SFC DS-OL Level 2 Validations

Annex 3

1. Purpose of Document

This document references the DS-OL Specification and provides more in-depth descriptions and examples in performing level 2 validations of DS-OL data submitted by in-scope brokers. Level 2 validations serve the purpose to ensure quality and integrity of the trade data being submitted for inspection.

This document is not intended to provide an exhaustive list of validations but only an aid for in-scope brokers to implement level 2 data validations. These rules may be added, removed or updated from time to time without prior notice.

2. Level 2 Validations

The appendix to the DS-OL Specification provides a validation guideline to in-scope brokers to perform validation of the data before submission, the guideline outlined 2 levels of validation check that can be performed. Level 1 validations are simple checks against data type and format requirements which are straightforward and need no further explanation. Level 2 validations are checks that involve comparing data fields within and/or between different events.

3. Types of Checks

The following categories represent the types of checks in Level 2 validations:

Type of Check	Descriptions	Rule Prefix
Cross Reference	Validates the linkage fields used to link up events and/or orders in the presented data	CR
Referential Integrity	Validates the fields with dictionary key data type, verifies the values are those used in the reference data dictionary	RI
Data Consistency	Validates consistency of data field values within and/or between events	DC
Event Construct	Validates if the events are constructed according to the specification	EC

3.1. Cross Reference

Summary of rules under the Cross Reference category:

Rule ID	Descriptions
CR01	All events should be linked to the corresponding Order New (ONEW) event by the logicalOrderID or LOID [1]
CR02	The logicalOrderID needs to be unique per client order
CR03	The orderID should be unique during a trading day
CR04	For aggregated orders (orders having values in aggregatedOrders, including ONEW & OMOD), the list of orders being aggregated should be present in the data with corresponding logicalOrderIDs
CR05	For allocation (AALC) events, the constituent client orders with LOIDs which are shown in the allocToLOID field should be present in the data
CR06	For multi-day (e.g. GTC/GTD) orders, the logicalOrderID should remain unchanged for events on each day until the order is expired or cancelled

Rule ID	**CR01**
Description	All events should be linked to the corresponding Order New (ONEW) event by the logicalOrderID
Related field(s) [2]	logicalOrderID from all events
Additional comment	● An event is invalid if there is no ONEW event in the data with the same logicalOrderID

Rule ID	**CR02**
Description	The logicalOrderID needs to be unique per client order
Related field(s)	logicalOrderID, actualOrderDate, eventResponseType etc. in ONEW events
Additional comment	● All client orders should be uniquely identified by the logicalOrderID. An ONEW event with response type ACK or UNS is invalid if it has the same logicalOrderID as another ONEW in the data, unless these ONEW events are presenting the same multi-day order on each trading day

[1] logicalOrderID is the actual field name in the DS-OL Specification. This field is sometimes referred to as LOID for the sake of simplicity. These terms are used interchangeably throughout this document.

[2] Implicitly related fields are not always listed, e.g. event, logicalOrderID and eventDateTime. If fields are not listed, that does not necessarily mean they are not used for validation if needed.

Rule ID	CR03
Description	The orderID should be unique during a trading day
Related field(s)	orderID, eventDateTime in ONEW event
Additional comment	● orderID assigned to an order should not be reused during a trading day. ● An ONEW event is invalid if any other ONEW events exist in the data with the same orderID on the same trading day

Rule ID	CR04
Description	For aggregated orders (orders having values in aggregatedOrders, including ONEW & OMOD), the list of orders being aggregated should be present in the data with corresponding logicalOrderIDs
Related field(s)	logicalOrderID, aggregatedOrders in ONEW/OMOD event
Additional comment	● An order’s ONEW event having valid values in the aggregatedOrders field is considered an aggregated order ● For all the logicalOrderIDs listed in the aggregatedOrders field in an ONEW event, the corresponding order’s ONEW event should be provided in the data ● The check includes modifications of the aggregated order (with OMOD event) with the addition of new order(s) being aggregated ● An ONEW/OMOD event is invalid if any of the above criteria are not met

Rule ID	CR05
Description	For allocation (AALC) events, the constituent client orders with LOIDs which are shown in the allocToLOID field should be present in the data
Related field(s)	logicalOrderID, allocToLOID in AALC event
Additional comment	● An AALC event is invalid if the orders with LOIDs presented in the allocToLOID field (i.e. the constituent client orders) do not exist in the data

