Derivative Clearing System Open Interface V1.4.5 Extended Interface Functionality - Technical Overview Guide

Table of Contents

1. Introduction

The Message Clearing Module (MCM) release V1.4.5 provides Extended Interface Functionality (EIF). This provides further mechanisms for clearing participants and third party vendors to interface with the Derivatives Clearing System (DCS). The new functionality is additional to the current interface options, namely read-access to the MCM databases and interface to the full set of DCS messages using the ActiveX components.

EIF is made available at no extra charge.

External systems that interface to MCM using EIF will initially be provided with seven separate procedures that can be used to pass instructions via DCS messages to ASX Clear (ASXCL). A brief description of each of these procedures is contained in the table below.

No.	Procedure	Definition
1	Exercise Quantity Procedure	A single stored procedure is provided to allow an external system to submit instructions to exercise open long option contracts.
2	Exclude Quantity Procedure	A single stored procedure is provided to allow an external system to exclude open long option contracts from the automatic exercise process.
3	Order Allocation Procedure	Two stored procedures are provided that in combination allow an external system to supply instructions for the allocation of contracts to accounts, for requests to be submitted for contracts to be 'prices averaged' and for contracts to be 'given-up' to a clearing participant. The instructions will relate to a specific order.
4	Trade Allocation Procedure (*two versions)	A stored procedure is provided that allows an external system to supply instructions for the allocation of contracts to accounts and for contracts to be 'given-up' to clearing participants. The instructions will relate to a specific trade. *There are two versions of this procedure. One version uses internal ID's for Accounts and CP's and the other version uses external ID's. The external ID flavour will suit callers who are not tightly coupled with the MCM databases.
5	Account Details Procedure	A stored procedure is provided that allows an external system to supply instructions to create new accounts, or edit and delete existing accounts.
6	Delete Allocations Procedure	A stored procedure is provided that allows an external system to supply instructions to delete previous allocations.
7	Reject Take-up Procedure	A stored procedure is provided that allows an external system to supply instructions to reject a take-up.

When a stored procedure is executed, parameters passed to the stored procedure are subject to initial validation (termed 'phase 1 validation'). If any errors are detected, the detected error is returned. This error can be returned as a string or a record set (refer to section three for more details). This behaviour is controlled by a parameter of the stored procedure.

Importantly, further and subsequent validations (termed 'phase 2 validation') are performed and details of any errors detected during phase 2 validation are contained in a row written to a new table (called "EIFErrorMsg") held on the MCM database (refer to Section three for further details).

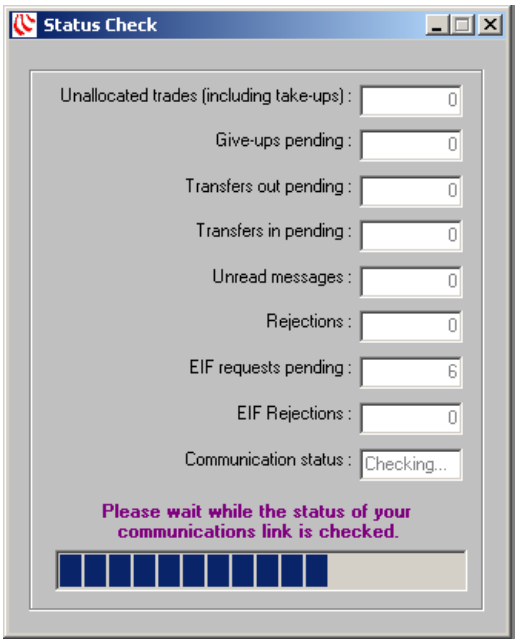
If no errors are detected in the parameters, then the instruction is automatically processed. The local MCM database is updated with the instruction upon successful execution of the stored procedure.

Successful processing of an instruction causes the MCM database to be updated and an appropriate DCS message to be sent to ASXCL.

Unsuccessful processing of an instruction results in an error message written to EIFErrorMsg table (refer to section three for further details).

All EIF tables (refer to Appendix 1 – Data Model for EIF Tables) contain current business day information.

The MCM application status check allows the participant to monitor the EIF Requests pending (i.e. number of EIF instructions still awaiting completion) as well as the EIF Rejections (i.e. number of EIF error messages received during processing of EIF Instructions).



Note:
Under normal circumstances, instructions are processed immediately. However, occasionally there may be a delay in processing. For example, the communication between the MCM and the ASXCL may be down and therefore a trade may have not been received. In this situation, the instruction is processed when communications are restored (and the trade is available to MCM).

The next section details the seven procedures (and the five stored procedures) in greater detail.

2. EIF Procedures

External systems interfacing with the MCM using an EIF can use the following seven procedures to pass instructions via DCS to ASXCL. This includes:

- Exercise Quantity Procedure
- Exclude Quantity procedure
- Order Allocation procedure
- Trade Allocation procedure
- Account Details procedure
- Delete Allocations procedures
- Reject Take-up Procedure.

2.1. Exercise Quantity Procedure

The Exercise Quantity Procedure when correctly executed conveys instructions to ASXCL via DCS messaging for the exercise of open long option contracts held in a specified open position. The table below lists the parameters that are passed.

Parameter	Comment
Account Id	This is a number that uniquely identifies an account (refer to 'AccId' on the 'Acc' table in the pa_mcmc database).
Entity Id	This is a number that uniquely identifies a traded entity (refer to 'EntId' on the 'TradedEntity' table in the pa_mcmc database).
Number of Contracts to be Exercised	This is the number of open contracts to be exercised. The number specified replaces any previous exercise instruction for the specified open position.
Reference	A reference value associated with this instruction and allocated by the external system. It is recommended that this value is unique for the day.

Stored Procedure name is: **pa_sp_EIF_ExerciseQuantity**

Parameter	SQL Server Type
@pAccountId	Integer
@pEntityId	Integer
@pNumberofExerciseContracts	Integer
@pReference	Varchar (10)
@pErrorInRecordSetFormat	Varchar(01) (optional) **Notes** -'Y' or 'N'. If not provided then Defaults to 'N'.

