M7 API Presentation

« An introductory training session on our Intraday Market API » - Light Version

16.01.2023

Sales & Market Data – API Customer Services

part of eex group

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

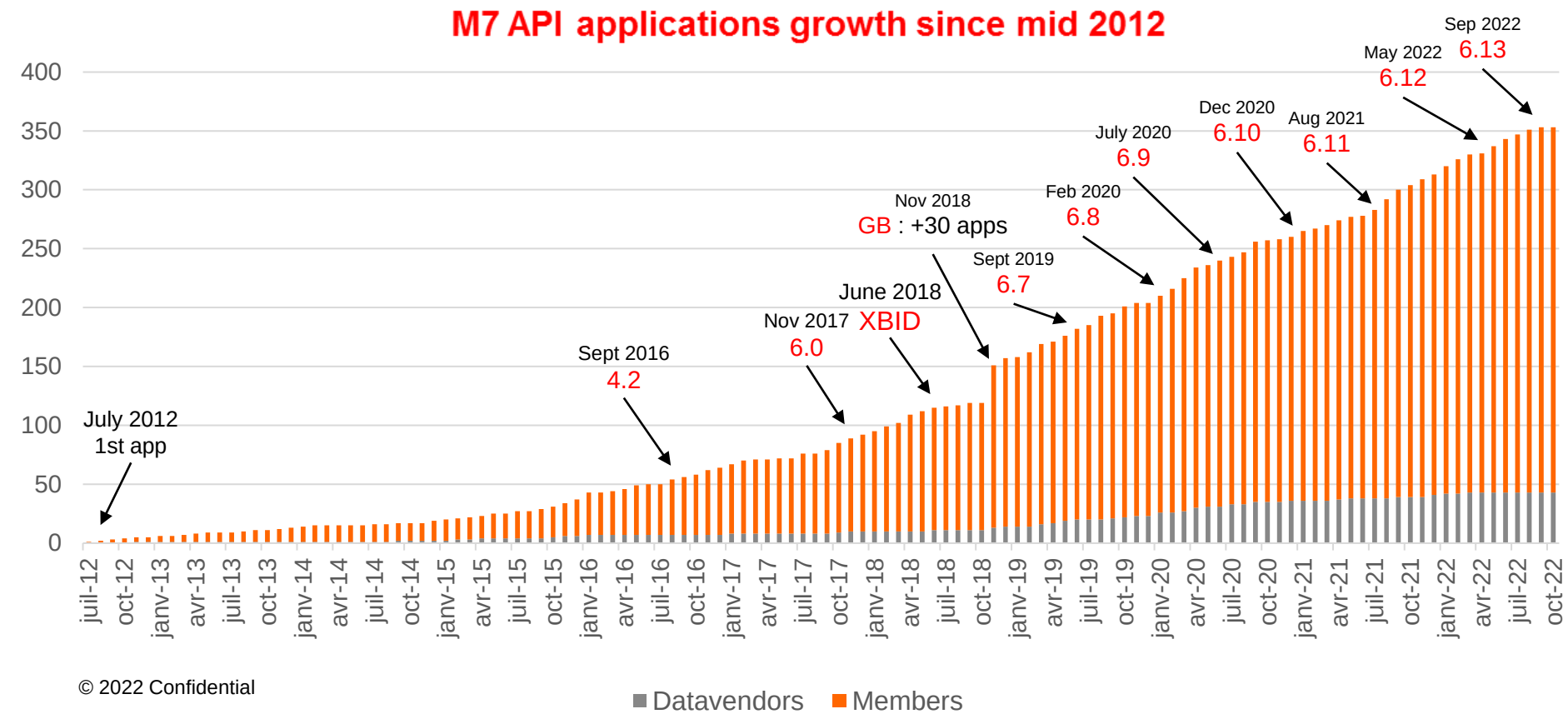
Agenda

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

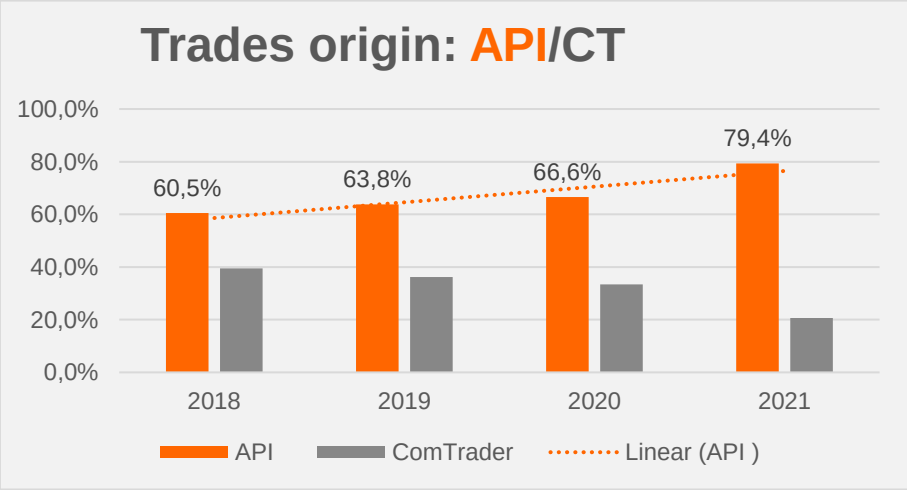
API Background and achievements

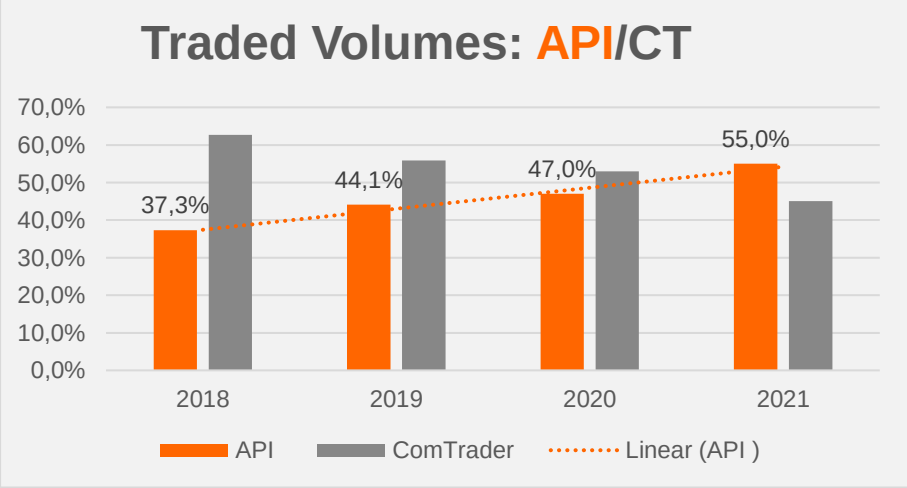
- Today **353 API applications** and **310 members (88%)** are connected to the exchange:
  - following the market and positions (pure read only: 35%)
  - and/or **automating trading strategies**



API Background and achievements

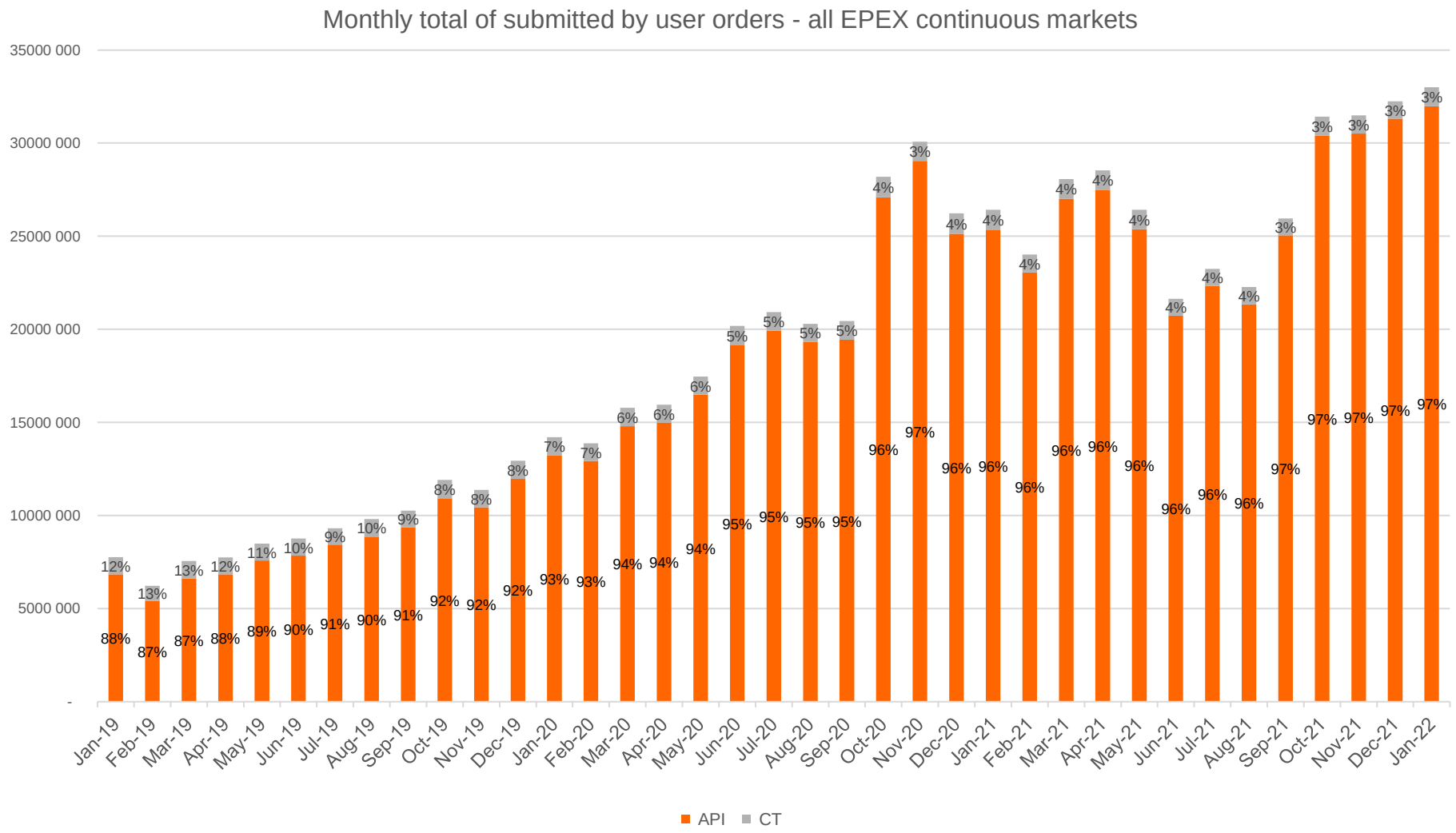
- EPEX has been offering an **API-friendly ecosystem** for almost 10 years.
- **~80% of M7 traded orders come from API apps** (+19% in 2021)
- **~55% of M7 volumes are traded directly via a custom API app** (+17% in 2021)
  - API apps ISV/In-House: 60%-40% (25 M7 “Software Providers”, listed on Epex website)