Rule ID	CR06
Description	For multi-day (e.g. GTC/GTD) orders, the logicalOrderID should remain unchanged for events on each day until the order is expired or cancelled
Related field(s)	logicalOrderID, timeInForce, eventDateTime, accountID, clientID, side, securityID, orderQty, nDayOrderQty in ONEW event

Additional comment	● Multi-day orders are identified using the timeInForce field for GTC and GTD orders ● With reference to additional fields such as eventDateTime, accountID, clientID, side, securityID, orderQty, nDayOrderQty, etc. to identify related ONEW events of a multi-day order ● Related events of a multi-day order are invalid if the logicalOrderID is not the same

3.2. Referential Integrity

Summary of rules under Referential Integrity category:

Rule ID	Descriptions
RI01	Dictionary key value of accountID should be present in the corresponding Account Details reference data dictionary with accountID dictionary key
RI02	Dictionary key value of clientID should be present in the corresponding Client Details reference data dictionary with clientID dictionary key
RI03	Dictionary key value of salesID and traderID should be present in the corresponding User Details reference data dictionary with userID dictionary key
RI04	Dictionary key value of executionVenue and sourceExecVenue should be present in the corresponding Execution Venue reference data dictionary with executionVenue dictionary key
RI05	Dictionary key value of algoStrategyID should be present in the corresponding Algo Strategy reference data dictionary with algoStrategyID dictionary key
RI06	Dictionary key value of orderInst should be present in the corresponding Order Instruction reference data dictionary with orderInst dictionary key
RI07	In Client Details reference data, full name of the client or entity given in entityName should be unique, i.e. no same name in multiple clientIDs
RI08	In User Details reference data, full name of the user in userName should be unique, i.e. no same user name in multiple userIDs, except if dateModified is different and dateOpen is the same (indicates an update of the user record)

Rule ID	RI01
Description	Dictionary key value of accountID should be present in the corresponding Account Details reference data dictionary with accountID dictionary key
Related field(s)	accountID in applicable events accountID in reference data dictionary

Additional comment	● An event is invalid if the accountID value is not found against accountID dictionary key in the Account Details reference data dictionary

Rule ID	**RI02**
Description	Dictionary key value of clientID should be present in the corresponding Client Details reference data dictionary with clientID dictionary key
Related field(s)	clientID in applicable events clientID in reference data dictionary
Additional comment	● The clientID field may contain multiple values with '|' as delimiter ● An event is invalid if the clientID value is not found against clientID dictionary key in the Client Details reference data dictionary

Rule ID	**RI03**
Description	Dictionary key value of salesID and traderID should be present in the corresponding User Details reference data dictionary with userID dictionary key
Related field(s)	salesID, traderID in applicable events userID in reference data dictionary
Additional comment	● An event is invalid if the salesID or traderID value is not found against userID dictionary key in the User Details reference data dictionary

Rule ID	**RI04**
Description	Dictionary key value of executionVenue and sourceExecVenue should be present in the corresponding Execution Venue reference data dictionary with executionVenue dictionary key
Related field(s)	executionVenue, sourceExecVenue in applicable events executionVenue in reference data dictionary
Additional comment	● An event is invalid if the executionVenue or sourceExecVenue value is not found against executionVenue dictionary key in the Execution Venue reference data dictionary

Rule ID	**RI05**
Description	Dictionary key value of algoStrategyID should be present in the corresponding Algo Strategy reference data dictionary with algoStrategyID dictionary key

Related field(s)	algoStrategyID in applicable events algoStrategyID in reference data dictionary
Additional comment	● An event is invalid if the algoStrategyID value is not found against algoStrategyID dictionary key in the Algo Strategy reference data dictionary

Rule ID	**RI06**
Description	Dictionary key value of orderInst should be present in the corresponding Order Instruction reference data dictionary with orderInst dictionary key
Related field(s)	orderInst in applicable events orderInst in reference data dictionary
Additional comment	● The orderInst field may contain multiple values with ‘|’ as delimiter ● An event is invalid if the orderInst value is not found against orderInst dictionary key in the Order Instruction reference data dictionary

Rule ID	**RI07**
Description	In Client Details reference data, the full name of the client or entity given in entityName should be unique, i.e. no same name in multiple clientIDs
Related field(s)	clientID, entityName in Client Details reference data
Additional comment	● A Client Details reference data could be invalid if the same entityName is found in more than one clientID key

