

# M7 API Presentation

## « An introductory training session on our Intraday Market API »

29.04.2021

Market Data – API Customer Services

# Presentation objectives

## « An introductory training session on our Intraday Market API »

### 1. Technico-Functional presentation of the API

- Understand how the API works in conjunction with M7
- Become autonomous in using API specifications (DFS180)
- What are the key recovery procedures to implement (gaps, etc.)
- M7 Roadmap and EPEX objectives

### 2. Implementation Guidelines

- Exchange on Best practices

# Agenda

## 1. Functional part

- API Background and achievements
- Overview of the API process
- Test environments and processes enhancements
- API functional overview
- M7 Roadmap
- EPEX API support organization

## 2. Technical part

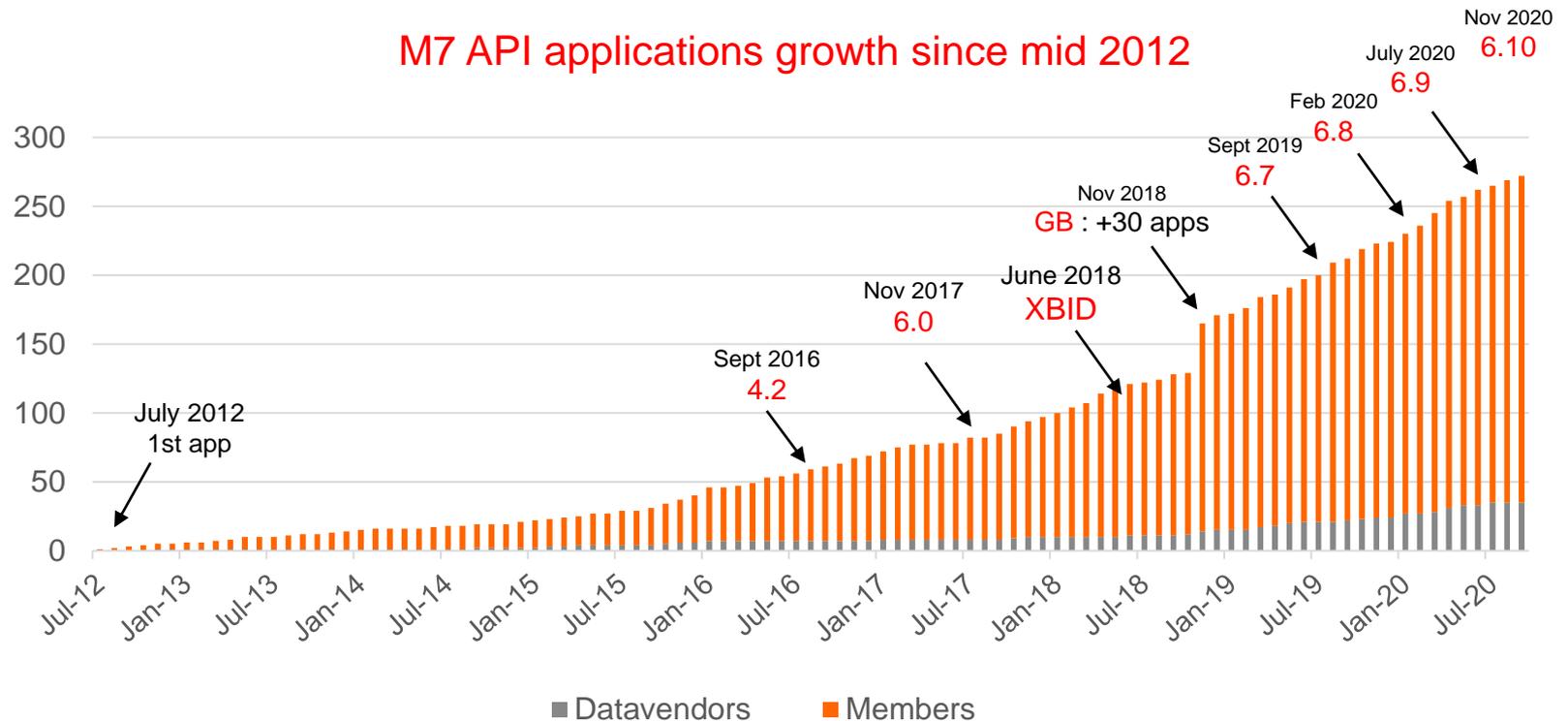
- M7 API technical overview
- API functionalities and messages
- Supported technologies

# 1. API Background and achievements

# API Background and achievements

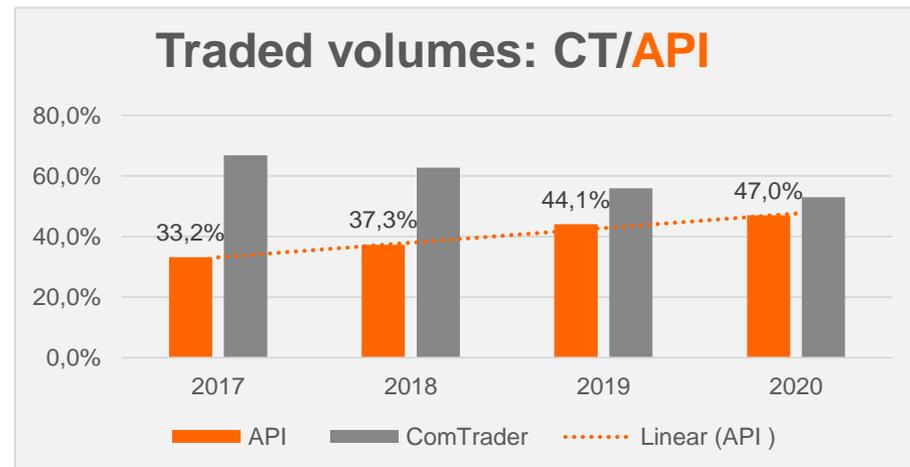
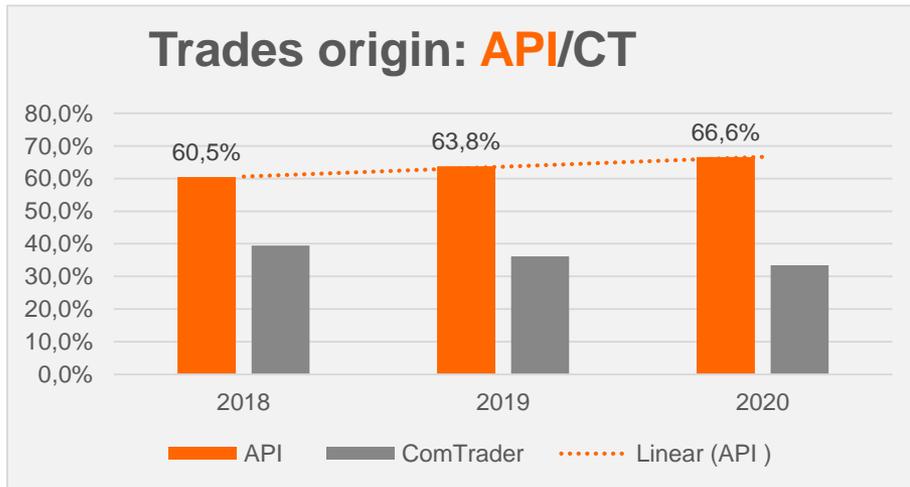
- Today **300 API applications** and **260 members** are connected to the exchange:
  - following the market and positions (pure read only: 40%)
  - and/or **automating trading strategies**

M7 API applications growth since mid 2012



# API Background and achievements

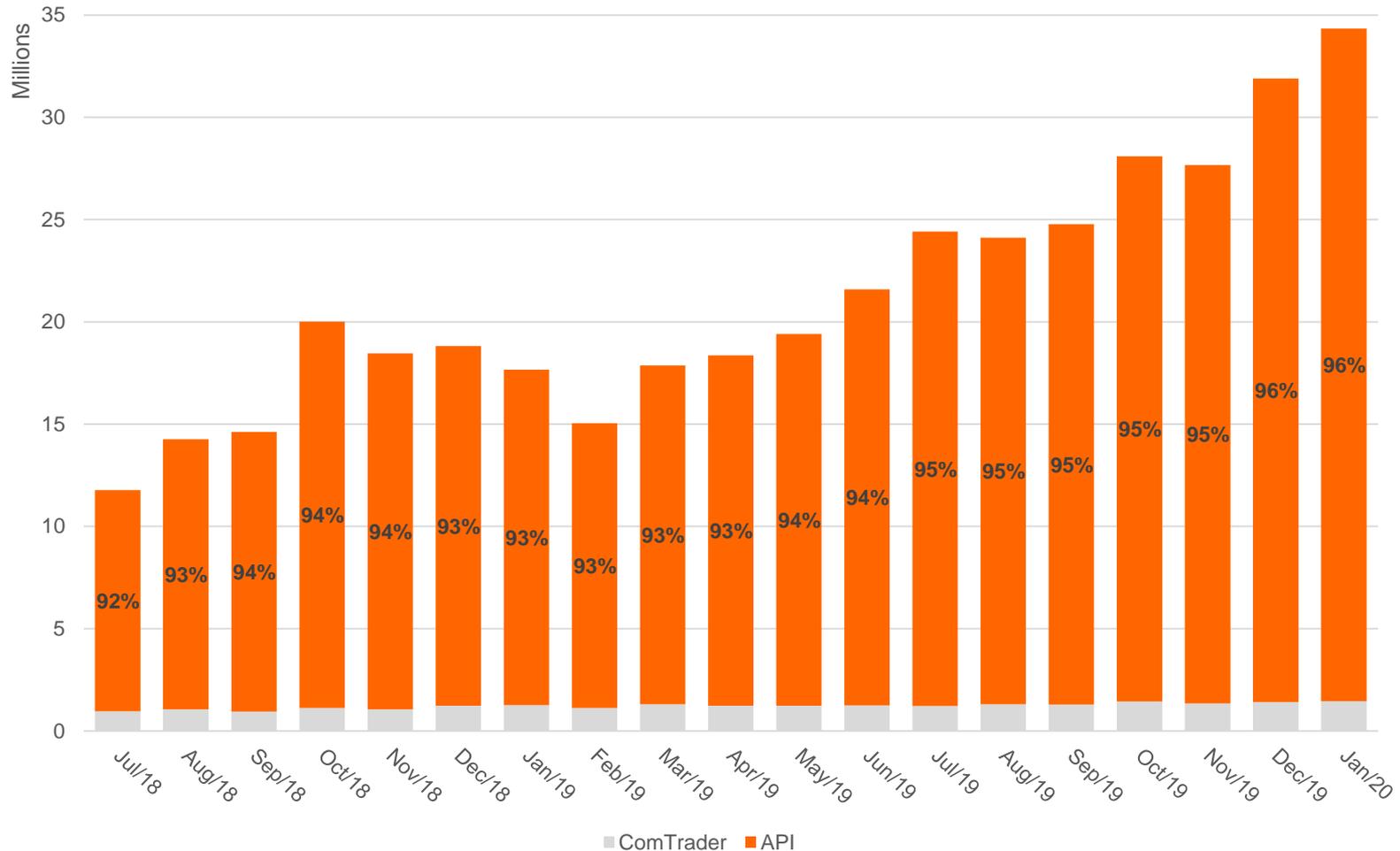
- EPEX has been offering an **API-friendly ecosystem** for more than 7 years.
- **~65% of M7 traded orders come from API apps** (+5% in 2019)
- **~45% of M7 volumes are traded directly via a custom API app** (+18% in 2019)
  - API apps ISV/In-House: 50%-50% (21 M7 “Software Providers”, listed on Epex website)



# API Background and achievements

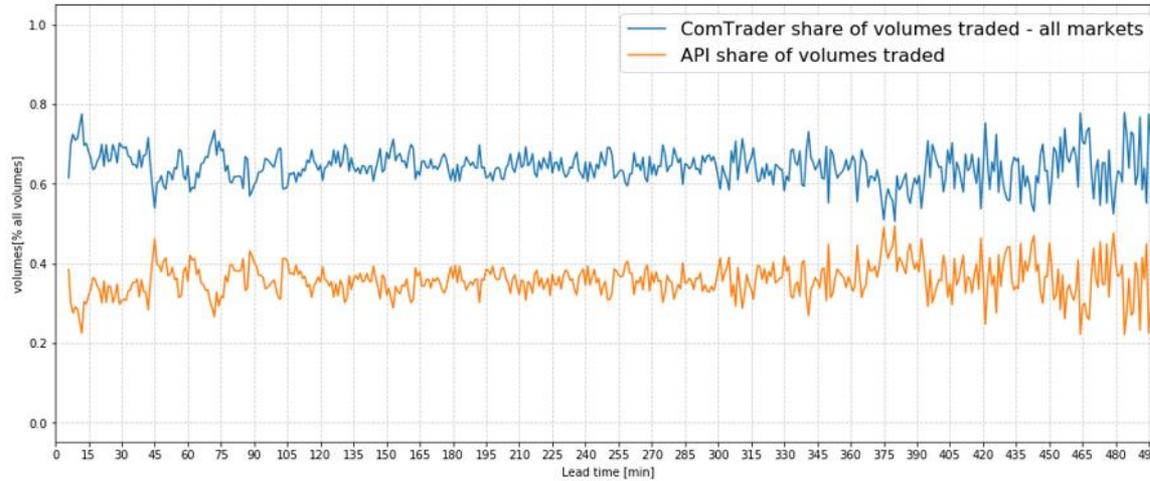
- 95% of orders come from API applications

Monthly total of submitted orders  
(all EPEX continuous markets)

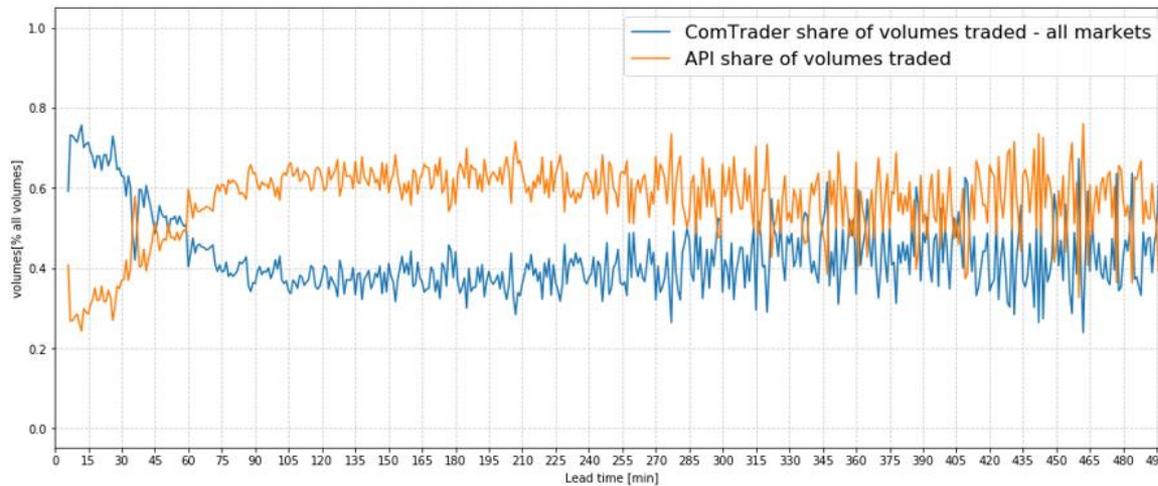


# Split of volumes between APIs and ComTrader

1-hour products per lead time



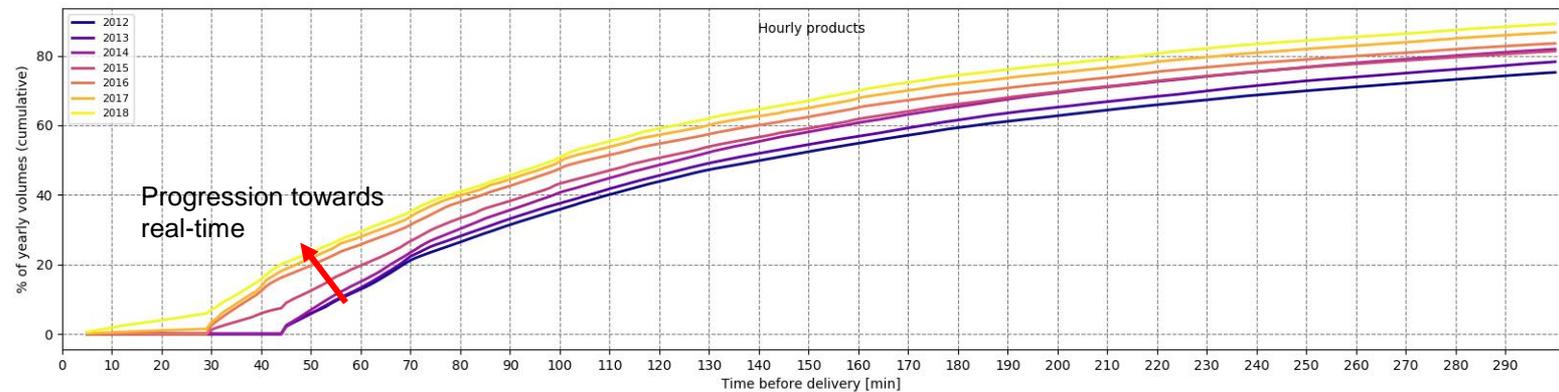
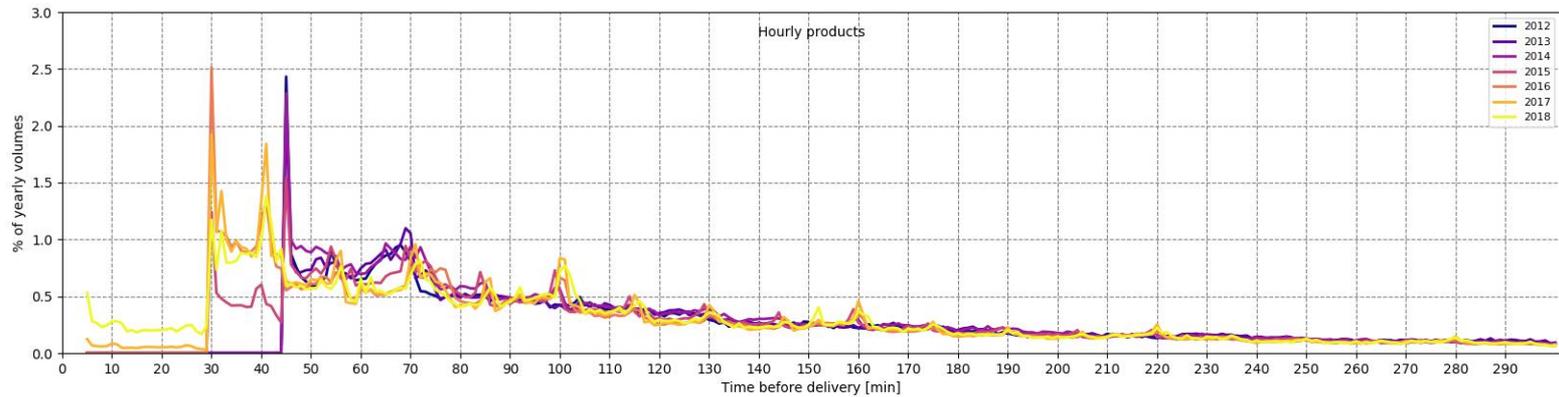
15-min products per lead time



Source: EPEX SPOT

# Trading takes place closer to delivery

Lead time of all trades of 1-hour products with at least 1 leg in the German intraday market



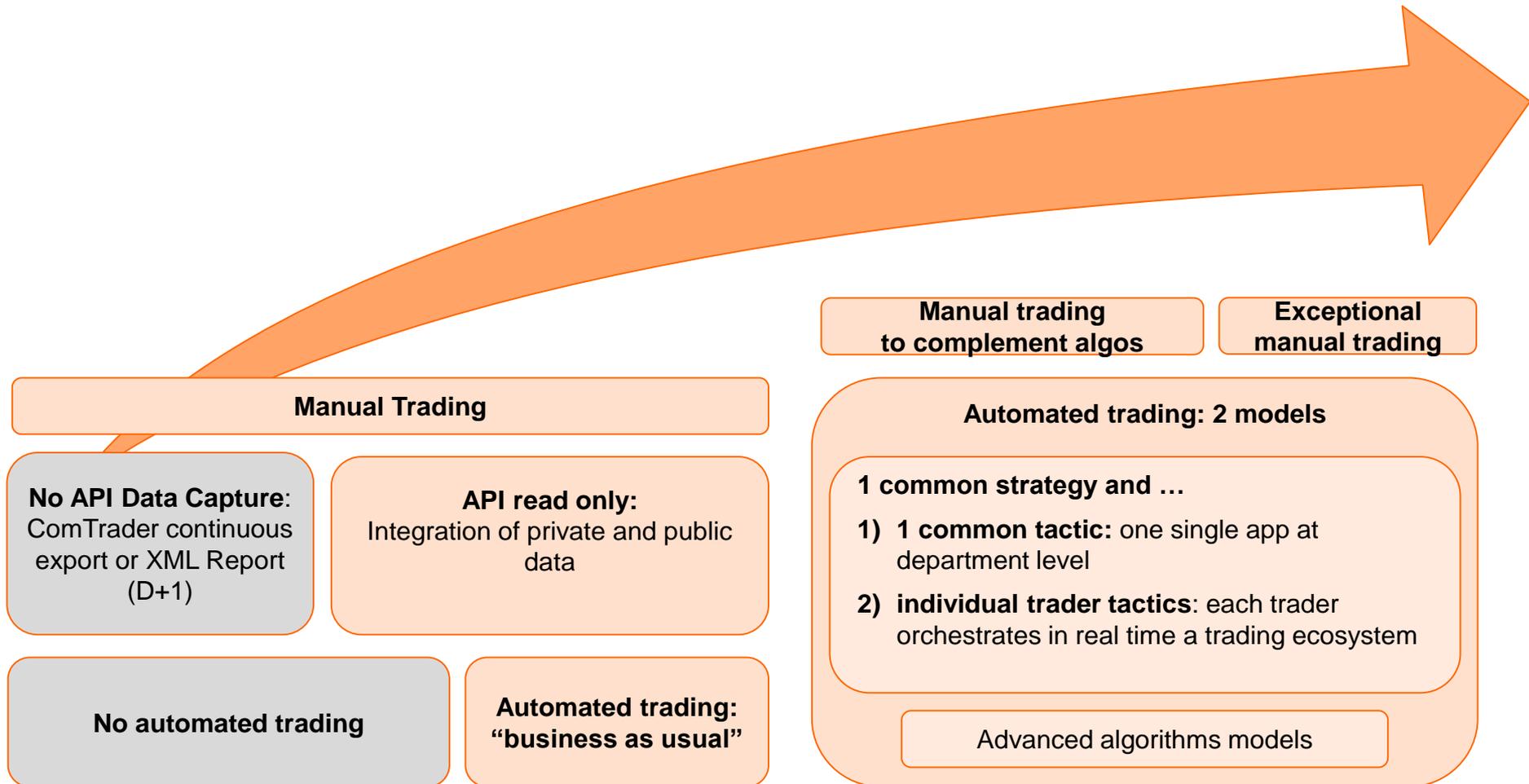
Source: EPEX SPOT

## Certified ISVs offer

- **Certified ISV list** : [https://www.epexspot.com/en/membership/list\\_isv](https://www.epexspot.com/en/membership/list_isv)
- ISVs propose advanced trading solutions, such as:
  1. **Alternative Trading GUIs** (includes additional order possibilities like custom iceberg orders with variable peak quantities, charts and trading indicators)
  2. **Predefined parameter-based trading algorithms**
  3. **Custom trading algorithms** programmable via the ISV software
  4. **Back testing** possibilities
  5. **Integrated suites** featuring solutions on top of AutomaticTrading for:
    - Optimization
    - Logistics ( )
    - Risk managemebalancing, nomination, dispatching (ETRM)

**Several customers combine an own app + ISV(s)**

# Classic path of API applications



# Your trusted Pan-European Power Spot Exchange

**593 TWh**  
traded in 2019 on  
all spot markets

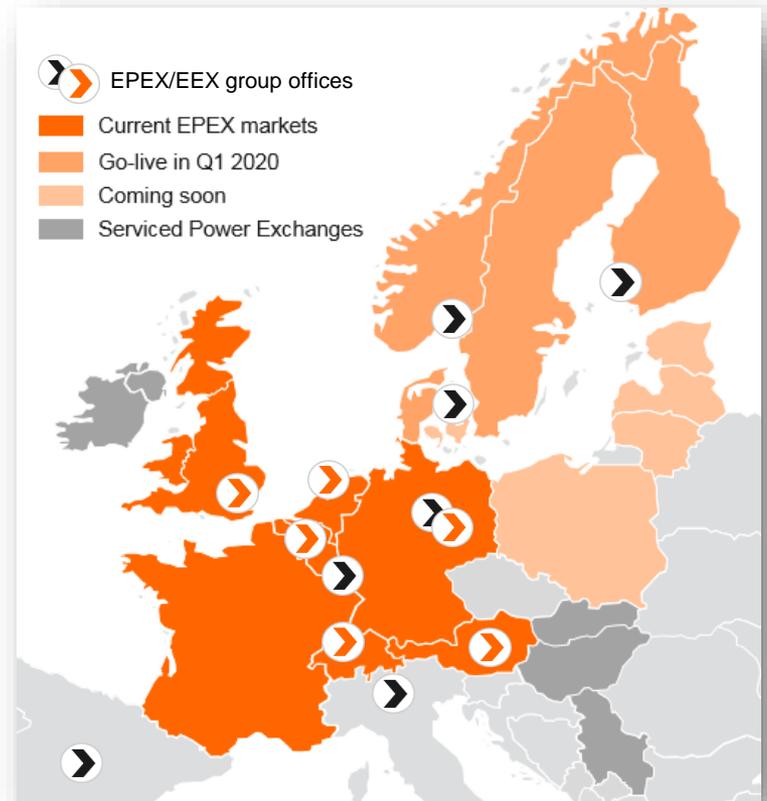
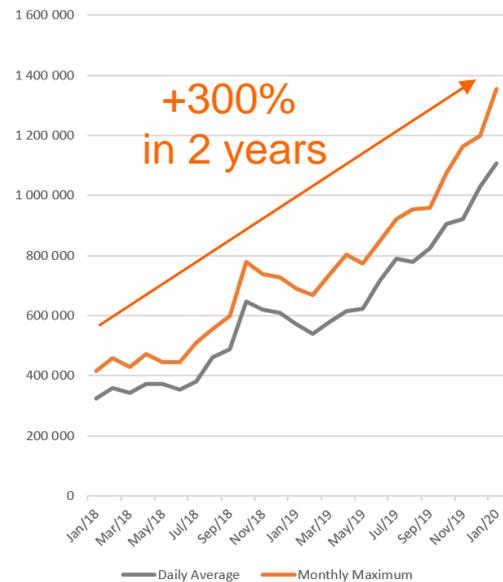
+5% Y2Y

Largest Intraday  
market in Europe  
with  
**91 TWh**  
traded in 2019

+11% Y2Y

Proven technical  
**performance**  
adapted to new market behavior

M7 Order Submissions 2018 - 2019

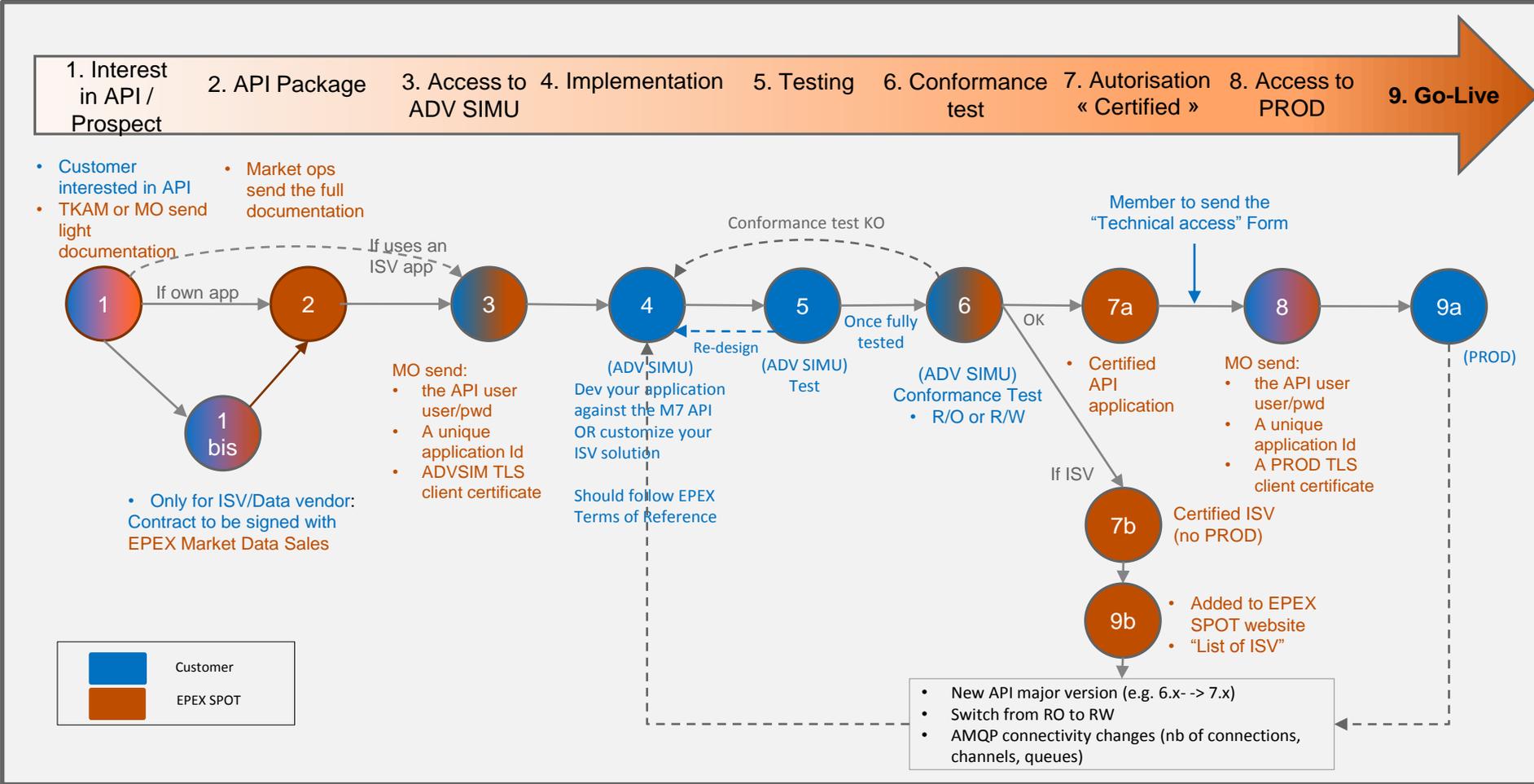


# Automation Stakes for EPEX?

- EPEX is **committed to accompanying the development of trade automation** in the power market.
- For the exchange, the most important is to maintain a:
  1. **Reliable,**
  2. **Orderly Market.**
- **Market surveillance**
  - Manipulation strategies (e.g. spoofing, layering) are monitored
  - Analytical tools/models are used to detect:
    - Abnormal trading behavior,
    - Unusual patterns, etc.