When the stored procedure is executed, validation is performed and any error detected results in a return error indicating the reason. The table below lists examples of error reason.

Reason
Shutdown required, flag is set to Yes. Database cannot be accessed at this time. Try again later.
Reference cannot be blank.
Account ID cannot be less than or equal to zero.
Entity ID cannot be less than or equal to zero.
The 'Account Id' does not exist on the Acc table.
The 'Entity Id' does not exist on the TradedEntity table or is not for an option series.
The selected Entity ID is not an option series that can be exercised.

Reason
The selected Entity ID has already expired.
The Entity does not expire today. No Exercise will take place for this Entity.
The Number of contracts to be Exercised is not a valid number or is less than zero.
The Number of contracts to be Exercised is not a valid number accepted by SQL Server.

Successful processing of the exercise instruction causes the MCM database to be updated and an appropriate DCS message sent to the ASXCL.

Unsuccessful processing of the exercise instruction results in an error message written to the EIFErrorMsg table (refer to the following section for more details).

2.2. Exclude Quantity Procedure

The Exclude Quantity Procedure when correctly executed conveys instructions to ASXCL via DCS messaging for open long option contracts held in a specified open position to be excluded from the automatic exercise process. The table below lists the parameters that are passed.

Parameter	Comment
Account Id	This is a number that uniquely identifies an account (refer to 'AccId' on the 'Acc' table in the pa_mcmc database).
Entity Id	This is a number that uniquely identifies a traded entity (refer to 'EntId' on the 'TradedEntity' table in the pa_mcmc database).
Number of Contracts to be Excluded (from the automatic exercise process)	This is the number of open contracts to be excluded from the automatic exercise process. The number specified replaces any previous value for the specified open position passed in previous calls of the stored procedure.
Reference	A reference value associated with this instruction and allocated by the external system. It is recommended that this value is unique for the day.

Stored Procedure name is: **pa_sp_EIF_ExcludeQuantity**

Parameter	SQL Server Type
@pAccountId	Integer
@pEntityId	Integer
@pNumberofExcludeContracts	Integer
@pReference	Varchar (10)
@pErrorInRecordSetFormat	Varchar(01) (optional) **Notes** -'Y' or 'N'. If not provided then Defaults to 'N'.

When the stored procedure is executed, validation is performed, and any error detected results in a return error indicating the cause. The table below lists examples of error reasons.

Reason
Shutdown required, flag is set to Yes. Database cannot be accessed at this time. Try again later.
Reference cannot be blank.
Account ID cannot be less than or equal to zero.
Entity ID cannot be less than or equal to zero.
The 'Account Id' does not exist on the Acc table.
The 'Entity Id' does not exist on the TradedEntity table or is not for an option series.
The selected Entity ID is not an option series that can be exercised.
The selected Entity ID has already expired.

Reason
The Entity does not expire today. No Exclusion will take place for this Entity.
The Number of contracts to be Excluded is not a valid number or is less than zero.
The Number of contracts to be Excluded is not a valid number accepted by SQL Server.

Successful processing of the exercise exclusion instruction causes the MCM database to be updated and an appropriate DCS message to be sent to ASXCL.

Unsuccessful processing of the exercise exclusion instruction results in an error message written to EIFErrorMsg table (refer to the following section for more details).

2.3. Order Allocation Procedure

Two stored procedures are executed to convey instructions to ASXCL via DCS messaging for the allocation or give-up of contracts traded during the current day for an order. These procedures are:

- pa_sp_EIF_OrderAllocation
- pa_sp_EIF_OrderEntities.

Additionally, external systems are able to request that contracts be 'price averaged' prior to allocation or give-up.

The stored procedures should only be executed when an order has been fully executed or has been partly filled and trading has ceased for the current clearing cycle. As soon as either condition has been satisfied (and not before), the first of the two stored procedures should be executed for the order. It should be noted that for one order, this stored procedure may be called multiple times if the order is to be allocated to multiple accounts. In other words, it should be called once for each account allocation. The table below lists the parameters passed.

Parameter	Comment
Allocation Type	"G" indicates a give-up; else "A".
Account Id	If 'Allocation Type' is "A" then the account ID is the number that uniquely identifies an account (refer to 'AccId' on the 'Acc' table in the pa_mcmc database); or else it is zero.
Clearing Participant	If 'Allocation Type' is "G", then the Clearing Participant is the value that is used by ASX to identify the Clearing Participant to whom the order is to be given-up; or else it is "000".
Number of Order Units to be Allocated.	This is the number of 'order units' to be allocated to the account or given-up to the Clearing Participant (as determined by 'Allocation Type'). Note that one order unit for a single-legged order equates to one contract, however for a combination order it represents the number of contracts for each leg after taking into account the 'relativity' value supplied to the trading system.
Commission	If 'Allocation Type' is "G", then the Commission is the commission to be charged to the Clearing Participant.
Commission Basis	If 'Allocation Type' is "G" then either 'P', 'R' or 'A' (to indicate that the Commission is expressed as a percentage, a rate or an amount respectively); or else it is " ".
Reference	A reference that uniquely identifies the particular order and is the value that was submitted to CLICK (in the Client field) when the order was submitted. It is recommended that this field is prefixed with '#' to prevent inadvertent automatic allocations by the ASXCL.
Allocation Reference	A string set to a value determined by the external system.

Stored Procedure name is: **pa_sp_EIF_OrderAllocation**.

Parameter	SQL Server Type
@pOrderReference	Varchar (10)
@pAllocType	Varchar(01)
@pAccountID	Integer
@pClearingParticipantID	Integer
@pOrderUnits	Integer
@pCommBasis	Integer
@pCommBasisVal	Money
@pAllocReference	Varchar(10)
@pErrorInRecordSetFormat	Varchar(01) (optional) Notes -'Y' or 'N'. If not provided then Defaults to 'N'.

When the stored procedure is executed, validation is performed, and any error detected results in a return error as listed in the table below.