Rule ID	**RI08**
Description	In User Details reference data, the full name of the user in userName should be unique, i.e. no same user name in multiple userIDs, except if dateModified is different and dateOpen is the same (indicates an update of the user record)
Related field(s)	userID, userName, dateOpen, dateModified in User Details reference data
Additional comment	● A User Details reference data could be invalid if the same userName is found in more than one userID key ● An exception is that if these records have the same dateOpen value but different dateModified values, which indicates an update of the user record

3.3. Data Consistency

Summary of rules under the Data Consistency category:

Rule ID	Descriptions
DC01	The sum of executed quantity (executionQty) in all executions (EXEC, and EXCR if applicable) of an order should not exceed its final order quantity (finalOrderQty) in its Order Summary (OSUM); If the order was cancelled (OCXL exists), however, it is acceptable to have finalOrderQty equals ‘zero’
DC02	The sum of executed quantity (executionQty) in all executions (EXEC, and EXCR if applicable) of an order should match its total executed quantity (totalExecutedQty) in its Order Summary (OSUM)
DC03	In Order & Split Modify events (OMOD & SMOD), the provided original order ID (origOrderID) should match a corresponding orderID in the data
DC04	For multi-day (e.g. GTC/GTD) orders, the provided ‘order quantity of a day’ (nDayOrderQty, if used) on Day N should be consistent with corresponding order quantity (orderQty) and executed quantity in previous N-1 days from related EXEC events. i.e. nDayOrderQty = orderQty - (sum of executionQty in related EXEC events in previous N-1 days)
DC05	The average executed price (avgExecutedPrice) and total executed quantity (totalExecutedQty) in OSUM should match with the calculated average from its EXEC or EXCR events. Reasonable rounding error is acceptable
DC06	The initial order price (initialOrderPrice) and quantity (initialOrderQty) in OSUM should match with the price and quantity values in the ONEW event
DC07	The final order price (finalOrderPrice) and quantity (finalOrderQty) in OSUM should match with the price and quantity in the last OMOD event if present, otherwise (when there is no OMOD), should match with those in the ONEW event; If the order was cancelled (OCXL exists), however, it is acceptable to have finalOrderQty equals ‘zero’
DC08	The sum of allocated quantities (allocQty) from an aggregated order’s AALC events should not exceed its total executed quantity
DC09	For cross trades (crossTrade is True), the execution capacity (executionCapacity) must be either 'XA' or 'XP'
DC10	For cross trades (crossTrade is True), the trade session (tradeSession) should be provided
DC11	For cross trades (crossTrade is True), the counterparty (counterpartyID) should be provided
DC12	Order side ('buy' or 'sell', short and long sell are considered the same side) should be consistent among Order and Split events, as well as all related events with the same LOID
DC13	The same instrument symbology should be used within the dataset

DC14	In an Execution Correction (EXCR) event, the sum of corrected execution quantity (executionQty) and cancelled quantity (execCxlQty) should be the same as before the correction
DC15	Order trade events submitted should fall within the submission date range inclusive of start and end dates. For day orders, overnight or queued orders received before the start date can be accepted. For multi-day orders, at least one day of events should be within the range
DC16	Order trade events timestamp should fall within business days, i.e. excluding public holidays and weekends
DC17	Execution (EXEC) events timestamp should fall within trading hours based on market and security type
DC18	Short sell execution (EXEC) should not be a cross trade (crossTrade is True)
DC19	Execution (EXEC) events of the same logicalOrderID (LOID) should have the same tradeBookID
DC20	All execution events with execution capacity of proprietary (P or XP) must have a non-null value in the tradeBookID field
DC21	ONEW/OMOD/SNEW/SMOD event which have a client type of INSI (institutional internal) or PROP (proprietary) should not have an orderCapacity of A
DC22	The executionVenue field in Execution (EXEC) events and Split New events (SNEW) shouldn't be N/A
DC23	SNEW events should not have orderType = MKT or orderPrice = 0