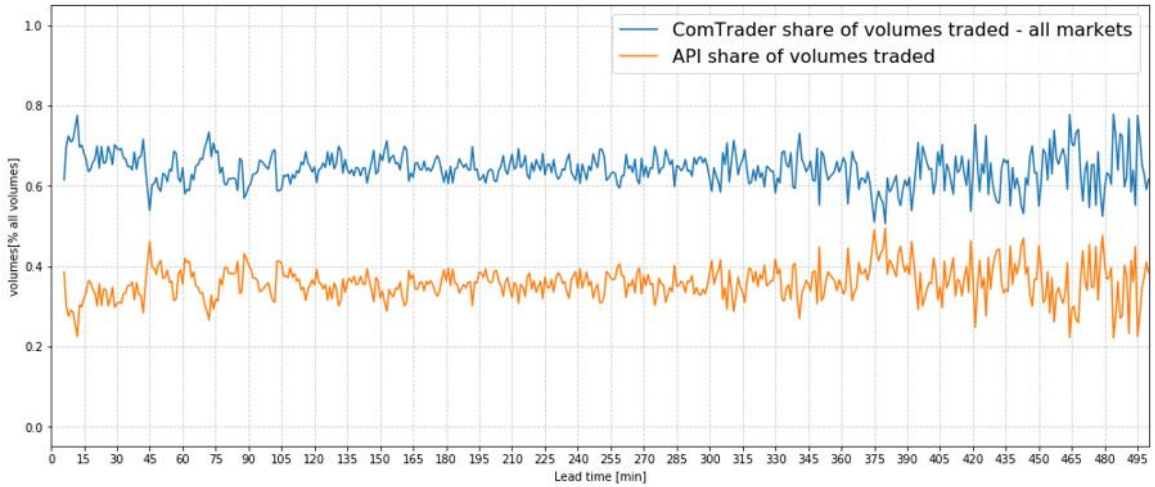
API Background and achievements

- 2019 → 2022: 88% → 97% of orders coming from API applications

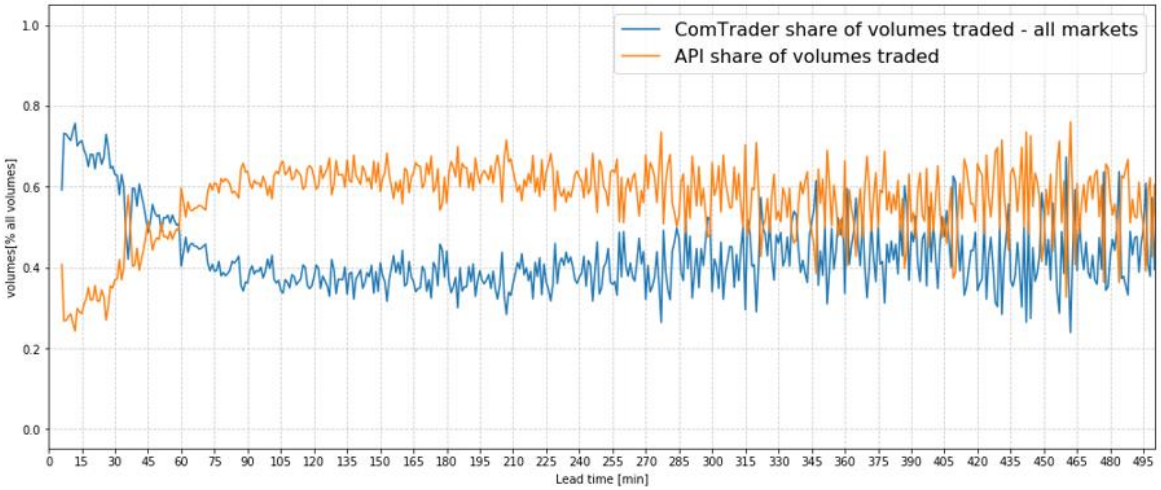


Split of volumes between APIs and ComTrader

1-hour products per lead time



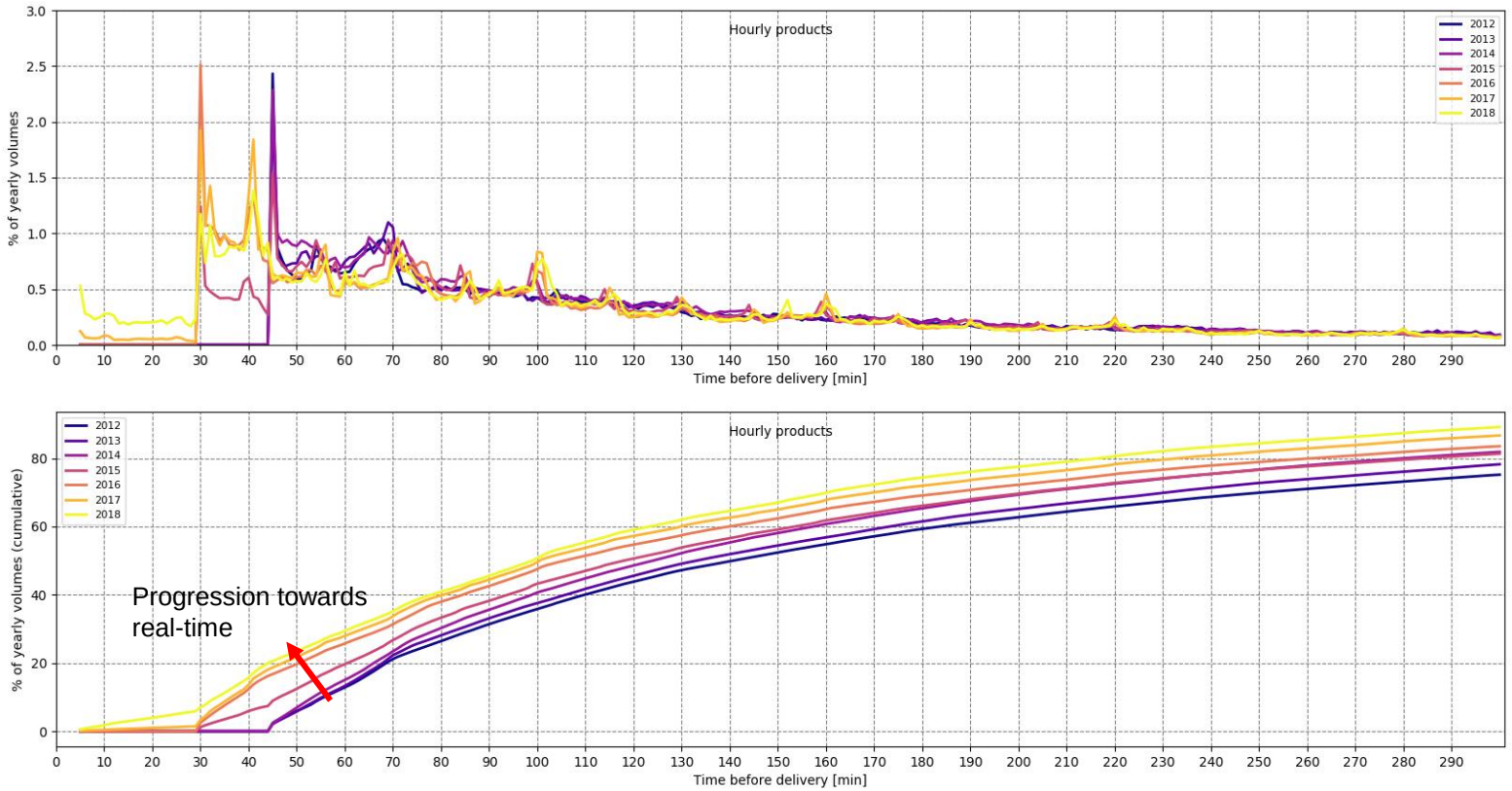
15-min products per lead time



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

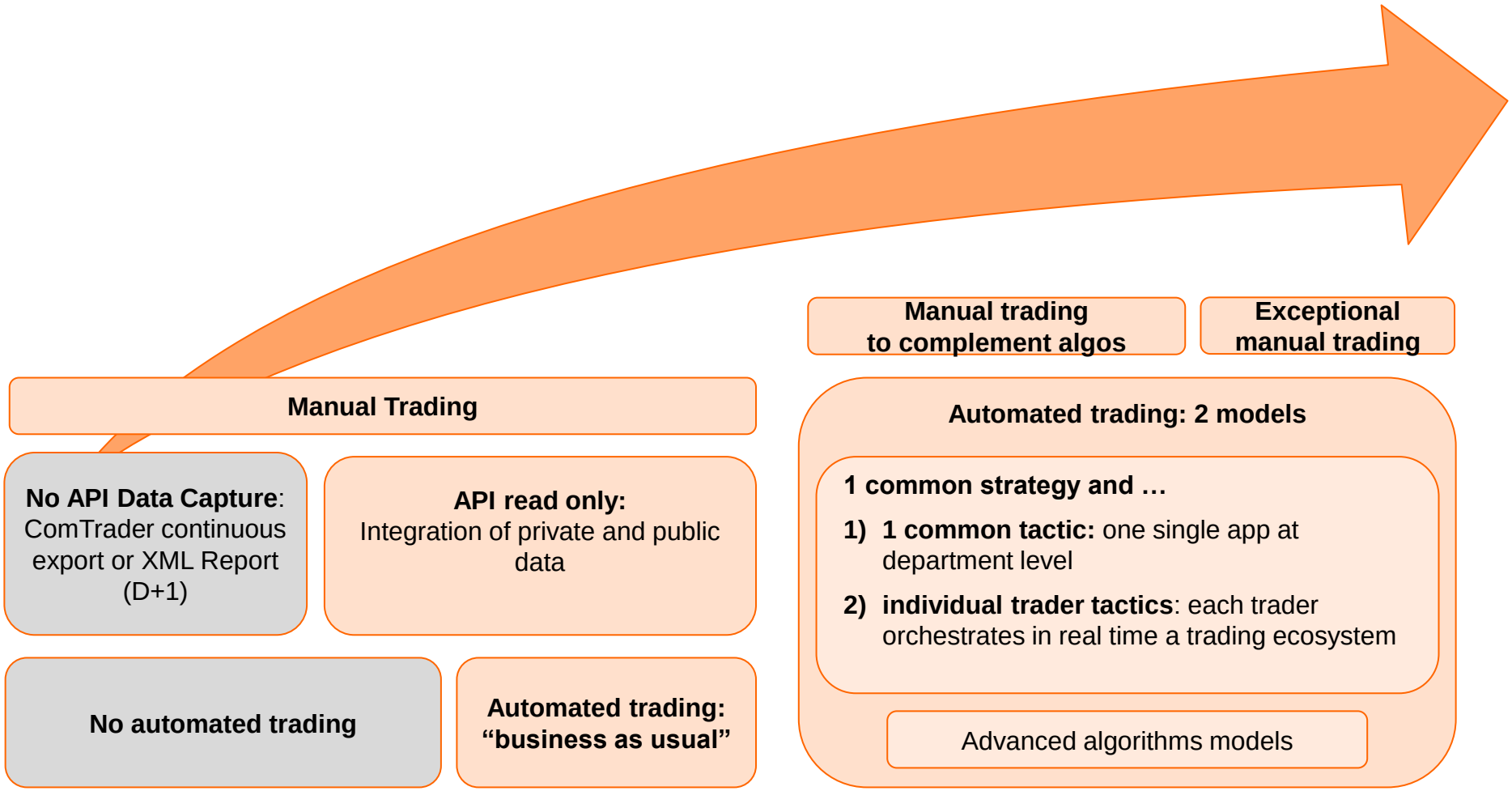
Certified ISVs offer

- **Certified ISV list** : https://www.epexspot.com/en/membership/list_isv