## 2. Overview of the API process

# M7 API Application Process: from Interest to Go-Live

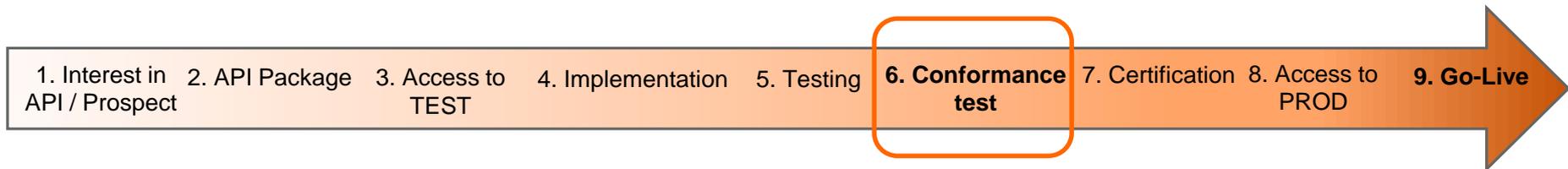


Continuous EPEX SPOT support

# M7 API Implementation Package content

Document	Description
<b>1. DFS180 M7 API</b>	M7 PMI specifications: <ul style="list-style-type: none"> <li>• AMQP principles</li> <li>• Detailed list of messages</li> </ul>
<b>2. M7 – PMI - Terms of Reference</b>	<ul style="list-style-type: none"> <li>• To ensure a fair and stable use of the M7 Trading System</li> <li>• To prevent from any incorrect implementation that might endanger the M7 Trading System stability.</li> </ul>
<b>3. EPEX Environment Details</b>	<ul style="list-style-type: none"> <li>• Production</li> <li>• Advanced simulation</li> <li>• Simulation</li> </ul>
<b>4. M7 API - member procedure</b>	<ul style="list-style-type: none"> <li>• Describe the actions to be taken by Exchange Members during the Software Implementation Process: focus on Conformance test</li> </ul>
<b>5. M7 Public Message Interface FAQ</b>	<ul style="list-style-type: none"> <li>• This document is an amendment of the DFS180 M7 Public API specification.</li> <li>• It takes over topics and terminology of this specification document.</li> </ul>
<b>6. M7 6.0 Static Data</b>	<ul style="list-style-type: none"> <li>• Market areas</li> <li>• Delivery areas</li> <li>• Products</li> </ul>
<b>7. Software Conformance Sheet_60</b>	<ul style="list-style-type: none"> <li>• Business details               <ul style="list-style-type: none"> <li>• R/O or R/W, automated trading or not, in parallel of manual trading,</li> <li>• Purpose the application, etc.</li> </ul> </li> </ul>
<b>8. Technical_Terms of Reference Compliance_60</b>	<ul style="list-style-type: none"> <li>• Customer to confirm having respected points of Terms of Reference (app is fully tested in SIMU env., exp. Back off after connection loss, etc.)</li> </ul>
<b>9. Java Sample Code</b>	<ul style="list-style-type: none"> <li>• Basic API implementation in Java: login + queues management + Log recording broadcasts</li> </ul>
<b>10. This Powerspoint Presentation</b>	<ul style="list-style-type: none"> <li>• An introduction to the M7 API to ease the DFS180 understanding</li> </ul>

# API Applications Conformance Test (1/3)



## Conformance Test Goals:

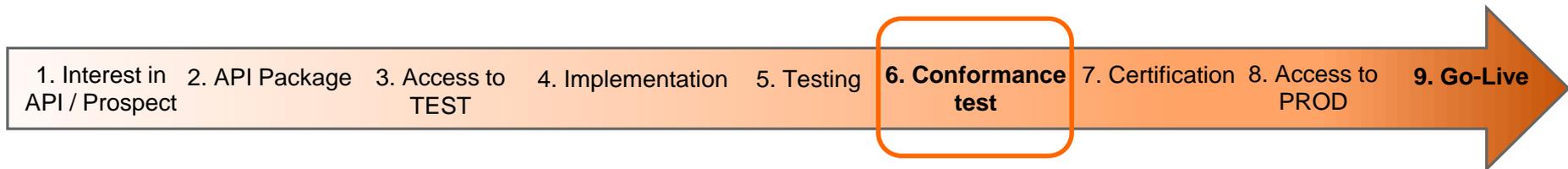
### 1. Accompany our customers to have the best API experience once in PROD

- **by detecting before production any wrong implementation breaching our ToR (implementation principles):**
  - a) that could lead to service interruptions in production:
    - customer app hitting its quotas of inquiry requests: be unable to trade for up to 60mn
    - non optimal connectivity (no logout after each action, no weird logout/login behavior, etc.)
  - b) that would degrade the API customer experience:
    - a proper usage of requests/broadcasts (ex: if only uses requests: customer would miss all the benefits from information “push”)
    - Missing mandatory requests compared to the app description (during the initialization phase or after)

### 2. Ensure customers technical readiness for a new project

(e.g. for a new M7 major release, a new logic like XBID, etc.)

# API Applications Conformance Test (2/3)



- **During 24 hours, we make sure that the tested API application interacts properly with M7 (i.e. respect our **Terms of Reference: will be reviewed**):**
  - Technical checks (e.g. AMQP nb of queues)
  - Connectivity (login/logout policy matching ToR)
  - Reaction to specific events (Market Halt/Trading, etc.)
  - Reaction to a automated continuous flow of orders and trades
- **The functional testing responsibility is on the customer side**
- **Members trading via an ISV:**
  - Thanks to the conformance test ISVs pass, members can use their software without having to pass one as well (ISV standardization process).

# API Applications Conformance Test (2/3)

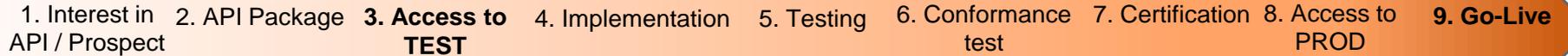


- **Criteria to go through a conformance test**

- New API major version (e.g. 4.2 -> 6.x, 6.x- -> 7.x)
- Switch from Read-Only to Read-Write
- AMQP connectivity changes (nb of connections, channels, queues)

### 3. Test environments and processes enhancements

# Overview of the API process - Test



← **Continuous EPEX SPOT support** →

## ADVANCED SIMULATION - ASIM = main test environment

- **test new M7 and XBID versions** during customer testing periods,
  - iso-production the rest of the year
- **Connected to an XBID platform:** independent from the XBID project tests
- **Used for API Conformance tests**

## SIMULATION - SIMU = secondary test environment

- SIMU upgrades are requested by the XBID project

Only available part time

# Customer Test and Conformance Test enhancements

- **Automatic monitoring of test environments is in place (same as in PROD)**
  - Surveillance of nb unacknowledged messages
  - Surveillance of nb of AMQP channels and queues

## 4. API Functional Overview

# API functional overview (trader perspective)

## 1. Login / Logout

## 2. Order management (management requests):

- **submission, modification, cancellation etc.** : 1 or several at once
- **On behalf** of (obh) another trader colleague

## 3. Trade Management: recall requests (obh), informed about state changes

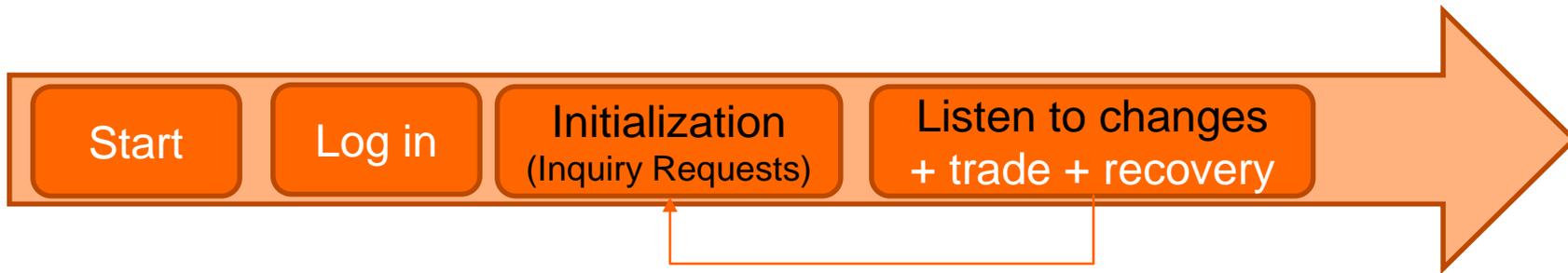
## 4. Get information about (inquiry requests):

- **System:** request rate limits per minute and hour, nb of days during which contracts are stored
- **Reference data:** BGs and users, products description (qty min size), contracts (IDs, state)
- **Market data:**
  - Last state: Order books (incl. Last trade info), own orders, member trading limit, Auction Reference prices, H2H Capacity matrix (XBID), own and public trades (D-7), last public and private messages
  - No Historical: cannot retrieve missed order books or trade states, just the final version

## 5. Be aware of all changes (broadcasts)

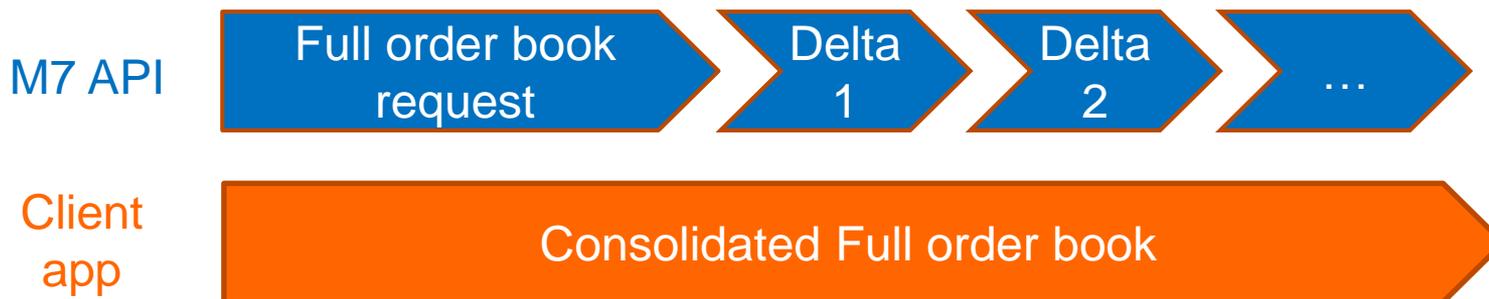
## 6. Change password

# API communication principles: “request and then listen”



1. Use Inquiry Requests to initialize your application
2. Then:
  - Listen to changes : « broadcast messages »
  - Manage orders and trades : « Management requests »

## Order book example:



# Typical API client implementation – RO or Read/Write

API client phase	Functionally
<b>Initialization</b>	<ul style="list-style-type: none"> <li>• <b>Login</b> (including failover logic and disconnection preferences)</li> <li>• <b>Initialization: Get up-to-date data:</b> <ul style="list-style-type: none"> <li>• system info + referential data + market/trading data</li> </ul> </li> </ul>
<b>Running phase</b>	<ul style="list-style-type: none"> <li>• <b>Listen to changes: Keep data up-to-date:</b> <ul style="list-style-type: none"> <li>• Integrate referential data changes (e.g. new area assigned to BG)</li> <li>• Integrate market data changes:               <ul style="list-style-type: none"> <li>• <i>Public data:</i> <i>Contracts, Public Trades, Public Order books, H2H ATC</i></li> <li>• <i>Private data:</i> <i>Own Trades, Own Orders, Trading limit updates</i></li> </ul> </li> </ul> </li> <li>• <b>Trade :</b> <ul style="list-style-type: none"> <li>• <b>Trading strategy:</b> order calculation (algorithms)</li> <li>• <b>Orders and trades management</b> (life cycle: entry -&gt; modif -&gt; etc.)</li> </ul> </li> <li>• <b>Recovery processes</b> (Heartbeats loss, gaps, disconnections)</li> </ul>
<b>Closure</b>	<ul style="list-style-type: none"> <li>• <b>Logout action</b></li> <li>• <b>Cleaning of the session context</b> (if a new user logs in, there should be no leftovers from the previous user session)</li> </ul>

# Order Books

**Order book = list of active orders for 1 contract + area**

- **Order Queue concept:**
  - For order with partial matching: Price-time queue
  - For order with All-Or-Nothing matching: Price-quantity-time queue

- **Illustration of the queue concept:**

► Order Book Details

EON T00-01 (CET) Hi/Low: 56.40 / 55.60 Last: 0.1 @ 56.40 ClosingPx: -

CHs	VWAP	Acc	Qty	Bid	Ask	Qty	Acc	VWAP
		11.6	11.6	55.60	56.40	24.7 (42.7)	24.7	56.40
		31.6	20.0	55.60	61.30	23.0	47.7	58.76
	55.22	37.6	6.0	53.20	65.70	24.0	71.7	61.08
	52.65	62.6	25.0 (26.0)	48.80	66.80	25.0 (36.0)	96.7	62.56
	52.41	64.6	2.0	44.60	69.60	25.0 (38.0)	121.7	64.01
	49.36	89.6	25.0 (38.0)	41.50				

Higher qty but entered after the 11.6MW order: priority #2

Desc. Priority

## Order books movements fundamentals

- **Orders can get in (>0 qty) and out (qty =0) of the order book because:**
  - a) **Intrinsic reason: the order has been updated**
  - b) **OR because of an external reason:**
    - Cross border capacity increasing or decreasing
    - XBID order book depth/volume limitations
- **All orders do not appear in the order book: only the non-traded portion gets displayed**
- You will never see in an obk an order ID trace of an aggressor:
  - fully matched
  - OR with an Execution Restriction = IOC or FOK

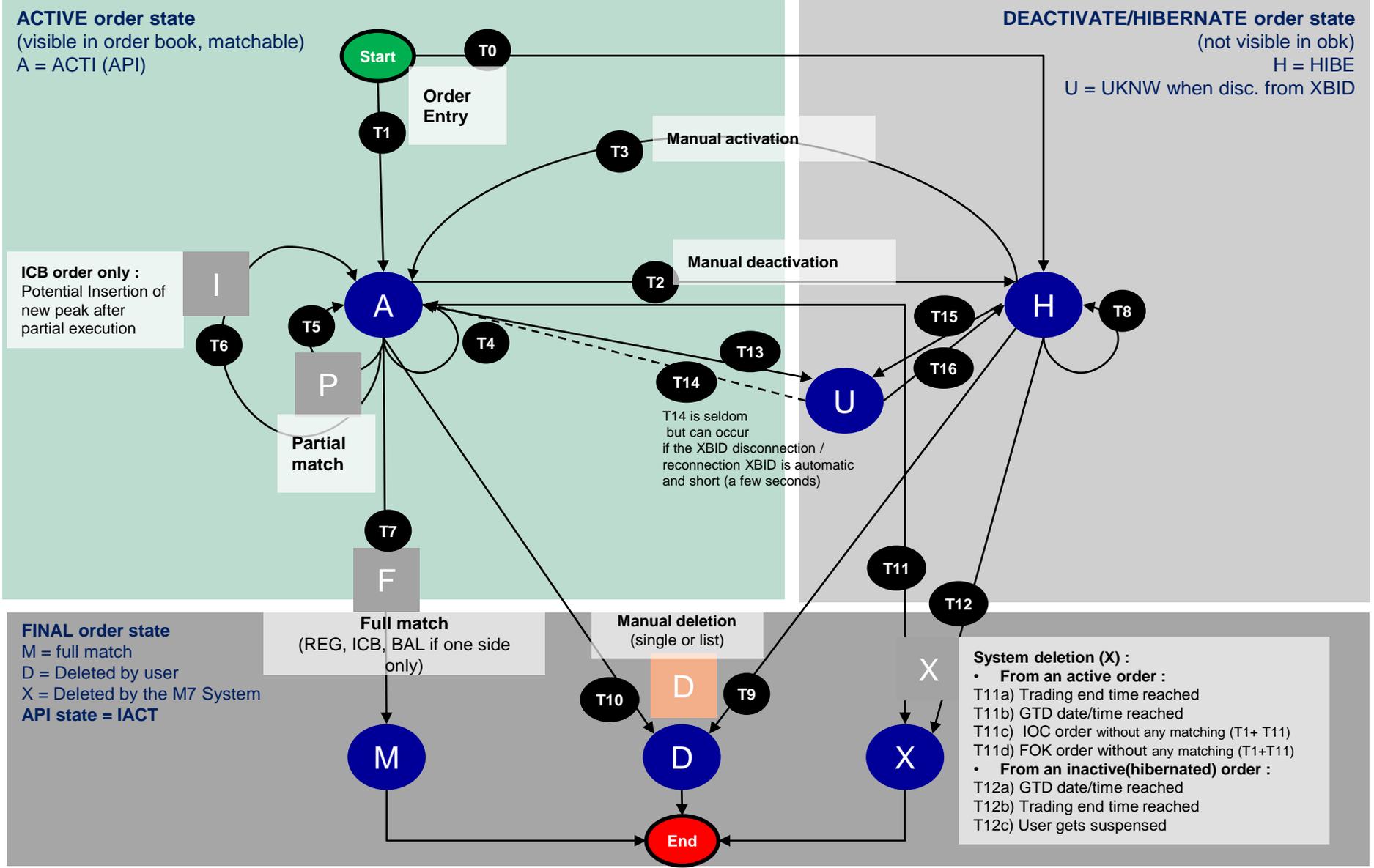
## Order management : which actions?

Action	Description
<b>Entry</b>	creation / submission of an active or deactivated order (1 to 100)
<b>Modification</b>	a subset of order characteristics can be modified (price, quantity, etc) (1 to 100)
<b>Deactivation</b>	(hibernation): removes the order from order books and from being executable
<b>Activation</b>	order gets displayed in order books and gets executable
<b>Cancellation</b>	Deletion by users or M7 system (e.g. contract expiry, GTD is reached)

# Order state diagram

## Exchange Orders

<span style="color: blue;">●</span> Order state	<span style="background-color: #f4a460; border: 1px solid black; display: inline-block; width: 15px; height: 10px;"></span> User triggered Order management action
<span style="border: 1px solid black; border-radius: 50%; padding: 2px;">Tx</span> Transition reference number	<span style="background-color: #cccccc; border: 1px solid black; display: inline-block; width: 15px; height: 10px;"></span> System triggered Order management action



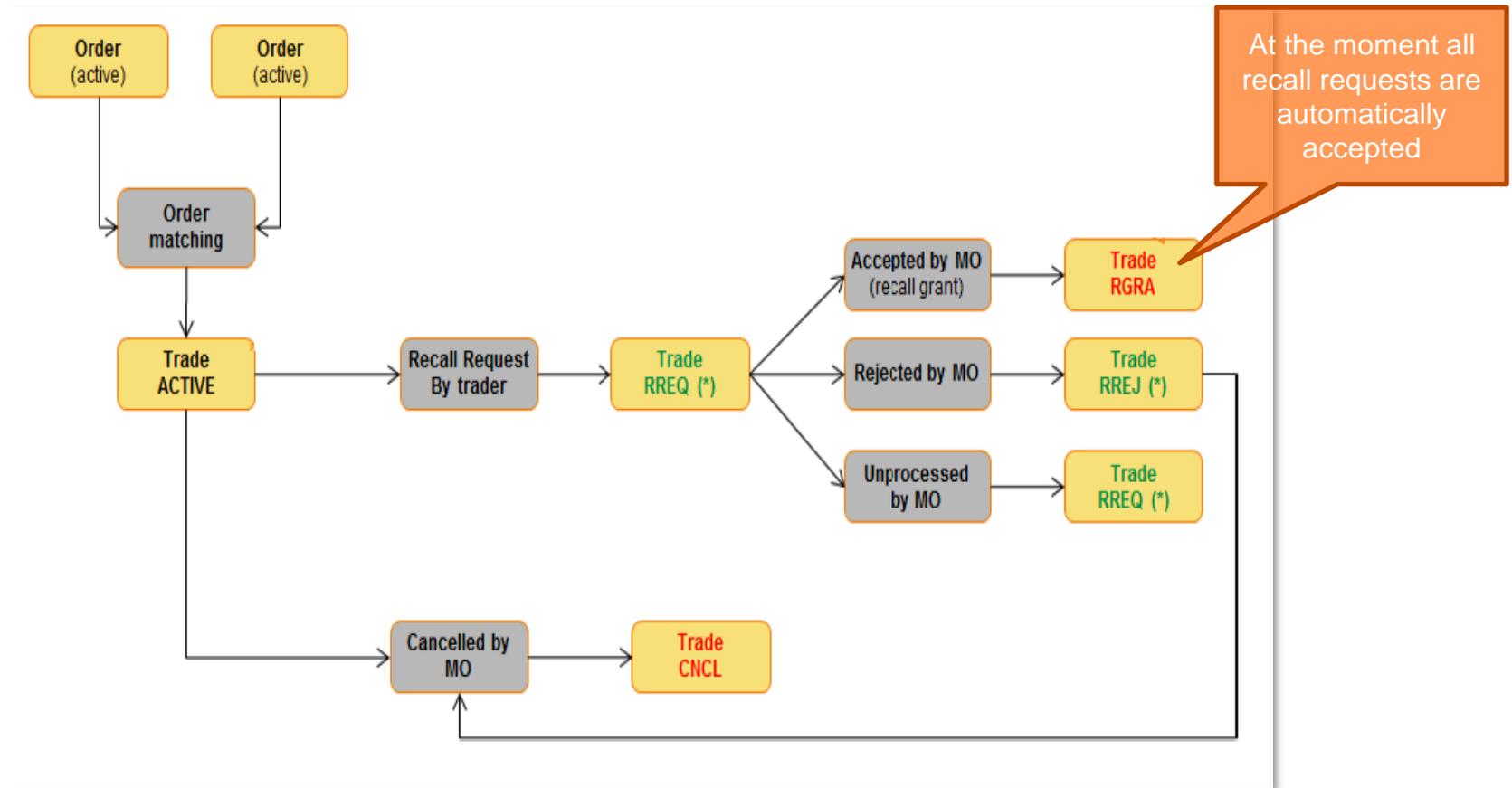
## OTR (Order-To-Trade Ratio) and API

- The M7 API does not calculate the OTRs.
- EPEX publishes on members FTP server the OTR the next day after contract closure.
- **More information in the M7 FAQ**
- **Reminder of EPEX Market Rules – Code of conduct:**
  - **§11 : Manipulation of the OTR:**

EPEX SPOT SE has implemented Intraday Trading System usage fees. The calculation of fees is based on the computation of the order-to-trade ratio (“OTR”), as described in EPEX SPOT Operational Rules. Exchange Members undertake not to manipulate the OTR by placing Orders and executing Transactions so as to reduce their Trading System usage fees. Notwithstanding the application of the Code of Conduct, EPEX SPOT SE reserves the right to exclude manipulated Transactions from the statistics. This rule applies to EPEX SPOT continuous Market Segments as defined in the paragraph related to the Intraday Trading System usage fees in EPEX SPOT Operational Rules.

# Trades state diagram

- **Trade status:**
  - The **recall process** can only be initiated **by Traders**
  - Trade **cancellation** can only be done **by Market operations**.