Reason
Shutdown required, flag is set to Yes. Database cannot be accessed at this time. Try again later.
Order reference cannot be blank.
Order Reference is not unique.
Allocation Type is not A or G.
Allocation type is A and Account ID is not a numeric value greater than zero.
Allocation type is G and Clearing Participant ID has not been supplied.
Allocation type is A and CommBasisVal is not blank or NULL.
Allocation type is G and CommBasisVal is not a numeric value or is less than zero.
Allocation type is G and CommBasisVal Basis is not "P" or "A" or "R".
The Number of contracts to be allocated is not greater than 0 and less than 100000.
The Account ID does not exist on the Acc table.
Allocation type is G and Clearing Participant is not a valid code used by ASX.

After the first stored procedure has been executed once for each allocation to an account or once for each give-up to a Clearing Participant, then the second stored procedure should be executed. It should be executed once only for the particular order. The table below lists the parameters that should be passed.

Number of Legs	Number of Legs for the Order
Order Units	The number of 'order units' filled. Note that one order unit for a single-legged order equates to one contract, however for a combination order it represents contracts for each leg after taking into account the 'relativity' value supplied to the trading system.
Price Average	Set to one if price averaging of contracts is required, or else is set to zero. Note that if there are multiple allocations, then price averaging is always performed.
Relativity – Leg 1	If 'Number of Legs' is one, then the number one; or else the 'relativity' value supplied to the trading system for the first leg.
Entity Id – Leg 1	This is a number that uniquely identifies the traded entity (refer to 'EntId' on the 'TradedEntity' table in the pa_mcmc

Number of Legs	Number of Legs for the Order
	database) of the first leg. For an equity leg (e.g. resulting from a buy/write combination) then set to zero.
Relativity – Leg 2	If 'Number of Legs' is greater than one, then the 'relativity' value supplied to the trading system for the second leg ;or else zero.
Entity Id – Leg 2	If 'Number of Legs' is one, then set to zero; if this is an equity leg then set to zero; or else the number that uniquely identifies the traded entity (refer to 'EntId' on the 'TradedEntity' table in the pa_mcmc database) of the second leg.
Relativity – Leg 3	If 'Number of Legs' is greater than two, then the 'relativity' value supplied to the trading system for the third leg ;or else 0.
Entity Id – Leg 3	If 'Number of Legs' is less than three then set to zero; if this is an equity leg then set to zero; or else the number that uniquely identifies the traded entity (refer to 'EntId' on the 'TradedEntity' table in the pa_mcmc database) of the third leg.
Relativity – Leg 4	If 'Number of Legs' is four, then the 'relativity' value supplied to the trading system for the fourth leg ;or else 0.
Entity Id – Leg 4	If 'Number of Legs' is less than four, then set to zero; if this is an equity leg then set to zero; or else the number that uniquely identifies the traded entity (refer to 'EntId' on the 'TradedEntity' table in the pa_mcmc database) of the fourth leg.
Reference	A reference that uniquely identifies the particular order and is the value that was submitted to CLICK (in the Client field) when the order was submitted.

Stored Procedure name is: **pa_sp_EIF_OrderEntities**.

Parameter	SQL Server Type
@pOrderReference	Varchar (10)
@pOrderUnits	Integer
@pNumberofLegs	Integer
@pIsPriceAverage	Varchar(01)
@pRelativity_1	Integer
@pEntityID_1	Integer
@pRelativity_2	Integer (optional)
@pEntityID_2	Integer (optional)
@pRelativity_3	Integer (optional)
@pEntityID_3	Integer (optional)
@pRelativity_4	Integer (optional)
@pEntityID_4	Integer (optional)
@pErrorInRecordSetFormat	Varchar(01) (optional) Notes -'Y' or 'N'. If not provided then Defaults to 'N'.

When the stored procedure is executed, validation is performed and any error detected results in a return error indicating the reason. Examples of error causes are listed in the table below.

Reason
Shutdown required, flag is set to Yes. Database cannot be accessed at this time. Try again later.
Order reference cannot be blank.
"Is Price Average" flag is blank or invalid.
Order Allocation has not been created for this order reference.
Order Reference is not unique.
The Number of Legs is not a number or is outside the range of one to four.
The Number of contracts to be allocated is not greater than 0 and less than 100000.
The Entity ID for leg one does not exist, has expired, or could not be used to locate an entity on the table TradedEntity.
The Relativity value for leg one is not a valid numeric value or is less than or equal to zero.
The Entity ID for leg two does not exist, has expired, or could not be used to locate an entity on the table TradedEntity.
The Relativity value for leg two is not a valid numeric value or is less than or equal to zero.
The Entity ID for leg three does not exist, has expired, or could not be used to locate an entity on the table TradedEntity.
The Relativity value for leg three is not a valid numeric value or is less than or equal to zero.
The Entity ID for leg four does not exist, has expired, or could not be used to locate an entity on table TradedEntity.
The Relativity value for leg four is not a valid numeric value or is less than or equal to zero.
The Entity ID selected has to be unique for each leg.
The Order units do not equal the sum of order units passed to the first stored procedure for the same order reference.

When EIF detects that the correct number of fills for an order (identified by the 'Reference' contained in the trade details) have been received from the central DCS server, an MCM process begins that causes allocation, give-up and price-averaging requests to be submitted to ASXCL for the particular order. This process is actioned once only. For example if allocation details are subsequently deleted manually, then the automatic allocation process is not re-activated.

Successful processing of the order allocation or give ups instructions causes the MCM database to be updated and an appropriate DCS message sent to ASXCL.

Unsuccessful processing of the order allocation or give up instructions result in an error message written to the EIFErrorMsg table (refer to the following section for further details).

2.4. Trade Allocation Procedure (using internal Account and CP ID's)

NOTE: Use the alternate version of this EIF procedure described in the next section if you do not have access to the internal ID's used by MCM for account and CP.

A stored procedure is executed to convey instructions to ASXCL via DCS messaging for the allocation or give-up of contracts for a single trade.

The stored procedure may be executed multiple times for a single trade. The table below lists parameters passed to the store procedure.