Rule ID	**DC01**
Description	The sum of executed quantities (executionQty) in all executions (EXEC, and EXCR if applicable) of an order should not exceed its final order quantity (finalOrderQty) in its Order Summary (OSUM); If the order was cancelled (OCXL exists), however, it is acceptable to have finalOrderQty equals 'zero'
Related field(s)	executionQty in EXEC & EXCR event execCxlQty in EXCR event finalOrderQty in OSUM event
Additional comment	● The total executed quantity of an order is calculated by the sum of the executionQty field of all executions in EXEC events with the same LOID ● In case an order's execution is corrected, the total executed quantity after correction is from the executionQty field of EXCR event with the same LOID ● An OSUM event and/or related EXEC/EXCR events are invalid if the calculated total executed quantity exceeds the finalOrderQty in OSUM event ● One exception is that if an order was cancelled, with the existence of OCXL event in the same LOID, finalOrderQty can be 'zero'

Rule ID	**DC02**
Description	The sum of executed quantities (executionQty) in all executions (EXEC, and EXCR if applicable) of an order should match its total executed quantity (totalExecutedQty) in its Order Summary (OSUM)
Related field(s)	executionQty in EXEC & EXCR event execCxlQty in EXCR event totalExecutedQty in OSUM event
Additional comment	● The total executed quantity of an order is calculated by the sum of the executionQty field of all executions in EXEC events with the same LOID ● In case an order's execution is corrected, the total executed quantity after correction is from executionQty field of EXCR event with the same LOID ● An OSUM event and/or related EXEC/EXCR events are invalid if the calculated total executed quantity does not match with the totalExecutedQty in OSUM event

Rule ID	**DC03**
Description	In Order & Split Modify events (OMOD & SMOD), the provided original order ID (origOrderID) should match a corresponding orderID in the data
Related field(s)	orderID in ONEW/SNEW/OMOD/SMOD event origOrderID in OMOD/SMOD event
Additional comment	● An OMOD or SMOD event is invalid if an 'order ID' in the origOrderID provided is not found in any ONEW or SNEW event with the same LOID

Rule ID	**DC04**
Description	For multi-day (e.g. GTC/GTD) orders, the provided 'order quantity of a day' (nDayOrderQty, if used) on Day N should be consistent with corresponding order quantity (orderQty) and executed quantity in previous N-1 days from related EXEC events. i.e. nDayOrderQty = orderQty – (sum of executionQty in related EXEC events in previous N-1 days)
Related field(s)	orderQty in ONEW/OMOD event nDayOrderQty in ONEW event executionQty in EXEC event
Additional comment	● Applicable to in-scope brokers which keep the same original orderQty for multi-day orders, when the nDayOrderQty field is used to indicate the actual quantity for the rolled-over order on a given day ● When the original order's quantity has been modified at Day N-1, the nDayOrderQty on Day N should be updated accordingly

- For a multi-day order on Day N before any new execution, the executed quantity so far can be calculated by the sum of executionQty in related EXEC events from previous N-1 days
- A good example of in-scope broker keeping the same original orderQty for multi-day order (only relevant events and fields are illustrated):

Day	**event**	**orderQty**	**nDayOrderQty**	**executionQty**
1	ONEW	10,000	10,000	
1	EXEC			2,000
2	ONEW	10,000	8,000	
2	OMOD	12,000		
2	EXEC			3,000
2	EXEC			1,000
3	ONEW	12,000	6,000	

- An ONEW event is invalid if the nDayOrderQty field (when used for a multi-day order) does not add up to the latest order quantity and the executed quantity so far before any new execution

Rule ID	**DC05**
Description	The average executed price (avgExecutedPrice) and total executed quantity (totalExecutedQty) in OSUM should match with the calculated average from its EXEC or EXCR events. Reasonable rounding error is acceptable
Related field(s)	executionPrice, executionQty in EXEC/EXCR event avgExecutedPrice, totalExecutedQty in OSUM event
Additional comment	● The average executed price of an order is calculated using the executionPrice and executionQty fields of all executions in EXEC events with the same LOID ● In case an order's execution is corrected, the average price after correction is from the executionPrice field of EXCR event with the same LOID ● An OSUM event and/or related EXEC/EXCR events are invalid if the calculated total executed quantity does not match with the totalExecutedQty, or the calculated average executed price does not match with the avgExecutedPrice in OSUM event. Rounding errors from averaging are acceptable if the differences are not significant

Rule ID	**DC06**
Description	The initial order price (initialOrderPrice) and quantity (initialOrderQty) in OSUM should match with the price and quantity values in the ONEW event
Related field(s)	orderPrice, orderQty in ONEW event initialOrderPrice, initialOrderQty in OSUM event
Additional comment	● By the definition of initialOrderPrice and initialOrderQty in OSUM event, the values should always agree with the orderPrice and orderQty in ONEW event respectively with the same LOID ● An OSUM event is invalid if these values do not match with those in the corresponding ONEW event