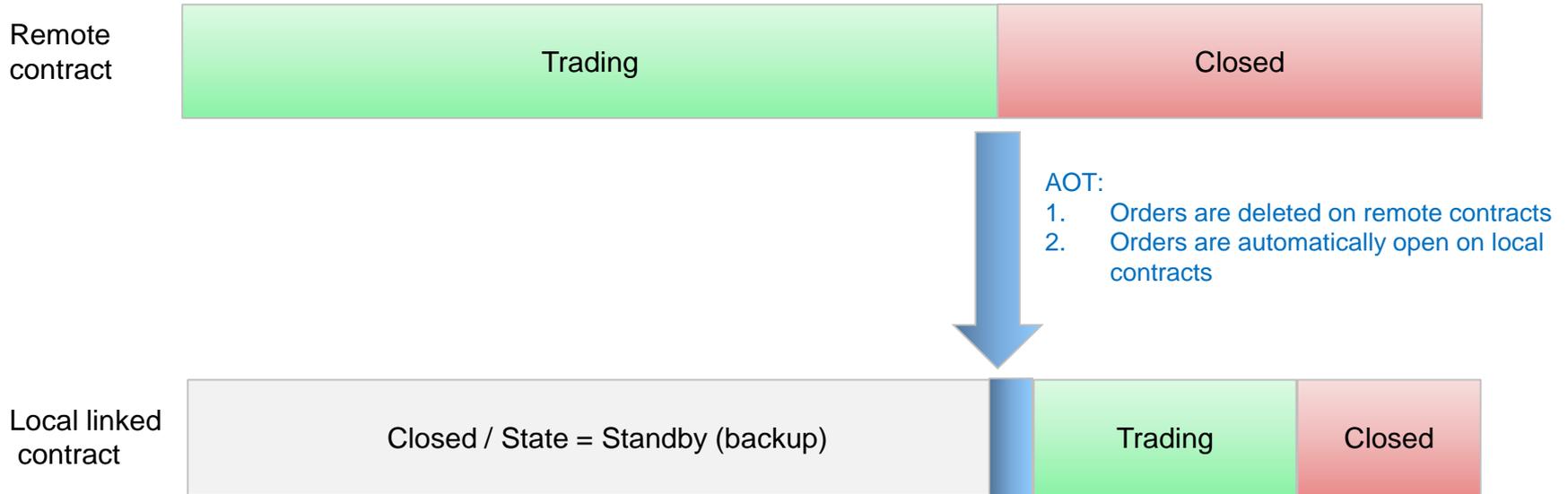


## 5. M7 Roadmap

# M7 Roadmap

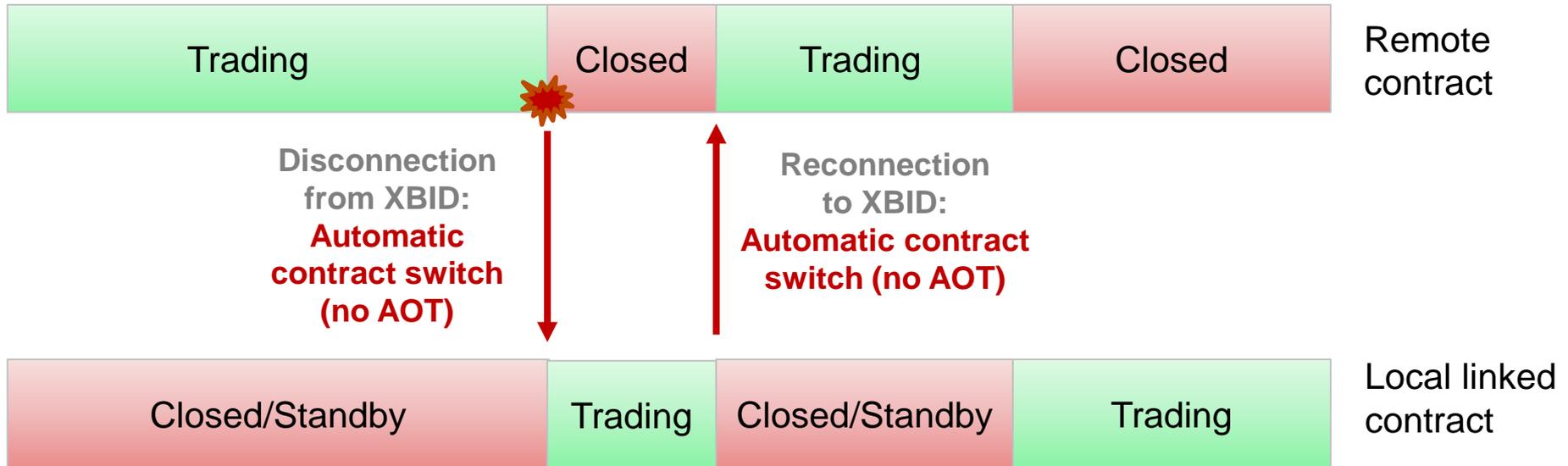
M7 release	Planned Go-Live	Content
<b>M7 6.8</b>	25 Feb 2020	<ul style="list-style-type: none"> <li>• <b>Cross trade flagging functionality</b> (new attribute in Public Trade msg)</li> <li>• <b>Internal stability and performance improvements</b> (no impact on customers API implementations)</li> <li>• <b>Java 11 ComTrader upgrade</b></li> </ul>
<b>M7 6.9</b>	15 July 2020	<ul style="list-style-type: none"> <li>• <b>New Unknown state for remote orders</b> when M7 disconnects from XBID</li> </ul>
<b>M7 6.10</b>	8 Dec. 2020	<ul style="list-style-type: none"> <li>• API « Send Request » panel in ComTrader (for testing purposes)</li> <li>• <b>Significantly faster responses times to inquiry requests</b></li> </ul>
<b>M7 6.11</b>	Q2 2021	<ul style="list-style-type: none"> <li>• Accompany the market activity exponential growth - Performance enhancements</li> <li>• <b>Private response queues :</b> <ul style="list-style-type: none"> <li>• <b>Automatic expiry (deletion) after 3 minutes</b> without a consumer (to avoid zombie queues)</li> <li>• New <u>optional</u> private response <b>queue naming convention</b> to limit their number to a maximum of ten. Will become mandatory only with M7 6.12.</li> </ul> </li> <li>• <b>Broadcast queues Time To Live (TTL) reduction:</b> automatic expiry (deletion) after 3 minutes without a consumer (instead of 6 mn today) to match the messages TTL in the queue (3mns). Recommendation to register a consumer as of login is done (reception of a User Report), before sending starting inquiry request (to avoid losing broadcasts that might have been stored during an unexpected disconnection period)</li> </ul>
<b>M7 6.12</b>	Q4 2021	<ul style="list-style-type: none"> <li>• Accompany the market activity exponential growth - Performance enhancements</li> <li>• <b>Mandatory: max 10 private response queues + new naming convention</b> (queues with other names cannot be created anymore)</li> </ul>
<b>M7 7.0</b>	No scheduled date	<ul style="list-style-type: none"> <li>• <b>New HTTP API (web sockets for broadcasts)</b> in parallel of the AMQP API</li> <li>• The AMQP API will not decommissioned.</li> </ul>

# Automatic Order Transfer (AOT/Bid Elevator)



AOT applies	AOT does not apply
At usual remote contract expiration	Any planned XBID maintenance
	Any unplanned XBID maintenance
	From local products to remote products

# Automatic switch of local contracts



At each contract switch:

- Remote and local contracts change of phase and state
- Orders of the closing contract are deactivated
- Existing deactivated orders are not reactivated automatically

## 6. EPEX Support organization

# API Customer support organization

- **24/7 Support from Market operations**
  - One central contact e-mail: [powerspot@epexspot.com](mailto:powerspot@epexspot.com)
  - Continuous Intraday Hotlines
    - FR: +33 1 73 03 77 00 / DE: +49 341 21 56 234
    - NL: +31 20 305 5079 / UK: +44 207 220 3444
  - Contact for Production issues
  - Operate API Conformance test in ASIMU (business hours)
  - Operate the SIMU and ASIMU environments:
    - User requests, password reset, connectivity issues, etc.
    - on demand: market halt, data in order books, etc.
- **API Technical Key Account Manager support:**  
[marketdata.technical@epexspot.com](mailto:marketdata.technical@epexspot.com) +33 1 73 03 61 81
  - Support along your API app implementation and test process
  - API Webinars, Workshops, bilateral meetings on demand
  - Only Business hours

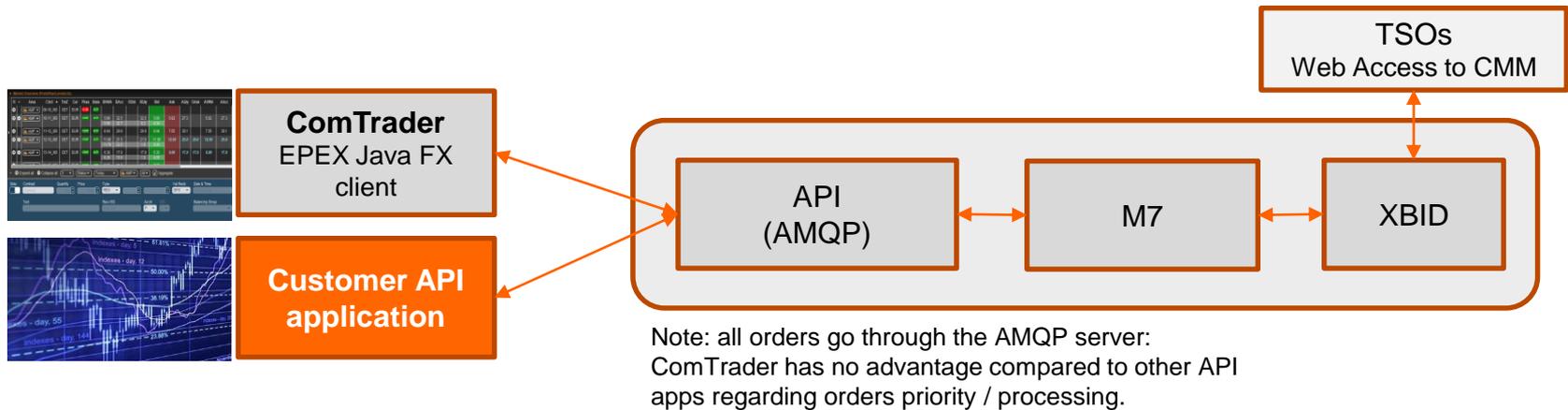
END OF PART 1

## Part 2

# 1. M7 API Technical Overview

# M7 PMI Introduction - AMQP protocol

- The M7 **Public Message Interface** allows clients to communicate with the backend system via a programmable interface.
- based on **Advanced Message Queuing Protocol (AMQP)** as the transport layer.



- AMQP:
  - Was originated in 2003 at JPMorgan Chase in London
  - is an open standard for **passing business messages between applications or organizations**
- The AMQP server is currently using the **AMQP implementation from RabbitMQ** (Erlang language)

# What is mandatory to be implemented?

In order to properly operate an API application (RO or R/W), you need the following « bricks » (on top of the functional content you need):

- **Ensure that your implementation respects our Terms of Reference**
- **Implement the Client Failover**
- **Implement the Starting sequence described in the slides**
  - From the AMQP connection, channels creation, private response Q creation,
  - Respect the Inquiry requests rate limits per minute and per hour
- **Manage all events leading to a recovery procedure:**
  - AMQP shutdown signal (can be caused by a network glitch)
  - M7 heartbeat loss (3 missing heartbeats in a row)
  - Gaps: Gap detection mechanism + data re-inquiry procedure
  - Reception of a Logout report (because of a concurrent access or if logged out by M7)
  - Inconsistent Revision No (see FAQ)
- **If your application submits orders:**
  - Ensure it **respects the Order-to-Trade Ratio (OTR)**, to avoid extra fees.
- Once you have fully tested your API app on the ASIM env., **register for an API conformance test 2 weeks in advance**

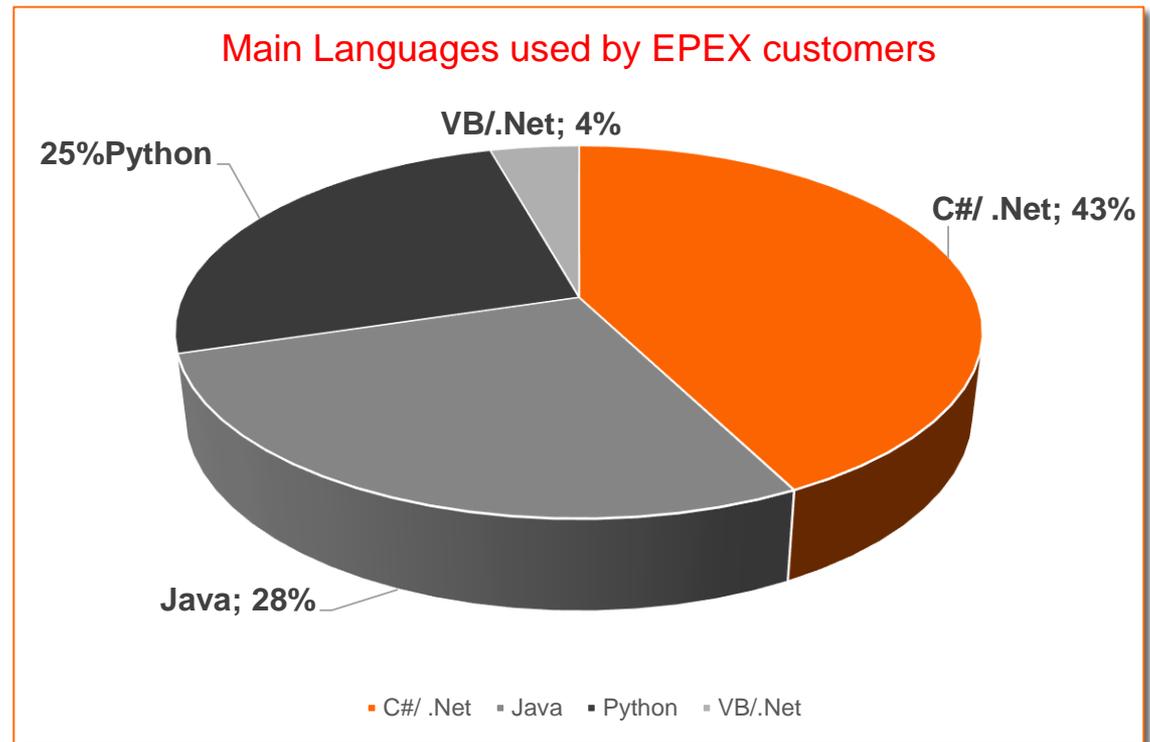
*Please check our DFS180 specs and the M7 API FAQ to see more detail which recovery procedure should be implemented*

# AMQP Rabbit MQ – supported languages

See <https://www.rabbitmq.com/devtools.html>:

## Languages:

- Java and Spring Framework
- .NET
- Ruby
- Python
- PHP
- Objective-C and Swift
- Java script
- C / C++
- ...



## AMQP Rabbit MQ – supported platforms

The following platforms are supported by Erlang and could therefore run RabbitMQ:

Linux

Windows, NT through 10

Windows Server 2003 through 2016

Mac OS X

Solaris

FreeBSD

TRU64

VxWorks

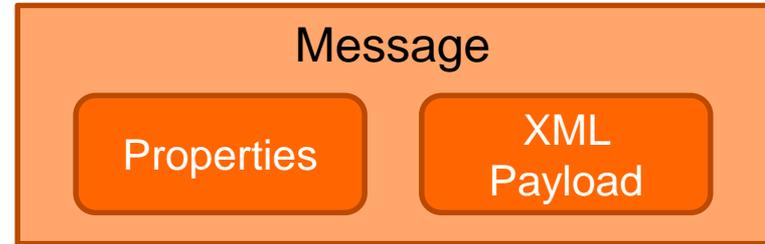
# Communication type and Request types

There are 2 kinds of communication between clients and backend:

- **Management and inquiry request:**
  - client issues a request and waits for a response from the backend system
- **Broadcast messages** sent by the backend to all or specific clients

Communication type	Description
<b>Inquiry Request</b>	<ul style="list-style-type: none"> <li>• <b>to obtain information on the current state of the market or reference data.</b></li> <li>• should only be used at the start of a new session to obtain an initial view of the market or when recovering from communication failures.</li> </ul>
<b>Management request</b>	<ul style="list-style-type: none"> <li>• <b>used to enter, modify or delete orders or trades</b></li> <li>• <b>Or to change the connected API user password</b></li> </ul>
<b>Broadcast</b>	<ul style="list-style-type: none"> <li>• the backend system <b>publishes notifications</b> that are:           <ul style="list-style-type: none"> <li>• <b>public:</b> addressed to all traders</li> <li>• or <b>private:</b> only receivable by privileged traders (see Routing Keys).</li> </ul> </li> <li>• <b>initiated by the backend system</b> (e.g. contract opening, order book change)</li> <li>• <b>Sequence counting:</b> (recovery mechanism to be implemented on client side)           <ul style="list-style-type: none"> <li>• to identify the order of the broadcasts</li> <li>• and to find out if some broadcasts have been lost (“real Gap”)</li> </ul> </li> </ul>

# Message structure



- All messages have:
  - An XML-encoded payload
  - Specific AMQP properties: it is not required to open the XML part to access those properties

AMQP Message Property	Description
content-type	Contains information about the used XML payload version as well as the used message type. Valid content-type definitions are (version number has to be filled with the used version): <ul style="list-style-type: none"> <li>▪ x-m7/request; version=x (Used by the clients when sending requests)</li> <li>▪ x-m7/response; version=x</li> <li>▪ x-m7/broadcast; version=x</li> <li>▪ x-m7/heartbeat; version=x</li> <li>▪ x-m7/error; version=x</li> </ul>
reply-to	contains the predefined trader's queue name a response has to be sent to (See 3.1)
user-id	contains the login-id of the logged in trader
app-id	contains the application id given by the granting authority
correlation-id	contains the request message id generated by each client
expiration	contains an optional entry specifying if the request should be deleted if not executed within the specified time
contentEncoding	contains <b>gzip</b> , if messages are compressed (content is encrypted using gzip method); property is null if messages are not compressed
header	can contain connecting application version in format (optional) <b>app-version</b> :version where version must be in integer(s) format optionally separated by dots (ie. 10 or 4.1.5)

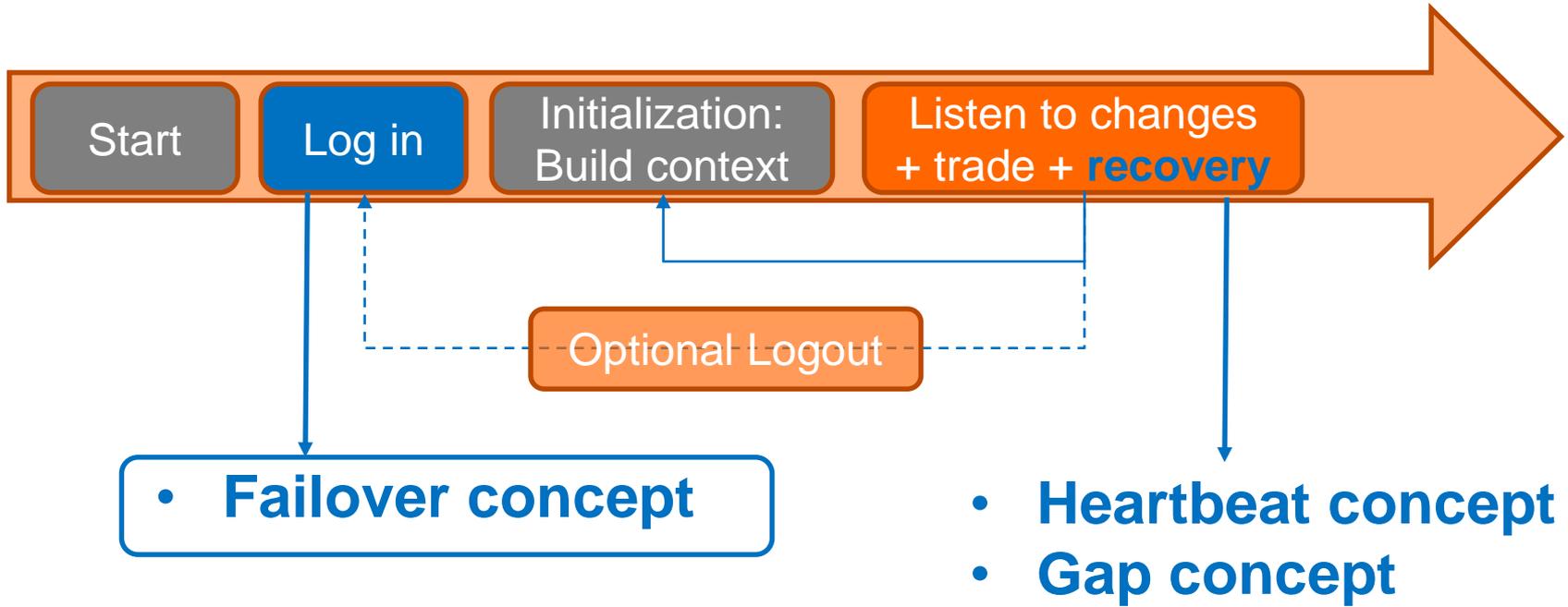
## API XML schema version

- Every message sent or received by the backend system must specify in its metadata the **XML schema major version** that was used to encode its payload (v6).
- This information is stored in the message properties in the contentType field as a string.

*Example: For XSD schema version 6.5.3 the correct contentType = 6.0 because the major schema version in this case is 6.0.*

- M7 new versions:
  - EPEX always supports 2 versions:
    - *when 7.0 is introduced in 2020, latest 6.x will still be supported*
  - You do not necessary need to upgrade to the latest API schema version
    - *e.g. 6.7 only required if using AOT*

# Key technical concepts (1/3)



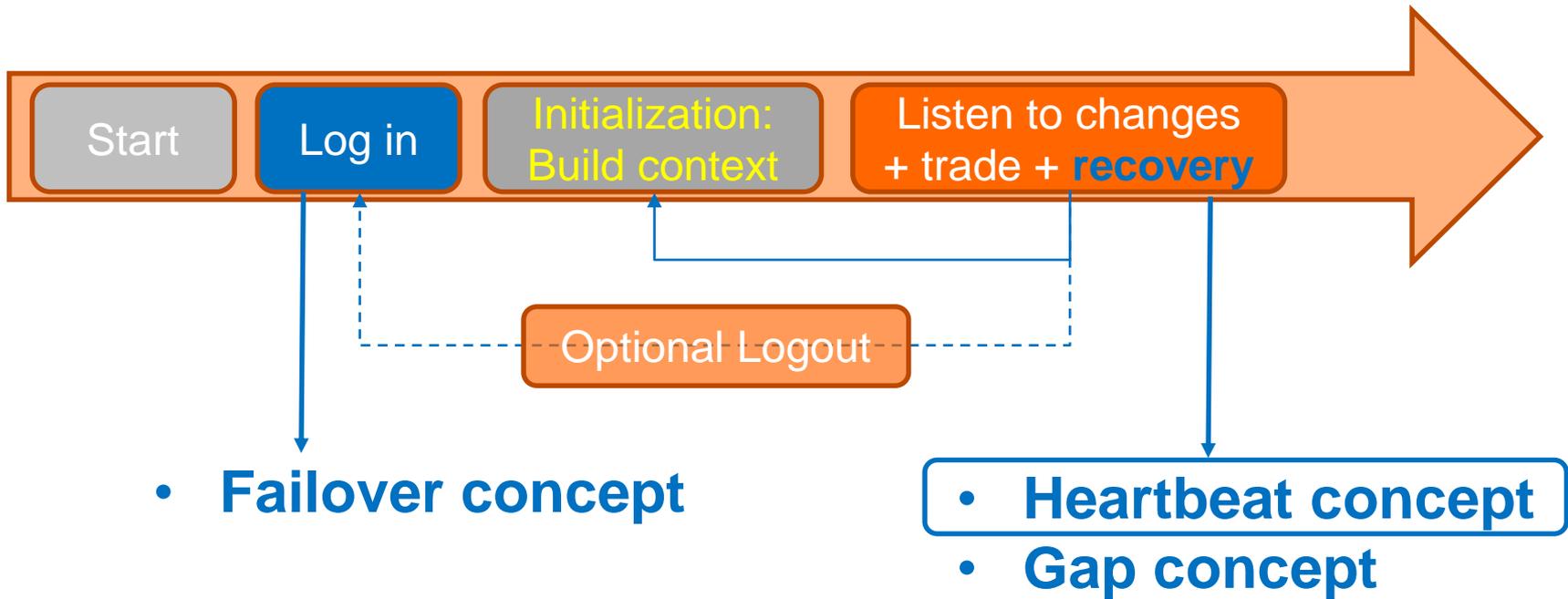
## Failover implementation

- Access to the AMQP server is possible via **2 different data centers**:
  - **In standard operating phase, both data centers are active and can be used to connect.**
- **The URL or IP address is the only difference between both ways to connect**
  - all other connection details (port, username, password, client certificate) stay the same.



- **Failover needs to be implemented on client side:**
  - In case the connection through the first URL is not possible, the client needs to try to connect to the second URL.

## Key technical concepts (2/3)

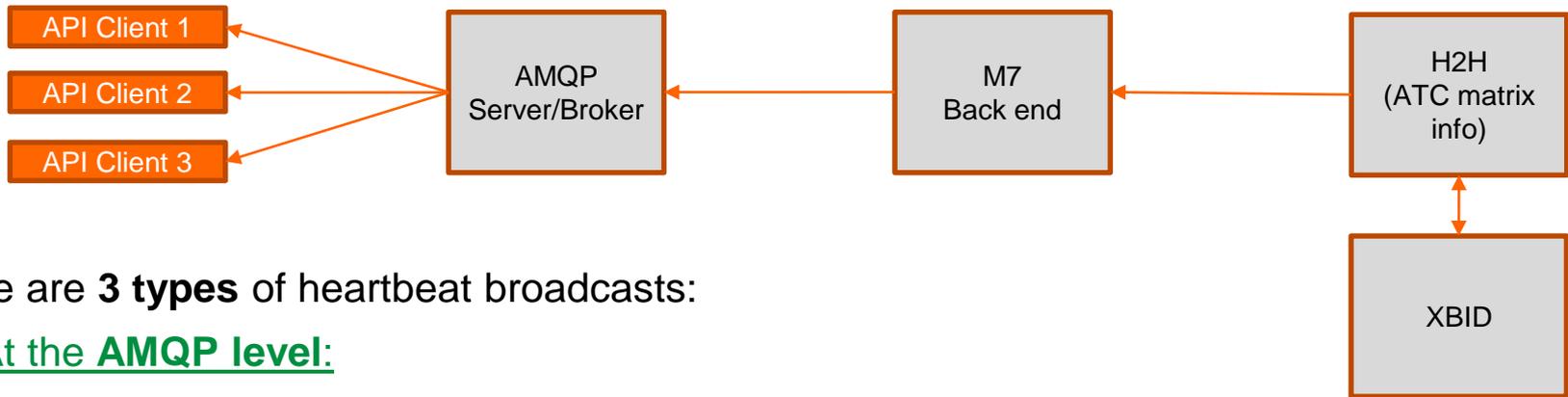


# Heartbeats 1/2

**AMQP heartbeats (30-60s):**  
« still connected to AMQP »

**M7 heartbeats (every 5s):**  
« M7 Still alive »

**H2H heartbeats (5s)**  
« H2H Still connected to XBID »



There are **3 types** of heartbeat broadcasts:

1. At the **AMQP level**:

- defined by the AMQP protocol specification (timeout / interval)
- allows the client to monitor the existence of a connection client-AMQP server.
- Need to reconnect to the broker when lost

2. At the **M7 level**:

- **allows the API client to monitor the availability of the backend system.**
- **Heartbeats are independent from other broadcasts, and should be monitored independently**

3. At the H2H Capacity level:

- informs that the Hub-to-Hub module is connected to XBID and running.

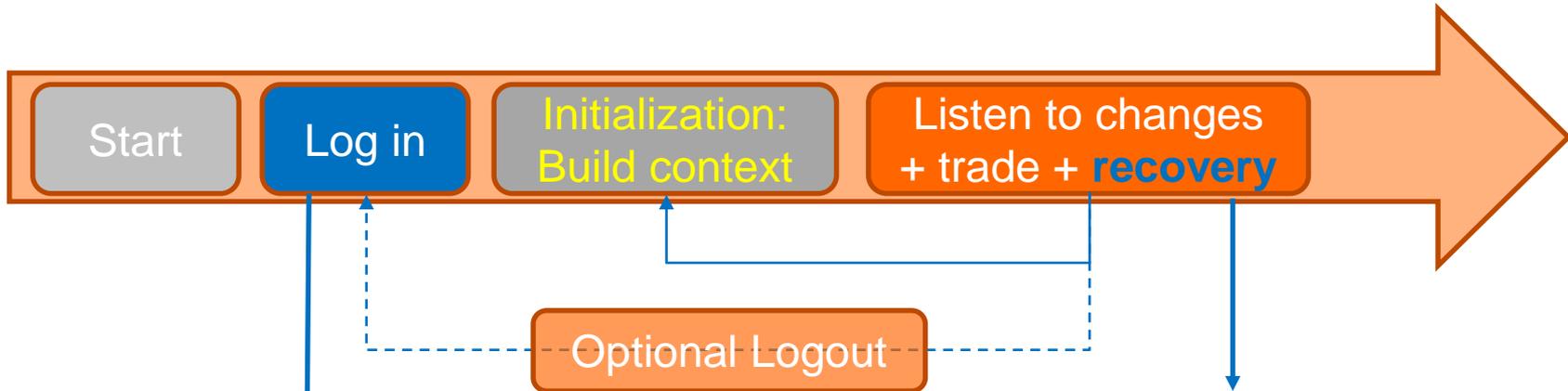
## Heartbeats 2/2

What should be done when your application detects an **M7 heartbeat loss**?

