**Interface Control System - Documentation**

1. **Interface Processing Package**
   1. Inbound Interfaces
      1. Processing package name. The package must implement the following procedures that will be executed from the \*INBOUND processing job.

procedure on\_start;

procedure on\_data(par\_record in varchar2);

procedure on\_end;

* + 1. \*SPLIT\_INTO ***outbound interface*** WHEN\_START\_WITH ***string***
    2. \*SPLIT\_INTO ***outbound interface*** WHEN\_SAME\_SUBSTRING ***from Index for Length***
    3. \*SPLIT\_INTO ***outbound interface*** FOR\_EACH\_ROW

* 1. Passthru Interfaces
     1. Outbound scripts to send the interface file via the AMI platform.
  2. Outbound Interfaces
     1. Outbound scripts to send the interface file via the AMI platform.

1. **Interface Processing Group**  
     
   The job group that identifies the \*INBOUND, \*PASSTHRU and \*OUTBOUND jobs that will be used to process inbound, passthru and outbound interfaces. The job group must match the required \*INBOUND, \*PASSTHRU or \*OUTBOUND job group code up to the parallel marker (i.e. the # hash character). For example, there are four \*INBOUND jobs with the following job groups; IB\_HIGH#01, IB\_HIGH #02, IB\_HIGH #03, IB\_HIGH #04. The interface loading group for interfaces that are attached to these jobs would be IB\_HIGH.
2. **Interface Loading Type**
   1. Inbound Interfaces  
        
      \*PUSH or \*POLL
   2. Passthru Interfaces  
        
      \*PUSH or \*POLL
   3. Outbound Interfaces  
        
      \*NONE
3. **Interface Loading Group**  
     
   The job group that identifies the \*FILE jobs that will be used to load inbound and passthru interfaces when the interface loading type is set to \*POLL. The job group must match the required \*FILE job group code up to the parallel marker (i.e. the # hash character). For example, there are four \*FILE jobs with the following job groups; GROUP#01, GROUP#02, GROUP#03, GROUP#04. The interface loading group for interfaces that are attached to these jobs would be GROUP.
4. **Interface Search Procedure**  
     
   The search package name when required (optional). The package must implement the following procedure that will be executed from the \*INBOUND, \*PASSTHRU and \*OUTBOUND loaders.  
     
    procedure on\_data(par\_record in varchar2);  
     
   The interface search package allows the interface developer to extract search strings from the interface file and store the string against the current interface instance with a tag that can be used in the interface monitor advanced search to locate specific interface files based on the stored search strings. For example, an Atlas control recipe could store the IDOC number, the process order identifier and the plant code as individual search strings that could then be used by the users to locate specific interface file for a plant.

**Interface Control System – Interface Overview Diagram**

ATLAS

ICS Installations

(Plant Databases, Care, Promax, etc.)

External Business Partners

(Warehouses, etc.)

Internal Applications

(SAP BW, Lotus Notes, etc.)

**FILE SYSTEM**

Passthru

Load Scripts

Inbound

Load Scripts

INBOUND

Directory

INTERFACE

Polling Directories

BIN

Directory

Inbound/Passthru

Files

Inbound/Passthru

Files

OUTBOUND

Directory

Outbound

Files

**LICS and LICS\_APP Schemas**

**SERVER**

**ICS – Interface Control System**

**ORACLE**

**Local Application Schemas (LADS, BDS, etc.)**

**FILE SYSTEM**

Outbound

Send Scripts

Passthru

Files

Outbound

Files

INBOUND

Directory

BIN

Directory

OUTBOUND

Directory

**AMI**

Parallel

Parallel

Serial

SEND INTERFACE FILES

MQFT

MQIF

Data Junction

Inbound Interface

Loader Package

Outbound Interface

Creation Package

Schema

Tables

**Interface Loading**

LICS\_FILE

Passthru Loader

Inbound Loader

Outbound Loader

File

Processors

Parallel

File Poller

Serial

**Interface Processing**

Passthru Processors

Parallel

Outbound Processors

Parallel

LICS\_DATA

Inbound

Processors

Parallel

UNTRIGGERED

TRIGGERED

**AMI**

Parallel

Serial

Parallel

RECEIVE INTERFACE FILES

Data Junction

MQIF

MQFT

LICS\_HEADER

Inbound Only

JAVA (lics\_filesystem)

JAVA (lics\_filesystem)

**Interface Control System – File Polling Configuration and Processing**

**Processing**

**LICS\_FILE\_POLLER**

1. Retrieve the list of files within the inbound directory interface sub directories using the LICS\_FILESYSTEM java stored procedure.
2. Creates a new LICS\_FILE row for each file where the interface directory name becomes the interface identifier.
3. Wakes up all file processing jobs.