Parameter	Comment
Trade Id	The number that DCS uses to uniquely identify a trade (refer to 'TrId' on the 'Trade' table in the pa_mcmc database).
Allocation Type	"G" indicates a give-up; else "A"

Parameter	Comment
Account Id	If 'Allocation Type' is "A" then the number that uniquely identifies an account (refer to 'AccId' on the 'Acc' table in the pa_mcmc database); or else zero.
Clearing Participant	If 'Allocation Type' is "G" then the value that is used by ASX to identify the Clearing Participant to whom the order to is to be given-up; or else "000".
Number of contracts to be Allocated	This is the number of contracts to be allocated to the account or given-up to the Clearing Participant (as determined by 'Allocation Type').
Commission	If 'Allocation Type' is "G" the commission to be charged to the Clearing Participant
Commission Basis	If 'Allocation Type' is "G" then either 'P', 'R' or 'A' (to indicate that the Commission is expressed as a percentage, a rate or an amount); else " ".
Allocation Reference	A string set to optional reference to be associated with this allocation.
Reference	A reference value associated with this instruction and allocated by the external system. It is recommended that this value is unique for the day.

Stored Procedure name: **pa_sp_EIF_TradeAllocation**.

Parameter	SQL Server Type
@pTrID	Integer
@pAllocType	Varchar(01)
@pAccountID	Integer
@pClearingParticipantID	Integer
@pQty	Integer
@pCommBasis	Varchar(01)
@pCommBasisVal	Money
@pAllocRef	Varchar(10)
@pReference	Varchar(10)
@pErrorInRecordSetFormat	Varchar(01) (optional) *Notes* -'Y' or 'N'. If not provided then Defaults to 'N'.

When the stored procedure is executed, validation is performed and any error detected results in a return error indicating the reason. The table below lists error reasons.

Reason
Shutdown required, flag is set to Yes. Database cannot be accessed at this time. Try again later.
Reference cannot be blank.
Trade ID is not a numeric value greater than zero.
Allocation Type is not A or G.
Allocation type is A and Account ID is not a numeric value greater than zero.
Allocation type is G and Clearing Participant ID has not been supplied.
Allocation type is A and CommBasisVal is not blank or NULL.
Allocation type is G and CommBasisVal is not a numeric value or is less than zero.
Allocation type is G and CommBasisVal Basis is not "P" or "A" or "R".
The Number of contracts to be allocated is not greater than zero, and less than 100000.

The Account ID does not exist on the Acc table.
Allocation type is G and Clearing Participant is not a valid code used by ASX.

When EIF detects that the correct number of fills for an order (identified by the 'Reference' contained in the trade details) have been received from the central DCS server, an MCM process begins that causes allocation, give-up and price-averaging requests to be submitted to ASXCL for the particular order. This process is actioned once only. For example if allocation details are subsequently deleted manually, then the automatic allocation process is not re-activated.

Successful processing of the order allocation or give-up instructions cause the MCM database to be updated and an appropriate DCS message to be sent to ASXCL.

Unsuccessful processing of the trade allocation or give-up instructions result in an error message written to the EIFErrorMsg table (refer to the following section for further details).

2.5. Trade Allocation Procedure (using external Account and CP ID's)

A stored procedure is executed to convey instructions to ASXCL via DCS messaging for the allocation or give-up of contracts for a single trade.

The stored procedure may be executed multiple times for a single trade. The table below lists parameters passed to the store procedure.

Parameter	Comment
Trade Id	The number that DCS uses to uniquely identify a trade (refer to 'TrId' on the 'Trade' table in the pa_mcmc database).
Allocation Type	"G" indicates a give-up; else "A"
AccountCode	If 'Allocation Type' is "A" then the number that uniquely identifies an account or else zero.
Clearing ParticipantCode	If 'Allocation Type' is "G" then the value that is used by ASX to identify the Clearing Participant to whom the order to is to be given-up; or else "000".
Number of contracts to be Allocated	This is the number of contracts to be allocated to the account or given-up to the Clearing Participant (as determined by 'Allocation Type').
Commission	If 'Allocation Type' is "G" the commission to be charged to the Clearing Participant
Commission Basis	If 'Allocation Type' is "G" then either 'P', 'R' or 'A' (to indicate that the Commission is expressed as a percentage, a rate or an amount); else " ".
Allocation Reference	A string set to optional reference to be associated with this allocation.
Reference	A reference value associated with this instruction and allocated by the external system. It is recommended that this value is unique for the day.

Stored Procedure name: **pa_sp_EIF_TradeAllocation2**.

Parameter	SQL Server Type
@pTrID	Integer
@pAllocType	Varchar(01)
@pAccountCode	Varchar(10)
@pClearingParticipantICode	Varchar(4)
@pQty	Integer

Parameter	SQL Server Type
@pCommBasis	Varchar(01)
@pCommBasisVal	Money
@pAllocRef	Varchar(10)
@pReference	Varchar(10)
@pErrorInRecordSetFormat	Varchar(01) (optional) *Notes* -'Y' or 'N'. If not provided then Defaults to 'N'.

When the stored procedure is executed, validation is performed and any error detected results in a return error indicating the reason. The table below lists error reasons.

Reason
Shutdown required, flag is set to Yes. Database cannot be accessed at this time. Try again later.
Reference cannot be blank.
Trade ID is not a numeric value greater than zero.
Allocation Type is not A or G.
Allocation type is A and Account ID is not a numeric value greater than zero.
Allocation type is G and Clearing Participant ID has not been supplied.
Allocation type is A and CommBasisVal is not blank or NULL.
Allocation type is G and CommBasisVal is not a numeric value or is less than zero.
Allocation type is G and CommBasisVal Basis is not "P" or "A" or "R".
The Number of contracts to be allocated is not greater than zero, and less than 100000.
The Account Code does not exist on the Acc table.
Allocation type is G and Clearing Participant is not a valid code used by ASX.