- 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

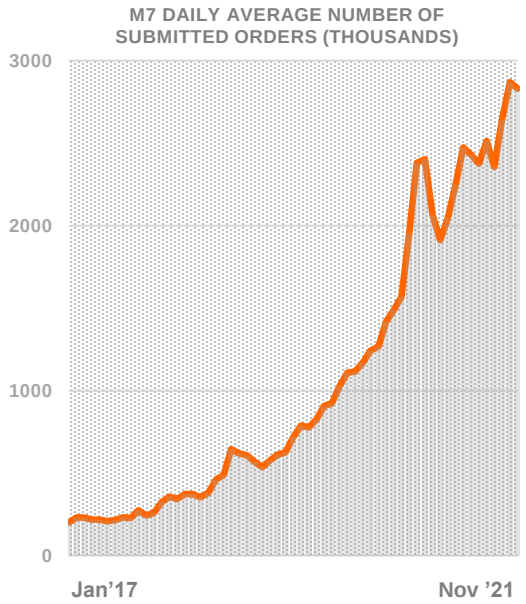
620 TWh
traded in 2021 on all spot markets

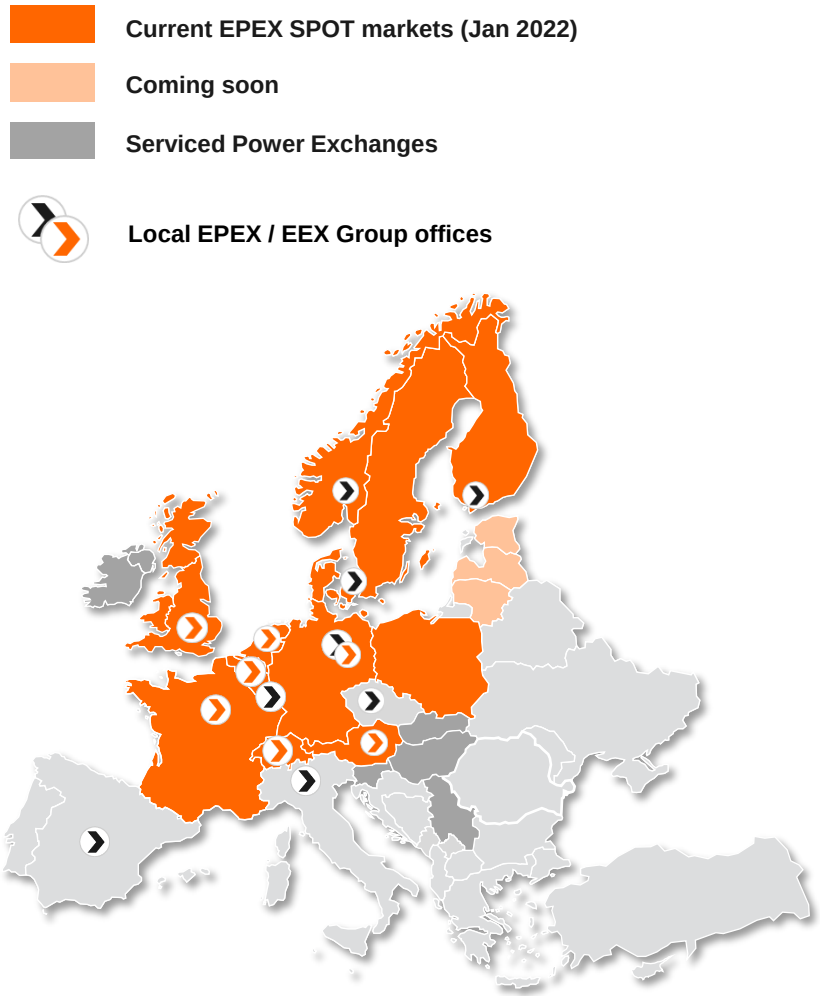
+0.8% Y2Y

Largest Intraday market in Europe with
122 TWh
traded in 2021

+9.9% Y2Y

Proven technical
performance
adapted to new market behavior





Automation Stakes for EPEX?

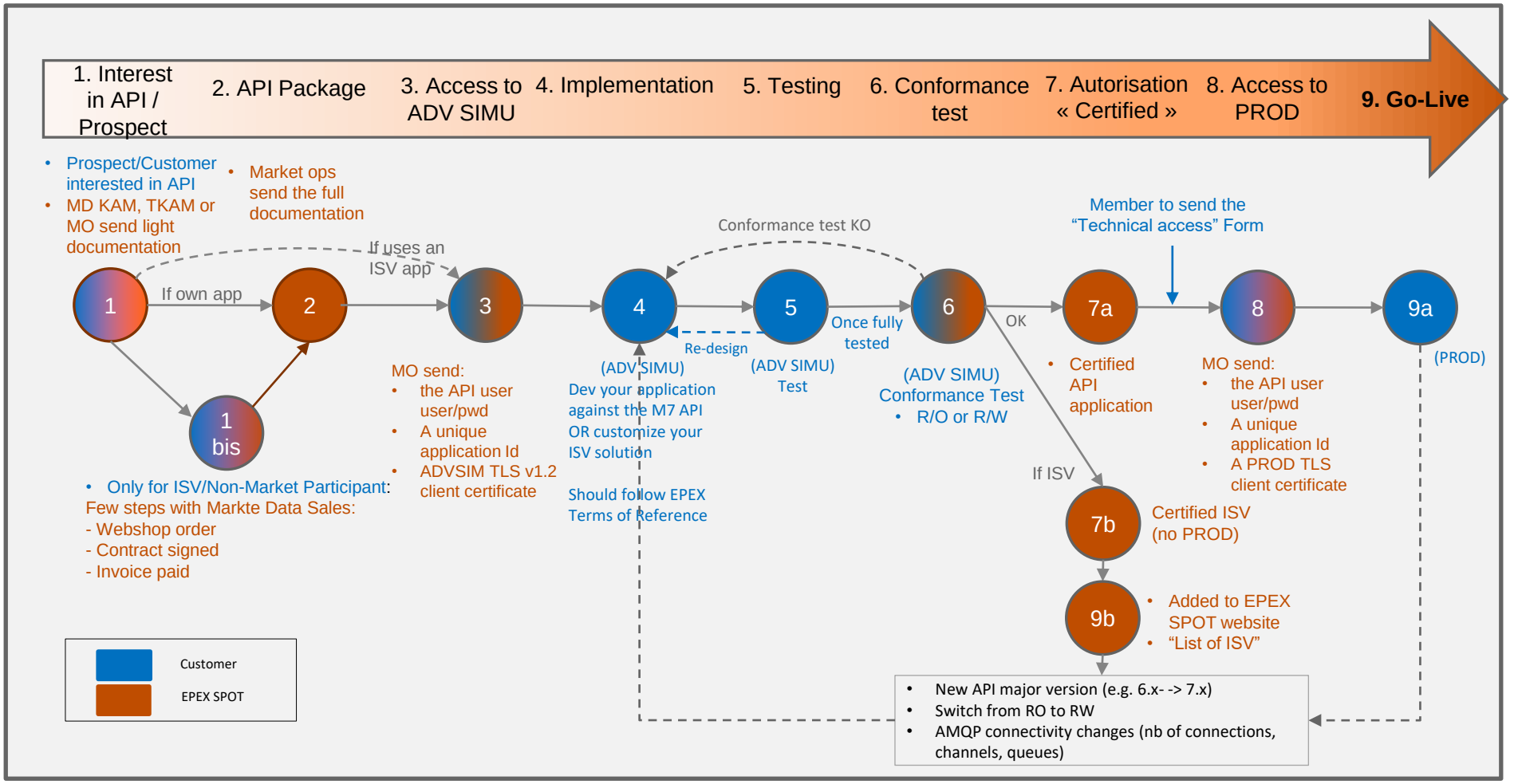
- EPEX is **committed to accompanying the development of trade automation** in the power market.