Rule ID	**DC07**
Description	The final order price (finalOrderPrice) and quantity (finalOrderQty) in OSUM should match with the price and quantity in the last OMOD event if present, otherwise (when there is no OMOD), should match with those in the ONEW event; If the order was cancelled (OCXL exists), however, finalOrderQty can be ‘zero’
Related field(s)	orderPrice, orderQty in ONEW/OMOD event finalOrderPrice, finalOrderQty in OSUM event
Additional comment	● For orders that have not been modified throughout their life cycles, the finalOrderPrice and finalOrderQty in OSUM event should agree with the orderPrice and orderQty in ONEW event with the same LOID ● If an order has been modified once or more throughout its life cycle, the finalOrderPrice and finalOrderQty in OSUM event should agree with the orderPrice and orderQty in the last OMOD event with the same LOID. ● An OSUM event is invalid if these values do not match with those in the corresponding ONEW/ OMOD event ● If an order was cancelled, with the existence of OCXL event in the same LOID, finalOrderQty can be ‘zero’

Rule ID	**DC08**
Description	The sum of allocated quantities (allocQty) from an aggregated order’s AALC events should not exceed its total executed quantity
Related field(s)	executionQty in EXEC event allocQty in AALC event
Additional comment	● The total executed quantity of an order is calculated by the sum of the executionQty field of all executions in EXEC events with the same LOID

	● The total allocated quantity is calculated by the sum of allocQty fields of all AALC events with the same LOID ● AALC events are invalid if the total allocated quantity exceeds the total executed quantity

Rule ID	**DC09**
Description	For cross trades (crossTrade is True), the execution capacity (executionCapacity) must be either ‘XA’ or ‘XP’
Related field(s)	crossTrade, executionCapacity in EXEC event
Additional comment	● By the definition of the executionCapacity field, for any internally crossed trade, i.e. the crossTrade field is True, its value must be either ‘XA’ or ‘XP’ ● An EXEC event is invalid if this rule is not met

Rule ID	**DC10**
Description	For cross trades (crossTrade is True), the trade session (tradeSession) should be provided
Related field(s)	crossTrade, tradeSession in EXEC event
Additional comment	● By the definition of the tradeSession field, for any internally crossed trade, i.e. the crossTrade field is True, a valid value (‘PO’, ‘CT’, ‘CA’ or ‘OF’) should be provided ● An EXEC event is invalid if this rule is not met

Rule ID	**DC11**
Description	For cross trades (crossTrade is True), the counterparty (counterpartyID) should be provided
Related field(s)	crossTrade, counterpartyID
Additional comment	● By the definition of counterpartyID, for any internally crossed trade, i.e. the crossTrade field is True, the counterparty’s ID should be provided ● An EXEC event is invalid if this rule is not met

Rule ID	**DC12**
Description	Order side (‘buy’ or ‘sell’, short sell and long sell are considered the same side) should be consistent among Order and Split events, as well as all related events with the same LOID

Related field(s)	logicalOrderID, side in applicable events
Additional comment	● Order side should be consistent when it is modified or split. Short sell and long sell are considered the same side ● An event is invalid if the side is not consistent with its ONEW event with the same LOID

Rule ID	**DC13**
Description	The same instrument symbology should be used within the dataset
Related field(s)	securityID, securitySource, underlyingSecurityID, underlyingIDSource in all events
Additional comment	● An event is invalid if different instrument symbologies are used within the dataset

Rule ID	**DC14**
Description	In an Execution Correction (EXCR) event, the sum of corrected execution quantity (executionQty) and cancelled quantity (execCxlQty) should be the same as that before the correction
Related field(s)	executionQty in EXEC/EXCR event execCxlQty in EXCR event
Additional comment	● The total executed quantity of an order is calculated by the sum of executionQty field of all executions in EXEC events with the same LOID ● An EXCR event is invalid if the sum of executionQty and execCxlQty does not match with the total executed quantity of an order

Rule ID	**DC15**
Description	Order trade events submitted should fall within the submission date range inclusive of the start and end dates. For day orders, overnight or queued orders received before the start date can be accepted. For multi-day orders, at least one day of events should be within the range
Related field(s)	eventDateTime, eventRespDateTime in all events
Additional comment	● For day orders, an event is invalid if the date in eventDateTime and eventRespDateTime fall outside the start and end dates found in the submission file’s header record, subject to a reasonable threshold ● For multi-day orders, an event is invalid if none of the related events linked by LOID is within the date range