When EIF detects that the correct number of fills for an order (identified by the 'Reference' contained in the trade details) have been received from the central DCS server, an MCM process begins that causes allocation, give-up and price-averaging requests to be submitted to ASXCL for the particular order. This process is actioned once only. For example if allocation details are subsequently deleted manually, then the automatic allocation process is not re-activated.

Successful processing of the order allocation or give-up instructions cause the MCM database to be updated and an appropriate DCS message to be sent to ASXCL.

Unsuccessful processing of the trade allocation or give-up instructions result in an error message written to the EIFErrorMsg table (refer to the following section for further details).

2.6. Account Details Procedure

A stored procedure is provided so that when correctly executed, conveys instructions to ASXCL via DCS messaging to create new accounts, edit accounts or delete existing accounts. The table below lists the parameters that are passed.

Parameter	Comment
Amendment Type	Valid values are: "N" for New "E" for Edit "D" for Delete.
Account Id	If 'Amendment Type' is "E" or "D" then the unique ID of the account to be edited or deleted; otherwise zero.
Segregation type	"S" for Segregated; "U" for Unsegregated. Not used if 'Amendment Type' is "D".

Parameter	Comment
Auto Match Out	A code indicating whether or not long and short positions for the same Traded Entity should be automatically liquidated to the fullest extent possible. "N" = No "Y" = Yes. Not used if 'Amendment Type' is "D".
Auto Exercise	"Y" or "N" indicates whether or not exercise notices can be automatically generated for the account. Not used if 'Amendment Type' is "D".
Account Type	The account type. Valid values are: "H" = House account "P" = Private account "I" = Institution. Not used if 'Amendment Type' is "D".
Acc Status	Account Status flag "A" = Active "I" = Inactive "S" = Suspended. Not used if 'Amendment Type' is "D".
Account Code	A unique code that is commonly used to identify the account. Not used if 'Amendment Type' is "D".
Account Name	Name of the account. Not used if 'Amendment Type' is "D".
Specific Cover	Set to "Y" if scrip collateral lodged is to be treated as specific cover, or else "N". Not used if 'Amendment Type' is "D".
Member Info	Free-format information relating to the account. Not used if 'Amendment Type' is "D".
Address Line 1	Account address – Line one. Not used if Amendment Type is "D".
Address Line 2	Account address – Line two. Not used if Amendment Type is "D".
Address Line 3	Account address – Line three Not used if Amendment Type is "D".
Address Line 4	Account address – Line four Not used if Amendment Type is "D".
Account Name Confirm	Used to indicate whether or not confirmation of an account name change is expected by ASXCL. Valid values are: "Y" = Yes "N" = No Not used if 'Amendment Type' is "N" or "D".
Reason	If 'Account Name' is changed, reason for the Account Name Change.

Entries that are not used when the Amendment Type is "D" are ignored. Null values can be passed in this situation.

Stored Procedure name is: **pa_sp_EIF_Account.**

Parameter	SQL Server Type
@pAmendmentType	Varchar(01)
@pAccId	Integer
@pSegType	Varchar(01)

Parameter	SQL Server Type
@pAutoMatchout	Varchar(01)
@pAutoExer	Varchar(01)
@pAccType	Varchar(01)
@pAccStat	Varchar(01)
@pAcc	Varchar(10)
@pAccName	Varchar(250)
@pSpecificCover	Varchar(01)
@pMbrInfo	Varchar(50)
@pAddress1	Varchar(50)
@pAddress2	Varchar(50)
@pAddress 3	Varchar(50)
@pAddress4	Varchar(50)
@pAccNameConfirm	Varchar(01)
@pReason	Varchar(250)

When the stored procedure is executed, validation is performed and any error detected results in a return error as seen in the table below.

Reason
Shutdown required, flag is set to Yes. Database cannot be accessed at this time. Try again later.
Invalid Amendment Type.
Account ID cannot be less than zero.
The 'Account Id' does not exist on the Acc table.
The 'Account Id' not zero for new Account.
Invalid `Account Id` of zero.
Invalid Segregation Type
Auto Match Out not Y or N
Auto Exercise not Y or N
Invalid Account type
Invalid Account Status
Account Code not unique
Specific Cover not Y or N
Account Name Confirm not Y or N
No Reason for Account Name Change supplied

Successful processing of the account instruction causes the MCM database to be updated and an appropriate DCS message to be sent to ASXCL.

Unsuccessful processing of the account instruction results in an error message written to EIFErrorMsg table (refer to the following section for further details).

2.7. Delete Allocations Procedure

A stored procedure is executed to convey instructions to ASXCL via DCS messaging for the deletion of an allocation.

The parameters passed to the store procedure are listed in the table below.

Parameter	Comment
Trade Id	The number that DCS uses to uniquely identify a trade (refer to 'TrId' on the 'Trade' table in the pa_mcmc database).
Allocation Sequence	The Allocation Sequence number of the allocation to be deleted.
Reference	A reference value associated with this instruction and allocated by the external system. It is recommended that this value is unique for the day.

Stored Procedure name is: **pa_sp_EIF_DeleteAllocation**.

Parameter	SQL Server Type
@pTrID	Integer
@pAllocSeq	Integer
@pReference	Varchar(10)
@pErrorInRecordSetFormat	Varchar(01) (optional) Notes -'Y' or 'N'. If not provided then Defaults to 'N'.

When the stored procedure is executed, validation is performed and any error detected results in a return error indicating the reason. Examples of error reasons are listed in the table below.

Reason
Shutdown required, flag is set to Yes. Database cannot be accessed at this time. Please try again later.
Alloc Seq is not a numeric value greater than zero.
Reference cannot be blank.
Trade ID is not a numeric value greater than zero.
Allocation to delete does not exist.

Successful processing of the order allocation or give-up instructions cause the MCM database to be updated and an appropriate DCS message to be sent to ASXCL.

Unsuccessful processing of the trade allocation or give-up instructions result in an error message written to EIFErrorMsg table (refer to the following section for further details).

2.8. Reject Take-up Procedure

A stored procedure is executed to convey instructions to ASXCL via DCS messaging for the rejection of a take-up.

