinvested interest. The Transaction Type in the file (column N) will be “Purchase”. It is only the presence of the TYPE argument that tells Bridge that this should be processed as an interest depposit
- Notes
  - Bridge depends on a naming convention being followed for MMF account-numbers in Orbit. We expect that an Orbit MMF account-number will be composed of the ICD account number for the entity followed by a hyphen (-), followed by the Bloomberg Ticker Symbol for the fund

Orbit Bridge - Actions

- ICD_ORDER_VERIFIED (continued)

- Notes
  - –A sample ICD verified Order File is shown below. The actual data file is aCSV formatted text file
  - –In the Example below, Bridge will look for an Orbit Bank Account whose account-number = 75280692-CJPXX or 75280692-CILXX (columns D and I)
  - –Orbit will look to column N to indicate if the order is a Purchase or Redemption
  - –ICD may also report interest re-invested in FUNDS. They will provide verified order files in a separate directory for reinvested-interest. The Transaction Type will be Purchase. In order for a ICD_ORDER_VERIFIED to process this file properly, the TYPE=I command argument must be coded
  - –For a Purchase: Bridge creates a Cash IN to the MMF account and a Cash OUT from the Treasury concentration account. Bridge will net all purchases in a single order file into a single wire
  - –For a Redemption: Bridge creates a Cash OUT from the MMF account and a Cash IN to the Treasury concentration account. Bridge expect separate wires in for redemptions from each fund
  - –For Reinvested Interest: Bridge creates a Cash IN to the MMF account

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
TradeDate	EntityName	ExternalId	AccountId	AccountNick	TradeTicketIc	OrderId	TransactionIc	Bloomberg	CUSIP	ISIN	SEDOL	FundName	TransactionT	Currency	TransactionA
25-Feb-14	Bristol Meye	2382813	752-80692	BMS&GSLLC	1059306	68915	92241	CJPXX	4812A0367	US4812A03674		JPMorgan Pr	Purchase	USD	10000000
25-Feb-14	Bristol Meye	2382814	752-80692	BMS&GSLLC	1059306	68915	92242	CILXX	52470G882	US52470G8823		Western Ass	Purchase	USD	10000000

Orbit Bridge - Actions

- TRIGGER_MISYS_FX_CONFIRM
- What it Does
  - The watch folder contains a “trigger” file. This is a simple text file whose presence triggers the execution of the action. This action queries the Orbit database for FX Trades whose trade status is “EXECUTED”. The action prepares MT300 confirmation messages for transmission to the trade confirmation service called Misys. MT300’s are written to a folder that is monitored by Misys FTS. When FTS sees a file in its folder, it uploads that file to Misys
- Command Arguments
- Notes

Orbit Bridge - Actions

- MISYS_DOWNLOADS_WATCH
- What it Does
  - The Misys FTS program downloads MT300 message files from Misys. Those files are written to the folder the MISYS_DOWNLOAD_WATCH action is watching. The action examines files and detects messages denoting trade confirmation from Misys. The FX Trade status of confirmed trades is updated in Orbit
- Command Arguments
- Notes

Orbit Bridge - Actions

- MISYS_CHECK_CONFIRM_STATUS
- What it Does
  - The watch folder contains a “trigger file”. The action executes once each cycle. The purpose of this action is to look for any FX Trades that remain unconfirmed by a certain point in time (usually later in the day on the day the trade was originally executed). Message Board Notices are generated for trades remaining unconfirmed. The Orbit FX Trade status is set to “U” – meaning “NOT CONFIRMED”

- Command Arguments
- Notes

Orbit Bridge - Actions

- TRIGGER_PAYMENTS
- What it Does
  - The watch folder contains a “trigger file”. The action executes once each cycle. The action queries the TCash_WTR_Que table and selects all payments whose status = APPROVED
  - Approved payments are processed creating MT101/MT103 message files
  - Message files are written to the AutoClient “emission” folder for upload to SWIFT

- Command Arguments
- Notes
  - I work closely with SWIFT_ARCHIVE_FIN_WATCH, SWIFT_ERROR_FIN_WATCH and SWIFT_FIN_RECEPTION_WATCH

Orbit Bridge - Actions

- SWIFT_ARCHIVE_FIN_WATCH
- What it Does
  - Watch the AutoClient Archive folder for FIN (MT101) messages that have been successfully uploaded by AutoClient to the Alliance Lite II gateway server