When our API app **misses 3 heartbeats in a row**:

- 1. raise a functional alert** (email to IT and operations team):
  - if your operations are aware of an M7 maintenance window, maybe can they stop your application and restart it as soon as EPEX market ops announce the end of the maintenance window?
- 2. Try to log out**
- 3. Try to close all AMQP objects**
- 4. Restart the app** (or just restart the connection + login procedure), **with an exponential back off** mechanism, **keeping in mind that the Login Inquiry request rate limit per hour (= 70) should not be hit**
  - Back off example: wait for 10s before the 1<sup>st</sup> start, then 20s, 40s, then every minute

# Key technical concepts (3/3)



- **Failover concept**

- **Heartbeat concept**
- **Gap concept**

• **Routing Keys**

• **Broadcasting rules**

• **Broadcasting Queue**

• **Queue basics**

# AMQP Queues basics

AMQP concepts:

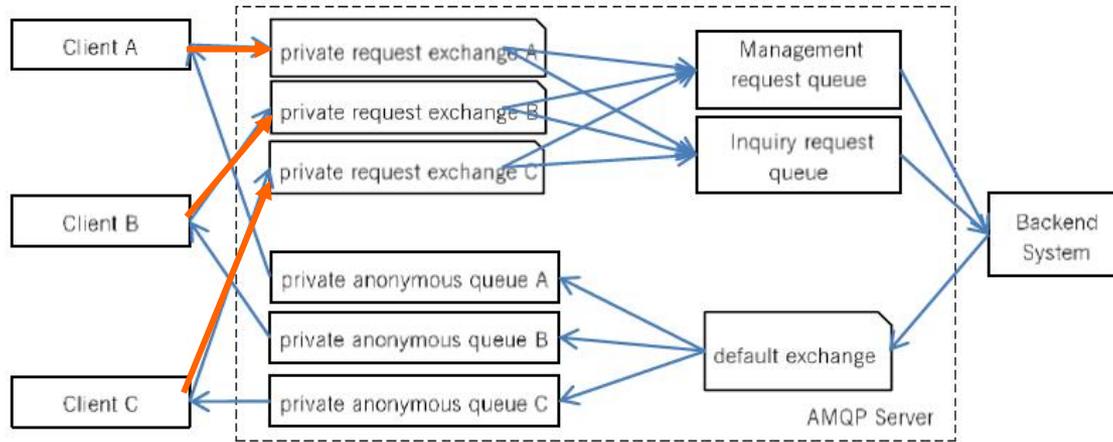
- The “**exchange**” receives messages from publisher applications and routes these to “message queues” based on arbitrary criteria, usually message properties or content
- • The “**message queue**” stores messages until they can be processed by a “consuming” application
- NB: messages have a Time To Live value =180 seconds in the M7 broadcast queue

**AMQP PRINCIPLE:** messages must be:

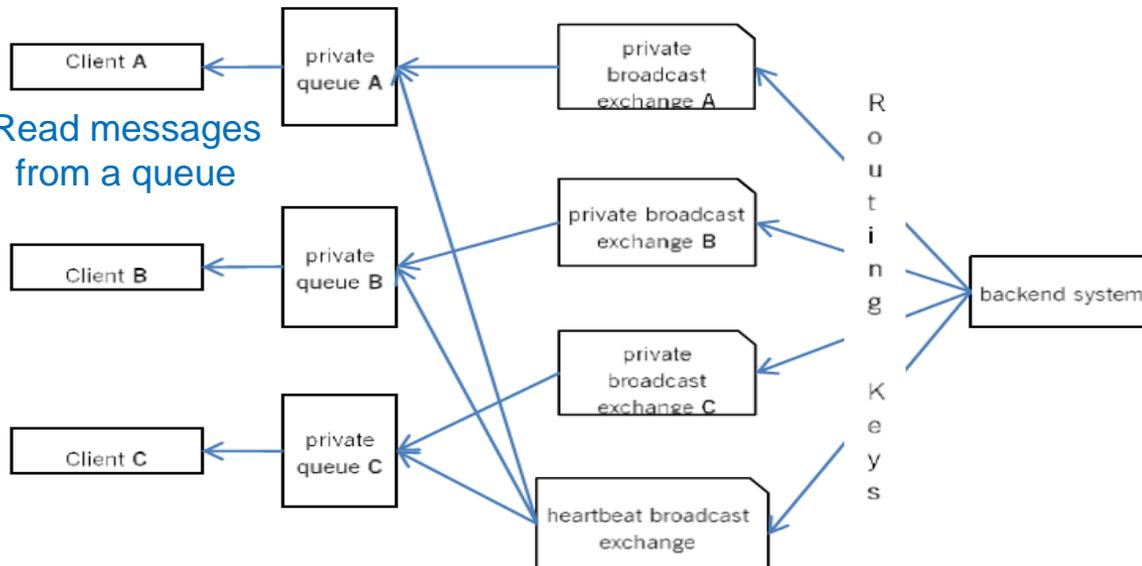
- **sent to an Exchange**
- **read from a Queue**

# AMQP Queues basics

Requests:  
sent to an  
exchange



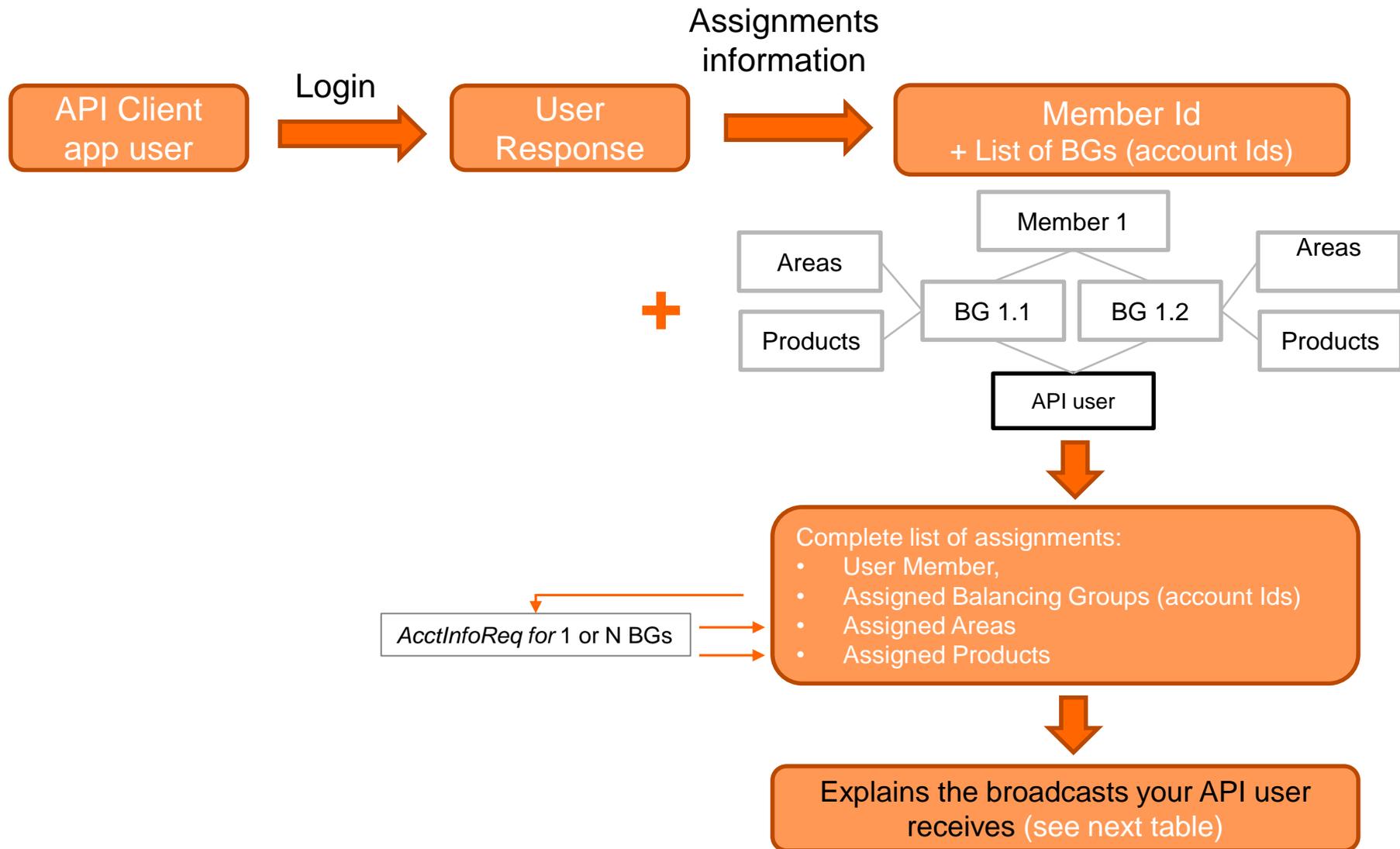
Read messages  
from a queue



Broadcasts  
TTL = 180s

# Information distribution rules: « Broadcasting rules »

An API user receives broadcasts based on its assignments



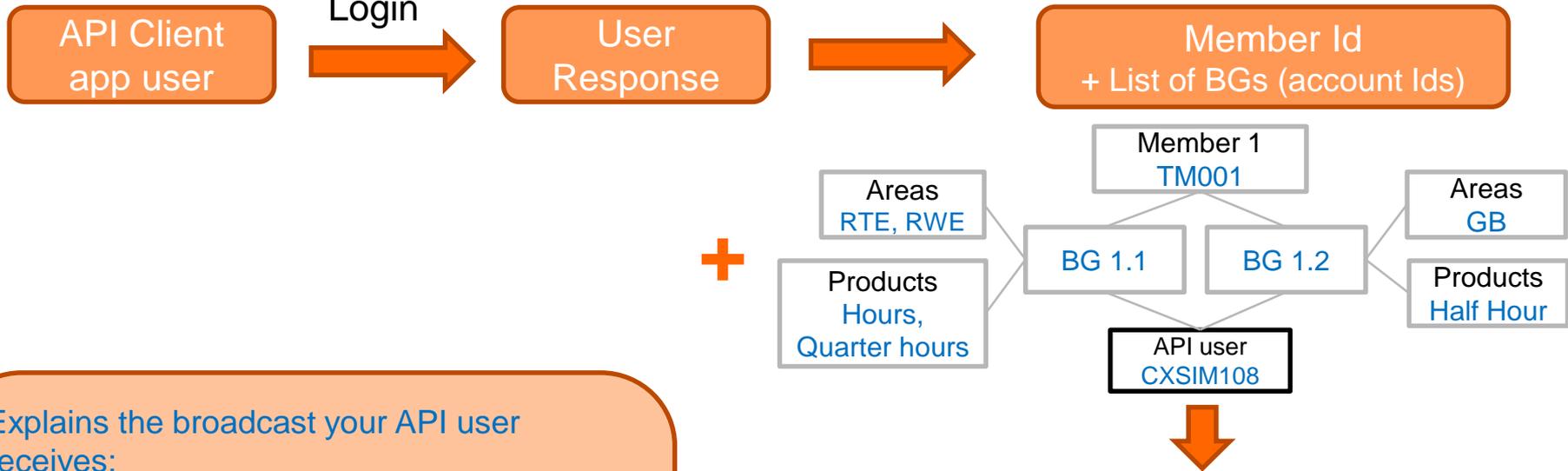
# Information distribution rules: « Broadcasting rules »

An API user receives broadcasts based on its assignments

e.g. CXSIM108

Assignments information

e.g. mbrld = TM001  
AssgAcctId = BG 1.1, BG 1.2



Explains the broadcast your API user receives:

- e.g.
- Trading limits for TM001
  - Order Execution Report for BG1.1 and BG1.2
  - Trade Capture Reports for BG1.1 and BG1.2
  - Public Trades for Hours, QH, HH
  - Order books for RTE.Hours, RWE.Hours, RWE, Quarter hours, GB.Half Hours
  - Etc.

Complete list of assignments:

- User Member = **TM001**
- Assigned BGs = **BG1 (11XTM1-TM-----1), BG2 (11XTM1-TM-----2)**
- Assigned Areas = **RTE, RWE, GB**
- Assigned Products = **Hours, Half Hours**

# Information distribution rules: « Broadcasting rules »

## Routing Keys

- **Broadcasting rules are implemented via “Routing Keys”**
  - **Example with “Order Execution Report”:**

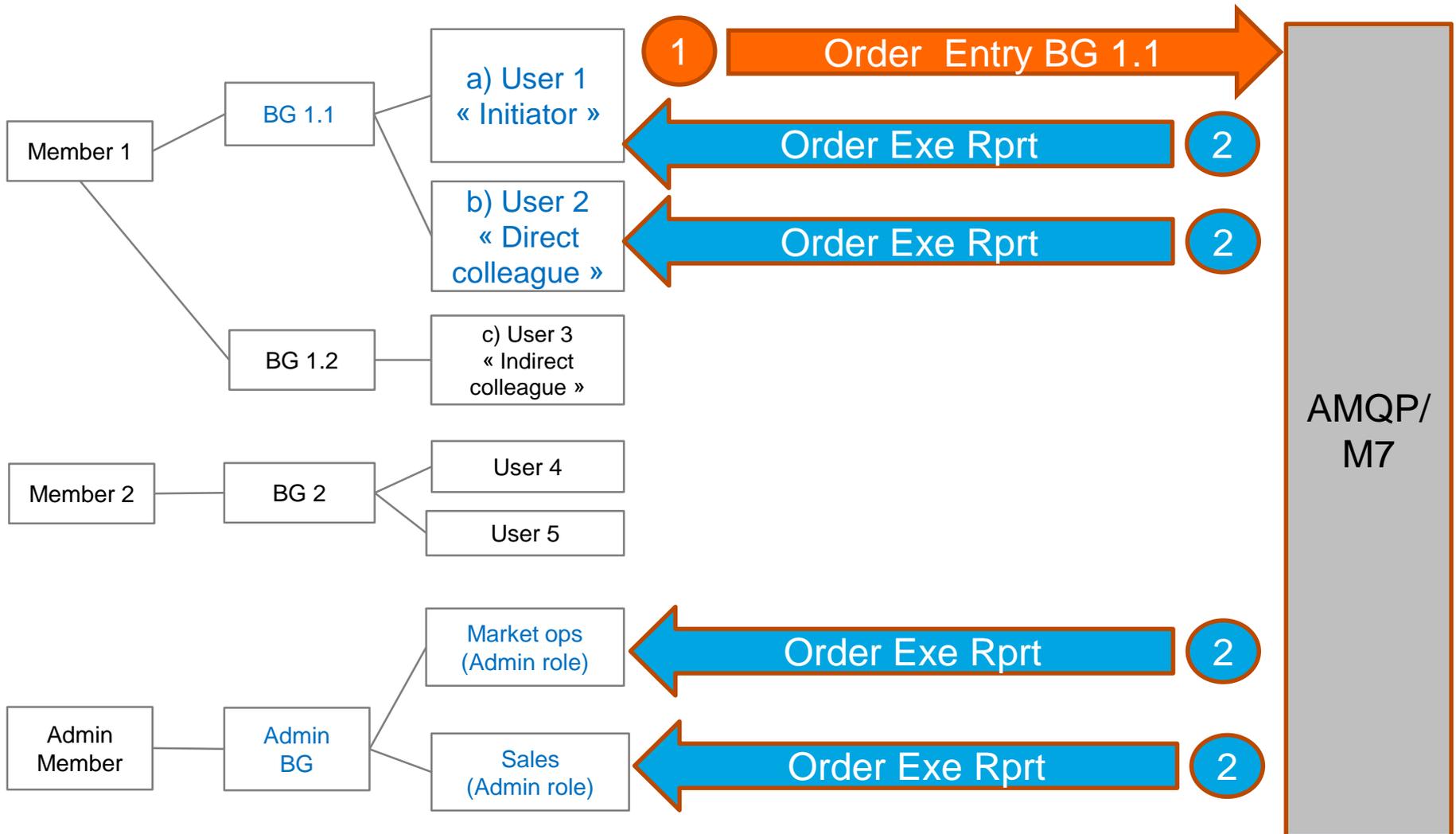
### 6.2.4 Order Execution Report (OrdExeRprt)

OrdExeRprt	
Type:	Management Response; Broadcast
Response to:	OrdEntry; OrdModify; OrdReq; ModifyAllOrders; (sent to private response queue see 3.1)
Broadcast:	Yes
Broadcast Routing Keys:	<code>&lt;schema-version&gt;.bg.&lt;acctId&gt;</code>
Broadcast audience:	Trader (owner of the order) and traders from his Balancing groups. Admins Brokers with assignment to Trader(owner of the order).
Roles:	Trader, Market Operation

# Where to find Broadcasting rules in DFS180?

Routing Keys: Order Execution report example

**RK = “Trader (owner of the order) and traders from his Balancing groups. + Admins »**



## Routing Keys List: public, product and area related (1/5)

Message	Generic Routing Key	Broadcasted to
MktStateRprt	<schema-version>.public	All users
MbrChangeRprt		
DivryAreaInfoRprt		
MktAreaInfoRprt		
AcctChangeRprt		
MsgRprt		
ProdInfoRprt	<schema-version>.prd.<prodName>	All users assigned to this product
ContractInfoRprt		
RefPxRprt		
MsgRprt		
PblcTradeConfRprt	<schema-version>.pblc.trd.<prodName>	All users assigned to this product
PblcOrdrBooksDelta Rprt	<schema-version>.prddlvr.<prodName>.<divryAreald> example: 6_0.prddlvr.GB_Hour_Power.10YGB-----A Is an instance of that generic routing key	Users assigned to this product and delivery area

## Routing Keys List: member, BG and user related (2/5)

Message	Generic Routing Key	Broadcasted to
MsgRprt	<schema-version>.mkt.<marketAreald>	All users assigned to the market area (via one of its delivery area)
MsgRprt	<schema-version>.dlvr.<dlvryAreald>	All users assigned to the delivery area
MsgRprt	<schema-version>.mbr.<mbrld>	All users of the member
UserRprt		
CashLmtRprt		
CashLmtDeltaRprt		
OrdExeRprt		
MsgRprt	<schema-version>.prvt.bg.msg.<acctld>	All users assigned to the BG
TradeCaptureRprt	<schema-version>.hlfrtd.<acctld>	All users assigned to the BG of the own order that got traded
LogoutRprt	<schema-version>.trdr.<login-id>	The connected API user
ErrResp		The API user who sent the request

## Routing Keys List: member, BG and user related (3/5)

Message	Generic Routing Key	Broadcasted to
M7 Heartbeat	<schema-version>.m7.heartbeat	
HubToHubHeartbeat	<schema-version>.hubToHub	M7 6.6
HubToHubNtf		M7 6.6

# Routing Keys List and GAP sequence numbers (4/5)

## Examples of x-m7-group-sequence & x-m7-group-id

### Example with M7 heartbeats

Gap monitoring  
via sequence nb

Routing Key

- 2019/06/25 19:56:33
  - getProperties = #contentHeader<basic>(content-type=x-m7/heartbeat; version=6.0, content-encoding=gzip,
  - headers={**x-m7-group-sequence=193825, x-m7-group-id= 6\_0.m7.heartbeat**, server-timestamp=1561485353058}, delivery-mode=1, priority=0, correlation-id=null, reply-to=null, expiration=null, message-id=null, timestamp=Tue Jun 25 19:55:53 CEST 2019, type=null, user-id=null, app-id=null, cluster-id=null)
  - XML = SYSTEM\_ALIVE:5000
  
- 2019/06/25 19:56:38
  - getProperties = #contentHeader<basic>(content-type=x-m7/heartbeat; version=6.0, content-encoding=gzip,
  - headers={**x-m7-group-sequence=193826, x-m7-group-id= 6\_0.m7.heartbeat**, server-timestamp=1561485358046}, delivery-mode=1, priority=0, correlation-id=null, reply-to=null, expiration=null, message-id=null, timestamp=Tue Jun 25 19:55:58 CEST 2019, type=null, user-id=null, app-id=null, cluster-id=null)
  - XML = SYSTEM\_ALIVE:5000
  
- 2019/06/25 19:56:43
  - getProperties = #contentHeader<basic>(content-type=x-m7/heartbeat; ...
  - headers={**x-m7-group-sequence=193827, x-m7-group-id= 6\_0.m7.heartbeat**, ...
  - XML = SYSTEM\_ALIVE:5000

## Routing Keys List and GAP sequence numbers (5/5)

### Gap detection: a gap is a discontinuity in broadcasts sequence nb

#### 2 attributes work in conjunction in the message header:

- ***x-m7-group-sequence*** : sequence number, counted per routing key
- ***x-m7-group-id*** : the routing key
- DFS180: In case of a gap (received sequence nb  $\neq$  last nb + 1) your app must send an inquiry request for all functional areas / objects that are used (e.g. Contracts + Products), and then deduplicate/ discard known data (using Revision No and IDs).
- **Sequence nbs are reset to 0 when M7 shutdowns or terminates** (e.g. in case of a maintenance window or disconnection from XBID).
- Even if you app is not disconnected, a gap recovery procedure should be launched when receiving 0.
- **Sequence nbs are per “instance” of RK:** for order book delta reports RK <schema-version>.prddlvr.<prodName>.<dlvryAreald>:
  - 6\_0.prddlvr.GB\_2\_Hour\_Power.10YGB-----A has a different sequence nb than 6\_0.prddlvr.GB\_Hour\_Power.10YGB-----A

# How to handle gaps

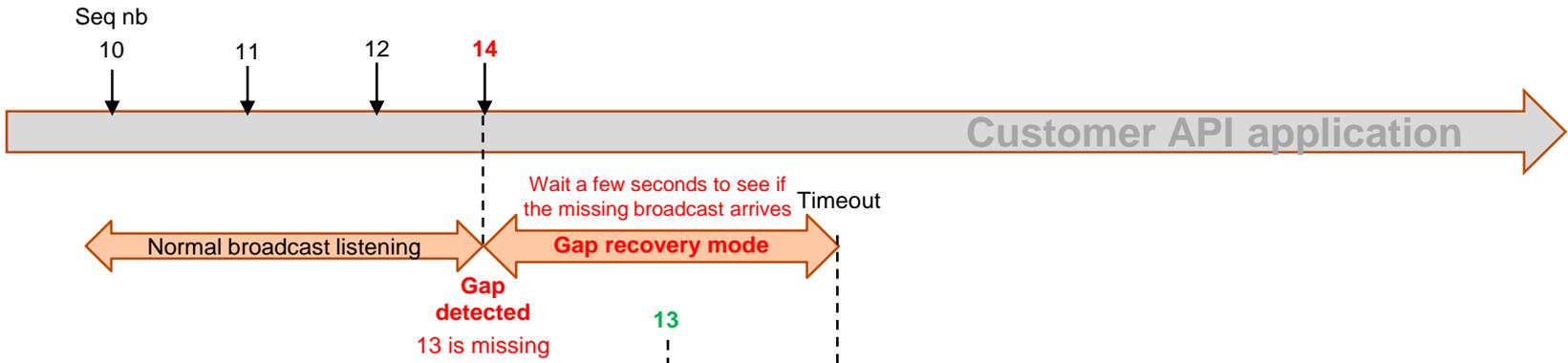
There are **3 types of gaps**:

- a) [new sequence nb for a given routing key] > [Last one + 1]: **“Real gap”**
  - *Example: 10, 11, 13*
- b) [new sequence nb for a given routing key] = 0 : **“Reset”**
- c)  $0 < [\text{new sequence nb for a given routing key}] < [\text{Last one} + 1]$ : **these gaps should be ignored**, that is the broadcast should not be processed (no functional added value)
  - *Example: 10, 11, 12, 11 => ignore 11*

**For a) and b) a recovery procedure must be implemented, compound of:**

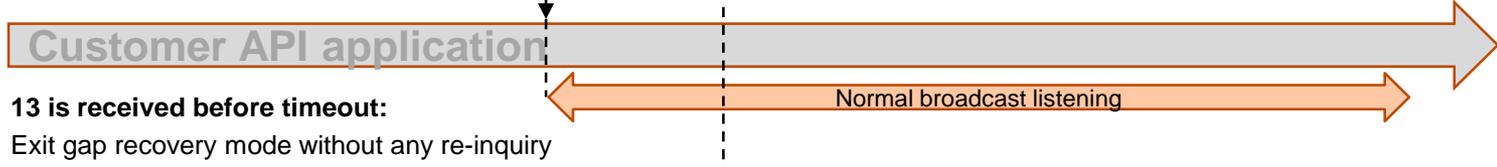
- **A gap recovery procedure:** wait for maximum a couple of seconds to see if the missing broadcast is just a bit late,
- If not received in the given timeline, launch **a re-inquiry procedure**:
  - You cannot request the sequence ID you missed.
  - The only possibility is to send inquiry requests for the RK on which a gap occurred.

# Seq Nb diagram – how to handle Gaps 1/2



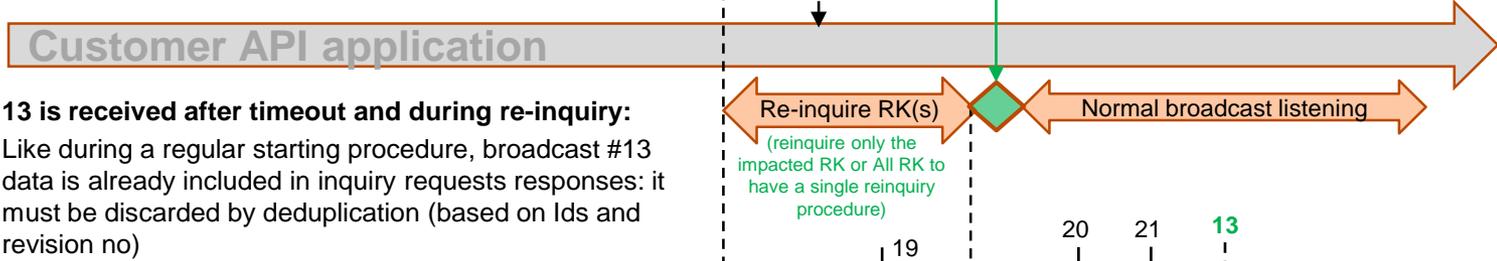
Case 1

Gap



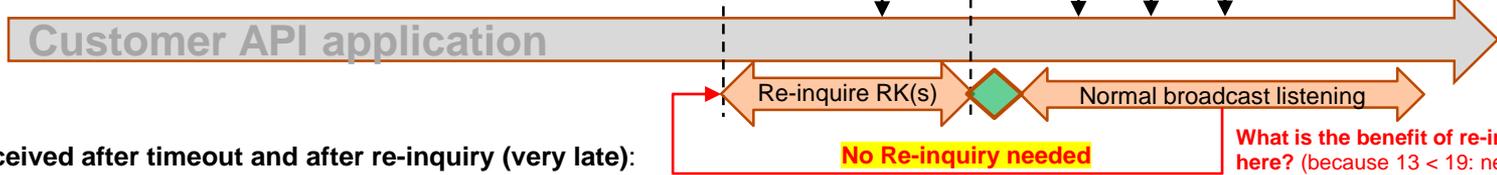
Case 2

Gap



Case 3

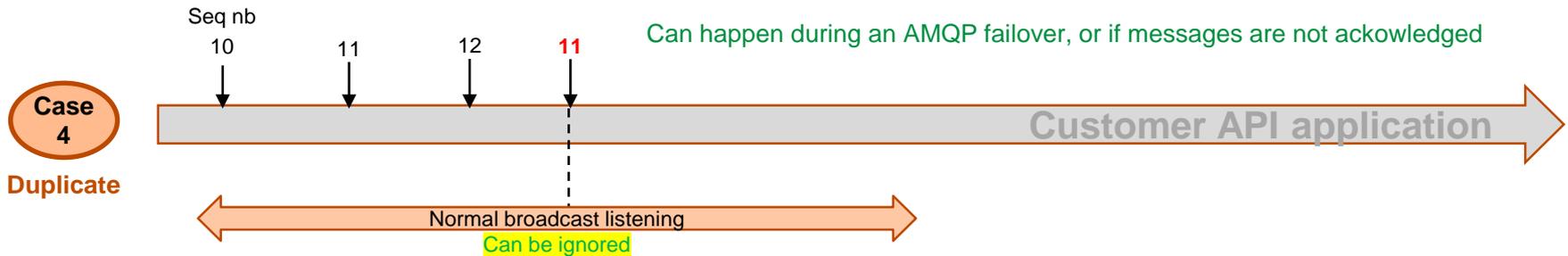
Gap



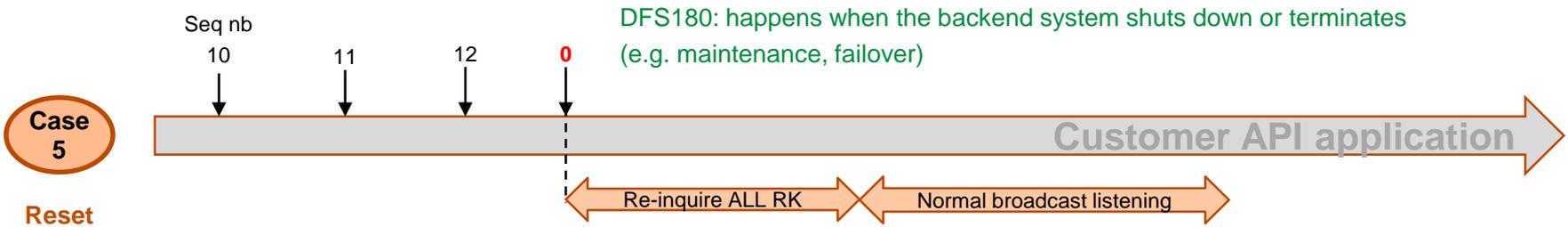
**What is the benefit of re-inquiring data here?** (because 13 < 19: never-received broadcast since last re-inquiry): **None**

In case of 6.0 launch (lots of delays in gap broadcasts) this can lead to « infinite » loops when 13 delay time is higher than [gap timeout + re-inquiry + deduplication time]

# Seq Nb diagram – Duplicate and Reset 2/2



11 was already received since as a broadcast since the last inquiry request: it should be ignored



Seq Nb gets reset for one Routing Key: this means that all Seq Nb have been reset for all RK

## Broadcast messages order and M7 6.6 / 6.7

- Sequence nb existence implies that messages are expected in a certain order, per routing key.
- **BUT there is no guarantee about messages reception order across several routing keys**
- M7 6.6 increased the situation where the sending order might change:
  - e.g. *Order Execution Report* received after *Public Order Books Delta reports*,
  - or e.g. *Order Execution Report* received after *Trade Capture Report*
  - Or any other message swap

# Broadcast messages order and M7 6.6 / 6.7

- if we take the messages is the following but can change:  
 some broadcasts can arrive **faster** or **later**, frequently enough to interfere with algorithmic trading in case of dependencies in your app :