The parameters passed to the store procedure are listed in the table below.

Parameter	Comment
Trade Id	The number that DCS uses to uniquely identify a trade (refer to 'TrId' on the 'Trade' table in the pa_mcmc database).
Reject Reason	The reason for the rejection of the take-up.
Reference	A reference value associated with this instruction and allocated by the external system. It is recommended that this value is unique for the day.

Stored Procedure name is: **pa_sp_EIF_RejectTakeup**.

Parameter	SQL Server Type
@pTrID	Integer
@pRejectReason	Varchar(255)

Parameter	SQL Server Type
@pReference	Varchar(50)
@pErrorInRecordSetFormat	Varchar(01) (optional) Notes -'Y' or 'N'. If not provided then Defaults to 'N'.

When the stored procedure is executed, validation is performed and any error detected results in a return error indicating the reason. Examples of error reasons are listed in the table below.

Reason
Shutdown required, flag is set to Yes. Database cannot be accessed at this time. Try again later.
Reference cannot be blank.
Reject Reason cannot be blank.
Trade ID is not a numeric value greater than zero.

Successful processing of the order allocation or give-up instructions cause the MCM database to be updated and an appropriate DCS message to be sent to ASXCL.

Unsuccessful processing of the trade allocation or give-up instructions result in an error message written to the EIFErrorMsg table (refer to the following section for further details).

3. EIF Retrieval Procedures

External systems interfacing with the MCM using an EIF can use the following two procedures to retrieve selected information from MCM. This includes:

- Trade and MCM-initiated Allocation Retrieval Procedure
- Traded Entity Retrieval procedure

3.1. Trade and MCM-initiated Allocation Retrieval Procedure

The Trade and MCM-initiated Allocation Retrieval Procedure can be used to retrieve all trades and MCM-initiated Allocations. Data can be retrieved for the current business day only.This informations includes:

- New Trades
- Take-up Trades
- Trades resulting from a price average
- Trade deletions
- Allocations entered in to MCM including give-ups
- Give-up advice – acceptance or rejection (from other party)
- Take-up advice – acceptance or rejection (MCM-initiated)

Parameter	Comment
Transaction Id	The transaction id of the last transaction received. At the start of a new day, this should be set to 0.

Stored Procedure name is: **pa_sp_EIF_RetrieveTrades**

Parameter	SQL Server Type
@pTransactionId	Integer

The stored procedure will return the following information for each transaction

Column	Comment	SQL Server Type
Transaction Id	Transaction Id	Integer
TransactionType	Transaction types: TR –Trade TD – Trade Deletion AL – Allocation AD – Allocation Deletion GA – Give-up Advice TA – Take-up Advice	Char(2)
TradeId	TR, TD, AL, AD, GA, TA: A number that uniquely identifies the trade.	Integer
Origin	TR: Origin of the trade: A – Manually entered trade G – Take-up trade (resulting from give-up) P – Price average trade i.e. trade replacing a number of trades that were price averaged T – Trade from ASX Trade	Char(1)
EntityId	TR: Indicates the traded entity to which this trade refers.	Integer

ExecutedBy	TR: Executing Participant (CP or Registered Trader)	Varchar(4)
OtherCP	TR: Other party to the trade. For take-ups, the CP giving the trade up. AL: For give-ups, the CP to whom the trade is given up	Varchar(4)
ExchangeRef	TR:ASX Trade Sales Slip Number	Varchar(10)
BuySell	TR: Indicates whether it is a Buy (B) or Sell (S) trade	Char(1)
TradedPrice	TR: Traded price of the trade (premium for options)	Money
Quantity	TR: Quantity (lots) Traded AL: Quantity (lots) allocated	Integer
PriceAverageId	TR; origin: P: Price average trade Id, which will correspond with the price average id on: AL: for price averaged trades, this will contain the price average Id	Integer
UnitContractValue	Unit contract value of the trade	Money
Reference	TR: Customer reference field (customer_info_s) entered with order in ASX Trade	Varchar(40)
ConditionCodes	TR: Condition codes associated with the trade.	Varchar(16)
Trader	TR: The ASX Trade trader	Varchar(10)
OrderNumber	TR: The ASX Trade order number	Varchar(17)
AllocationSequence	AL, GA: The allocation sequence number for the allocation.	Integer
Account	TR, AL: Account code to with the trade was allocated. Empty on TR rows if the trade was not automatically allocated, and on AL rows for give-ups and price averaged trades	Varchar(10)
AllocationRef	TR: Client fiels (ex_client_s) entered with order in ASX Trade AL: Account refence entered with allocation in MCM	Varchar(50)
Commission	AL (Give-ups only) and TR, origin: G: Total commission for the give-up/take-up.	Money
CommissionBasis	AL (Give-ups only) and TR, origin: Commission basis: A – Amount P – Percentage R – Rate	Char(1)
CommissionBasisValue	AL (Give-ups only) and TR, origin: Value assoc	Money
Taken	GA, TA: Y for give-up/take-up acceptance N – for give-up/take-up rejection	Char(1)

The following table indicates which columns may contain data for each transaction type

Column	Trade (TR)	Trade Deletion (TD)	Allocation (AL)	Allocation Deletion (AD)	Give-up Advice (GA)	Take-up Advice (TA)
TransactionId	•	•	•	•	•	•

TransactionType	•	•	•	•	•	•
TradeId	•	•	•	•	•	•
Origin	•					
EntityId	•					
ExecutedBy	•					
OtherCP	•		•			
ExchangeRef	•					
BuySell	•					
TradedPrice	•					
Quantity	•					
PriceAverageId	•					
UnitContractValue	•					
Reference	•					
ConditionCodes	•					
Trader	•					
OrderNumber	•					
AllocationSequence	•		•	•	•	
Account	•		•			
AllocationRef	•		•			
Commission			•			
CommissionBasis			•			
CommissionBasisValue			•			
Taken					•	•

3.2. Traded Entity Retrieval Procedure

The Traded Entity Retrieval Procedure can be used to retrieve the details of new traded entities created on that business day (usually OTCs) or details of an existing entity or entities.Three optional parameters are provided of which one and only one must be entered