- 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

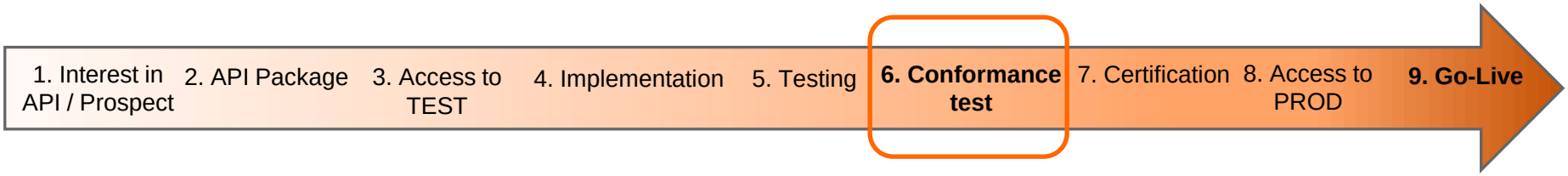
M7 API Implementation Package content

Document	Description
1. DFS180 M7 API	M7 PMI specifications: • AMQP principles • Detailed list of messages
2. M7 – PMI - Terms of Reference	• To ensure a fair and stable use of the M7 Trading System • To prevent from any incorrect implementation that might endanger the M7 Trading System stability.
3. EPEX Environment Details	• Production • Advanced simulation • Simulation
4. M7 API - member procedure	• Describe the actions to be taken by Exchange Members during the Software Implementation Process: focus on Conformance test
5. M7 Public Message Interface FAQ	• This document is an amendment of the DFS180 M7 Public API specification. • It takes over topics and terminology of this specification document.
6. M7 6.0 Static Data	• Market areas • Delivery areas • Products

7. Software Conformance Sheet_60	• Business details
	  • R/O or R/W, automated trading or not, in parallel of manual trading,
	  • Purpose the application, etc.

8. Technical_Terms of Reference Compliance_60	• Customer to confirm having respected points of Terms of Reference (app is fully tested in SIMU env., exp. Back off after connection loss, etc.)
9. Java Sample Code	• Sample API implementation in Java for pedagogic purposes: illustration of main API concepts, including load management (Throttling data)
10. This Powerspoint Presentation	• An introduction to the M7 API to ease the DFS180 understanding

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):

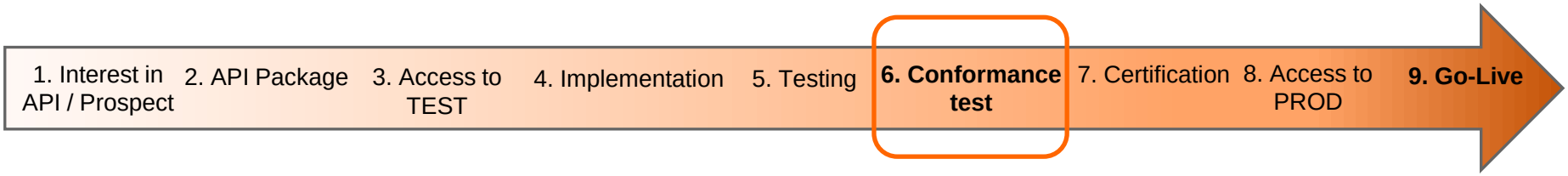
  a) that could lead to service interruptions in production:
  - customer app hitting its quotas of inquiry requests: be unable to trade for up to 60mn
  - non optimal connectivity (no logout after each action, no weird logout/login behavior, etc.)

  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

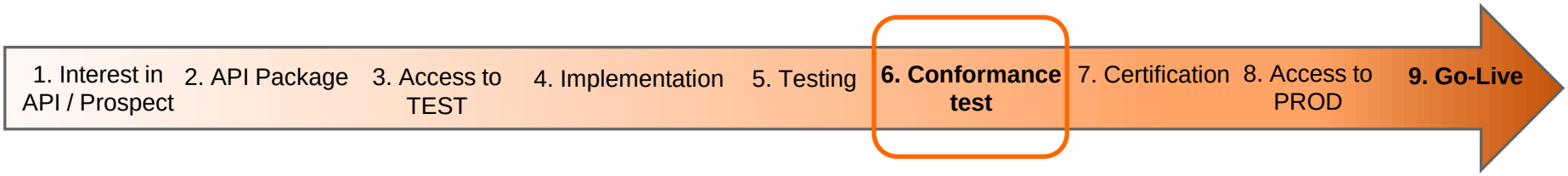
(e.g. for a new M7 major release, a new logic like XBID, etc.)

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)**:**
  - Technical checks (e.g. AMQP nb of queues)
  - Connectivity (login/logout policy matching ToR)
  - Reaction to specific events (Market Halt/Trading, etc.)
  - Reaction to an automated continuous flow of orders and trades
- The **functional testing responsibility** is **on 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).

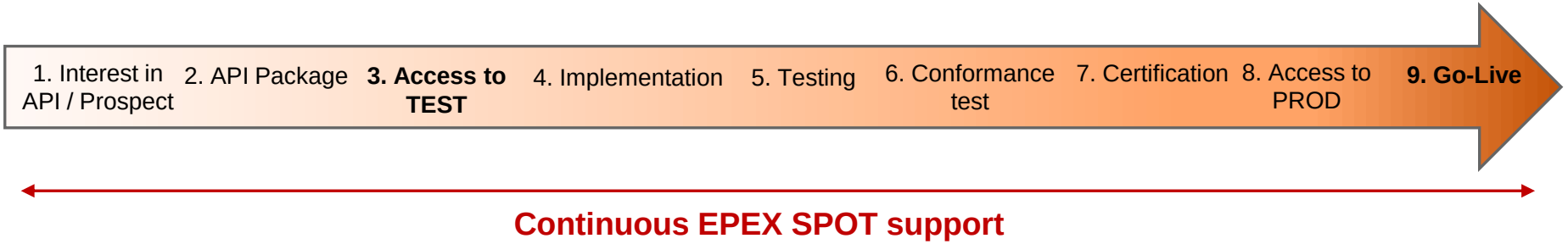
API Applications Conformance Test (2/3)



- **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



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 - XSIM = secondary test environment

- Local trading with the same version as PROD

Iso prod version wise
Local trading first
Connection to XBID in Q1 2023

Customer Test and Conformance Test enhancements

- **Automatic monitoring of test environments is in place**
  - nb of unacknowledged messages, nb of consumers
  - nb of AMQP channels and queues

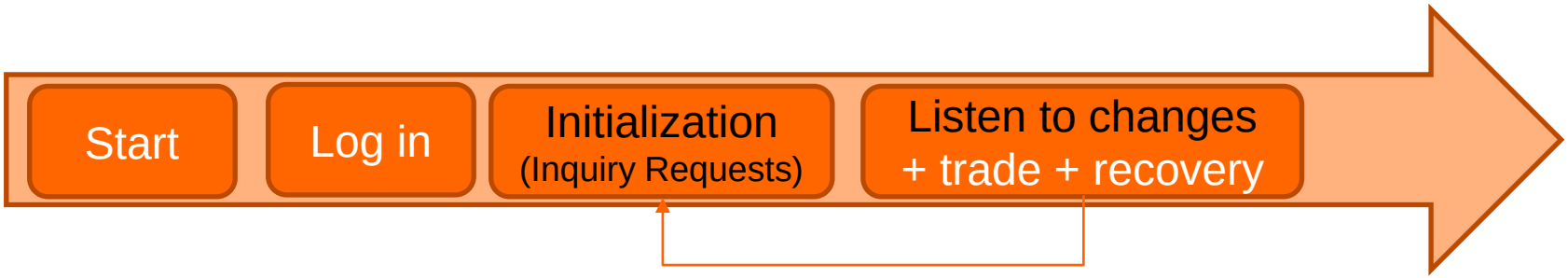
Thresholds have been lowered to ease the detection of potential AMQP resources consumption issue in test.

4. API Functional Overview

API functional overview (trader perspective)

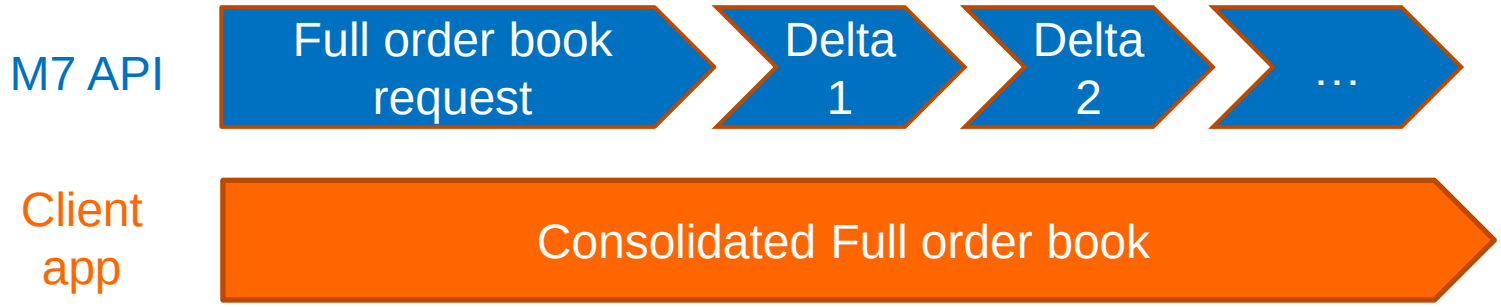
1. **Login / Logout**
2. **Order management (management requests):**
   - **submission, modification, cancellation etc. :** 1 or several at once
   - **On behalf** of (obh) another trader colleague
3. **Trade Management:** recall requests (obh), informed about state changes
4. **Get information about (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
Initialization	• **Login** (including failover logic and disconnection preferences) • **Initialization: Get up-to-date data**: • system info + referential data + market/trading data

Running phase	• **Listen to changes: Keep data up-to-date:**
	  • Integrate referential data changes (e.g. new area assigned to BG)
	  • Integrate market data changes:
	  • *Public data: Contracts, Public Trades, Public Order books, H2H ATC*
	  • *Private data: Own Trades, Own Orders, Trading limit updates*
	• **Trade :**
	  • **Trading strategy:** order calculation (algorithms)
	  • **Orders and trades management** (life cycle: entry -> modif -> etc.)
	• **Recovery processes** (Heartbeats loss, gaps, disconnections)