Routing Key	Correlation ID	Request Type
6_0.prddlvr.XBID_Hour_Power.10YFR-RTE-----C	23465073-d573-4d9a-ad50-e30f28f9c668	PblcOrdrBooksDeltaRprt
6_0.prddlvr.XBID_Hour_Power.10YBE-----2	23465073-d573-4d9a-ad50-e30f28f9c668	PblcOrdrBooksDeltaRprt
6_0.prddlvr.XBID_Hour_Power.10YNL-----L	23465073-d573-4d9a-ad50-e30f28f9c668	PblcOrdrBooksDeltaRprt
6_0.pblc.trd.XBID_Hour_Power	23465073-d573-4d9a-ad50-e30f28f9c668	PblcTradeConfRprt
6_0.mbr.TM001	23465073-d573-4d9a-ad50-e30f28f9c668	CashLmtDeltaRprt
6_0.hlftrd.11XTM1-TM-----1	23465073-d573-4d9a-ad50-e30f28f9c668	TradeCaptureRprt
6_0.mbr.TM001	23465073-d573-4d9a-ad50-e30f28f9c668	CashLmtDeltaRprt
6_0.mbr.TM001	23465073-d573-4d9a-ad50-e30f28f9c668	CashLmtDeltaRprt
6_0.bg.11XTM1-TM-----1	23465073-d573-4d9a-ad50-e30f28f9c668	OrdrExeRprt
m7.private.responseQueue.CXSIM107.c7e7dd24-7e70-4dec-a28f-52	23465073-d573-4d9a-ad50-e30f28f9c668	AckResp
m7.request.management	23465073-d573-4d9a-ad50-e30f28f9c668	OrdrEntry

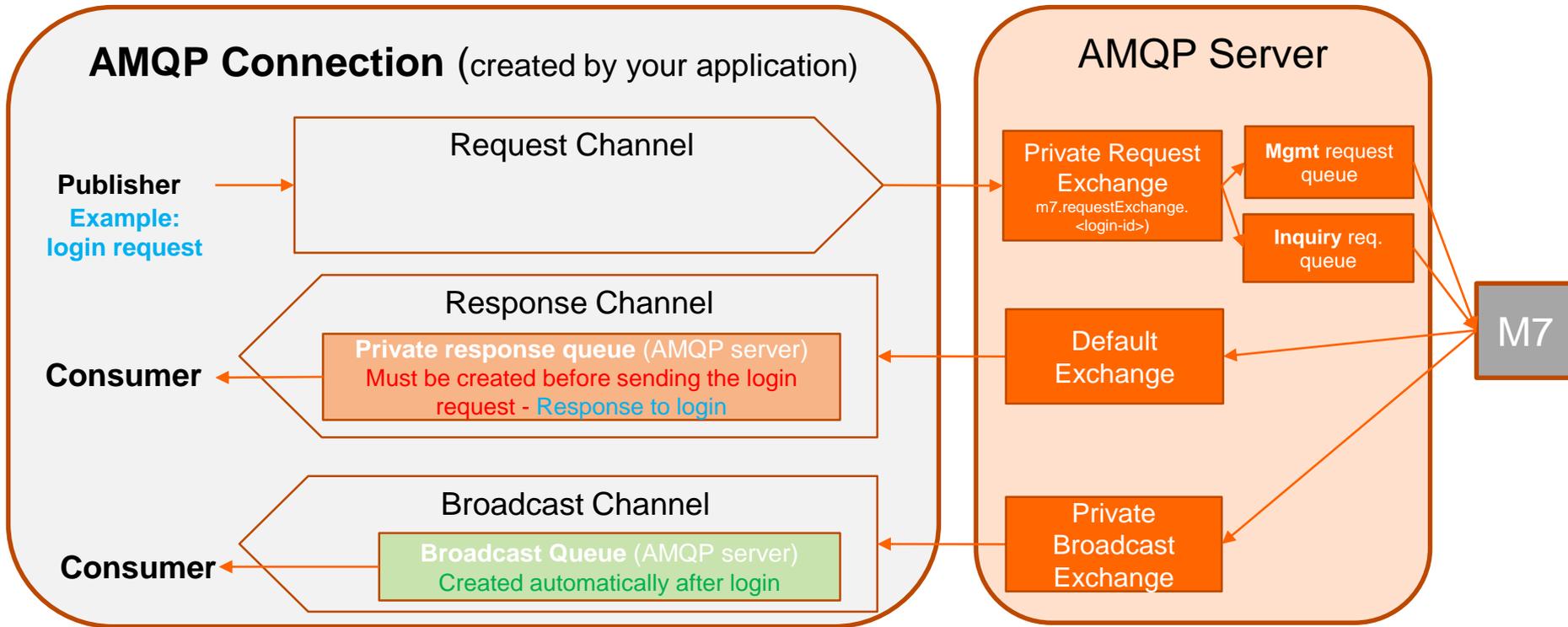


# Continuation from AMQP Queues basics

## AMQP objects:

- **Connections:** The first step every client program needs to take is to establish a connection to the AMQP server. **3 connections max per API user**  
Connection details for the ADV SIMU environment:
  - Exchange: **EPEX**
  - Rabbit Host: **advsimu1.epex.m7.deutsche-boerse.com** (or advsimu2....)
  - Rabbit User: **your M7 API user (CX....)**
  - Rabbit password: **your M7 API pwd**
  - Rabbit Vhost: **app**
  - Rabbit Port: **50240**
  - Application ID: **the app Id provided by market operations**
- **Channels:** used for communication between the client application and the backend server
  - can all share the same connection.
  - **5 channels max per connection (multi threading: channels cannot be shared)**
- **Queues:** stores messages until they can be safely processed by a “consuming” application
  - **1 private response queue per connection.**

# AMQP Queues basics



# Recurrent questions from customers

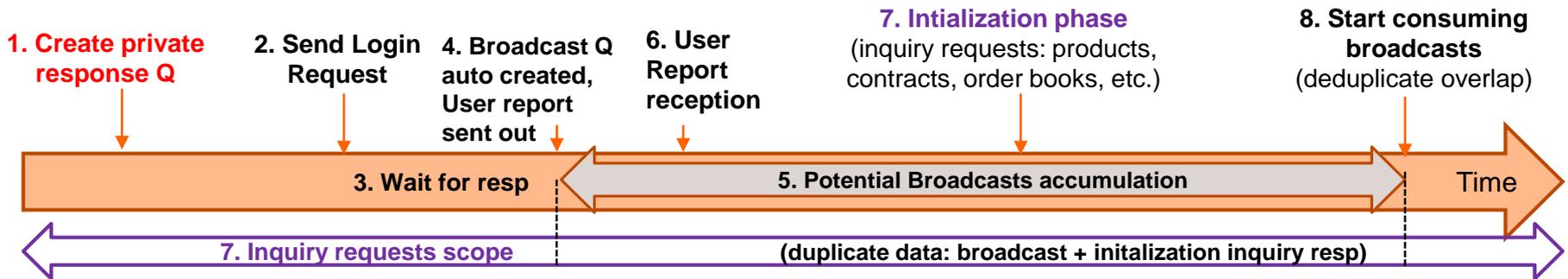
- **What is a proper login/start procedure?**
- **When should an application:**
  - Create queues? Which one?
  - Send inquiry requests?
  - Start consuming broadcasts?
  - Etc.
- **Should I avoid consuming broadcasts before sending my request?**
- **If I start consuming broadcast after the response can I lose a broadcast between my request and its response?**

# Key technical concepts



- **Proper login/start Procedure**

# Proper Login/start Procedure



- 1. Before logging in, the private response queue must be created** so that when sending the login request the software can get the response
  - named `m7.private.responseQueue.<login-id>.<unique-id>`
  - Must be created again only when the connection with the AMQP server is lost
- 2. Send the login request** to the M7 request exchange (`m7.requestExchange.<login-id>`)
- 3. Wait for a positive login response** (User Report message reception), or an Error response
- In case of successful login, the **broadcast queue is created automatically** (`m7.broadcastQueue.<login-id>`) just before M7 sends out the User Report
- As a result, the broadcast queue gets immediately filled in with potential broadcast messages. Immediately register a consumer on that queue. (before sending any “initialization” inquiry request)
- Upon User report reception, **subscribe to the broadcast queue**
- Perform your “Initialization phase”** with inquiry requests to build a context
- Start consuming the broadcast queue, eliminating potential duplicates** (overlap with initialization)

# Handling errors

## ErrResp message

- **XML format error:** synchronous error in the private response queue
- **M7 back end error (ex: invalid order):** Error response in the broadcast queue
- Example with an invalid order (negative peak quantity):

```

<?xml version="1.0" encoding="UTF-8" standalone="true"?>
<ErrResp xmlns="http://www.deutsche-boerse.com/m7/v6">
  <StandardHeader marketId="EPEX"/>
  <Error clOrdId="dbfa04ed-4d40-424a-87a5-3d41ac3355d7" errorCode="1021"
    err="Peak quantity must be positive for iceberg orders.">
  <VarList/>
</Error>
</ErrResp>

```

- DFS200 contains the error code list:

peak_quantity_must_be_positive	Peak quantity must be positive for iceberg orders	1021	The error is sent when an iceberg order is sent with a non-positive Peak quantity.
--------------------------------	---	------	--

## Date/time in messages

- **All Date/Time values are given in GMT time zone (UTC format).**
  - To get local times, the client have to add or remove hours based on the local time zone.
- **The Product time zone only affects the Contract naming patterns**
  - CET (default value)
  - Europe/London
    - » GB Contract example:
      - 1H 180718-14 : for delivery 14:00 to 15:00 London time

## 2. API Functionalities and messages

- Requests sequence to initialize your applications
- Products, Contracts and phases (zoom on 6.7)
- Back end events: waterfall impacts including contracts

# API functional overview (trader perspective)

## 1. Login / Logout

## 2. Order management:

- **submission, modification, cancellation etc.** : 1 or several at once
- **On behalf** of (obh) another trader colleague

## 3. Trade Management: recall requests (obh), informed about state changes

## 4. Get information about (inquiry requests):

- **System:** request rate limits per minute and hour, nb of days during which contracts are stored
- **Reference data:** BGs and users, products description (qty min size), contracts (IDs, state)
- **Market data:**
  - last state: Order books, own orders, member trading limit, own and public trades (D-7), public and private messages,

## 5. Be aware of all changes (broadcasts)

- System
- Reference data
- Market data

## 6. Change password



Implementing the API:  
in **which order** to use those messages?

# Why do we need an initialization phase? (1/3)

Your application needs to know how to interpret the content of broadcast messages:

Example #1 with an order book message:

- Reference to which « functional » contract?
- Which product?

Note: references are used to minimize data load

```

PblcOrdrBooksDeltaRprt
1 <?xml version="1.0" encoding="UTF-8"?>
2 <PblcOrdrBooksDeltaRprt xmlns="http://www.deutsche-boerse.com/m7/v6">
3   <StandardHeader marketId="EPEX"/>
4   <OrdrbookList>
5     <OrdrBook contractId="10490150" dlvrAreaId="10YDE-RWENET---I" pxDir="0" revisionNo="5">
6       <BuyOrdrList>
7         <OrdrBookEntry ordId="111951663" qty="5000" px="1000" ordEntryTime="2018-06-23T20:35:59.473Z"/>
8       </BuyOrdrList>
9     </OrdrBook>
10  </OrdrbookList>
11 </PblcOrdrBooksDeltaRprt>
12
  
```

- Price, Quantity: how many decimals?

## Why do we need an initialization phase? (2/3)

Example #2 with a new Public Trade message:

- Reference to which « functional » contract?
- Which product?

```

PblcTradeConfRprt
1 <?xml version="1.0" encoding="UTF-8"?>
2 <PblcTradeConfRprt xmlns="http://www.deutsche-boerse.com/m7/v6">
3   <StandardHeader marketId="EPEX"/>
4   <TradeList>
5     <PblcTradeConf tradeId="12060114" state="ACTI" contractId="10490738"
6     qty="5000" px="1000" tradeExecTime="2018-06-24T08:25:59.020Z" revisionNo="1"
7     buyDlvryAreaId="10YDE-RWENET---I" sellDlvryAreaId="10YDE-RWENET---I"/>
8   </TradeList>
9 </PblcTradeConfRprt>
  
```

- Price, Quantity: how many decimals?

## Why do we need an initialization phase? (3/3)

### Conclusion:

- Since the most basic read only application needs to « listen » to:
  - new orders
  - and/or new public trades

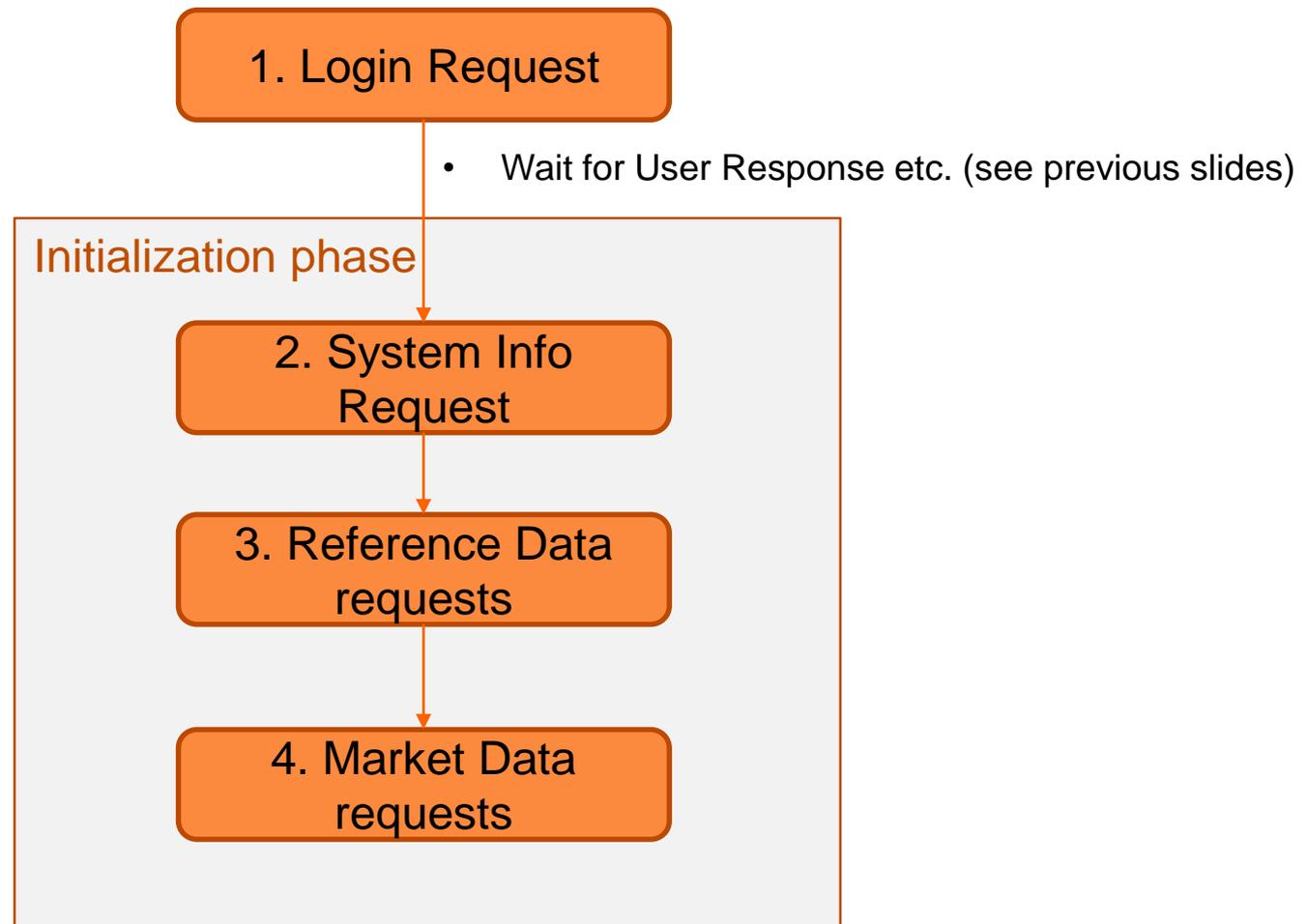
- **Any application needs to build a « context » when it starts:**

- **Get information about products** it needs to follow:
  - get product characteristics, including the number of decimal for prices and quantities
- **Get information about contracts for those products**
- **and then listen to changes / new events** (process incoming broadcasts)



# Which message sequence to initialize your application? (1/3)

During its initialization phase your application should **send inquiry requests in the following order**:



# Which message sequence to initialize your application? (2/3)

1. **LoginReq**: log in M7 + get the UserRprt response that contain all the connected user characteristics: user Id and user code, list of assigned accounts (Balancing Groups “BG”) and user roles (e.g. trading on behalf)
2. **SystemInfoReq**: get the high level M7 parameters + inquiry request rate limit per request
3. **Referential data**:
  - **AllUsersReq**: to know all users of the connected user member, **mandatory if trading on behalf of other users or need to follow-up trades done by other API or ComTrader users** (User name-User Code correspondance)
  - **ProdInfoReq**: get all the product characteristics, which are required for instance to interpret price and quantity numbers in terms of decimals in other messages
  - **ContractInfoReq**: to know (\*) to which contract each contract Id corresponds in order books, trades and Contract info reports messages (updates on a given contract because its state or one of its phase state changes). Retrieve D-1, D , D+1.

---

  - **AcctInfoReq**: get the state, areas and products assignments of your BG(s)
  - **MktStateReq: (for GUIs)** know the state of the market ACTI (trading possible) or HIBE (no trading is possible and obks are empty)
  - **MktAreaInfoReq: (for GUIs)** get the list and state of market areas assigned to your account (BG)
  - **DivryAreaInfoReq: (for GUIs)** get the local and remote state of each area assigned to your account (BG) + the list of tradable products of each of these areas
  - **MsgReq: (for GUI)** to get all recent private and public message (up to 5 days)
  - **RefPxReq**: to know all reference opening price of contracts (auction prices uploaded in M7 for T-contracts between the auction results and T-contracts opening)

# Which message sequence to initialize your application? (3/3)

## 4. Market data:

- **CashLmtReq**: to know your member initial trading limit
- **for public info (\*)**:
  - **PblcOrdRBooksReq** (public order book),
  - **PblcTradeConfReq** (public trades), ←
  - **HubToHubReq** (H2H Capacity)
- **for own info (\*)**:
  - **OrdRReq** (own orders),
  - **TradeCaptureReq** (own trades) ←

**As of M7 6.10** reduction of the private and public trade retrieval period to 7 hours (instead of 25)  
Inquiry request rate limits increased to 56/280 (instead of 14/70)

(\*): up to SystemInfoReq.contractStoreTimeInDays days in the past (configured to 7), mandatory is functionally covered by the app

**Color code: Mandatory, Recommended, optional**

# “Inquiry requests rate limits” and initialization phase

## SystemInfoReq: limits per message

Message	Limit per 60s	Limit per Hour
HubToHubReq	50	500
SystemInfoReq	14	70
LoginReq	14	70
LogoutReq	14	70
ProdInfoReq	14	70
AcctInfoReq	14	70
ContractInfoReq	14	70
MktAreaInfoReq	14	70
PblcOrdrBooksReq	14	70
OrdrReq	14	70
PblcTradeConfReq / TradeCaptureReq	14->56	70->280
CashLmtReq	14	70
MsgReq	14	70
MktStateReq	14	70
RefPxReq	14	70
AllUsersReq	14	70

**Caution:** the limits in the documentation are default values, not EPEX specific (e.g LoginReq 3/20 instead of 14/70)

**Move from 14/70 to 56/280 as of M7 6.10** to accompany the reduction of the trade retrieval period to 7 hours

# Best practice

## Observe messages through ComTrader

In ComTrader: use **CTRL+ALT+R** to display the “Recorded messages” panel:

- Observe the **initialization phase**
- Observe the **broadcast behavior**

Client Date	Process Date	RTT	Direction	Routing Key	Correlation ID	Request Type	XML
23.05.2018 15:37:25	23.05.2018 15:37:25	32	↓	6_0.prddlvr.XBID_Hour_Power.10YDE-EON-----1	48eac199-516a-	PblcOrdRBooksDeltaRprt	Q
23.05.2018 15:37:25	23.05.2018 15:37:25	29	↓	6_0.prddlvr.XBID_Hour_Power.10YDE-VE-----2	48eac199-516a-	PblcOrdRBooksDeltaRprt	Q
23.05.2018 15:37:25	23.05.2018 15:37:25	28	↓	6_0.prddlvr.XBID_Hour_Power.10YDE-RWENET---I	48eac199-516a-	PblcOrdRBooksDeltaRprt	Q
23.05.2018 15:37:25	23.05.2018 15:37:25	26	↓	6_0.prddlvr.XBID_Hour_Power.10YAT-APG-----L	48eac199-516a-	PblcOrdRBooksDeltaRprt	Q
23.05.2018 15:37:25	23.05.2018 15:37:25	24	↓	6_0.prddlvr.XBID_Hour_Power.10YDE-ENBW-----N	48eac199-516a-	PblcOrdRBooksDeltaRprt	Q
23.05.2018 15:37:25	23.05.2018 15:37:25	22	↓	6_0.mbr.CENEX	48eac199-516a-	CashLmtDeltaRprt	Q
23.05.2018 15:37:25	23.05.2018 15:37:25	18	↓	6_0.bg.CENEXxxxxxxxxxx	48eac199-516a-	OrdRExeRprt	Q
23.05.2018 15:37:25	23.05.2018 15:37:25	17	↓	m7.private.responseQueue.CXOWEG02.6ada4630-d2da-42b3-bd18-694f323f34dc	48eac199-516a-	AckResp	Q
23.05.2018 15:37:25	23.05.2018 15:37:25	0	↑	m7.request.management	48eac199-516a-	OrdREntry	Q

```
<?xml version="1.0" encoding="UTF-8"?>
<OrdREntry xmlns="http://www.deutsche-boerse.com/m7/v6">
  <StandardHeader marketId="EPEX"/>
  <OrdRList>
    <OrdR clearingAcctType="P" acctId="TESTXXXXXXXXXXXX" contractId="47994" prod="XBID_Hour_Power"
      side="BUY" px="1000" qty="5000" ordRExeRestriction="NON" dlvrAreaId="10YDE-RWENET---I"
      clOrdId="03461d25-68fd-4481-9d5b-bedf11b7963e" type="O" validityRes="GFS" state="ACTI"/>
  </OrdRList>
</OrdREntry>
```

# A few typical mistakes 😊

## 1. Connection to the AMQP server:

- Mistake in technical data:
  - try to log in ComTrader to make that your user is still valid (revoked if too many wrong passwords)
  - Double check in the environment details document

## 2. Different starting sequence than the recommended one (private queue creation, send login, etc.):

- Ex: Do not wait for the response to LoginReq (User Report) to subscribe to the broadcast queue

## 3. To send a request with the wrong routing key: “unknown request” error

- management request with routing key = `m7.request.inquiry` => NOK
- or an inquiry request with RK = `m7.request.management` => NOK

## 4. To rely only on inquiry request without using broadcast: forbidden (ToR) + not efficient

- contract info request mandatory at the app start (+ any request related to a functional area your app will follow via broadcasts)

## 5. Use messages revision numbers to detect gaps in broadcast messages:

- sequence counting must be used for gaps

# Products, Contracts and phases

This section develops the following **principles**:

- Contracts (e.g.12-13) are instances of products (e.g. Intraday\_Hour\_Power).
- Each product schedule for a given delivery area orchestrates the different phases (e.g.Closed-> Continuous Trading -> Closed) of contracts life cycles.
- In M7 API Contract Info Report messages, only the delivery area part should be considered to determine the tradability of a contract for that delivery area, based on its *state* and *phase*.

# Products, Contracts and Phases – GB market

- **Product:** generic description used to generate contracts

- Characteristics: min price, max price, iceberg or not, etc.
- Example: Hourly Product « GB\_Hour\_Power »
- Schedule: set of phases per area, with an offset



- **Contract = instance of a product**

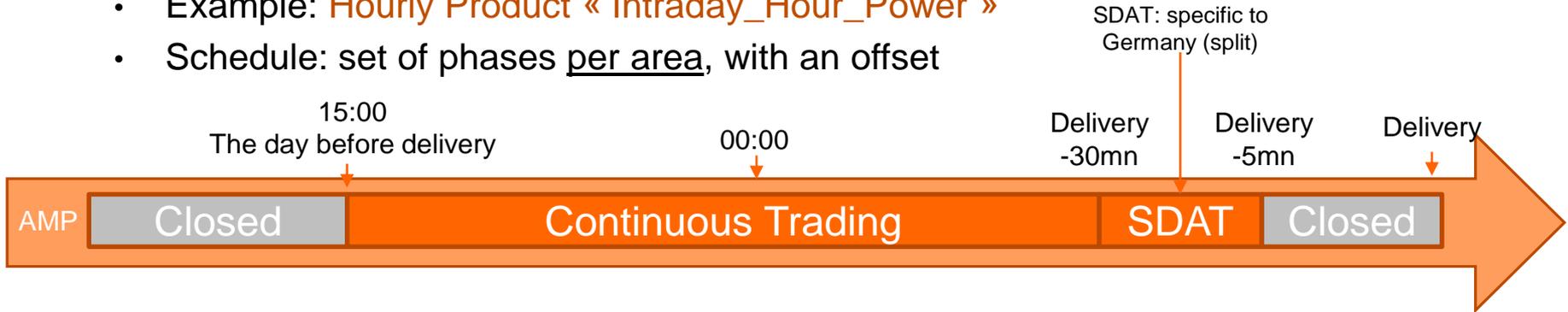
- Example: for delivery 1H180718-10



# Products, Contracts and Phases – German market **before** XBID

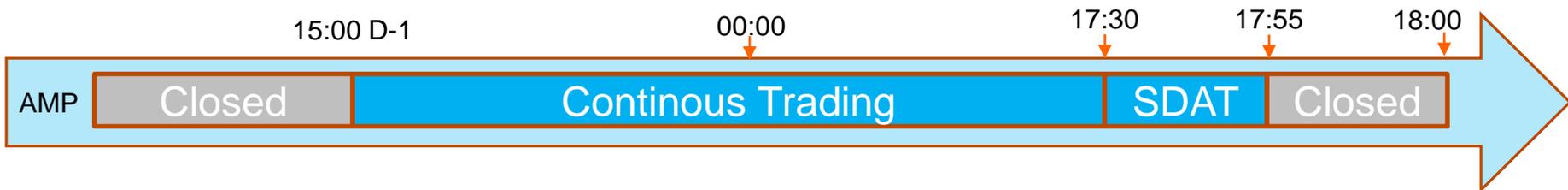
- **Product:** generic description used to generate contracts

- Characteristics: min price, max price, iceberg or not, etc.
- Example: Hourly Product « Intraday\_Hour\_Power »
- Schedule: set of phases per area, with an offset



- **Contract:** instance of a product

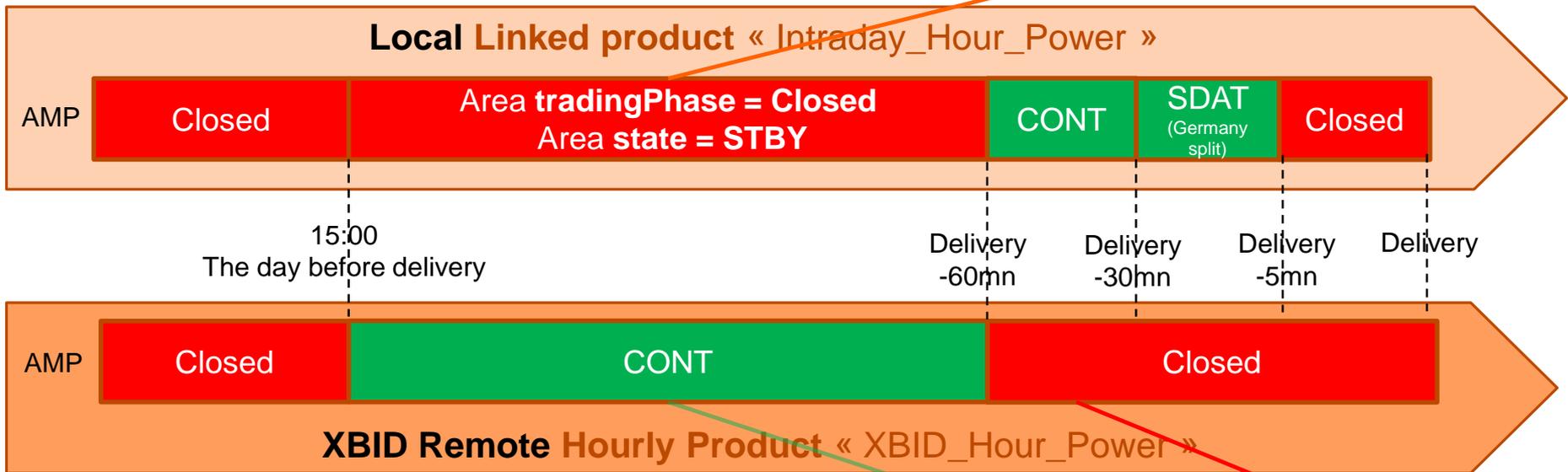
- Example: for delivery 18-19 on AMP



# Products, Contracts and Phases – German market since XBID

Note: when M7 gets connected to XBID, local contracts switch to a « Standby » phase state during the local CLOSED phase.