Rule ID	**DC16**
Description	An order trade event’s timestamp should fall within business days, i.e. excluding public holidays and weekends
Related field(s)	eventDateTime, eventRespDateTime in all events
Additional comment	● An event might be invalid if the date in eventDateTime and eventRespDateTime fall in public holidays or weekends ● Exceptions may apply, e.g. if orders are received during holidays or weekends and executed on the next business day

Rule ID	**DC17**
Description	An execution (EXEC) event timestamp should fall within trading hours based on market and security type
Related field(s)	eventDateTime in EXEC events
Additional comment	● An EXEC event is invalid if the time in eventDateTime falls outside of trading hours and the execution venue is SEHK (on-exchange executions should be within 9:00 to 16:10 HKT) ● Exceptions may apply for executions happening off-hours within a reasonable timeframe

Rule ID	**DC18**
Description	Short sell execution (EXEC) should not be a cross trade (crossTrade is True)
Related field(s)	side, crossTrade in EXEC events
Additional comment	● An EXEC event is invalid if the side is sell short (ie, (5) or sell short exempt (ie, 6) and crossTrade is True

Rule ID	**DC19**
Description	Execution (EXEC) events of the same logicalOrderID (LOID) should have the same tradeBookID
Related field(s)	tradeBookID in EXEC events
Additional comment	● All EXEC events under the same LOID might be invalid if more than one values are reported in tradeBookID field

Rule ID	**DC20**
Description	All execution events (EXEC) with execution capacity of proprietary (P or XP) must have a non-null value in the tradeBookID field
Related field(s)	executionCapacity, tradeBookID
Additional comment	● An execution event (EXEC) is invalid if executionCapacity is 'P' or 'XP', and tradeBookID is Null (__null__)

Rule ID	**DC21**
Description	ONEW/OMOD/SNEW/SMOD event which have a client type of INSI (institutional internal) or PROP (proprietary) should not have an orderCapacity of ‘A’
Related field(s)	clientType, orderCapacity in ONEW/OMOD/SNEW/SMOD events
Additional comment	● An event might be invalid if clientType is INSI or PROP but orderCapacity is ‘A’

Rule ID	**DC22**
Description	The executionVenue field in Execution (EXEC) events and Split New events (SNEW) should not be N/A
Related field(s)	executionVenue in EXEC/SNEW events
Additional comment	● An event (EXEC/SNEW) is invalid if its executionVenue field value is N/A (__na__)

Rule ID	**DC23**
Description	SNEW events should not have orderType = MKT or orderPrice = 0
Related field(s)	orderType, orderPrice in SNEW events
Additional comment	● An SNEW event is invalid if - orderType = MKT OR - orderPrice = 0

3.4. Event Construct

Summary of rules under the Event Construct category:

Rule ID	Descriptions
EC01	Timestamps of the events should be able to be reconstructed in the correct sequence using eventDateTime with the eventSequence field if provided, i.e. a combination of eventDateTime and eventSequence should always be unique
EC02	Aggregation of orders should not have a mix of orderCapacity into the same aggregated order
EC03	ONEW and OSUM events should be present in overnight orders in a manner consistent with the specific scenarios given
EC04	There should be at most one Execution Correction (EXCR) event per order
EC05	There should be no additional EXEC event after EXCR event
EC06	AALC events should only be used for aggregated orders
EC07	ONEW event should always be the first event of an order's life cycle at the start of a day
EC08	OSUM event should be the last event of an order's life cycle except aggregated orders
EC09	OCXL event should always come after ONEW and OMOD events except if the OCXL event is rejected
EC10	A supplementary event record must immediately follow its corresponding event record
EC11	ONEW events of the constituent client orders should happen before the corresponding ONEW event of the aggregated order
EC12	Aggregation of client orders should not have a mix of order types (MKT and LMT) into the same aggregated order

Rule ID	EC01
Description	Reconstruction of sequence of events should be made possible using eventDateTime with eventSequence field if provided, i.e. a combination of eventDateTime and eventSequence should always be unique
Related field(s)	eventDateTime, eventSequence in all events
Additional comment	● An event is invalid if a combination of the eventDateTime and eventSequence fields happens to be the same with any other event in the same dataset