Closure	• **Logout action** • **Cleaning of the session context** (if a new user logs in, there should be no leftovers from the previous user session)

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: -

	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

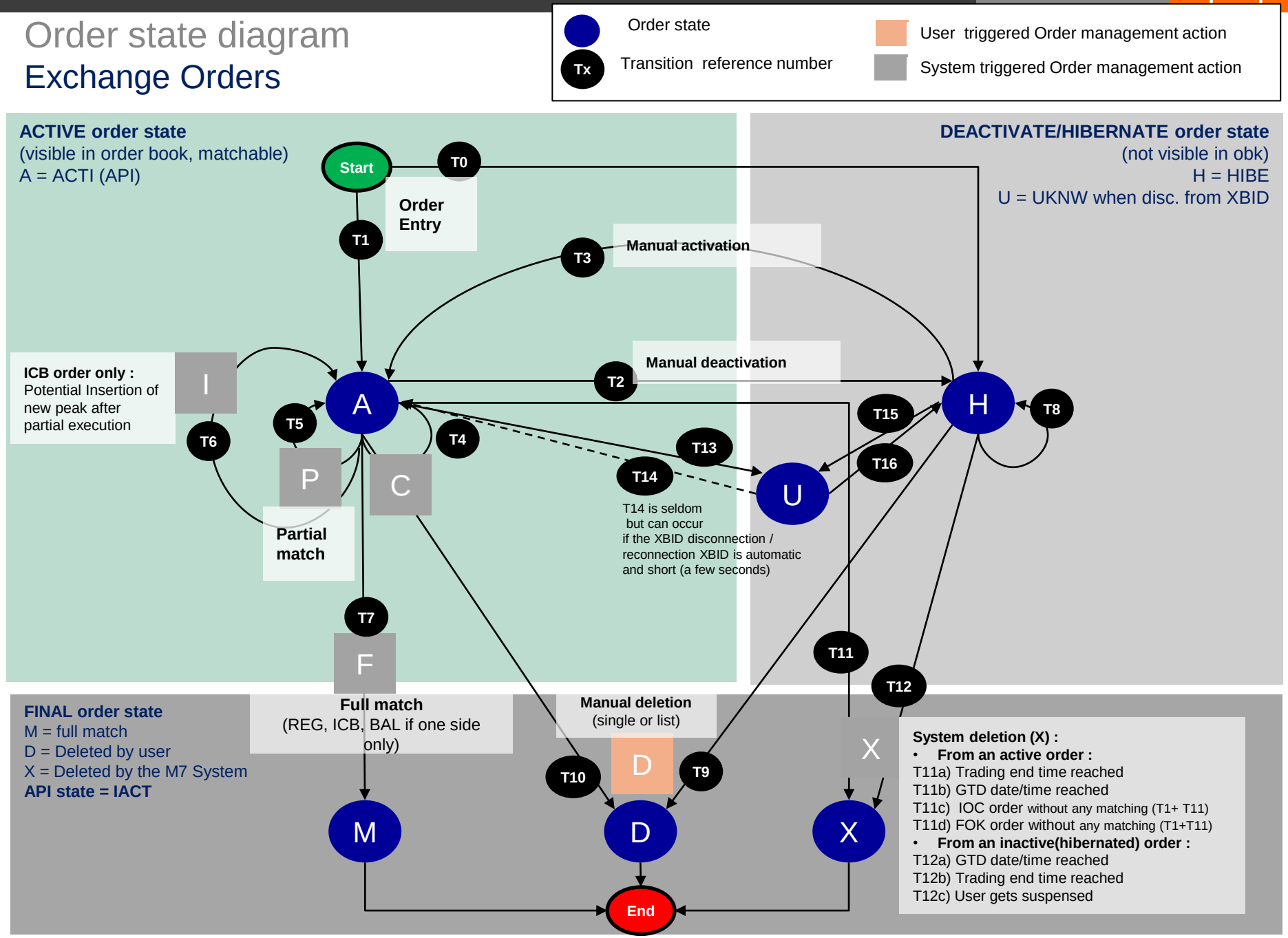
- **Orders can get in (>0 qty) and out (qty =0) of the order book because:**
  - **a) Intrinsic reason: the order has been updated (not applicable to SEMOpx)**
  - **b) OR because of an external reason:**
    - Cross border capacity increasing or decreasing
    - XBID order book depth limitations

- **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?

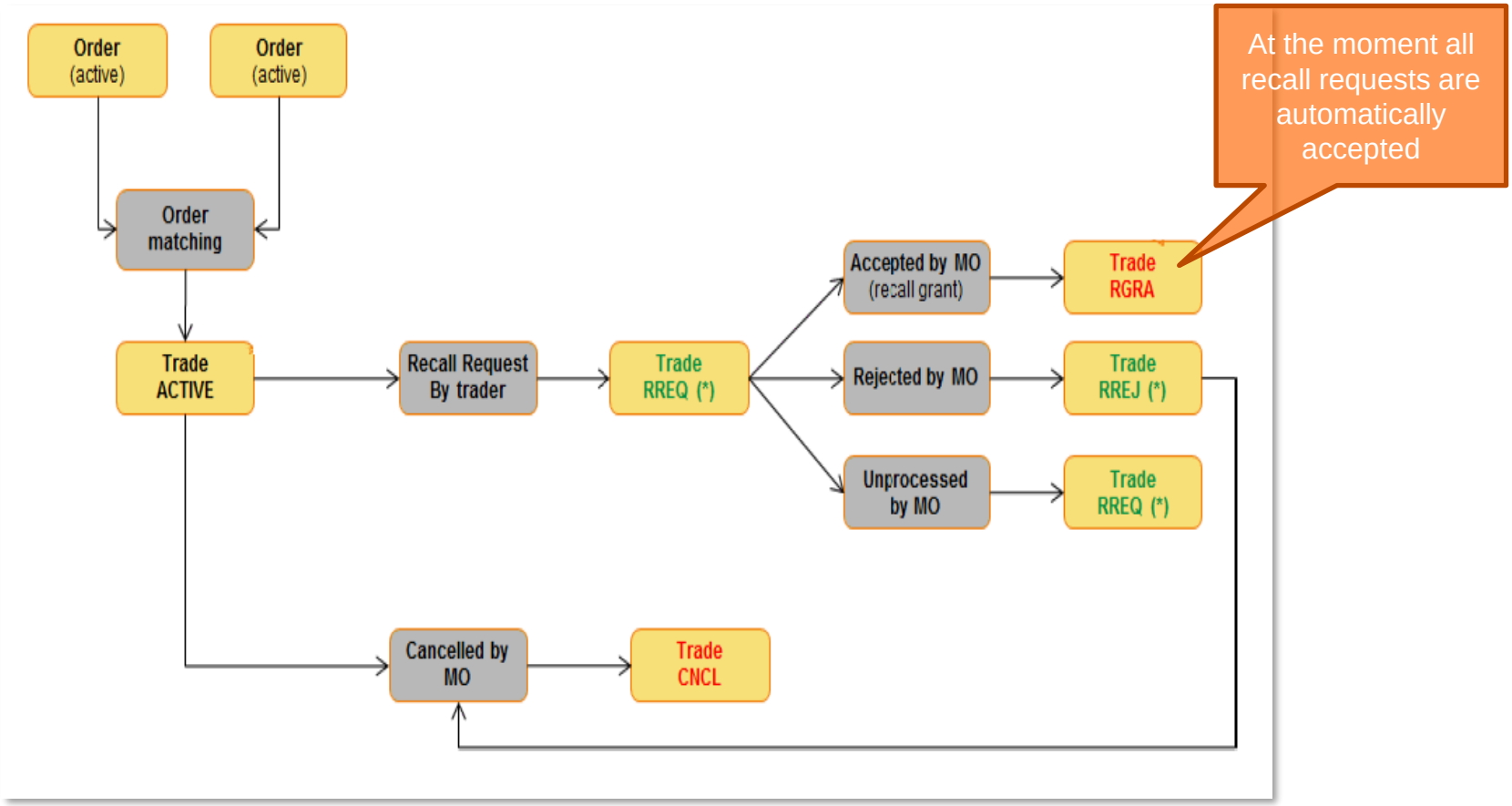
Action	Description
Entry	creation / submission of an active or deactivated order (1 to 100 at once ‘basket’ concept)
Modification	a subset of order characteristics can be modified (price, quantity, etc) (1 to 100 orders at once)
Deactivation	(hibernation): removes the order from order books and from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