R	+	Area	Ctrct	TmZ	Cur	Phas	State
⊗		AMP	12-13	CET	EUR	CONT	ACTI
⊗		AMP	13-14	CET	EUR	CLSD	STBY
⊗		AMP	14-15	CET	EUR	CLSD	STBY
⊗		AMP	15-16	CET	EUR	CLSD	STBY



contracts

R	+	Area	Ctrct	TmZ	Cur	Phas	State
⊗		AMP	12-13_XB	CET	EUR	CLSD	ACTI
⊗	+	AMP	13-14_XB	CET	EUR	CONT	ACTI
⊗	+	AMP	14-15_XB	CET	EUR	CONT	ACTI
⊗	+	AMP	15-16_XB	CET	EUR	CONT	ACTI

# Each product has a Schedule compound of different phases

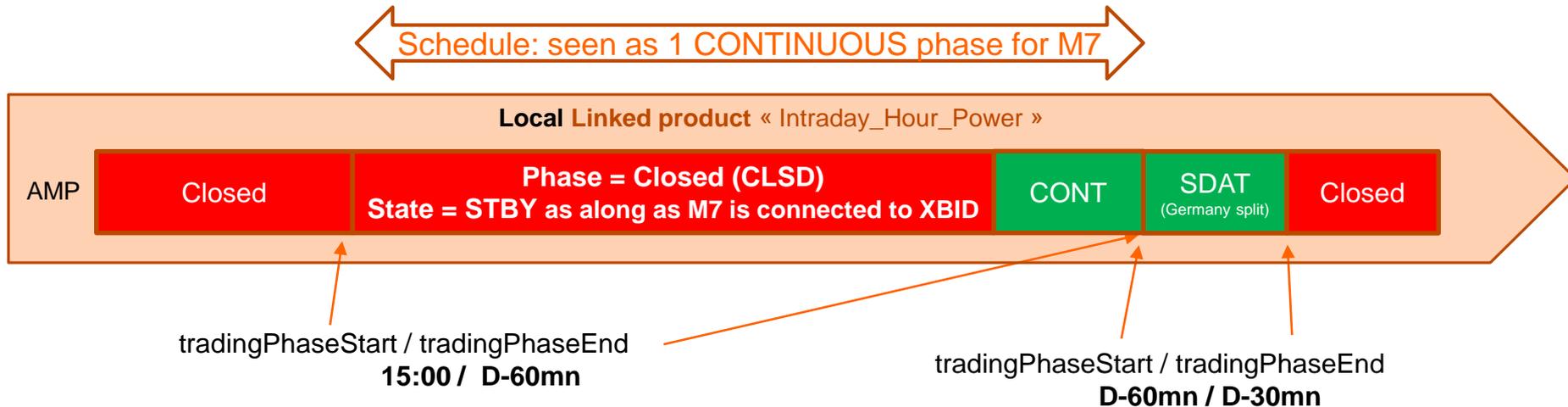
- **A Schedule is a sequence of phases** (e.g. Closed-> Continuous Trading -> Closed) that is setup for a given product/delivery area combination.
- In other words each product schedule for a given delivery area (DA) orchestrates the life cycle of contracts generated from this products for this DA.
- The schedule details for each product and delivery area can be found in the **TRA 100 document** in the API Package.
- An ABSOLUTE\_START offset refers to a duration before 00:00.
- A RELATIVE\_START offset refers to a duration before delivery start.

Schedule name	Phase name	Phase Type	Phase Reference	Offset
XBID4 Linked Local 15h 5mn (SDAT)	Implicitely closed	Closed	ABSOLUTE_START	0s
	CONTINUOUS TRADING	Continuous Trading	ABSOLUTE_START	-9h
	SDAT_TRADING	Same Delivery Area Trading	RELATIVE_START	-30m
	CLOSING_TIME	Closed	RELATIVE_START	-5m

# Trading phases start and end times

## XBID trading: specific case of Local Linked products

- **Trading on XBID requires to trade « remote contracts »** (e.g. 12\_13\_XB of the XBID\_Intraday\_Power product) **and and Local contracts** working in conjunction with remote products, called « Local Linked Contracts » (see « Back-end events: waterfall impacts”)
- **Local Linked contracts are used in 2 situations:**
  - as a backup when M7 disconnects from XBID (in general because XBID is not available)
  - or the final local trading phases (CONT + SDAT in Germany).
- Please note that local contracts remain in a closed phase/ Stby state as an XBID backup trading solution. They become automatically tradable when XBID becomes unavailable.



Contract Info Report message: Until the SDAT phase starts:

- tradingPhaseStart does not change
- tradingPhaseEnd indicates the very end of the Continuous phase

# Zoom on contracts life cycle and related broadcast messages

# Contract broadcast message

## 6.4.17 Contract Information Report (ContractInfoRprt)

ContractInfoRprt	
Type:	Inquiry Response, Broadcast
Response to:	ContractInfoReq (sent to private response queue see 3.1)
Broadcasted:	Yes
Routing Keys:	<schema-version>.prd.<prodName>
Broadcast audience:	All users with assignment to particular product.
Roles:	<ALL>

Topic	Comment
Benefit of implementing Contract requests and broadcasts	<ul style="list-style-type: none"> <li>• Only way to understand Contract Id references in other messages.</li> <li>• Enables to maintain a contract list (Contract Id – Contract Info)</li> </ul>
Main rules	<p>Response to a Contract Info Request or broadcasted for every contract or phase state</p> <p><b><u>Contract Creation / generation:</u></b></p> <ul style="list-style-type: none"> <li>• <b>Standard</b> contracts are created days in advance</li> <li>• <b>User Defined Block</b> contracts are created “on the fly”:                             <ul style="list-style-type: none"> <li>• May not present in the initial request</li> <li>• Created with the 1<sup>st</sup> order on block delivery period</li> </ul> </li> </ul> <p><b><u>Phases:</u></b></p> <ul style="list-style-type: none"> <li>• only show the <b>current phase</b> for each area</li> <li>• Before trading opens and as of delivery time no area record is featured</li> </ul>

# Contract Info Report - Example

```

<?xml version="1.0" encoding="UTF-8"?>
<ContractInfoRprt xmlns="http://www.deutsche-boerse.com/m7/v6">
  <StandardHeader marketId="EPEX"/>
  <ContractList>
    <Contract contractId="10520338" prod="GB_Hour_Power" name="1H180719-10"
      longName="20180719 10:00-20180719 11:00"
      dlvrystart="2018-07-19T09:00:00.000Z" dlvrystend="2018-07-19T10:00:00.000Z"
      predefined="true" revisionNo="3" state="ACTI" tradingPhase="CONT" duration="1.0"
      actPoint="2018-07-17T23:00:00.000Z" expPoint="2018-07-19T09:00:00.000Z" delUnits="1.0">
      <DlvryAreaState dlvrystart="2018-07-17T23:00:00.000Z" dlvrystend="2018-07-19T08:44:00.000Z"
        state="ACTI" tradingPhase="CONT"
        tradingPhaseStart="2018-07-17T23:00:00.000Z" tradingPhaseEnd="2018-07-19T08:44:00.000Z"/>
    </Contract>
  </ContractList>
</ContractInfoRprt>

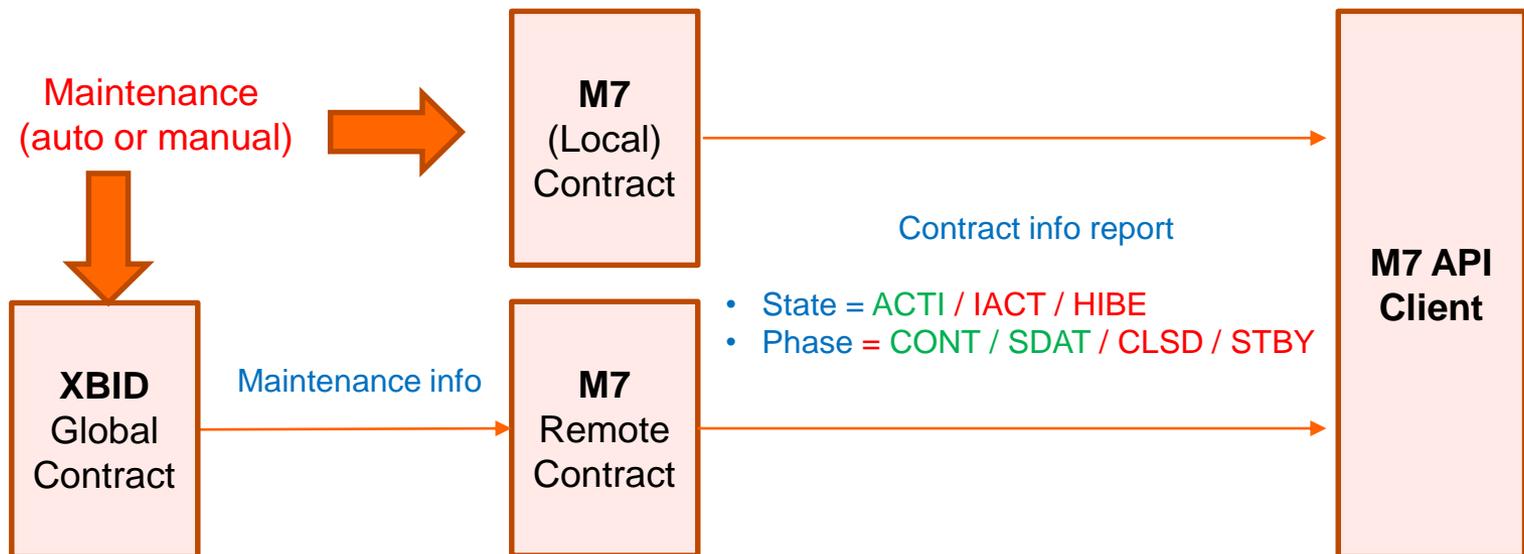
```

Use « Delivery area » information:

- To determine if the contract is tradable
- To know the end of the current trading phase

# Contract and Phase States

- To know if trading is possible, rely on contract State and phase at the area level (DivryAreaState tag)
  - Contracts are NOT tradable when in state:
    - IACT (auto deactivation)
    - or HIBE (manually deactivated)
  - Only Contract in ACTI state and CONT or SDAT phase are tradable
  - Every Product or Contract change is communicated in real time:





# M7 Contract life cycle and messages

## Local and remote contracts (*hourly example*)

Contract Step D= Delivery day	Area	LOCAL Contract State (for the indicated area)	LOCAL Phase (for the indicated area)	XBID Remote Contract State (for the indicated area)	XBID Remote Phase (for the indicated area)
From contract generation to 8am D-1	AMP	N/A. <contract> record without any <delivery area state> record			
8am D-1	AMP	N/A <contract> record without any <delivery area> record		ACTI	CLSD
3pm D-1	AMP	STBY	CLSD	ACTI	CONT
6pm D-1	AMP				
60 mn before delivery	AMP	ACTI	CONT	ACTI	CLSD
30mn before delivery	AMP	ACTI	SDAT	ACTI	CLSD
5mn before delivery	AMP	ACTI	CLSD	ACTI	CLSD
Delivery	AMP	N/A. <contract> record without any <delivery area> record			

## M7 Contract life cycle and messages

GB contracts (*hourly example*) – *not affected by the STBY state*

Contract Step D= Delivery day	Area	Contract State (for the indicated area)	Phase (for the indicated area)
From contract generation to 1:00 am D-1	NGET	N/A. <contract> record without any <delivery area state> record	
1:00 am D-1	NGET	ACTI	CONT
16mn before delivery	NGET	ACTI	CLSD
Delivery	NGET	N/A. <contract> record without any <delivery area> record	

**Note: Contract Info Report: when there are delivery area records:**

- only the contract and phase states for a given area should be considered
- the « Global » contract state should be ignored (technical state)

# GB Half Hour product & contract example

Note : each CIR is received either when starting/recovering your API app (as a response to a ContractInfoReq) or broadcasted by M7



CONTRACT

**1st CIR: Contract Creation**  
 Your API app  
 > stores ContractID  
 > stores Contract Information

**CIR: Contract becomes tradable**  
 Your API app  
 > updates contract information

**CIR: Contract closes**  
 Your API app  
 > updates contract information

**Last CIR: Delivery Start**  
 Your API app  
 > updates contract information  
 > will not receive any additional broadcasts concerning this contractID

Example with HH200818-47:

```

<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<ContractInfoRprt xmlns="http://www.deutsche-boerse.com/m7/v6">
  <StandardHeader marketId="EPEX"/>
  <ContractList>
    <Contract contractId="424280">
      <prod>GB Half Hour Power</prod>
      <name>20200818 23:00-20200818 23:30</name>
      <longName>20200818 23:00-20200818 23:30</longName>
      <divryStart>2020-08-18T22:00:00.000Z</divryStart>
      <divryEnd>2020-08-18T22:30:00.000Z</divryEnd>
      <predefined>true</predefined>
      <revisionNo>0</revisionNo>
      <state>IACT</state>
      <tradingPhase>CLSD</tradingPhase>
      <duration>0.5</duration>
      <actPoint>2020-08-17T23:00:00.000Z</actPoint>
      <expPoint>2020-08-18T22:00:00.000Z</expPoint>
      <delUnits>0.5</delUnits>
    </Contract>
  </ContractList>
</ContractInfoRprt>

<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<ContractInfoRprt xmlns="http://www.deutsche-boerse.com/m7/v6">
  <StandardHeader marketId="EPEX"/>
  <ContractList>
    <Contract contractId="424280">
      <prod>GB Half Hour Power</prod>
      <name>HH200818-47</name>
      <longName>20200818 23:00-20200818 23:30</longName>
      <divryStart>2020-08-18T22:00:00.000Z</divryStart>
      <divryEnd>2020-08-18T22:30:00.000Z</divryEnd>
      <predefined>true</predefined>
      <revisionNo>3</revisionNo>
      <state>IACT</state>
      <tradingPhase>CONT</tradingPhase>
      <duration>0.5</duration>
      <actPoint>2020-08-17T23:00:00.000Z</actPoint>
      <expPoint>2020-08-18T22:00:00.000Z</expPoint>
      <delUnits>0.5</delUnits>
      <DivryAreaState divryAreaId="10YGB-----A">
        <state>IACT</state>
        <tradingPhase>CONT</tradingPhase>
        <tradingPhaseStart>2020-08-17T23:00:00.000Z</tradingPhaseStart>
        <tradingPhaseEnd>2020-08-18T21:45:00.000Z</tradingPhaseEnd>
      </DivryAreaState>
    </Contract>
  </ContractList>
</ContractInfoRprt>

<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<ContractInfoRprt xmlns="http://www.deutsche-boerse.com/m7/v6">
  <StandardHeader marketId="EPEX"/>
  <ContractList>
    <Contract contractId="424280">
      <prod>GB Half Hour Power</prod>
      <name>HH200818-47</name>
      <longName>20200818 23:00-20200818 23:30</longName>
      <divryStart>2020-08-18T22:00:00.000Z</divryStart>
      <divryEnd>2020-08-18T22:30:00.000Z</divryEnd>
      <predefined>true</predefined>
      <revisionNo>4</revisionNo>
      <state>IACT</state>
      <tradingPhase>CONT</tradingPhase>
      <duration>0.5</duration>
      <actPoint>2020-08-17T23:00:00.000Z</actPoint>
      <expPoint>2020-08-18T22:00:00.000Z</expPoint>
      <delUnits>0.5</delUnits>
      <DivryAreaState divryAreaId="10YGB-----A">
        <state>IACT</state>
        <tradingPhase>CLSD</tradingPhase>
        <tradingPhaseStart>2020-08-17T23:00:00.000Z</tradingPhaseStart>
      </DivryAreaState>
    </Contract>
  </ContractList>
</ContractInfoRprt>

<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<ContractInfoRprt xmlns="http://www.deutsche-boerse.com/m7/v6">
  <StandardHeader marketId="EPEX"/>
  <ContractList>
    <Contract contractId="424280">
      <prod>GB Half Hour Power</prod>
      <name>HH200818-47</name>
      <longName>20200818 23:00-20200818 23:30</longName>
      <divryStart>2020-08-18T22:00:00.000Z</divryStart>
      <divryEnd>2020-08-18T22:30:00.000Z</divryEnd>
      <predefined>true</predefined>
      <revisionNo>5</revisionNo>
      <state>IACT</state>
      <tradingPhase>CLSD</tradingPhase>
      <duration>0.5</duration>
      <actPoint>2020-08-17T23:00:00.000Z</actPoint>
      <expPoint>2020-08-18T22:00:00.000Z</expPoint>
      <delUnits>0.5</delUnits>
    </Contract>
  </ContractList>
</ContractInfoRprt>
    
```

Technical contract status  
 Not to be used to determine  
 if contract is tradable

Area state  
 And phase

Area state  
 And phase

No tradingPhaseEnd as it's the last trading phase

DeliveryAreaState added upon contract opening

OBK

1st PblcOrdrBooksDeltaRprt (OBK Update) is received at contract creation With RevisionNo = 1

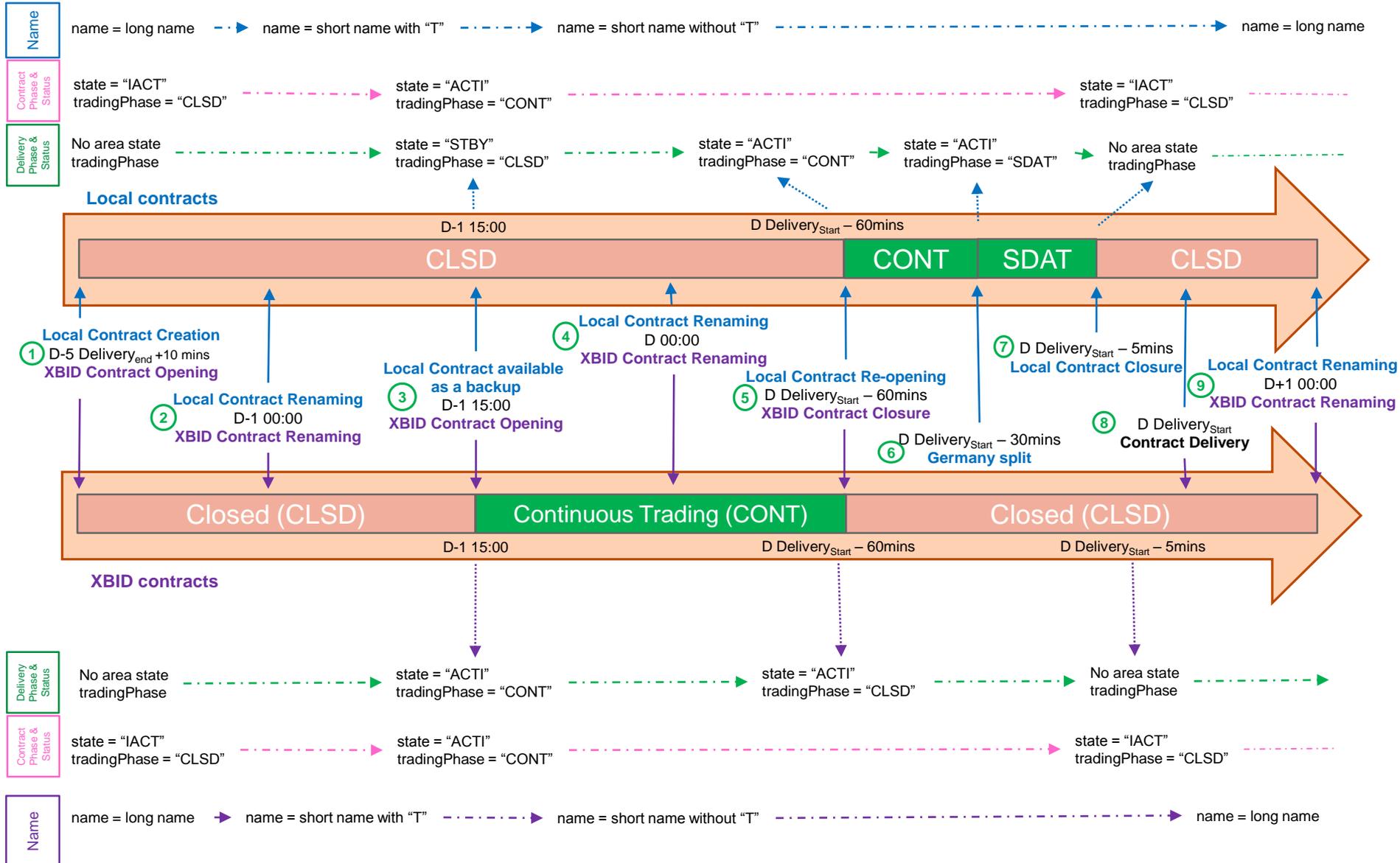
OBK UPDATE Accompanying the Contract change (contractID already known)

Updates due to order activity

OBK UPDATE Accompanying the Contract change

Last PblcOrdrBooksDeltaRprt

# Local and Remote (XBID) Hour products & contracts examples on the German market – Quick overview



## Points of Attention (1/2)

To ensure the completeness of contract data, the following process is recommended:

1. Your API application starts,
2. It sends (several) *ContractInfoReq* which provides all the information on contracts already generated by M7,
3. It listens to broadcasted *ContractInfoRprts* to receive the future contract information (existing/generated contracts updates + new contracts created later).

→ So when receiving the first *PblcOrderBooksDeltaRprt* for a given contract/area, all the contract information is already stored and known by the application because:

- Either the contract was already created when the application logged in and the contractID was collected in the response to the *ContractInfoReq* that was sent by the application when it starts (Initialization phase)
- Or the contract was not already created when the application logged in: the contract was created while the application was logged in and the contractID was sent by a *ContractInfoRprt* broadcast.

As a result, when a Public Order Books Delta Report is broadcasted, API apps already know the contract ID (\*), and should NOT systematically send an Contract Info Request (exception).

- (\*) except for UDBs, User Defined Blocks which do not exist on the GB market, and are created on the fly by M7 when a first order is entered
- The only authorized exception where an API app can send a *Contract Information Request* outside of the app start/recovery phase is if the Contract ID is unknown, which means that either the API app or M7 is bugged, or that messages did not arrive in the usual order (M7 or XBID stress):
  - please trace such events in your app logs and if you cannot identify an issue on the app please report your issue to EPEX.
- The only case where this historically happened is with User Defined Blocks where the contract ID is generated on the fly by M7 when the 1<sup>st</sup> order for the contract is submitted : M7 would sometimes send the Order book broadcast before the Contract Info broadcast, resulting in an unknown Contract ID. The issue is solved since M7 6.9.

## Points of Attention (2/2)

Certain information such as the contract name are updated during the trading session so it is **important for an application to be able:**

- **to update contract information upon broadcasts reception**
- **and to use an unique field to identify the contract.**
  - Only the *contractId* will remain unchanged during the whole contract life (including pre-trading and post-trading phase).

We only focused on 2 examples but the mechanism is the same:

- For the Swiss market as on the GB market
- For all CWE countries as well as the Nordics as on the German countries (except for the SDAT phase)

Note that the hours of opening/closure can be changed depending on the market areas. All information is available in the **product schedules document (TRA100)** in the API package.

In order to know if a contract is open for trading or not, the fields to take into account are the *state* and *tradingPhase* at the DeliveryArea level (**not** the ones at Contract level). Every time a change will occur on one of the delivery area, a new revision of the contract will be broadcasted.

### Caveats:

- The number of updates mentioned on diagrams may change depending on our provider implementation they cannot be used as a contractual basis. They can vary to provide the same functional content.
- For instance, for technical reasons, it is possible that a new revision will be broadcasted with no new information in it.

# Zoom on products messages

# Product broadcast message

## 6.4.19 Product Information Report (ProdInfoRprt)

ProdInfoRprt	
Type:	Inquiry Response
Response to:	ProdInfoReq (sent to private response queue see 3.1)
Broadcasted:	Yes
Broadcast Routing Keys:	<schema-version>.prd.<prodName>
Broadcast audience:	All users with assignment to particular product.
Roles:	<ALL>

Topic	Comment
Benefit of implementing product requests and broadcasts	to have a dynamic implementation / avoid having to change “manually” a static configuration
Main rules	<ul style="list-style-type: none"> <li>• returned as answer to the Product Information Request and also broadcasted in case of the product modification.</li> <li>• the user’s BG(s) product assignment are checked</li> <li>• If the prodName in Product Info Req is not set then the products which user’s BG(s) have assignment are returned</li> <li>• Note: Products changes can occur only when <u>Not</u> Active</li> </ul>

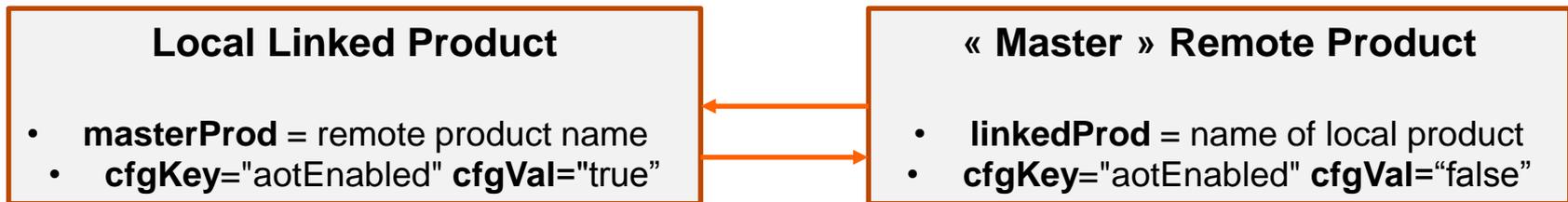
# XBID: Relation between local and remote products

1. An explicit relation between local and remote products
2. A new *Standby* phase state during local *Closed* phase
3. New Trading start and end time in Contract Info Reports

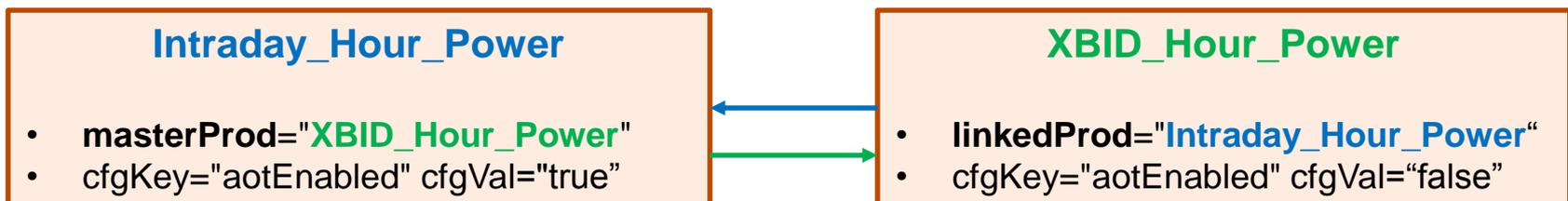
Note: irrelevant for After Market, CH, GB and markets

# Product change – M7 6.7 API

- This change impacts is independent from the API schema version:
  - **Whatever is your API schema version you need to migrate to new Linked products names, except for the GB market.**
- **Product Info Report are enriched for all API schemas:**
  - **Master / Linked product** : an explicit link between remote and local products
  - with a new key-value pairs in the ProdCfgs structure for non GB products:



Example:



# Product change – M7 6.7 API

REMOTE

```
<Prod prodType="COM" prodName="XBID Hour Power" dsplName="XBID_Hour_Power" revisionNo="158"
base="false" cashLmtEnabled="true" riskSetId="1" currency="EUR" minQty="100" maxQty="999000"
maxAmount="0" lotSize="100" decShftQty="3" qtyUnit="MW" delUnits="1.0"
minPx="-999900" maxPx="999900" tickSize="1" decShftPx="2" exchangeId="XSOB" assetType="COM"
executionRestriction="NON" contractsGenerationNumber="120" contActBusinessDay="false"
contExpiryBusinessDay="false" state="ACTI" timeZone="CET" linkedProd="Intraday_Hour_Power">
  <ProdCfgs cfgKey="autoOrderMatcher" cfgVal="true"/>
  <ProdCfgs cfgKey="quoteOrderProduct" cfgVal="false"/>
  <ProdCfgs cfgKey="onExchangePrearrangedTrade" cfgVal="false"/>
  <ProdCfgs cfgKey="icebergPriceDeltaRange" cfgVal="500"/>
  <ProdCfgs cfgKey="aotEnabled" cfgVal="false"/>
  <ProdCfgs cfgKey="blockOrderProduct" cfgVal="true"/>
```

LINKED LOCAL

```
<Prod prodType="COM" prodName="Intraday Hour Power" dsplName="Intraday_Hour_Power" revisionNo="144"
base="false" cashLmtEnabled="true" riskSetId="1" currency="EUR" minQty="100" maxQty="999000"
maxAmount="0" lotSize="100" decShftQty="3" qtyUnit="MW" delUnits="1.0"
minPx="-999900" maxPx="999900" tickSize="1" decShftPx="2" exchangeId="EPEX" assetType="COM"
executionRestriction="NON" contractsGenerationNumber="120" contActBusinessDay="false"
contExpiryBusinessDay="false" state="ACTI" timeZone="CET" masterProd="XBID_Hour_Power">
  <ProdCfgs cfgKey="autoOrderMatcher" cfgVal="true"/>
  <ProdCfgs cfgKey="quoteOrderProduct" cfgVal="false"/>
  <ProdCfgs cfgKey="onExchangePrearrangedTrade" cfgVal="false"/>
  <ProdCfgs cfgKey="icebergPriceDeltaRange" cfgVal="500"/>
  <ProdCfgs cfgKey="aotEnabled" cfgVal="true"/>
```

# Back-end events: waterfall impacts

# Contract HALT/TRADING

- **MktStateRprt** send the new state (ACTI/HIBE) when it changes.