Rule ID	**EC02**
Description	Aggregation of orders should not have a mix of orderCapacity in the same aggregated order
Related field(s)	event, aggregatedOrders, orderCapacity in ONEW event
Additional comment	● An order's ONEW event having valid values in the aggregatedOrders field is considered an aggregated order ● An ONEW event of an aggregated order should be invalid if its constituent orders have a mix of orderCapacity in their ONEW events

<table>
<tr><th>Rule ID</th><th>EC03</th></tr>
<tr><td>Description</td><td>
ONEW and OSUM events should be presented in overnight orders in a manner consistent with the scenarios below:

For the purposes of the cases, the terms C+0 (Calendar day +0) and C+1 are used. Calendar day is generally defined as the calendar day according to Hong Kong Standard Time. Trade day (T+0) refers to the <u><i>trading day</i></u>. <i>When T+0 is Friday or pre-public holiday, T+1 should be next Monday or the first business day after public holiday.</i>

<i>For the purposes of the scenarios below, C+0 refers to 8 July 2021 and C+1 refers to 9 July 2021.</i>
<table>
<tr><th>Time Period</th><th>Label</th></tr>
<tr><td>16:10:01 – 23:59:59 (e.g. July 8)</td><td>C+0 / T+0</td></tr>
<tr><td>00:00:00 – 16:10:00 (e.g. July 9, the next calendar day)</td><td>C+1 / T+0</td></tr>
<tr><td>16:10:01 – 23:59:59 (e.g. July 9)</td><td>C+1 / T+1</td></tr>
</table>
<table>
<tr><th>SCENARIO</th><th>EXAMPLE</th><th>ONEW/OSUM Required on C+0 (T+0)?</th><th>ONEW/OSUM Required on C+1 (T+0)?</th></tr>
<tr><td rowspan="2">(A) Order is received and accepted</td><td>Order arrives at 6pm on C+0 (T+0), and is accepted immediately. The order is rolled over or reinstated right before market open on C+1 (T+0), (e.g. 8am) and receives executions on C+1 (T+0)</td><td>Yes, both ONEW (eventDateTime=6pm) and OSUM events are required. For the avoidance of doubt, the time that the order is reinstated is not relevant</td><td>Only an OSUM event is required</td></tr>
<tr><td>Order arrives at 2am on C+1 (T+0), and is accepted immediately and receives executions on C+1 (T+0)</td><td>Not applicable because order is received on C+1 (T+0)</td><td>Yes, an ONEW is required with eventDateTime = 2am (the time that the order is</td></tr>
</table>
</td></tr>
</table>

			accepted), and an OSUM event is required for C+1 (T+0) day
	Order arrives at 6pm on C+0 (T+0), and is accepted. The order is subsequently cancelled on C+0 (T+0)	Yes, ONEW (eventDateTime=6pm), OCXL and OSUM events are required	Not applicable because the order is already cancelled
	Order arrives at 6pm on C+0 (T+0), and is accepted. The order is subsequently cancelled on C+1 (T+0) before market open	Yes, only ONEW (eventDateTime=6pm) and OSUM events are required	Yes, OCXL and OSUM events are required.
	Order arrives at 2am on C+1 (T+0), and is accepted. The order is subsequently cancelled on C+1 (T+0) before market open	Not applicable because the order is received on C+1 (T+0)	Yes, ONEW (eventDateTime=2am), OCXL and OSUM events are required.
(B) Order is received and rejected	Order arrives at 6pm on C+0 (T+0), and is rejected immediately	Yes, ONEW and OSUM events are required	Not applicable as the order is already rejected
	Order arrives at 2am on C+1 (T+0), and is rejected immediately	Not applicable because the order is received on C+1 (T+0)	Yes, an ONEW is required with eventDateTime = 2am (the time that the order is rejected), and an OSUM event is required for C+1 (T+0) day
(C) Order is received, and is neither accepted nor rejected until C+1 (T+0)	Order arrives at 6pm on C+0 (T+0), and is neither accepted nor rejected. Before market open on C+1 (T+0), the order is accepted at 8am. Executions received on C+1 (T+0) after market open	No, neither ONEW nor OSUM events are required	Yes, an ONEW is required with eventDateTime = 8am (the time that the order is accepted), and an OSUM event is required for C+1 (T+0) day
	Order arrives at 2am on C+1 (T+0), and is neither accepted nor rejected. Before market open on C+1 (T+0), the order is accepted at 8am. Executions received on C+1 (T+0) after market open	Not applicable because the order is received on C+1 (T+0)	Yes, an ONEW is required with eventDateTime = 8am (the time that the order is accepted), and an OSUM event is required for C+1 (T+0) day