Order state diagram
Exchange Orders



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

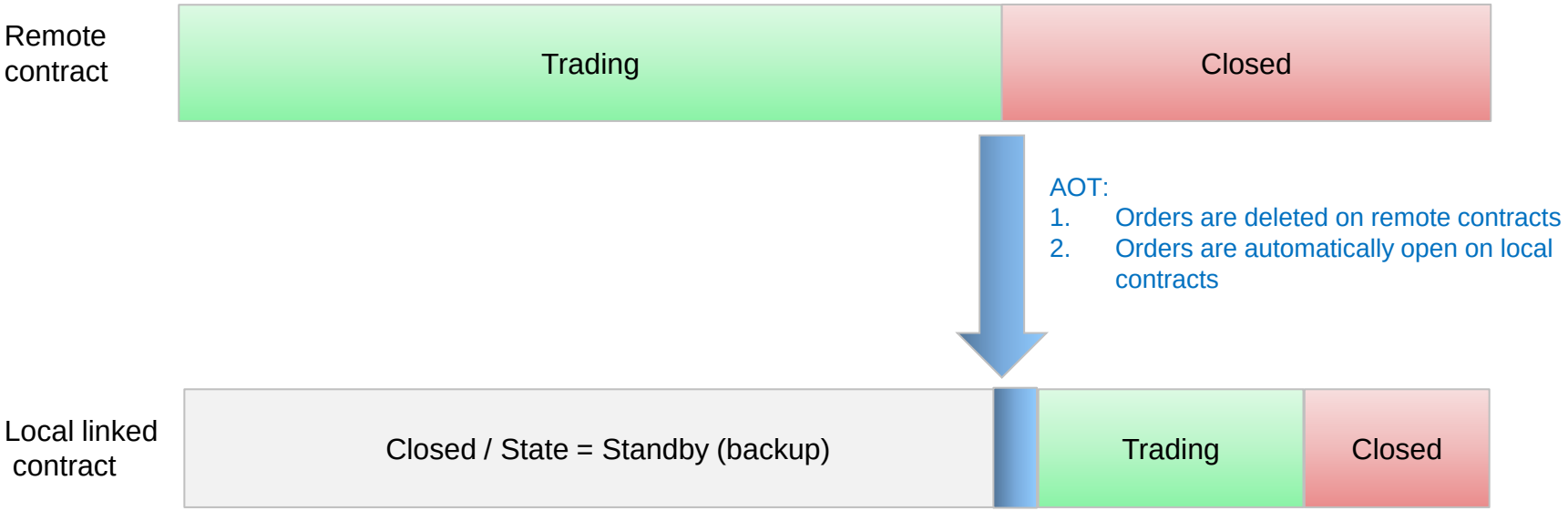
M7 Roadmap

M7 release	Planned Go-Live	Content

M7 6.11	August 2021	• Accompany the market activity exponential growth - Performance enhancements
		• **Private response queues :**
		  • **Automatic expiry (deletion) after 3 minutes** without a consumer (to avoid zombie queues)
		  • New optional private response **queue naming convention** to limit their number to a maximum of ten. Will become mandatory only with M7 6.12.
		• **Broadcast queues Time To Live (TTL) reduction**: 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)

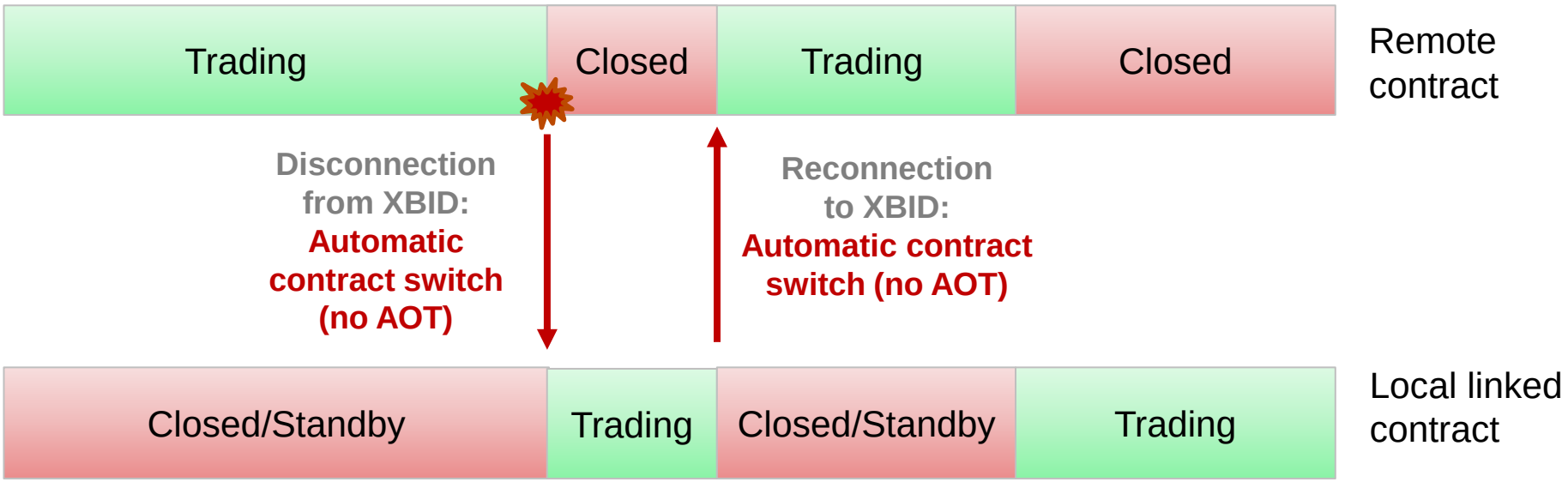
M7 6.12	• **ASIM**: 22 March 2022 • **Technical GL** :end of April / early May 2022 (no warning nor restriction) • **Trial Phase** Q3 2022	• **Accompany the market activity exponential growth: Load Management mechanism** • Please check the dedicated section in the 2nd part of the slides + webinar presentation • **Optional but recommended: max 10 private response queues + new naming convention** (queues with other names cannot be created anymore) **=> will become mandatory only in 6.13**
M7 6.13	13 Sept 2022	• **Mandatory** : **max 10 private response queues + new naming convention** • Throttling Status Req limit in M7 • Requests expiration default value to 45 seconds
M7 6.14	March 2023	• Failover improvement: historical data reduced to today + 4 days in the past (instead of 6) • New attribute in Product Info Reports => new 6.14 API schema/XSD
M7 6.15	Mid 2023	• Performance enhancements

Automatic Order Transfer (AOT/Bid Elevator) at the end of the XBID trading session



AOT applies	AOT does not apply (API apps have to transfer orders if required)
At usual remote contract expiration	Any planned XBID maintenance
	Any unplanned XBID maintenance
	From local products to remote products

Automatic switch of local contracts when XBID is not available



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
  - 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.

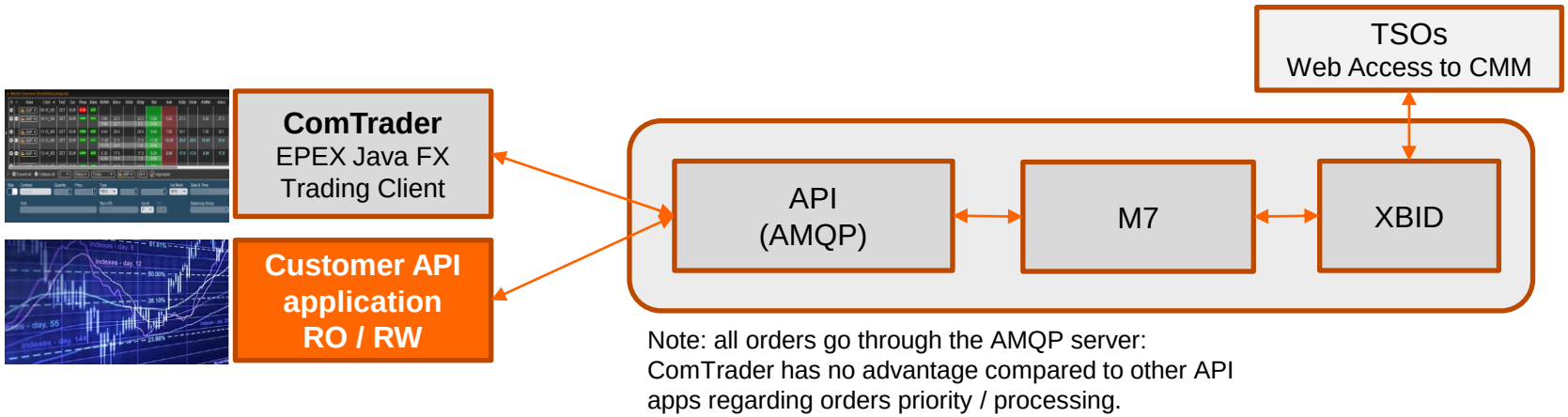
- **API Technical Key Account Manager support:**
  marketdata.technical@epexspot.com +33 1 73 03 61 81
  - Support along your API app implementation and test process
  - API Webinars, Workshops, bilateral meetings on demand
  - Only Business hours

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 application:**

- **Respects our Terms of Reference**
- **Implements the Client Failover**: connect to API end point 1 or 2 randomly
- **Implements the provided Starting sequence**
  - From the AMQP connection, channels creation, private response Q creation,
  - Respect the Inquiry requests rate limits per minute and per hour
  - Once your app start is completed, remain logged in and exclusively rely on real time broadcast.
- **Manages 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: Broadcast Gap detection mechanism + data re-inquiry procedure to find what you potentially missed
  - Reception of a Logout report (because of a concurrent access or if logged out by M7)
  - Inconsistent Revision No (see FAQ)
  - Change Password (automatic expiry every 90 days)

  Please check our DFS180 specs and M7 API FAQ to see more detail which recovery procedure should be implemented
- **If your application submits/manages orders:**
  - Implement the Load Management functionality (Throttling Status and notifciations, order rejection etc.)

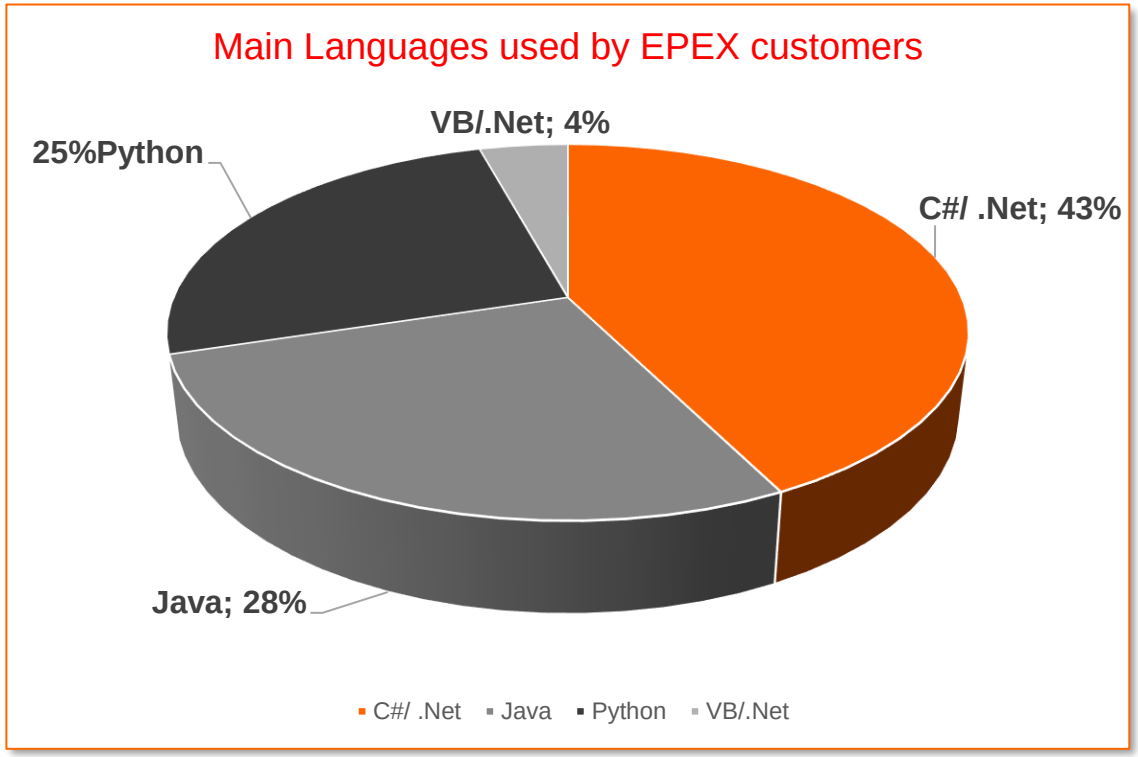
- Once you have fully tested your API app on the ASIM env., **register for an API conformance test 2 weeks in advance**