LTS Object	Market State transition <b>ACTI-&gt; HIBE</b>	Market State transition <b>HIBE-&gt;ACTI</b>
<b>Products</b>	<ul style="list-style-type: none"> <li>• Unchanged</li> </ul>	<ul style="list-style-type: none"> <li>• Unchanged</li> </ul>
<b>Contracts</b>	<ul style="list-style-type: none"> <li>• Contracts: deactivated</li> <li>• Contract Info Report: state="HIBE" tradingPhase="CLSD"</li> </ul>	<ul style="list-style-type: none"> <li>• Contracts: activated</li> <li>• Contract Info Report: state="ACTI" tradingPhase="CONT"</li> </ul>
<b>Order books</b> (Area+Contract)	<ul style="list-style-type: none"> <li>• Order books update</li> <li>• Order Book Delta Report: Qty = 0</li> </ul>	<ul style="list-style-type: none"> <li>• Unchanged</li> <li>• No broadcast</li> </ul>
<b>Own orders</b>	<ul style="list-style-type: none"> <li>• Orders on XBID contracts: deactivated</li> <li>• Order Execution report: state="HIBE"</li> </ul>	<ul style="list-style-type: none"> <li>• Unchanged</li> <li>• No broadcast</li> </ul>

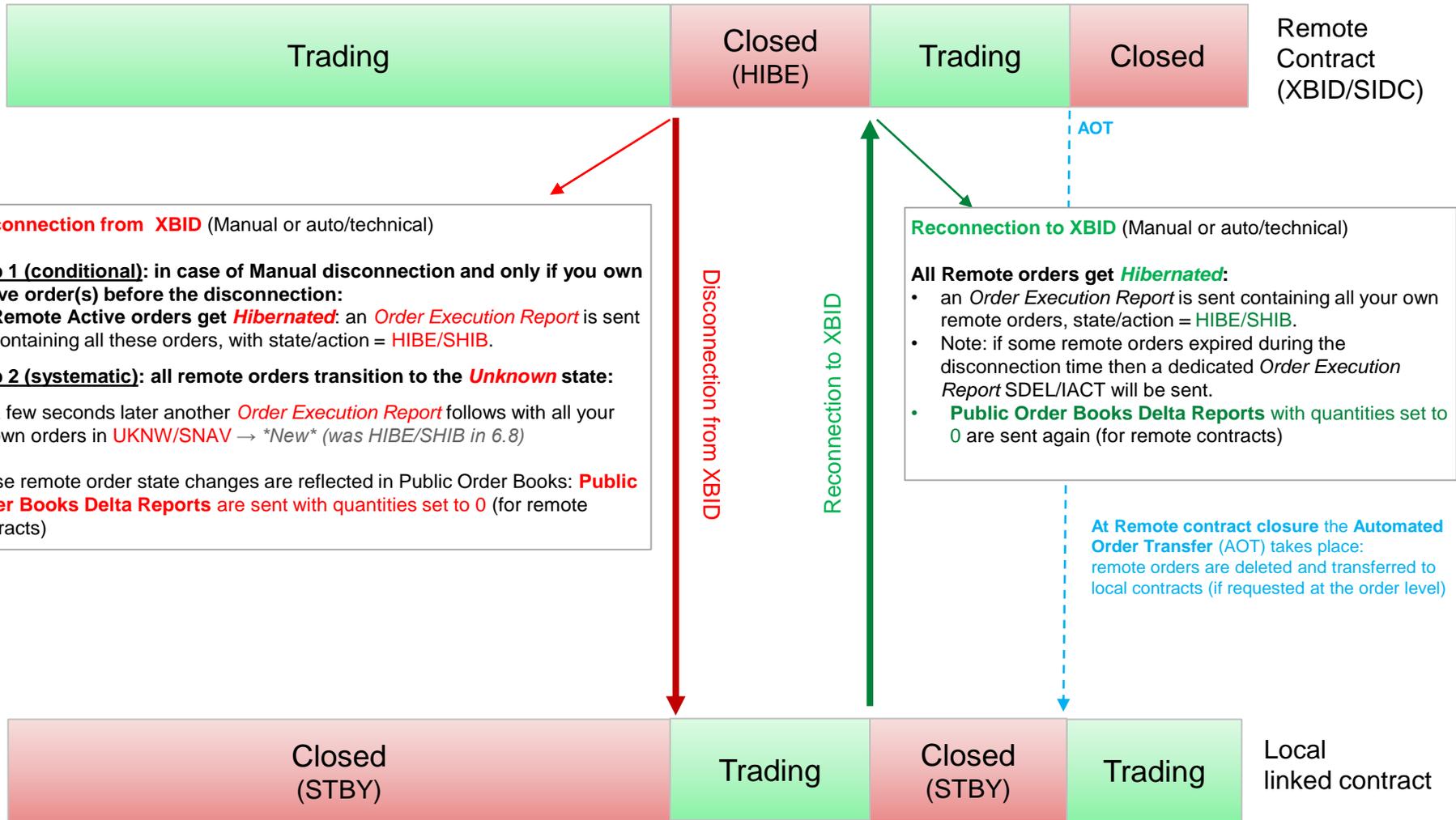
# XBID related back-end event

- **New back-end event:** LTS-XBID connection / disconnection
  - waterfall impacts on LTS objects:

M7 Object	M7-XBID <b>Disconnection</b>	M7-XBID <b>Reconnection</b>
<b>Products</b>	<ul style="list-style-type: none"> <li>• <u>XBID</u> Products: deactivated</li> <li>• Product info report: state="HIBE"</li> </ul>	<ul style="list-style-type: none"> <li>• XBID Products activated</li> <li>• Product info report: state="ACTI"</li> </ul>
<b>Contracts</b>	<ul style="list-style-type: none"> <li>• XBID: deactivated, Linked: activated</li> <li>• Contract Info Report:                      XBID: state=HIBE tradingPhase=CLSD                      Linked: state = ACTI tradingPhase= CONT</li> </ul>	<ul style="list-style-type: none"> <li>• XBID: activated, Linked: Standby</li> <li>• Contract Info Report:                      XBID: state="ACTI" tradingPhase="CONT"                      Linked: state = STBY tradingPhase= CLSD</li> </ul>
<b>Order book</b> (Area+Contract)	<ul style="list-style-type: none"> <li>• XBID Order books cleaned:</li> <li>• Remote Order Book Delta Report: Qty = 0</li> </ul>	<ul style="list-style-type: none"> <li>• XBID Unchanged (no activation)</li> <li>• Linked order books cleaned:</li> <li>• Linked Obk Delta Report: Qty = 0</li> </ul>
<b>Own orders</b>	<ul style="list-style-type: none"> <li>• Orders on XBID contracts: deactivated</li> <li>• Remote Order Execution report: state="HIBE"</li> </ul>	<ul style="list-style-type: none"> <li>• XBID Unchanged (no activation)</li> <li>• Local linked orders: hibernated</li> <li>• Linked Order Execution report: state="HIBE"</li> </ul>

# Local Products as a backup to remote products

UKNW/SNAV Order Execution Report while M7 is disconnected from XBID



# Thank you for your attention!

**EPEX SPOT Paris**

5 boulevard Montmartre  
75002 Paris  
France  
Tel +33 1 73 03 96 00  
sales@epexspot.com

**EPEX SPOT London**

11 Westferry Circus  
Canary Wharf  
London E14 4HE  
United Kingdom

**EPEX SPOT Bern**

Marktgasse 20  
3011 Bern  
Switzerland

**EPEX SPOT Amsterdam**

Quarter Plaza  
Transformatorweg 90  
1014 AK Amsterdam  
The Netherlands  
Tel +31 20 305 4000

**EPEX SPOT Berlin**

Regus at The Chancellor Office  
Rahel-Hirsch-Straße 10  
10557 Berlin  
Germany

**EPEX SPOT Brussels**

Boulevard de l'Impératrice 66  
1000 Bruxelles  
Belgium

**EPEX SPOT Wien**

Mayerhofgasse 1/19  
1040 Wien  
Austria

# APPENDIX

# Zoom on Login request / response

## 1. LoginReq:

```
LoginReq
1 <?xml version="1.0" encoding="UTF-8"?>
2 <LoginReq xmlns="http://www.deutsche-boerse.com/m7/v6" user="CXIEST01"
3   force="false" disconnectAction="NO">
4   <StandardHeader marketId="EPEX"/>
5 </LoginReq>
```

### DisconnectAction options (unexpected connection loss) :

- "NO": No action is executed.
- "DEACT\_USER\_ORDRS": All orders of this user will be deactivated.
- "DEACT\_ACCT\_ORDRS": All orders of the assigned trader accounts will be deactivated.

### Force: true / false

If the same user is already logged:

- If force = false: an ErrResp msg is replied « TraderAlreadyLoggedInException »
- If force = true: a Logout Report is broadcasted to the user who is logged out

# Zoom on Login request / response

1. If Login info is OK, a “User Report” is replied to the LoginReq

**usrCode:** Useful in Trade Capture reports, etc.

```
UserRprt
1 <?xml version="1.0" encoding="UTF-8"?>
2 <UserRprt xmlns="http://www.deutsche-boerse.com/m7/v6">
3   <StandardHeader marketId="EPEX"/>
4   <Usr sessionId="1529666329966" mbrName="Member Test For EPEX" usrId="1003864"
revisionNo="7" name="Test User Epex" usrCode="TRD001" mbrId="TEST1"
defaultAcctId="TESTBG1-----" state="ACTI">
5     <AssgAcctId>TESTBG2-----</AssgAcctId>
6     <AssgAcctId>TESTBG1-----</AssgAcctId>
7     <UsrRole>ROLE_PUBLIC_API</UsrRole>
8     <UsrRole>ROLE_CAPACITY_INFO</UsrRole>
9     <UsrRole>ROLE_USER</UsrRole>
10    <UsrRole>ROLE_ON_BEHALF</UsrRole>
11    <UsrRole>ROLE_SYSTEM_ACCESS</UsrRole>
12  </Usr>
13 </UserRprt>
```

**Note:** In case of connection failure, you must implement an exponential back-off (wait more and more between login attempts)

# Zoom on initialization phase requests / responses

- **SystemInfoReq:** identify high level parameters:

```
SystemInfoResp
1 <?xml version="1.0" encoding="UTF-8"?>
2 <SystemInfoResp xmlns="http://www.deutsche-boerse.com/m7/v6"
  backendVersion="m7-core-6.4.1.72-af5ded9715d272152c2437efc8e9a3c49997abe5"
  backendTimeZone="CET" backendMarketTimeZone="CET" contractStoreTimeInDays="7"
  maxOrders="100" maxQuotes="100" capabilities="TRADING-LIMIT LOCAL-EXCHANGE PNC-ORDERS
  SETTLEMENT" allowedClearingAcctTypes="A,P">
3   <StandardHeader marketId="EPEX"/>
4   <RequestLimitList>
5     <RequestLimit message="CashLmtReq" duration="60" rate="14"/>
6     <RequestLimit message="CashLmtReq" duration="3600" rate="70"/>
7     <RequestLimit message="PblcOrdRBooksReq" duration="60" rate="14"/>
8     <RequestLimit message="PblcOrdRBooksReq" duration="3600" rate="70"/>
9     <RequestLimit message="ChgPwdReq" duration="60" rate="1"/>
10    <RequestLimit message="ChgPwdReq" duration="3600" rate="10"/>
11    <RequestLimit message="MktStateReq" duration="60" rate="1"/>
12    <RequestLimit message="MktStateReq" duration="3600" rate="10"/>
13    <RequestLimit message="MbrChangeRprt" duration="60" rate="1"/>
14    <RequestLimit message="MbrChangeRprt" duration="3600" rate="10"/>
15    <RequestLimit message="MsgReq" duration="60" rate="14"/>
16    <RequestLimit message="MsgReq" duration="3600" rate="70"/>
```

# Zoom on initialization phase requests / responses

- **SystemInfoReq:** identify inquiry requests limit per minute (60s) and hour (3600s)

```
SystemInfoResp
50 <RequestLimit message="ProdInfoReq" duration="3600" rate="70"/>
51 <RequestLimit message="AllUsersReq" duration="60" rate="1"/>
52 <RequestLimit message="AllUsersReq" duration="3600" rate="10"/>
53 <RequestLimit message="ClgInfoReq" duration="60" rate="1"/>
54 <RequestLimit message="ClgInfoReq" duration="3600" rate="10"/>
55 <RequestLimit message="DlvryAreaInfoReq" duration="60" rate="14"/>
56 <RequestLimit message="DlvryAreaInfoReq" duration="3600" rate="70"/>
57 <RequestLimit message="AcctInfoReq" duration="60" rate="14"/>
58 <RequestLimit message="AcctInfoReq" duration="3600" rate="70"/>
59 <RequestLimit message="OrdrEntry" duration="60" rate="1"/>
60 <RequestLimit message="OrdrEntry" duration="3600" rate="10"/>
61 <RequestLimit message="ImplOrdrReq" duration="60" rate="1"/>
62 <RequestLimit message="ImplOrdrReq" duration="3600" rate="10"/>
63 <RequestLimit message="TradeCaptureReq" duration="60" rate="14"/>
64 <RequestLimit message="TradeCaptureReq" duration="3600" rate="70"/>
65 <RequestLimit message="LoginReq" duration="60" rate="14"/>
66 <RequestLimit message="LoginReq" duration="3600" rate="70"/>
67 </RequestLimitList>
68 </SystemInfoResp>
```

Example: An user can log in:

- 14 times per mn
- 70 times per hour
- Then an Error Response message is sent

# Zoom on initialization phase requests / responses

## SystemInfoReq: limits per message

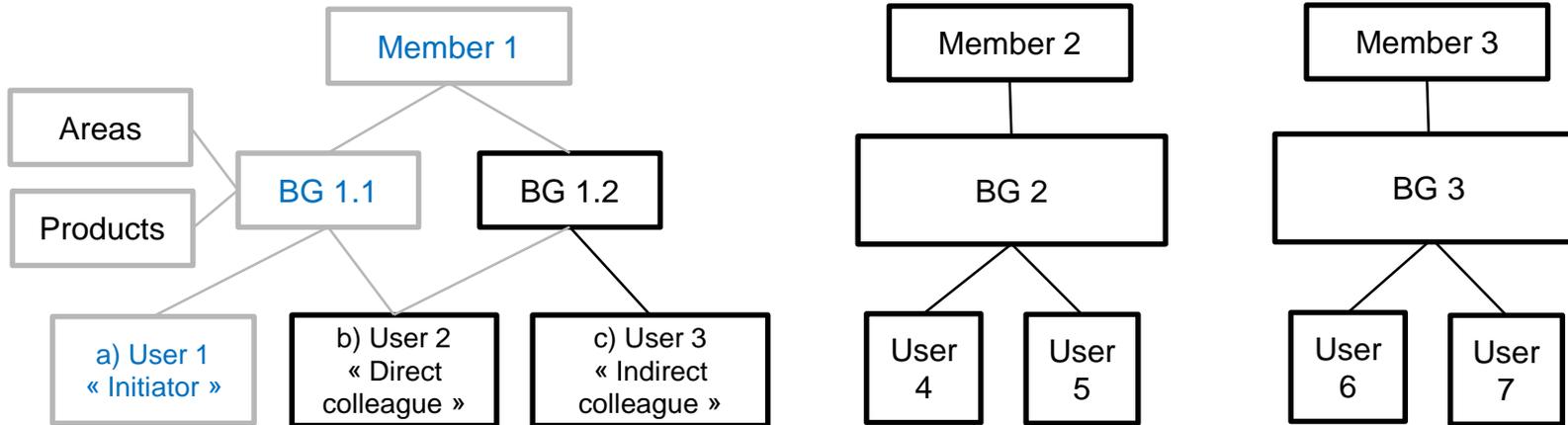
Message	Limit per 60s	Limit per Hour
HubToHubReq	50	500
SystemInfoReq	14	70
LoginReq	14	70
LogoutReq	14	70
ProdInfoReq	14	70
AcctInfoReq	14	70
ContractInfoReq	14	70
MktAreaInfoReq	14	70
PblcOrdrBooksReq	14	70
PblcTradeConfReq	14	70
CashLmtReq	14	70
MsgReq	14	70
MktStateReq	14	70
RefPxReq	14	70
AllUsersReq	14	70

**Caution:** the limits in the documentation have NOT been updated with those new System Info Req values (e.g LoginReq 3/20 instead of 14/70)

# **APPENDIX – Ownship categories**

# Member, balancing group and users

Personal data vs Own data vs public data



1. **Personal:** user related (e.g. order owner)
2. **Own:** BG related: **personal + my direct colleagues**
3. **Public:** all users

**APPENDIX –  
Order books  
and  
Order management**

## Order management : State transitions

- **An order can be created as active or hibernated**
- **It can be modified** (Active -> Active, Hibe -> Hibe)
- **It will get partially traded** (Active -> Active)
- **etc.**
  
- **Each of these state transitions will trigger:**
  - **an Order Execution Broadcast message:** new state and/or characteristic and Revision No and
  - **Potentially other broadcasts:**
    - Order book delta report (if the order is active)
    - Private and public Trade messages (partial exe, full exe)
    - Etc.

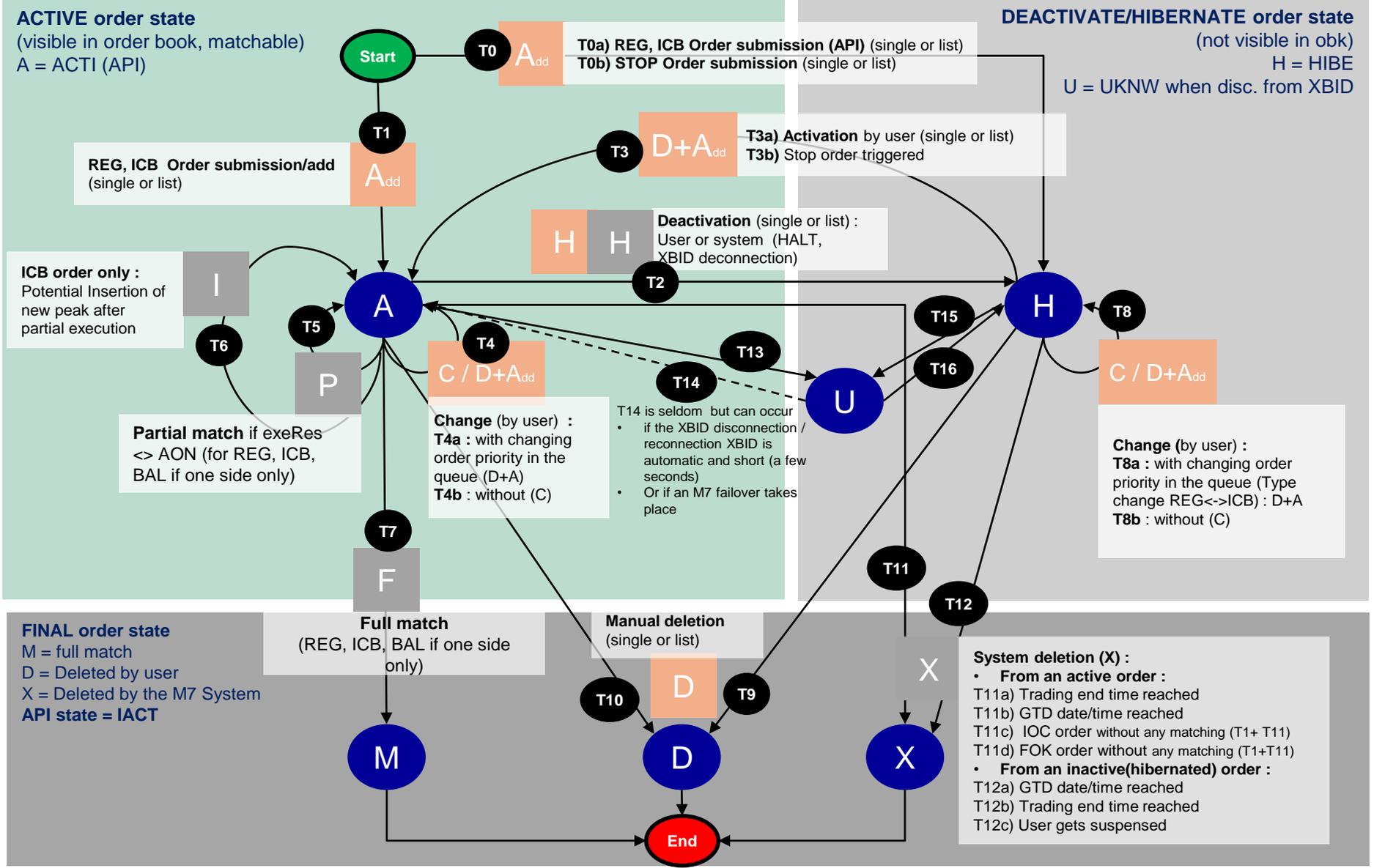
# Order state diagram

## Exchange Orders

<span style="color: blue;">●</span> Order state	<span style="background-color: #f4a460; border: 1px solid black; padding: 2px;"> </span> User triggered Order management action
<span style="border: 1px solid black; border-radius: 50%; padding: 2px;">Tx</span> Transition reference number	<span style="background-color: #cccccc; border: 1px solid black; padding: 2px;"> </span> System triggered Order management action

**ACTIVE order state**  
(visible in order book, matchable)  
A = ACTI (API)

**DEACTIVATE/HIBERNATE order state**  
(not visible in obk)  
H = HIBE  
U = UKNW when disc. from XBID



# Order management with change of order ID

- As showed on the previous diagram, several actions trigger an new Order ID, M7 first deleting/cancelling the order and then submitting it again (D+A).
- This is the case when:
  - The price of an active order is modified
  - The quantity (or Peak qty for icebergs) of an active order is increased,
  - The order execution restriction is changed (NON -> IOC or NON->FOK)
  - A deactivated order is activated
  - The type of a deactivated order is changed: REG <-> ICB

For instance if a public order price (order ID = 115848168) is changed you the following message would be sent, featuring the removed order ID with a zero Quantity

```
<?xml version="1.0" encoding="UTF-8"?>
<PblcOrdrBooksDeltaRprt xmlns="http://www.deutsche-boerse.com/m7/v6">
  <StandardHeader marketId="EPEX"/>
  <OrdrbookList>
    <OrdrBook contractId="56620" dlrvyAreald="10YNL-----L" lastPx="8256" lastQty="1000" totalQty="491800" lastTradeTime="2019-03-14T09:57:08.496Z"
    pxDir="0" revisionNo="149" highPx="8256" lowPx="-5050">
      <BuyOrdrList>
        <OrdrBookEntry ordId="115848272" qty="10000" px="3800" ordEntryTime="2019-03-14T09:58:20.453Z"/> => new order with new order ID and new price
        <OrdrBookEntry ordId="115848168" qty="0" px="3900" ordEntryTime="2019-03-14T09:58:13.282Z"/> => old order being removed automatically by M7: qty = 0
      </BuyOrdrList>
    </OrdrBook>
  </OrdrbookList>
</PblcOrdrBooksDeltaRprt>
```

# Order Entry and Maintenance

## Which request message for what?

Message	Nb of orders	Usage
<b>OrdEntry</b>	1 or List	<b>New order submission</b> Execution restriction at the list level: None, Valid, Linked (FOK). <b>The order execution report gives the Order Id and revision number.</b>
<b>OrdModify</b>	1 or List	Depends on <b>ordrModType</b> : <ul style="list-style-type: none"> <li>• <b>“ACTI”</b>: Activate all orders contained in this basket.</li> <li>• <b>“DEAC”</b>: Deactivates (hibernates) all orders contained in the basket.</li> <li>• <b>“MODI”</b>: Modifies all orders in the basket.</li> <li>• <b>“DELE”</b>: Deletes all orders in the basket.</li> <li>• <b>The latest revision number of the order must be provided by the client. In case the backend has another revision number, it will reject the request with an ErrResp.</b></li> </ul>
<b>ModifyAllOrdrs</b>	All in the scope	Depends on <b>ordrModType</b> : <ul style="list-style-type: none"> <li>• to delete or deactivate all orders belonging to a member, an account or a trader</li> <li>• <b>“ACTI”, “DEAC”, “DELE”</b></li> </ul>
<b>PreArrangedOrderProcess</b>	1	<b>Accept or Reject an OTC orders.</b> The latest revision number of the order must be provided by the client.
<b>OrdReq</b>	All in the scope	<b>Request all own orders</b> , which are currently active or hibernated for the given account and products.