**Configuration**

Logical Job Group

**FLL**

**File Processing Job L1** – Job Group = **FLL**#01

**LICS\_FILE\_PROCESSOR**

1. Wakes up.
2. Retrieves all LICS\_FILE rows with AVAILABLE status.
   1. Attempts to lock the LICS\_FILE row.
   2. When LICS\_FILE row locked and still AVAILABLE status executes the LICS\_FILESYSTEM java stored procedure to move the file from the inbound interface directory to the root inbound directory.
   3. Executes either LICS\_INBOUND\_LOADER or LICS\_PASSTHRU\_LOADER depending on the interface type.
   4. Executes the LICS\_FILESYSTEM java stored procedure to archive the file from the inbound interface directory to the archive directory for inbound interfaces only.
   5. Deletes the LICS\_FILE row.
3. Sleeps.

**File Processing Job L2** – Job Group = **FLL**#02

**File Processing Job L3** – Job Group = **FLL**#03

**File Processing Job L4** – Job Group = **FLL**#04

**Interface A** – Loading Group = **FLL**

**Interface C** – Loading Group = **FLL**

**Interface E** – Loading Group = **FLL**

**Interface B** – Loading Group = **FLL**

**Interface D** – Loading Group = **FLL**

LICS\_FILE

Logical Job Group

**FLH**

**File Processing Job H1** – Job Group = **FLH**#01

**File Processing Job H2** – Job Group = **FLH**#02

**Interface X** – Loading Group = **FLH**

**Interface Z** – Loading Group = **FLH**

**Interface Y** – Loading Group = **FLH**

**INBOUND Interface Directories**

Schedule LICS\_FILE\_POLLER as a \*POLLER job with an execution interval of 10 seconds.

**Interface Directory**

Interface Files

**Interface Directory**

Interface Files

Wakes up all active file jobs

(Where job type is equal to \*FILE)

**Interface Control System – Inbound Interface Configuration and Processing**

**Processing**

**LICS\_INBOUND\_LOADER**

1. Retrieve the interface configuration using the supplied interface identifier (e.g. **Interface B**).
2. Creates a new LICS\_HEADER row.
3. Loads the interface data into the LICS\_DATA table.
4. Executes the interface search when search procedure specified in the interface configuration.
   1. Executes LICS\_INTERFACE\_SEARCH.INITIALISE procedure
   2. Executes the Application Search Package ON\_DATA for each LICS\_DATA row
   3. Executes LICS\_INTERFACE\_SEARCH.FINALISE
5. Wakes up the associated processing jobs using the Processing Group from the interface configuration.

**Configuration**

Logical Job Group

**IBREF**

**Inbound Processing Job R1** – Job Group = **IBREF**#01

**Application**

**Inbound Loader Package**

ON\_START

ON\_DATA

ON\_END

**Application Search Package -** ON\_DATA

1. Executes LICS\_INTERFACE\_SEARCH.ADD\_SEARCH for each required search tag and value.

**LICS\_INTERFACE\_SEARCH**

INITIALISE

ADD\_SEARCH

FINALISE

**LICS\_INBOUND\_PROCESSOR**

1. Wakes up.
2. Retrieves all LICS\_HEADER rows with LOADED status.
   1. Attempts to lock the LICS\_HEADER row.
   2. When LICS\_HEADER row locked and still LOADED status executes the application inbound loader procedure specified in the interface configuration.
      1. Execute ON\_START
      2. Execute ON\_DATA for each LICS\_DATA row
      3. Execute ON\_END
3. Sleeps.