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

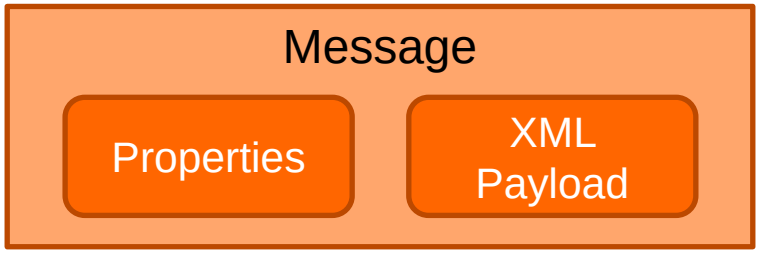
There are 2 kinds of communication between a client and the back-end:

- **Management and inquiry request: send questions to M7:**
  - API client apps send a request and wait asynchronously for a response from the backend system
- **Broadcast messages** sent by the backend to all or specific clients **in real time**

Communication type	Description
Inquiry Request	• **to obtain information on the current state of the market or reference data.** • 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.
Management request	• **used to enter, modify or delete orders or trades** • **Or to change the connected API user password**

Broadcast	• the backend system **publishes notifications** that are:
	  • **public**: addressed to all traders
	  • or **private**: only receivable by privileged traders (see Routing Keys).
	• **initiated by the backend system** (e.g. contract opening, order book change)
	
	• **Sequence counting**: (recovery mechanism to be implemented on client side)
	  • to identify the order of the broadcasts
	  • and to find out if some broadcasts have been lost (“real Gap”)

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): ▪ x-m7/request; version=x (Used by the clients when sending requests) ▪ x-m7/response; version=x ▪ x-m7/broadcast; version=x ▪ x-m7/heartbeat; version=x ▪ x-m7/error; version=x
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 ***gzip***, 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) **app-version**:version where version must be in integer(s) format optionally separated by dots (*ie. 10 or 4.1.5*)

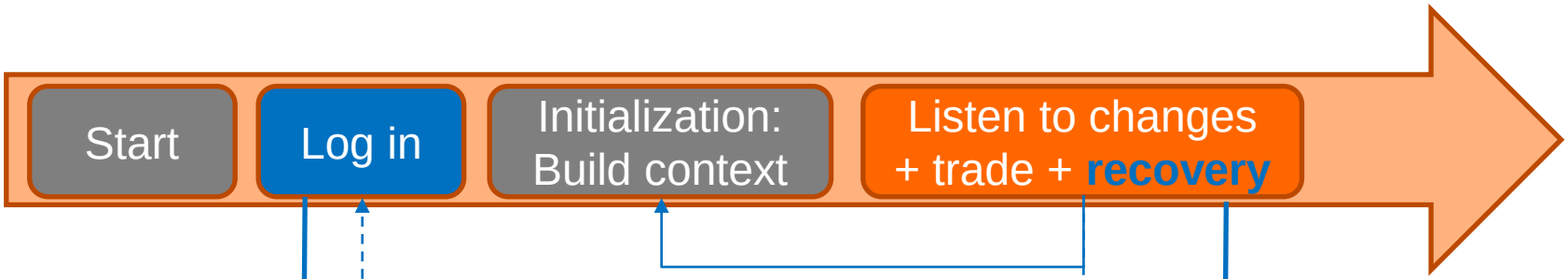
API XML schema version

- 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:
  - DFS180 explains how to implement a ***Forward Compatibility***: when a new version introduces a new attribute, it can be ignored when checking the message against the XSD file.
  - We explicitly mention in our communications which feature in a new version requires a mandatory implementation (functionality and XSD file).

Key technical concepts (1/3)



Optional Logout

- **Failover concept**

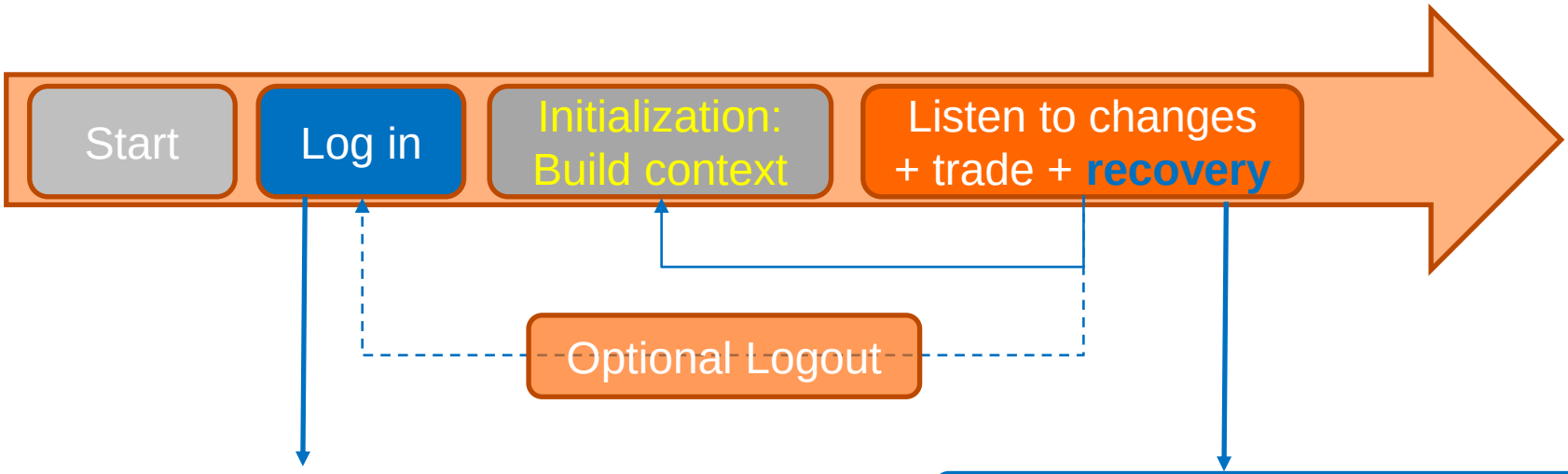
- **Heartbeat concept**
- **Gap concept**

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)



- **Failover concept**

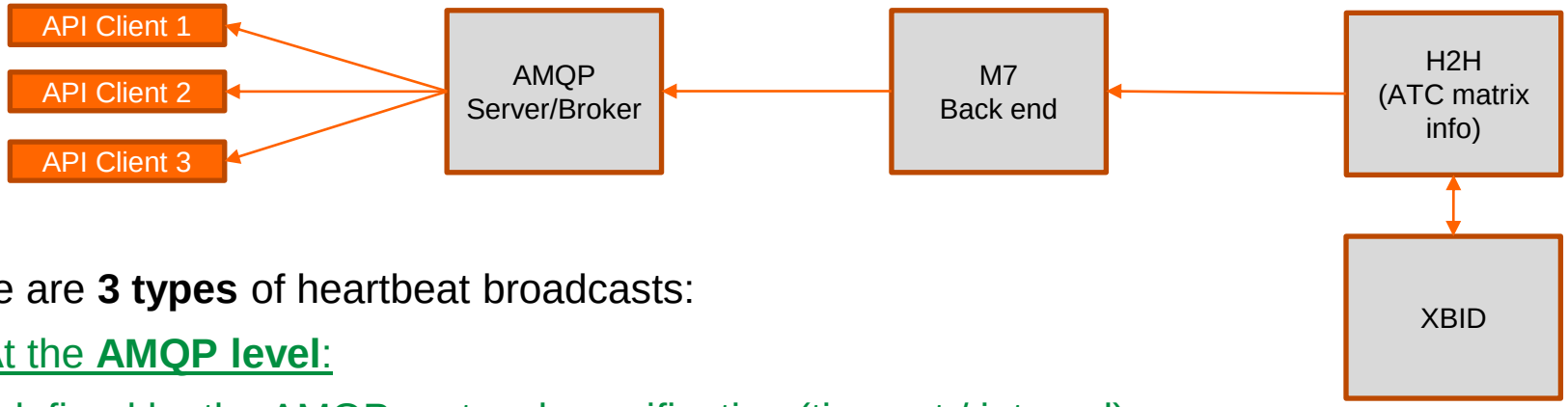
- **Heartbeat concept**
- **Gap concept**

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 AMQP 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.