Related field(s)	event, eventDateTime, eventSequence, timeInForce

Additional comment	● The firm should consistently report multi-day orders using one of the following methods: 1) one ONEW reported per day until the order is fully executed or cancelled: ● One ONEW and one OSUM event are expected to be reported per trading day, omission or excess instances are invalid 2) only one ONEW reported during the order life cycle: ● ONEW is reported once when the order is first acknowledged or accepted; and ● One OSUM is expected per trading day

Rule ID	**EC04**
Description	There should be at most one Execution Correction (EXCR) event per order
Related field(s)	event in EXCR event
Additional comment	● An EXCR event is invalid if there is more than one EXCR event with the same LOID

Rule ID	**EC05**
Description	There should be no additional EXEC event after EXCR event
Related field(s)	event, eventDateTime, eventSequence in EXEC/EXCR event
Additional comment	● The eventDateTime and eventSequence fields are used to sequence all events with the same LOID ● An EXEC event is invalid if it appears after an EXCR event with the same LOID

Rule ID	**EC06**
Description	AALC events should only be used for aggregated orders
Related field(s)	event, aggregatedOrders in ONEW event event in AALC event
Additional comment	● An order’s ONEW event having valid values in aggregatedOrders field is considered an aggregated order ● An AALC event is invalid if it appears under a non-aggregated order

Rule ID	EC07
Description	ONEW event should always be the first event of an order’s life cycle at the start of a day
Related field(s)	event, eventDateTime, eventSequence, timeInForce in all applicable events
Additional comment	● eventDateTime and eventSequence fields are used to sequence all events with the same LOID ● For a day order, an event is invalid if it appears before an ONEW event with the same LOID ● For a multi-day order, an event is invalid if it appears before an ONEW event with the same LOID

Rule ID	EC08
Description	OSUM event should be the last event of an order’s life cycle except aggregated orders
Related field(s)	event, aggregatedOrders, eventDateTime, eventSequence, timeInForce in all applicable events
Additional comment	● eventDateTime and eventSequence fields are used to sequence all events with the same LOID ● For a day order, an event is invalid if it appears after an OSUM event with the same LOID ● For a multi-day order, an event is invalid if it appears after an OSUM event with the same LOID

Rule ID	EC09
Description	OCXL event should always come after ONEW and OMOD events except if the OCXL event is rejected
Related field(s)	event, eventDateTime, eventSequence, eventResponseType in ONEW/OMOD/OCXL event
Additional comment	● eventDateTime and eventSequence fields are used to sequence all events with the same LOID ● An non-rejected OCXL event (eventResponseType is not REJ) is invalid if it appears before any ONEW or OMOD event with the same LOID

Rule ID	**EC10**
Description	A supplementary event record must immediately follow its corresponding event record
Related field(s)	event, eventDateTime, eventSequence in ONEWS/OMODS/OCXLS/SNEWS/SMODS/SCXLS event
Additional comment	● The eventDateTime and eventSequence fields are used to sequence all events with the same LOID, including corresponding supplementary event records ● A supplementary event record is invalid if it appears not immediately after the corresponding event with the same LOID and eventDateTime

Rule ID	**EC11**
Description	ONEW events of the constituent client orders should happen before the corresponding ONEW event of the aggregated order
Related field(s)	event, aggregatedOrders, eventDateTime, eventSequence in ONEW event
Additional comment	● An order’s ONEW event having valid values in the aggregatedOrders field is considered an aggregated order ● An ONEW event of an aggregated order is invalid if it appears before any of its constituent orders when sorted by the eventDateTime and/or eventSequence

Rule ID	**EC12**
Description	Aggregation of client orders should not have a mix of order types (MKT and LMT) in the same aggregated order
Related field(s)	event, aggregatedOrders, orderType in ONEW event
Additional comment	● An order’s ONEW event having valid values in the aggregatedOrders field is considered an aggregated order ● An ONEW event of an aggregated order might be invalid if its constituent orders have a mix of orderType (MKT and LMT) in their ONEW events