**Inbound Processing Job R2** – Job Group = **IBREF**#02

**Inbound Processing Job R3** – Job Group = **IBREF**#03

**Inbound Processing Job R4** – Job Group = **IBREF**#04

**Interface A** – Processing Group = **IBREF**

**Interface C** – Processing Group = **IBREF**

**Interface E** – Processing Group = **IBREF**

**Interface B** – Processing Group = **IBREF**

**Interface D** – Processing Group = **IBREF**

LICS\_DATA

LICS\_HDR\_SEARCH

LICS\_INT\_REFERENCE

LICS\_HEADER

Logical Job Group

**IBTRN**

**Inbound Processing Job T1** – Job Group = **IBTRN**#01

**Inbound Processing Job T2** – Job Group = **IBTRN**#02

**Interface X** – Processing Group = **IBTRN**

**Interface Z** – Processing Group = **IBTRN**

**Interface Y** – Processing Group = **IBTRN**

**INBOUND**

**Directory**

Interface File

Wakes up all related jobs for the Processing Group

(Where matches Job Group up to the parallel marker #)

Execute LICS\_INBOUND\_LOADER from either an inbound script or a File Processor using the interface identifier and the file path and name

**Interface Control System – Passthru Interface Configuration and Processing**

**Processing**

**LICS\_PASSTHRU\_LOADER**

1. Retrieve the interface configuration using the supplied interface identifier (e.g. **Interface B**).
2. Creates a new LICS\_HEADER row.
3. Executes the interface search when search procedure specified in the interface configuration.
   1. Executes LICS\_INTERFACE\_SEARCH.INITIALISE procedure
   2. Executes the Application Search Package ON\_DATA for each LICS\_DATA row
   3. Executes LICS\_INTERFACE\_SEARCH.FINALISE
4. Wakes up the associated processing jobs using the Processing Group from the interface configuration.

**Configuration**

Logical Job Group

**PTR**

**Passthru Processing Job R1** – Job Group = **PTR**#01

**Outbound Script**

**Application Search Package -** ON\_DATA

1. Executes LICS\_INTERFACE\_SEARCH.ADD\_SEARCH for each required search tag and value.

**LICS\_INTERFACE\_SEARCH**

INITIALISE

ADD\_SEARCH

FINALISE

**LICS\_PASSTHRU\_PROCESSOR**

1. Wakes up.
2. Retrieves all LICS\_HEADER rows with LOADED status.
   1. Attempts to lock the LICS\_HEADER row.
   2. When LICS\_HEADER row locked and still LOADED status executes the processing procedure (outbound script) specified in the interface configuration using the LICS\_FILESYSTEM java stored procedure.
3. Sleeps.

**Passthru Processing Job R2** – Job Group = **PTR**#02

**Passthru Processing Job R3** – Job Group = **PTR**#03

**Passthru Processing Job R4** – Job Group = **PTR**#04

**Interface A** – Processing Group = **PTR**

**Interface C** – Processing Group = **PTR**

**Interface E** – Processing Group = **PTR**

**Interface B** – Processing Group = **PTR**

**Interface D** – Processing Group = **PTR**

LICS\_HDR\_SEARCH

LICS\_INT\_REFERENCE

LICS\_HEADER

Logical Job Group

**PTT**

**Passthru Processing Job T1** – Job Group = **PTT**#01

**Passthru Processing Job T2** – Job Group = **PTT**#02

**Interface X** – Processing Group = **PTT**

**Interface Z** – Processing Group = **PTT**

**Interface Y** – Processing Group = **PTT**

**INBOUND**

**Directory**

Interface File

Wakes up all related jobs for the Processing Group

(Where matches Job Group up to the parallel marker #)

Execute LICS\_PASSTHRU\_LOADER from either a passthru script or a File Processor using the interface identifier and the file path and name

**AMI**

**Interface Control System – Outbound Interface Configuration and Processing**

**Processing**

**LICS\_OUTBOUND\_LOADER**

1. Execute one of the CREATE\_INTERFACE function overloads.
   1. Interface identifier only creates a new outbound interface with a generated file and the message name equal to the file name.
   2. Interface identifier and file name creates a new outbound interface with the supplied file name and the message name equal to the file name.
   3. Interface identifier, file name and message name creates a new outbound interface with the supplied file name and message name.
   4. Retrieve the interface configuration using the supplied interface identifier (e.g. **Interface B**).
   5. Creates a new LICS\_HEADER row.
   6. Executes LICS\_INTERFACE\_SEARCH.INITIALISE procedure.
2. Execute the APPEND\_DATA procedure for each data row to add to the interface file.
3. Execute the ADD\_SEARCH procedure for each required search tag and value.
4. Execute the FINALISE procedure to complete the interface loading.
   1. Creates a new LICS\_HEADER row
   2. Executes LICS\_INTERFACE\_SEARCH.FINALISE procedure
   3. Creates the interface file in the outbound directory.
   4. Wakes up the associated processing jobs using the Processing Group from the interface configuration.