Parameter	Comment	Usage
Transaction Id	The transaction id of the last transaction received. At the start of a new day, this should be set to 0.	To request details of the next set of new entities created today
Entity Id	The entity id of the particular entity required	To request details of a specific traded entity
ASX Code	The ASX Code required. This may contain standard SQL Server wild cards (e.g. %, _)	To request details of of a specific traded entity or entities

Stored Procedure name is: **pa_sp_EIF_RetrieveTradedEntity**

Parameter	SQL Server Type
@pTransactionId	Integer
@pEntityId	Integer
@pASXCode	Varchar(50)

The stored procedure will return all updates

Column	Comment	SQL Server Type
Transaction Id	Transaction Id – will non-zero for request using @pTransactionId only	Integer
EntityId	This is a number that uniquely identifies a traded entity.	Integer
DerivativeProduct	The derivative product e.g. BHP, BHPL, XJO, XPJF	Varchar(6)
ASXCode	ASX Code	Varchar(8)
DeliveryMonth	Delivery month. Format MMMYY for ETOs, DDMMMYY for OTCs	Varchar(7)
OptionType	Option Type C – Call, P- Put	Char(1)
Strike	Strike price for options 0 for futures	Char(1)
ExpiryDate	The expiry/maturity date	Date
ExerciseStyle	Exercise style for options A - American, E - European	Char(1)
Underlying	Undelying product e.g. BHP, XJO, XPJ	Varchar(6)
DerivativeProductType	Derivative product type: FU – Future LE – LEPO OF - Option on future OI – Option on index OS – Option on Stock (Equity option)	Char(2)
ContractSize	Contract size (units per lot)	Money
Multiplier	Multiplier used to calculate contract value	Money
CashSettDate	The cash settlement date for cash settled entities, otherwise NULL	Date
DeliveryStartDate	Delivery start date for deliverable futures	Date
SpecialProcess	Reserved for future use	Varchar(10)
IsOTC	Y – OTC, N – Exchange traded	Char(1)

4. EIFErrorMsg Table

As mentioned previously, a new table is added to the MCM database. This table holds details of errors detected during the second-phase validation for the current business date. The table contains the following columns, as listed in the table below.

Code	Reason
RecId	A number that uniquely identifies a row within the table.
InstructionType	A value that identifies the instruction type, e.g. 'Exercise', 'Exclude', 'Order Allocation', 'Order Allocation (Entities)' or 'Trade Allocation'.
ErrorCode	A value that can be associated with a reason for the error.
ErrorDescription	A short narrative indicating the likely cause of the error.
Reference	This is the unique value that the external system passed to the stored procedure to identify the instruction.

To illustrate the concept, the following possible entries into the EIFErrorMsg table are listed in the table below.

Id	Instruction Type	Error Code	Error Description	Reference
1	Exercise	101	Insufficient quantity open or position not found.	ABC123
2	Exclude	102	Insufficient quantity open or position not found.	DEF456
3	Trade Allocation	103	Insufficient unallocated quantity or trade not found	GHI789
4	Order Allocation	104	Insufficient fills found for the specified order reference.	XYZ321

As seen in the illustration above, the user can view all the instructions that are rejected by EIF – processing in the EIFErrorMsg table.

The Reference field along with the InstructionType field on EIFErrorMsg table assists the participant to uniquely identify the rejected instruction. Further action can be taken to rectify the problem by resubmitting the transaction to MCM.

The following table indicates the instruction status present on each of the relevant control tables.

Instruction Status	Description
N	Instruction Not processed
C	Instruction processed successfully
E	Instruction processing failed. Error description will be present in EIFErrorMsg table for the appropriate reference.

For more information on the EIF table's schema, refer to *Appendix 1 – Data Model for EIF Tables.*

5. Retrieving Error Messages from Stored Procedures

The stored procedures can now return the error number and messages using two methods. This includes:

- SQL Native Error and Description (default method)
- RecordSet – ErrorNumber, ErrorDescription.

Each stored RecordSet procedure has a parameter that allows the user to switch between the below specified methods. Each method provides different ways of accessing the error number and description. Participants can select the most convenient method.

The participants have to make sure the (@pErrorInRecordSetFormat)) in each stored procedure is set to:

- 'Y' - for error to be retrieved in record set
- 'N' - for the SQL Native Error and Description.

5.1. SQL Native Error and Description (Default Method)

The stored procedure returns a SQL Native Error with an Error Description. This only returns one error at a time.

5.2. RecordSet – ErrorNumber, ErrorDescription

The stored procedure returns a record set with the first data being the column ErrorNumber and the second data being the column ErrorDescription (Error Description). This only returns one error at a time.

	ErrorNumber	ErrorDescription
On Success	0	''
On Any Failure	50000	Error message according to the stored procedure error list.

6. Appendix 1 – Data Model for EIF Tables

EIFErrorMsg – (EIF Only) The EIFErrorMsg table contains details of errors encountered while processing EIF instructions.

Index Name	Property		Number of Fields
Primary Key	Unique: Fields:	True Unique number used to identify this record.	1

Column Name	Description	Type	Size
RecId	Unique number used to identify this record.	Number (Long)	4
InstructionType	The identifier for the type of EIF instruction.	Text	20
ErrorCode	Unique EIF error code for the instruction.	Number(Long)	4
Errordescription	Description of the EIF error.	Text	1000
Reference	Unique Reference for the EIF instruction.	Text	50

EIFExclInstruct – (EIF Only) The EIFExclInstruct table contains details of the instructions submitted via EIF to exclude open long option contracts from the automatic exercise process.