# GB Products: order types and Execution Restrictions (2/2)

Product	XBID/local	Product execution Restriction	PreDefined	UDB	Supported order types <ul style="list-style-type: none"> <li>• "O": Regular limit order.</li> <li>• "B": User defined block order.</li> <li>• "I": Iceberg order.</li> </ul>	Supported exeRes at order level per order type	Comment
GB_Half_Hour_Power	local	NON	Yes	No	O, I	Order type O: NON, IOC, FOK Order type I: NON	
GB_Hour_Power	local	NON	Yes	No	O, I	Order type O: NON, IOC, FOK Order type I: NON	
GB_2_Hour_Power	local	NON	Yes	No	O, I	Order type O: NON, IOC, FOK Order type I: NON	
GB_4_Hour_Power	local	NON	Yes	No	O, I	Order type O: NON, IOC, FOK Order type I: NON	
GB_Baseload	local	NON	Yes	No	O, I	Order type O: NON, IOC, FOK Order type I: NON	
GB_Peakload	local	NON	Yes	No	O, I	Order type O: NON, IOC, FOK Order type I: NON	
GB_3_Plus_4	local	NON	Yes	No	O, I	Order type O: NON, IOC, FOK Order type I: NON	
GB_Overnight	local	NON	Yes	No	O, I	Order type O:NON, IOC, FOK Order type I: NON	
GB_Extended_Peak	local	NON	Yes	No	O, I	Order type O:NON, IOC, FOK Order type I: NON	

# XBID API impact : Messages content or behavior

Message	Attribute	Comment
<b>DivvyAreaInfoRprt</b>	remoteState	<p>Current Remote XBID Status of the delivery area.</p> <ul style="list-style-type: none"> <li>• "ACTI": Delivery area is active and tradable.</li> <li>• "SUSP": Delivery area is inactive. Trading not possible.</li> <li>• "DELE": Delivery area was deleted. Trading not possible.</li> </ul>
<b>ContractInfoRprt</b>	remoteContractId	<p>XBID SOB Contract Id</p> <ul style="list-style-type: none"> <li>• Only mentioned for remote contracts</li> </ul>
<b>OrdRExeRprt</b>	aggressorIndicator	<ul style="list-style-type: none"> <li>• "Y" - Trade aggressor</li> <li>• "N" - Trade originator</li> <li>• "U" - Unknown, for executed orders of remote products</li> </ul>
<b>OrdRExeRprt</b>	remoteOrdId	<p>XBID SOB Order Id</p> <ul style="list-style-type: none"> <li>• remoteOrdId="0" for local products</li> </ul>
<b>TradeCaptureRprt</b> (own half trade trade)	remoteTradeId remoteOrdId aggressorIndicator	<p>XBID SOB Trade Id</p> <ul style="list-style-type: none"> <li>• remoteTradeId="0" for local products</li> </ul>
<b>PbIcOrdRBooksResp</b> <b>PbIcOrdRBooksDeltaRprt</b>	Behavior / OrdId	<ul style="list-style-type: none"> <li>• Updates beyond the « display » depth limitation do not trigger any <i>PbIcOrdRBooksDeltaRprt</i></li> <li>• When an order is submitted from EPEX: <ul style="list-style-type: none"> <li>• Order Id = M7 "local" ordId</li> </ul> </li> <li>• When an order is submitted from another NEMO: <ul style="list-style-type: none"> <li>• Order Id = XBID "remote" ordId</li> </ul> </li> </ul>
<b>PbIcTradeConfRprt</b>	Trade Id buyDivvyAreaId sellDivvyAreaId	<ul style="list-style-type: none"> <li>• M7 already generates a "local" trade Id: <ul style="list-style-type: none"> <li>• Trade Id = M7 "local" Trade Id (never the Remote Trade ID, even for trades on the same product which involve Epex on one side only or no side at all: ex: Epex Area-Spain or Spain-Spain)</li> </ul> </li> <li>• Buy and Sell Delivery Area Id are always filled in, even when those areas are not tradable via M7 (ex: Spanish area 10YES-REE-----0)</li> </ul>

# UDB submission with XBID

- User Defined Blocks in XBID are a functionality associated to XBID\_Hour\_Power
- Example: UDB on 12-15\_XB

▶ Market Overview (User Defined Delivery Period)

R	+	Area	Ctrl	▲	Cur	Phas	State	BAcc	BQty	BP-VWA	Bid
⊗		50HzT	12-15_XB		EUR	CONT	ACTI	50.0	50.0		10.00

```
<?xml version="1.0" encoding="UTF-8"?>
<OrdEntry xmlns="http://www.deutsche-boerse.com/m7/v6">
  <StandardHeader marketId="EPEX"/>
  <OrderList>
    <Order clearingAcctType="P" acctId="TESTXXXXXXXXXX" prod="XBID Hour Power" side="BUY" px="1000" qty="50000"
      orderExeRestriction="AON" txt="XBID UDB test order" dlvrAreaId="10YDE-VE-----2"
      clOrderId="1eb8c933-4715-4d46-9718-9e0c9c2eff71" type="B"
      dlvrStart="2018-05-24T10:00:00.000Z" dlvrEnd="2018-05-24T13:00:00.000Z" validityRes="GFS" state="ACTI"/>
  </OrderList>
</OrdEntry>
```

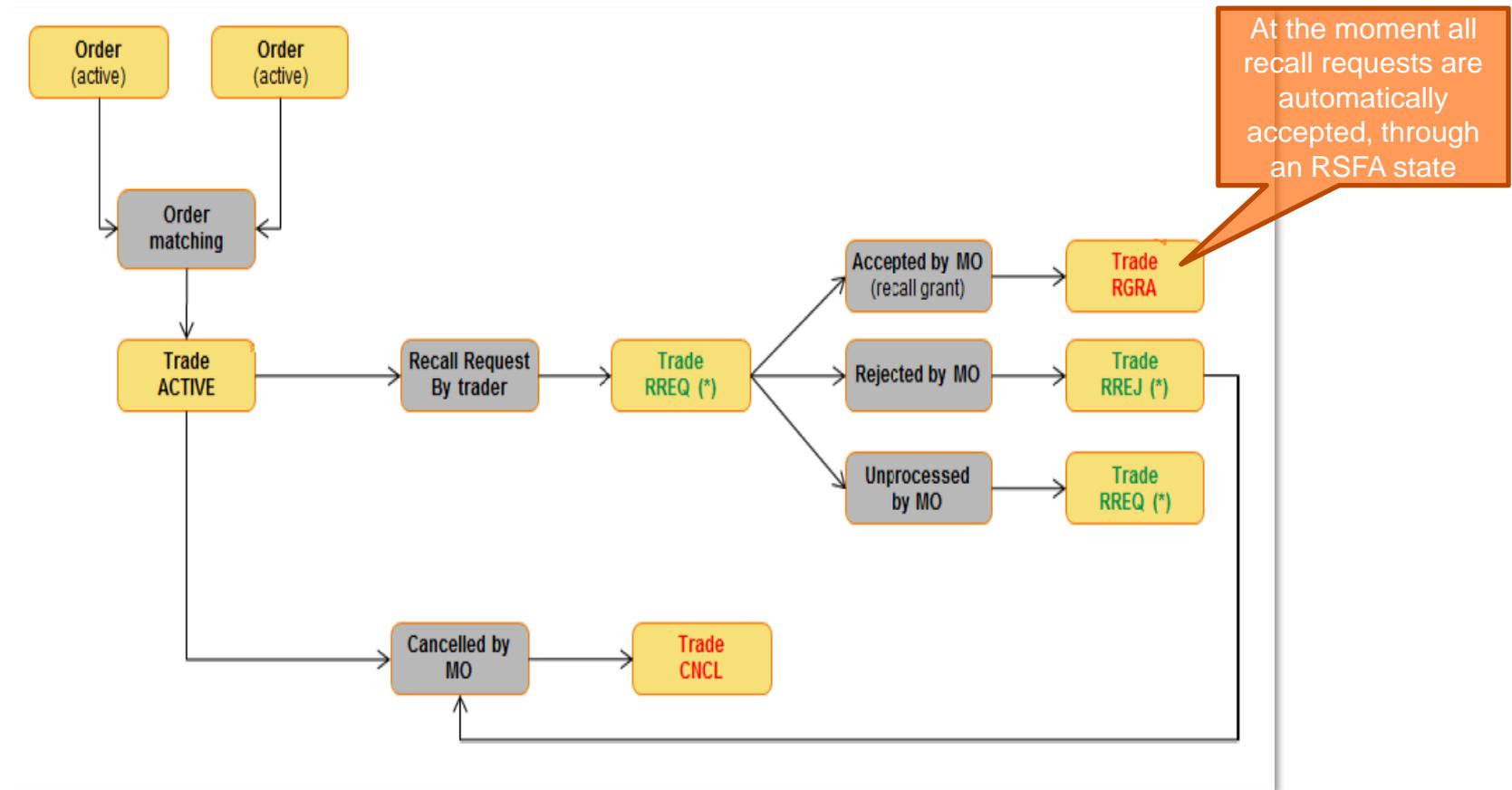
- As usual for UDBs, contracts are created on the fly with the first submission:

```
<?xml version="1.0" encoding="UTF-8"?>
<ContractInfoRpt xmlns="http://www.deutsche-boerse.com/m7/v6">
  <StandardHeader marketId="EPEX"/>
  <ContractList>
    <Contract contractId="50120" prod="XBID_Hour_Power" name="12-15_XB" longName="20180524 12:00-20180524 15:00"
      <DlvrAreaState dlvrAreaId="10YFR-RTE-----C" state="ACTI" tradingPhase="CONT" tradingPhaseStart="2018-05-
      <DlvrAreaState dlvrAreaId="10YDE-RWENET---I" state="ACTI" tradingPhase="CONT" tradingPhaseStart="2018-05-
```

# Trades

# Trades state diagram

- **Trade status:**
  - The **recall process** can only be initiated **by Traders**
  - Trade **cancellation** can only be done **by Market operations**.



- **Any Trade new state / transition will generate a private (if own trade) and public trade broadcast**

# Own Trade Broadcasts – Trade Capture Report

TradeCaptureRprt message can be sent:

1. **As a response to a TradeCaptureReq** inquiry request
2. **As a broadcast** when one of your own order is executed and changes of state

Please note that in case of a cross trade the **TradeCaptureRprt content differs a bit:**

1. **As a response to a TradeCaptureReq** inquiry request:
  - both sides of the trades are packaged in the same <Trade> tag
2. **As a broadcast:** half-trades are sent **separately**
  - 2 separate TradeCaptureRprt are sent: 1 for the buy side, 1 for the sell side

```

TradeCaptureRprt
1 <?xml version="1.0" encoding="UTF-8"?>
2 <TradeCaptureRprt xmlns="http://www.deutsche-boerse.com/m7/v6">
3   <StandardHeader marketId="EPEX"/>
4   <TradeList>
5     <Trade tradeId="12391677" state="ACTI" contractId="10612678" qty="14000" px="3904" execTime="2018-10-05T16:02:03.767Z" revisionNo="1" preArranged="false"
6     contractPhase="CONT" decomposed="false">
7       <Buy clearingAcctType="P" dlvrAreaId="10YDE-VE-----2" acctId="TESTBG1-----" ordId="114267767" txt="" aggressorIndicator="N" usrCode="TRD001"
8       clOrdId="6e472148-23cc-4530-9054-d916c06368af" mbrId="TEST1"/>
9     </Trade>
10  </TradeList>
11 </TradeCaptureRprt>
    
```

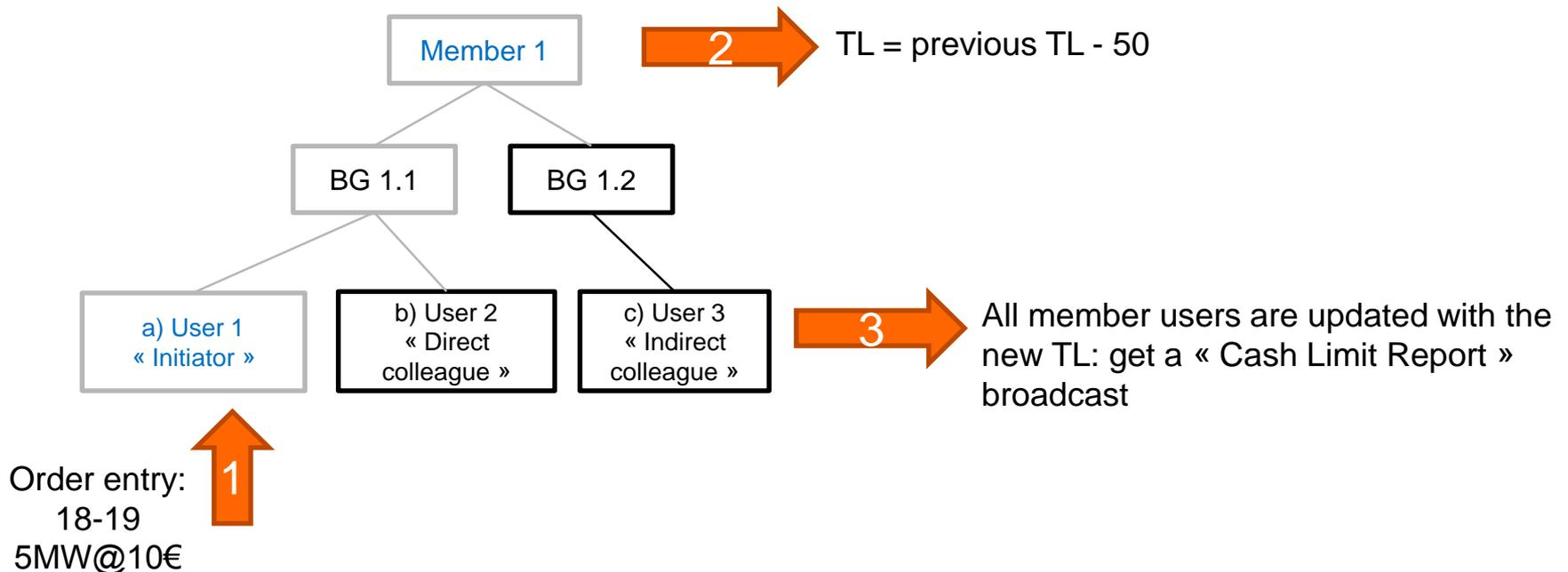
```

TradeCaptureRprt
1 <?xml version="1.0" encoding="UTF-8"?>
2 <TradeCaptureRprt xmlns="http://www.deutsche-boerse.com/m7/v6">
3   <StandardHeader marketId="EPEX"/>
4   <TradeList>
5     <Trade tradeId="12391677" state="ACTI" contractId="10612678" qty="14000" px="3904" execTime="2018-10-05T16:02:03.767Z" revisionNo="1" preArranged="false"
6     contractPhase="CONT" decomposed="false">
7       <Sell clearingAcctType="P" dlvrAreaId="10YDE-VE-----2" acctId="TESTBG1-----" ordId="114267769" txt="" aggressorIndicator="Y" usrCode="TRD001"
8       clOrdId="1d983af5-ee8a-412d-a18b-7ba551e818b5" mbrId="TEST1"/>
9     </Trade>
10  </TradeList>
11 </TradeCaptureRprt>
    
```

# Cash limit

# Trading limits (cash limit)

CashLmtRprt	
Type:	Inquiry Response; Broadcast
Response to:	CashLmtReq (sent to private response queue see 3.1)
Broadcast:	Yes
Broadcast Routing Keys:	<schema-version>.mbr.<mbrId>
Broadcast audience:	All users from particular member Admins Brokers with assigned members Clearing users
Roles:	Trader, Market Operation, Broker,



# Trading limits (cash limit)

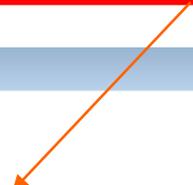
- **EUR and GBP trading limits are distinct**
- **Cash Limit Report:** both GBP and EUR current and initial TL are returned after a request, or in case the member TL is modified by market ops

```
CashLmtReq
1 <?xml version="1.0" encoding="UTF-8"?>
2 <CashLmtReq xmlns="http://www.deutsche-boerse.com/m7/v6" mbrId="TEST1">
3   <StandardHeader marketId="EPEX"/>
4 </CashLmtReq>
5
```



Current TL

```
CashLmtRprt
<?xml version="1.0" encoding="UTF-8"?>
<CashLmtRprt xmlns="http://www.deutsche-boerse.com/m7/v6" mbrId="TEST1">
  <StandardHeader marketId="EPEX"/>
  <CurrentCashLmtList>
    <CurrentCashLmt currentCashLmtRevision="1208" currentCashLmt="20000000000" currency="EUR"/>
    <CurrentCashLmt currentCashLmtRevision="169" currentCashLmt="599907534" currency="GBP"/>
  </CurrentCashLmtList>
  <CashLmtList>
    <CashLmt lmtId="507" state="ACTI" startDate="2016-12-14T23:00:00.000Z" revisionNo="1" cashLmt="20000000000" decShft="2" currency="EUR"/>
    <CashLmt lmtId="591" state="ACTI" startDate="2017-09-25T22:00:00.000Z" revisionNo="1" cashLmt="6000000000" decShft="0" currency="GBP"/>
  </CashLmtList>
</CashLmtRprt>
```



- **Broadcast:** only the current TL update is broadcasted « Cash Limit Delta Report »

GB order entry

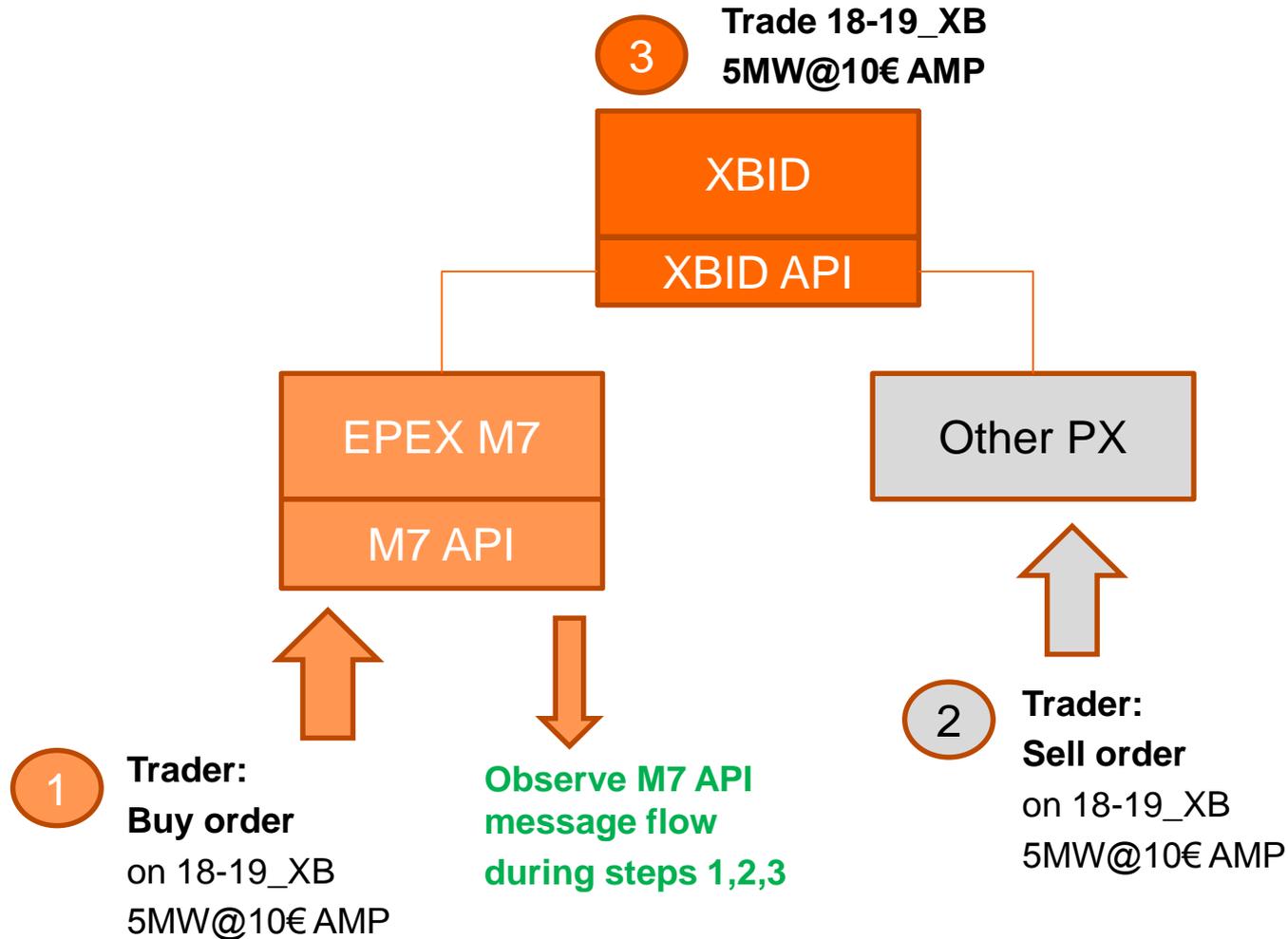


```
CashLmtDeltaRprt
<?xml version="1.0" encoding="UTF-8"?>
<CashLmtDeltaRprt xmlns="http://www.deutsche-boerse.com/m7/v6" mbrId="TEST1"
revisionNo="170" cashLmt="599820096" decShft="2" currency="GBP">
  <StandardHeader marketId="EPEX"/>
</CashLmtDeltaRprt>
```

# Appendix - Use case

# API Messages Illustration with a Use Case

EPEX-Other PX trade seen from M7 (1/7)



# API Messages Illustration with a Use Case

## EPEX-Other PX trade seen from M7 (2/7)

**1** In M7: Buy order 5MW@10€ AMP on 18-19\_XB (contractId="47994")

In ComTrader: use CTRL+ALT+R to display the “Recorded messages” panel:

Client Date	Process Date	RTT	Direction	Routing Key	Correlation ID	Request Type	XML	Size
23.05.2018 15:37:25	23.05.2018 15:37:25	32	↓	6_0.prddlvr.XBID_Hour_Power.10YDE-EON-----1	48eac199-516a-	PblcOrdrBooksDeltaRprt	Q	539
23.05.2018 15:37:25	23.05.2018 15:37:25	29	↓	6_0.prddlvr.XBID_Hour_Power.10YDE-VE-----2	48eac199-516a-	PblcOrdrBooksDeltaRprt	Q	539
23.05.2018 15:37:25	23.05.2018 15:37:25	28	↓	6_0.prddlvr.XBID_Hour_Power.10YDE-RWENET---I	48eac199-516a-	PblcOrdrBooksDeltaRprt	Q	539
23.05.2018 15:37:25	23.05.2018 15:37:25	26	↓	6_0.prddlvr.XBID_Hour_Power.10YAT-APG-----L	48eac199-516a-	PblcOrdrBooksDeltaRprt	Q	438
23.05.2018 15:37:25	23.05.2018 15:37:25	24	↓	6_0.prddlvr.XBID_Hour_Power.10YDE-ENBW----N	48eac199-516a-	PblcOrdrBooksDeltaRprt	Q	539
23.05.2018 15:37:25	23.05.2018 15:37:25	22	↓	6_0.mbr.CENEX	48eac199-516a-	CashLmtDeltaRprt	Q	250
23.05.2018 15:37:25	23.05.2018 15:37:25	18	↓	6_0.bg.CENEXxxxxxxxxxxxx	48eac199-516a-	OrdrExeRprt	Q	760
23.05.2018 15:37:25	23.05.2018 15:37:25	17	↓	m7.private.responseQueue.CXOWEG02.6ada4630-d2da-42b3-bd18-694f323f34dc	48eac199-516a-	AckResp	Q	152
23.05.2018 15:37:25	23.05.2018 15:37:25	0	↑	m7.request.management	48eac199-516a-	OrdrEntry	Q	449

# API Messages Illustration with a Use Case

## EPEX-Other PX trade seen from M7 (3/7)

1 Buy order 5MW@10€ AMP on 18-19\_XB (contractId="47994")

```
<?xml version="1.0" encoding="UTF-8"?>
<OrderEntry xmlns="http://www.deutsche-boerse.com/m7/v6">
  <StandardHeader marketId="EPEX"/>
  <OrderList>
    <Order clearingAcctType="P" acctId="TESTXXXXXXXXXXXX" contractId="47994" prod="XBID_Hour_Power"
      side="BUY" px="1000" qty="5000" orderExeRestriction="NON" dlvrAreaId="10YDE-RWENET---I"
      clOrderId="03461d25-68fd-4481-9d5b-bedf11b7963e" type="O" validityRes="GFS" state="ACTI"/>
  </OrderList>
</OrderEntry>
```

# API Messages Illustration with a Use Case

## EPEX-Other PX trade seen from M7 (4/7)

M7 answer response to order entry: **OrdExeRprt + PblcOrdBooksDeltaRprt:**

```
<?xml version="1.0" encoding="UTF-8"?>
<OrdExeRprt xmlns="http://www.deutsche-boerse.com/m7/v6">
  <StandardHeader marketId="EPEX"/>
  <OrdList>
    <Ordr ordId="52751" initialOrdId="52751" clearingAcctType="P" acctId="TESTXXXXXXXXXXXX" contractId="47994"
    side="BUY" px="1000" qty="5000" initialQty="5000" ordExeRestriction="NON" txt=""
    dlvrAreaId="10YDE-RWENET---I" clOrdId="03461d25-68fd-4481-9d5b-bedf11b7963e" preArranged="false" type="O"
    state="ACTI" usrCode="TRD001" revisionNo="1" timeStamp="2018-05-23T13:37:25.340Z"
    validityDate="2018-05-23T15:30:00.000Z" validityRes="GFS" action="UADD" lastUpdateUsrInfo="TMEPEX-BG1----XOPEPE2"
    counterOrder="false" remoteOrdId="466942" lastUpdateTm="2018-05-23T13:37:25.373Z"/>
  </OrdList>
</OrdExeRprt>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<PblcOrdBooksDeltaRprt xmlns="http://www.deutsche-boerse.com/m7/v6">
  <StandardHeader marketId="EPEX"/>
  <OrdbookList>
    <OrdBook contractId="47994" dlvrAreaId="10YDE-RWENET---I"
    lastPx="200" lastQty="12000" totalQty="12000" lastTradeTime="2018-05-23T13:16:39.617Z"
    pxDir="0" revisionNo="4" highPx="200" lowPx="200">
      <BuyOrdList>
        <OrdBookEntry ordId="52751" qty="5000" px="1000" ordEntryTime="2018-05-23T13:37:25.340Z"/>
      </BuyOrdList>
    </OrdBook>
  </OrdbookList>
</PblcOrdBooksDeltaRprt>
```

# API Messages Illustration with a Use Case

EPEX-Other PX trade seen from M7 (5/7)

**2** Via another PX: Sell order 5MW@10€ AMP on 18-19\_XB



**3** Trade 18-19\_XB  
5MW@10€ AMP

**Information received by M7 trader who owned the buy counterpart order:**

Client Date	Process Date	RTT	Direction	Routing Key	Correlation ID	Request Type	XML	Size
23.05.2018 15:48:47	23.05.2018 15:48:47	105	▼	6_0.prddlvr.XBID_Hour_Power.10YAT-APG-----L	51	PblcOrdRBooksDeltaRprt	Q	537
23.05.2018 15:48:47	23.05.2018 15:48:47	104	▼	6_0.prddlvr.XBID_Hour_Power.10YDE-EON-----1	51	PblcOrdRBooksDeltaRprt	Q	537
23.05.2018 15:48:47	23.05.2018 15:48:47	101	▼	6_0.prddlvr.XBID_Hour_Power.10YDE-ENBW-----N	51	PblcOrdRBooksDeltaRprt	Q	537
23.05.2018 15:48:47	23.05.2018 15:48:47	94	▼	6_0.prddlvr.XBID_Hour_Power.10YDE-VE-----2	51	PblcOrdRBooksDeltaRprt	Q	537
23.05.2018 15:48:47	23.05.2018 15:48:47	87	▼	6_0.prddlvr.XBID_Hour_Power.10YDE-RWENET--I	51	PblcOrdRBooksDeltaRprt	Q	537
23.05.2018 15:48:47	23.05.2018 15:48:47	70	▼	6_0.pblc.trd.XBID_Hour_Power	51	PblcTradeConfRprt	Q	404
23.05.2018 15:48:47	23.05.2018 15:48:47	70	▼	6_0.mbr.CENEX	51	CashLmtDeltaRprt	Q	250
23.05.2018 15:48:47	23.05.2018 15:48:47	63	▼	6_0.hlftrd.CENEXxxxxxxxxxxxx	51	TradeCaptureRprt	Q	640
23.05.2018 15:48:46	23.05.2018 15:48:46	45	▼	6_0.mbr.CENEX	51	CashLmtDeltaRprt	Q	250
23.05.2018 15:48:46	23.05.2018 15:48:46	38	▼	6_0.bg.CENEXxxxxxxxxxxxx	51	OrdRExeRprt	Q	757

# API Messages Illustration with a Use Case

## EPEX-Other PX trade seen from M7 (6/7)

### M7 answer response to Trade execution:

**Own info:** OrdExeRprt + TradeCaptureRprt (local trade Id)

```
<?xml version="1.0" encoding="UTF-8"?>
<OrdExeRprt xmlns="http://www.deutsche-boerse.com/m7/v6">
  <StandardHeader marketId="EPEX"/>
  <OrdList>
    <Order ordId="52751" initialOrdId="52751" clearingAcctType="P" acctId="TESTXXXXXXXXXXXX"
      contractId="47994" side="BUY" px="1000" qty="0" initialQty="5000" ordExeRestriction="NON"
      txt="" dlrvyAreaId="10YDE-RWENET---I" clOrdId="03461d25-68fd-4481-9d5b-bedf11b7963e"
      preArranged="false" type="O" state="IACT" usrCode="TRD001" revisionNo="2" timestmp="2018-05-23T13:37:25.340Z"
      validityDate="2018-05-23T15:30:00.000Z" validityRes="GFS" action="FEXE" lastUpdateUsrInfo="TMEPEX-BG1-----XOPEPE2"
      counterOrder="false" remoteOrdId="466942" lastUpdateTm="2018-05-23T13:48:46.696Z"/>
  </OrdList>
</OrdExeRprt>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<TradeCaptureRprt xmlns="http://www.deutsche-boerse.com/m7/v6">
  <StandardHeader marketId="EPEX"/>
  <TradeList>
    <Trade tradeId="2075" state="ACTI" contractId="47994" qty="5000" px="1000" execTime="2018-05-23T13:48:46.447Z"
      revisionNo="1" preArranged="false" contractPhase="CONT" decomposed="false" remoteTradeId="311916">
      <Buy clearingAcctType="P" dlrvyAreaId="10YDE-RWENET---I" acctId="TESTXXXXXXXXXXXX" ordId="52751" txt=""
        aggressorIndicator="U" usrCode="TRD001" clOrdId="03461d25-68fd-4481-9d5b-bedf11b7963e" mbrId="TESTX"
        remoteOrdId="466942"/>
    </Trade>
  </TradeList>
</TradeCaptureRprt>
```

# API Messages Illustration with a Use Case

## EPEX-Other PX trade seen from M7 (7/7)

### M7 answer response to Trade execution:

**Public info:** Public trade Confirmation report + Public Order book Delta Report

```

<?xml version="1.0" encoding="UTF-8"?>
<PblcTradeConfRprt xmlns="http://www.deutsche-boerse.com/m7/v6">
  <StandardHeader marketId="EPEX"/>
  <TradeList>
    <PblcTradeConf tradeId="2075" state="ACTI" contractId="47994" qty="5000" px="1000"
      tradeExecTime="2018-05-23T13:48:46.447Z" revisionNo="1"
      buyDlvryAreaId="10YDE-RWENET---I" sellDlvryAreaId="10YAT-APG-----L"/>
  </TradeList>
</PblcTradeConfRprt>
  
```

```

<?xml version="1.0" encoding="UTF-8"?>
<PblcOrdrBooksDeltaRprt xmlns="http://www.deutsche-boerse.com/m7/v6">
  <StandardHeader marketId="EPEX"/>
  <OrdrbookList>
    <OrdrBook contractId="47994" dlvryAreaId="10YDE-RWENET---I"
      lastPx="1000" lastQty="5000" totalQty="17000" lastTradeTime="2018-05-23T13:48:46.447Z"
      pxDir="1" revisionNo="5" highPx="1000" lowPx="200">
      <BuyOrdrList>
        <OrdrBookEntry ordrid="52751" qty="0" px="1000" ordrEntryTime="2018-05-23T13:37:25.340Z"/>
      </BuyOrdrList>
    </OrdrBook>
  </OrdrbookList>
  
```

# How to solve connectivity issues in ADV SIMU?