- Command Arguments
- Notes

Orbit Bridge - Actions

- SWIFT_ERROR_FIN_WATCH
- What it Does
  - Watch the AutoClient Error folder for FIN (MT101) messages that failed being uploaded by AutoClient to the Alliance Lite II gateway server

- Command Arguments
- Notes

Orbit Bridge - Actions

- SWIFT_FIN_RECEPTION_WATCH
- What it Does
  - Watch the AutoClient Reception folder for FIN ACK/NAK messages. Use these messages to update the status of Orbit generated MT101 payments
  - Also used to monitor the reception folder and clear out files we want to ignore

- Command Arguments
  - MSG_TYPE_COPY= MT199.COPY1 or MT199.COPY2 (MT199 = a message type we know how to handle)
  - IGNORE=COPY1 or COPY2 (MT999 free format messages and MT081 – Daily Status)
    - If IGNORE=COPY1 is present then message types that are marked as IGNORE will be moved to COPY1
    - If IGNORE= is not present as an Argument, IGNORED message types will be left alone
  - IGNORE_MT101 – mark MT101 files as IGNORE
  - IGNORE_ACKNAK – mark ACK/NAK files as IGNORE
  - NAK_NOTIFY=Y – If a NAK appears notify somebody
  - EMAIL=
- Notes
  - Valid Message Types:
  - MT081 (Daily System Messages) (IGNORE)
  - MT101
  - MT011 (AutoClient Delivery Notice)
  - MT900 (Debit Advice)
  - MT199 (Payment Format Error)
  - MT999 (Free Format Message) (IGNORE)

Orbit Bridge - Actions

- TRIGGER_FX_PREADVICE
- What it Does
  - The watch folder contains a “trigger file”. The action executes once each cycle. The action queries the Cash Ledger identifying FX Trade settlement receipts.

- Command Arguments
- Notes

Orbit Bridge - Actions

- TRIGGER_PAYMENTS_EXPORT
- What it Does
  - Export payments from Orbit for consumption by an external system
  - Data exported by this task may be
    - payment requests
    - debit/credit advice
    - GL Journal Entries

- Command Arguments
- Notes

Orbit Bridge - Actions

- IMPORT_EXTERNAL_INTERFACE_PAYMENT_STATUS
- What it Does
  - Read in a status file related to TRIGGER_PAYMENTS_EXPORT

- Command Arguments
- Notes

Orbit Bridge - Actions

- IMPORT_EXTERNAL_PAYMENTS_FILE
- What it Does
  - Read in a payment file from a external system

- Command Arguments
- Notes

Orbit Bridge - Actions

- AUTOCLIENT_HEARTBEAT
- What it Does
  - xxx

- Command Arguments
- Notes

Orbit Bridge - Actions

- RUN_REPORTS
- What it Does
  - This task runs a report based on information specified in the file that is handled by the task

- Command Arguments
  - RCACHE=COPY1 – Use the COPY1 folder as a work folder for report processing
  - CLEANUP=N – Clean up files created in the work folder (Yes or No)
  - CLEAR_RUN_FLAG=N – Clear the report run-timestamp from the session cache. If Y this allows us to run a one time only report over and over again. One time only reports will not run if a run-timestamp is present in the session cache.
  - TIME_TEST=N – This allows us to test the logic for determining if a report is scheduled to run. You will be prompted to enter a hour and minute that will be used in place of the machine hour and minute. This lets you test running a report that is scheduled to run at 11:00 AM even though the machine time is 10:00 am. When prompted you would enter 11:15 and the “IsRunnable” method would act as if the machine time is 11:15
- Report Request File Format
  - //Report Request Files should have the following form
  - JOBNAME="Tracked Changes to GL Account Strings"
  - DAYSOFWEEK=*****SS
  - //DAYSOFMONTH=1,15,LAST
  - HOUR=07
  - MINUTE=00
  - FILENAME=GL_CHGS_LOG
  - //OUTPUT=XLS