**Configuration**

Logical Job Group

**OBR**

**Outbound Processing Job R1** – Job Group = **OBR**#01

**Outbound Script**

**LICS\_INTERFACE\_SEARCH**

INITIALISE

ADD\_SEARCH

FINALISE

**LICS\_OUTBOUND\_PROCESSOR**

1. Wakes up.
2. Retrieves all LICS\_HEADER rows with LOADED status.
   1. Attempts to lock the LICS\_HEADER row.
   2. When LICS\_HEADER row locked and still LOADED status executes the processing procedure (outbound script) specified in the interface configuration using the LICS\_FILESYSTEM java stored procedure.
3. Sleeps.

**Outbound Processing Job R2** – Job Group = **OBR**#02

**Outbound Processing Job R3** – Job Group = **OBR**#03

**Outbound Processing Job R4** – Job Group = **OBR**#04

**Interface A** – Processing Group = **OBR**

**Interface C** – Processing Group = **OBR**

**Interface E** – Processing Group = **OBR**

**Interface B** – Processing Group = **OBR**

**Interface D** – Processing Group = **OBR**

LICS\_HDR\_SEARCH

LICS\_INT\_REFERENCE

LICS\_HEADER

Logical Job Group

**OBT**

**Outbound Processing Job T1** – Job Group = **OBT**#01

**Outbound Processing Job T2** – Job Group = **OBT**#02

**Interface X** – Processing Group = **OBT**

**Interface Z** – Processing Group = **OBT**

**Interface Y** – Processing Group = **OBT**

**OUTBOUND Directory**

Interface File

Wakes up all related jobs for the Processing Group

(Where matches Job Group up to the parallel marker #)

Execute LICS\_OUTBOUND\_LOADER from a stored

**AMI**

**Interface Control System – Stream and Triggered Overview Diagram**

**LICS and LICS\_APP Schemas**

**SERVER**

**ICS – Interface Control System**

**ORACLE**

**Local Application Schemas (LADS, BDS, etc.)**

Triggered Procedure

Stream Invocation

Procedure

Schema

Tables

**Stream and Triggered Processing**

Damon

Processors

Parallel

LICS\_TRIGGERED

LICS\_STR\_ACTION

Stream Poller

Serial

LICS\_STR\_HEADER

Stream Loader

Stream Configuration

Trigger Loader

**Interface Control System – Stream Configuration and Processing**

**Processing**

**LICS\_STREAM\_LOADER**

1. Supports the creation of runtime substitution parameters.
2. Creates a new stream action sequence and creates new LICS\_STR\_ACTION rows for each event in the selected stream header based on the current stream configuration data.

**Configuration**

Logical Job Group

**STR**

**Daemon Processing Job S1** – Job Group = **STR**#01

**Job Procedure**

lics\_trigger\_processor.execute\_from\_daemon

**LICS\_TRIGGERED\_PROCESSOR**

1. Wakes up.
2. Retrieves all LICS\_TRIGGERED rows for the logical job group.
   1. Attempts to lock the LICS\_TRIGGERED row.
   2. When LICS\_TRIGGERED row locked executes the trigger procedure.
   3. Deletes the LICS\_TRIGGERED row.
3. Sleeps.

LICS\_TRIGGERED

Schedule LICS\_STREAM\_POLLER as a \*POLLER job with an execution interval of 30 seconds.

Wakes up all related jobs for the Processing Group

(Where matches Job Group up to the parallel marker #)

**Daemon Processing Job S1** – Job Group = **STR**#02

**Job Procedure**

lics\_trigger\_processor.execute\_from\_daemon

**Daemon Processing Job S1** – Job Group = **STR**#03

**Job Procedure**

lics\_trigger\_processor.execute\_from\_daemon

**LICS\_TRIGGER\_LOADER**

1. Creates a new LICS\_TRIGGERED row using the stream action event configuration.
2. Wakes up the associated processing jobs using the Job Group from the stream action event configuration.

Stream Header

LICS\_STR\_ACTION

Stream Task

Stream Task

Stream Event

Stream Event

Stream Task

Stream Event

Stream Event

Stream Task

Stream Event

Stream Event

**LICS\_STREAM\_POLLER**

1. Retrieve the list of active streams and controls the event processing based on the stream task hierarchy and lock information.
2. Executes the LICS\_TRIGGERED\_LOADER for each event that is ready for processing using the stream action information, that is, the trigger procedure becomes the stream action event procedure.

Scheduled job or stored procedure executes LICS\_STREAM\_LOADER with the requested stream header identifier.