Index Name	Property		Number of Fields
Primary Key	Unique: Fields:	True AccID, Ascending EntID,	2

Column Name	Description	Type	Size
AccID	Unique number used to identify the account.	Number (Long)	4
EntID	Unique number used to identify the entity.	Number(Long)	4
Qty	Number of contracts to be excluded from Exercise.	Number(Long)	4
Reference	A unique reference for this instruction	Text	50
Status	Status of this transaction. 'N' – Not processed 'C' – Processed successfully 'E' – Error in processing. Error found in EIFErrorMsg table	Text	1

Column Name	Description	Type	Size
MsgSeq	The sequence number of the message associated with this transaction.	Number(Long)	4
LastUpdated	Date and time of last update to the record	Date/Time	8
UpdateCount	Count of number of updates to the record	Number(Long)	4

EIFExerInstruct – The EIFExerInstruct table contains details of the instructions submitted via EIF to exercise open long option contracts.

Index Name	Property		Number of Fields
PrimaryKey	Unique: Fields:	True AccID, Ascending EntID, Ascending	2

Column Name	Description	Type	Size
AccID	Unique number used to identify the account.	Number (Long)	4
EntID	Unique number used to identify the entity.	Number (Long)	4
Qty	Number of contracts to be excluded from Exercise.	Number (Long)	4
Reference	A unique reference for this instruction.	Text	50
Status	Status of this transaction: N' – Not processed 'C' – Processed successfully 'E' – Error in processing. Error found in EIFErrorMsg table.	Text	1
MsgSeq	The sequence number of the message associated with this transaction.	Number (Long)	4
LastUpdated	Date and time of last update to the record.	Date/Time	8
UpdateCount	Count of number of updates to the record.	Number(Long)	4

EIFOrderAlloc – The EIFOrderAlloc table contains details of all Order Allocation instructions submitted via EIF.

Index Name	Property		Number of Fields
PrimaryKey	Unique: Fields:	True OrderRef, Ascending RecID, Ascending	2

Column Name	Description	Type	Size
OrderRef	Unique identifier for a particular order.	Text	10
RecID	Unique number used to identify the record.	Number(Long)	4
AllocType	Allocation Type.	Text	1
AccID	Unique number used to identify the account.	Number(Long)	4
OtherMbrID	Unique number used to identify the other member.	Number(Long)	4
Units	Order Units to be allocated to the account.	Number(Long)	4
CommBasis	Commission basis (Points, Rate or Amount).	Text	1
CommBasisVal	Commission Value.	Currency	8
AllocRef	Allocation Reference.	Text	50

EIFOrderAllocControl – The EIFOrderAllocControl table contains control details for EIF Order Allocation instructions**.**

Index Name	Property		Number of Fields
PrimaryKey	Unique: Fields:	True OrderRef, Ascending	1

Name	Description	Type	Size
OrderRef	Unique identifier for a particular order.	Text	10
IsPriceAvg	A flag to utilise price average.	Number (Long)	4
OrderUnits	Total Number of units in a particular order.	Number (Long)	4
AccID	Unique number used to identify the account	Number (Long)	4
OtherMbrID	Unique number used to identify the other member.	Number (Long)	4

Name	Description	Type	Size
Units	Order Units to be allocated.	Number (Long)	4
CommBasis	Commission basis (Points, Rate or Amount).	Text	1
CommBasisVal	Commission Value.	Currency	8
AllocRef	Allocation Reference.	Text	50
Status	Status of this transaction. 'N' – Not processed 'C' – Processed successfully 'E' – Error in processing. Error found in EIFErrorMsg table.	Text	1
MsgSeq	The sequence number of the message associated with this transaction.	Number (Long)	4
LastUpdated	Date and time of last update to the record	Date/Time	8
UpdateCount	Count of number of updates to the record	Number (Long)	4

EIFOrderAllocRelativity – The EIFOrderAllocRelativity table contains details of the entities and their relativities relating to EIF Order Allocation instructions.

Index Name	Property		Number of Fields
PrimaryKey	Unique: Fields:	True OrderRef, Ascending EntID, Ascending	2

Column Name	Description	Type	Size
OrderRef	Unique identifier for a particular order.	Text	10
EntID	Unique number used to identify the entity.	Number (Long)	4
Relativity	Number to specify the relativity for the entity.	Number (Long)	4

EIFPriceAvgControl – EIFPPriceAvgControl table contains control details of the price averages generated as a result of EIF instructions.

Index Name	Property		Number of Fields
PrimaryKey	Unique: Fields:	True OrderRef, Ascending PriceAvgID, Ascending	2

Column Name	Description	Type	Size
OrderRef	Unique identifier for a particular order.	Text	10
PriceAvgID	Unique number used to identify the Price Average order allocation.	Number (Long)	4
Status	Status of this transaction. 'N' – Not processed 'C' – Processed successfully 'E' – Error in processing. Error found in EIFErrorMsg table	Text	1
TrID	Unique number used to identify the Trade.	Number (Long)	4
LastUpdated	Date and time of last update to the record	Date/Time	8
UpdateCount	Count of number of updates to the record	Number (Long)	4

EIFTradeAllocation – The EIFTradeAllocation table contains details of EIF Trade Allocation instructions.

Index Name	Property		Number of Fields
PrimaryKey	Unique: Fields:	True TrID, Ascending RecID, Ascending	2

Name	Description	Type	Size
TrID	Unique identifier used to identify the trade.	Number (Long)	4
RecID	Unique number used to identify the Record.	Number (Long)	4
AllocType	AllocationType	Text	1

Name	Description	Type	Size
AccID	Unique number used to identify the Account.	Number (Long)	4
Qty	Quantity of the trade to be allocated.	Number (Long)	4
CommBasis	Commission basis (Points, Rate or Amount).	Text	1
CommBasisVal	Commission Value.	Currency	8
AllocRef	Allocation Reference.	Text	50
Reference	Instruction Reference.	Text	50
IsReady	Flag to indicate if the state of the instruction.	Text	1
Status	Status of this transaction. 'N' – Not processed 'C' – Processed successfully 'E' – Error in processing. Error found in EIFErrorMsg table.	Text	1
MsgSeq	The sequence number of the message associated with this transaction.	Number (Long)	4
LastUpdated	Date and time of last update to the record.	Date/Time	8
UpdateCount	Count of number of updates to the record.	Number (Long)	4