Orbit Bridge - Actions

- RUN_REPORTS (continued)
- Report Request File Format (continued)
  - OUTPUT=PDF
  - //DELIVERY=FILE
  - DELIVERY=EMAIL
  - SUBJECT=Changes to GL Account Strings in Orbit [STARTDATE] to [ENDDATE]
  - MSGBODY=Attached is a PDF report showing changes to GL Account strings that occurred From [STARTDATE] to [ENDDATE]^^
  - EMAIL1=danf@walkingManSoftware.com
  - //EMAIL2=ykong@Brocade.COM
  - //EMAIL3=ar@brocade.com
  - //EMAIL4=creditadmin@brocade.com
  - //RCATEGORY=CM.CUSTOM
  - //RESTRICTED=TRUE
  - RWREPORT=TRUE
  - REPORTID=AUDT_002a
  - //
  - //entity
  - //entityids
  - //entityid
  - //currencyids
  - //bankids
  - //asofdate=YYYYMMDD or TD or TD + 10 or PD (previous date) FIRST_DAY_OF_MONTH, LAST_DAY_OF_MONTH, FIRST_DAY_OF_QUARTER, LAST_DAY_OF_QUARTER

Orbit Bridge - Actions

- RUN_REPORTS (continued)
- Report Request File Format (continued)
  - //asofdate=today()
  - startdate=TD - 5
  - enddate=TD
  - //fperiod
  - //regionids
  - //accountid
  - //accountids=01241,09833,41990,75014,79017,34023,75022,34015,31015
  - //txncodeids

Orbit Bridge - Actions

- CONFIRM_FILE_RECEIPT
- What it Does
  - xxx

- Command Arguments
- Notes

Orbit Bridge - Actions

- TRIGGER_PROCESS_UPLOADED_PAYMENT_FILES
- What it Does
  - Handle payment files that have been uploaded by clients using Orbit Lite. This task handles things for 2 steps in the overall process:
  - 1) Verify receipt of the file and ask the users to verify the control amounts Bridge computes for the file
  - 2) Process the uploaded files and create XML PAIN V3 payment files for transmission to the payment bank via AutoClient and FileAct

- Command Arguments
- Notes

Orbit Bridge - Actions

- HANDLE_XML_PAYMENT_CONFIRM_FILES
- What it Does
  - Handle files returned by the bank confirming receipt of XML PAIN V3 payment files

- Command Arguments
- Notes

Orbit Bridge - Actions

- MATCH_FILE_AND_PERFORM_ACTION
- What it Does
  - match a file based on a substring in the file-name. When a match occurs, perform a MOVE or COPY action

- Command Arguments

  - MATCH_STRING
    the string in the file name we will match on
  - MATCH_ACTION
    where we copy the file to (COPY1 or COPY2)
  - HANDLING_CODE
    MXQ = create a folder structure based on QNYY and YYYY_MM_MON,
    FMXQ = same as MXQ except we use fiscal periods
  - OUTPUT_NAME
    A pattern for a new file name. Symbols supported: [YYYY-MM-MON]
  - SEPARATOR
    Input file name uses the separator character for field separators. Used to look for particular information in the file name such as date or reporting period
  - FILEDATE_NODE
    A piece of information in the file name. In this case it is the FILEDATE. Usually expected in YYYYMMDD form. This tells us which file name field number to look into for the FILEDATE value

- Notes

Orbit Bridge - Actions

- IMPORT_EXTERNAL_TRADES
- What it Does
  - This task is used to import data from external sources where data formats and purpose may vary with different clients. Task arguments allow us to dispatch handling to code that is specific to a particular client

- Command Arguments

- IMPORT_TYPE
  FX_TRADES_SAP1 = SAP Format for importing FX Trade Information.
  FX_RATES_SAP1 = SAP Format for importing month end FX accounting rates
  FX_POINTS_SAP1 = SAP Format for importing month end FX forward points
  IC_BALANCE_SAP1 = SAP format for importing inter-company balances
  AGENTS = Import Sigue Agent update records
- DELM
  The character used in the import file to delimit fields
- NOTIFY
  EMAIL = If errors occur, send an email notice
- REPORT
  EMAIL = Send a report describing import file processing. If there are no errors, no imports and no new records, no report will be sent (for example, because all of the trades in the file have already be verified in earlier files)
- EMAIL1, EMAIL2…
  email recipients when NOTIFY and REPORT = EMAIL

Orbit Bridge - Actions

IMPORT_EXTERNAL_TRADES (continued)

- NUMBERS

  This argument tell Bridge how numbers in the import file are formatted. Valid arguments are

  AMERICA (999,999,999.00)

  EUROPEAN (999.999.999,00)

- DATES

  This argument tell Bridge how dates in the import file are formatted. Valid arguments are

  "MM/DD/YYYY", "DD/MM/YYYY", "YYYY/MM/DD", "MM.DD.YYYY", "DD.MM.YYYY", "YYYY.MM.DD"

  "MM-DD-YYYY", "DD-MM-YYYY", "YYYY-MM-DD", "MMDDYYYY","DDMMYYYY", "YYYYMMDD"

- FILENAME_LIKE=[MATCH_STRING]

  A string of characters Bridge will use to select files for processing. Only files in the watch directory that begin with the [MATCH_STRING] will be processed. Files that don't match will be ignored

- VERBOSE

  Y = more text in logs and email reports/notices

- RATE_IMPORT_FORMAT

  This argument is specific to import FX month end rates. It tells us what form those rates are expressed in. Valid values include

  FX = all rates are expressed in units of USD per 1 unit of foreign currency (EUR = 1.25, JPY = 0.0089)

  US = all rates are expressed in units of FX per 1 USD (EUR = 0.80, JPY = 89.88)

- RESTRICT_ACTION

  Used by AGENTS import to restrict the update action permitted: RESTRICT_ACTION=A only adds are p

Orbit Bridge - Actions

- Notes
  - Each Orbit client will likely have their own IMPORT_TYPE arguments to allow Bridge to dispatch to logic that is specific to that client and interface

Orbit Bridge – Local DB Storage

```
DROP TABLE TTableStamps;
CREATE TABLE TTableStamps (
  tableName         Varchar(60),
  updateStamp       Varchar(24),
  userID            Varchar(20)
);

DROP TABLE TText;
CREATE TABLE TText (
    private         Varchar(1),
    fileName     Varchar(60),
    seq              INTEGER,
    text             TEXT,
    tag1            Varchar(24),
    tag2            Varchar(24)
 );

DROP TABLE TCountries;
CREATE TABLE TCountries (
  rid                       INTEGER,
  country                   Varchar(30),
  name                      Varchar(60),
  ISOCode                   Varchar(10),
  ISONumber                 Varchar(10),
  active                    Varchar(1)
);
```

Orbit Bridge – Local DB Storage

```
DROP TABLE TCurrencies;
CREATE TABLE TCurrencies (
  rid                       INTEGER,
  currency                  VARCHAR(8),
  name                      VARCHAR(60),
  countryID                 INTEGER,
  precision                 INTEGER,
  stdSettlementDays         INTEGER,
  quoteTermUnits            VARCHAR(4),
  active                    VARCHAR(1),
  termdays                  INTEGER
);

DROP TABLE TCodes;
CREATE TABLE TCodes (
  rid                       INTEGER,
  codeType                  Varchar(50),
  code                          Varchar(80),
  description               Varchar(255),
  value                     Double,
  comments                  Varchar(512),
  active                    Varchar(1)
);

DROP TABLE TClient_Defaults;
CREATE TABLE TClient_Defaults (
  name                      Varchar(60),
  defaultValue              TEXT
);
```

Orbit Bridge – Local DB Storage

```
DROP TABLE TClient_Extended_Attributes;
CREATE TABLE TClient_Extended_Attributes (
  rid                      INTEGER,
  fname                   varchar(60),
  fvalue                   TEXT
);

DROP TABLE TCounters;
CREATE TABLE TCounters (
    name                VARCHAR(60),
    nextCounter         INTEGER
 );

DROP TABLE TWatchFolder;
CREATE TABLE TWatchFolder (
  rid                      INTEGER,
  watchName                VARCHAR(60),
  description              TEXT,
  pathName                 TEXT,
  itemNumber               INTEGER,
  actionCmd                TEXT,
  actionArgs               TEXT,
  actionURL                TEXT,
  actionUID                TEXT,
  actionPassword           TEXT,
  actionOtherCredential    TEXT,
  copyToFolderPath1        TEXT,
  copyToFolderPath2        TEXT,
  copyToFolderPath3        TEXT,
  copyToFolderPath4        TEXT,
  addFileContentToLog      VARCHAR(1),
  deleteFileAfterAction    VARCHAR(1),
  archiveAfterAction       VARCHAR(1),
  archiveFolderPath        TEXT,
  logRetentionDays         INTEGER,
  active                   VARCHAR(1)
);


DROP TABLE TWatchLog;
CREATE TABLE TWatchLog (
  rid                      INTEGER,
  watchID                  INTEGER,
  itemNumber               INTEGER,
  logDate                  VARCHAR(24),
  logTime                  VARCHAR(24),
  fileName                 TEXT,
  msgText                  TEXT
);
```