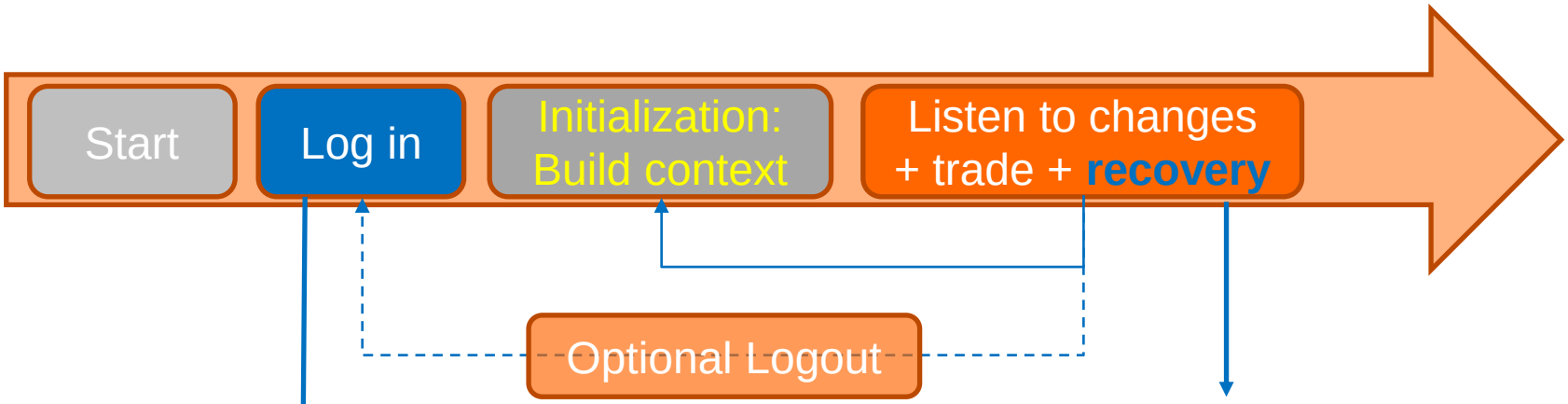
Heartbeats 2/2

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

When our API app **misses 3 heartbeats in a row (15 seconds + margin):**

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 1st 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

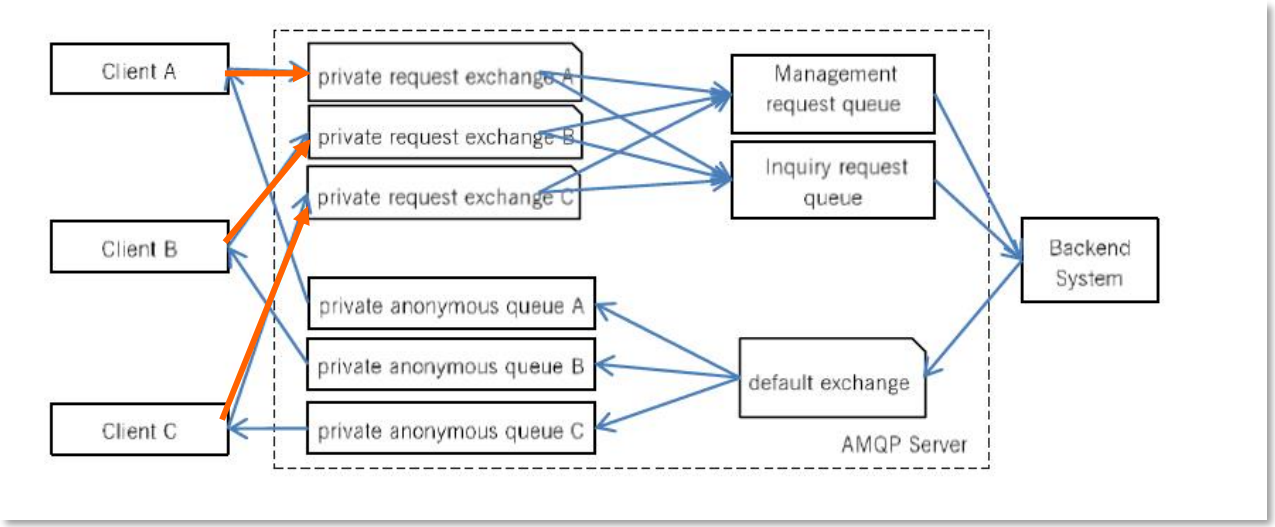
AMQP concepts:

- 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 are processed by a “consuming” application
  - NB: messages have a Time To Live value =180 seconds in the M7 broadcast queue

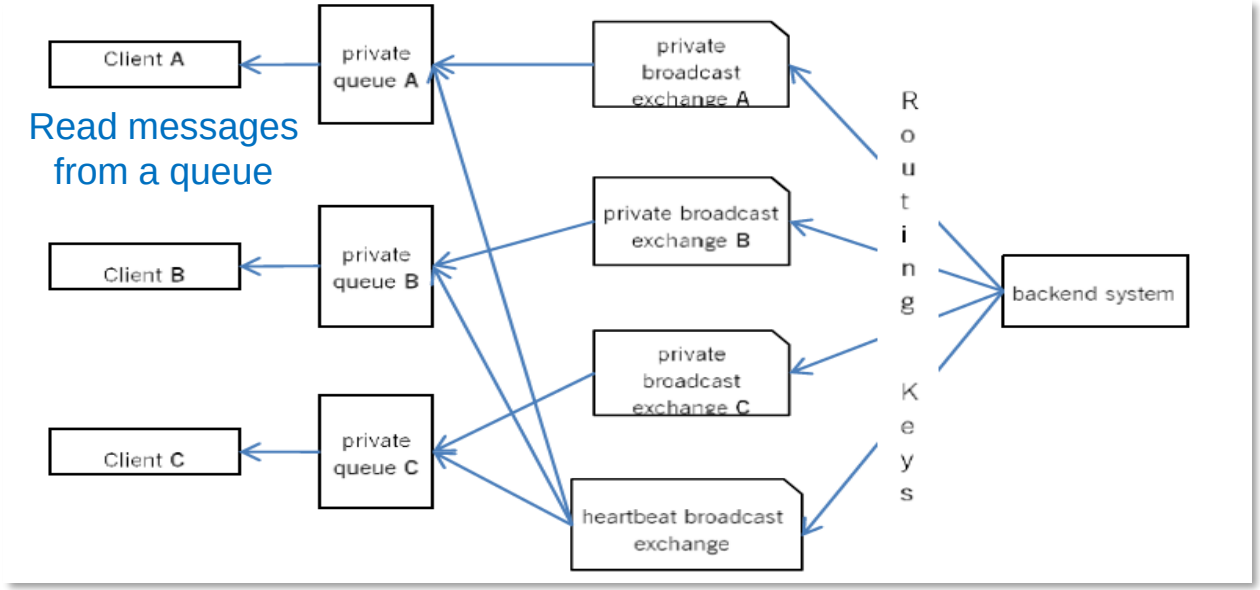
AMQP PRINCIPLE: messages must be:
- **sent to an Exchange**
- **read from a Queue**

AMQP Queues basics

Requests:
sent to an exchange



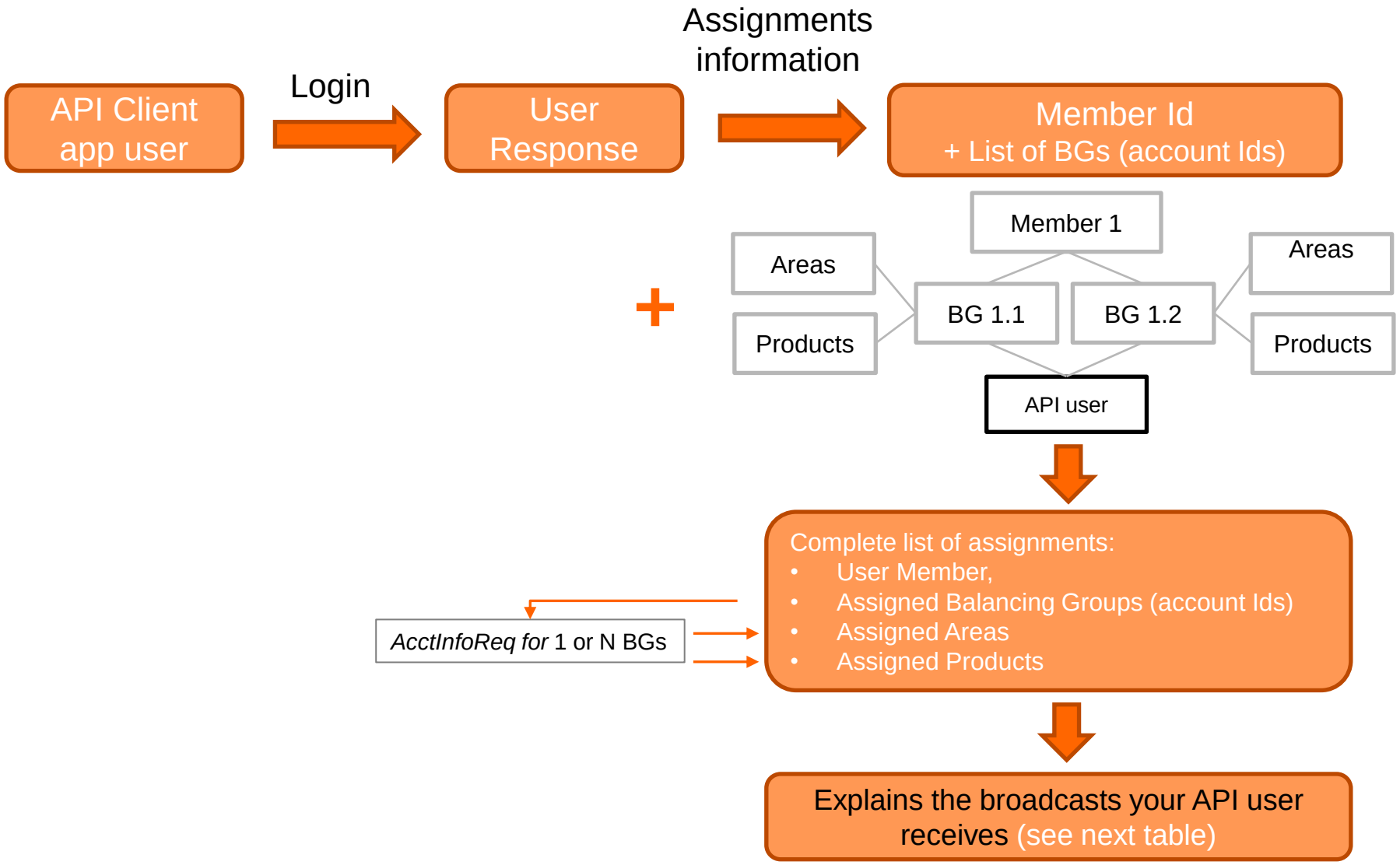
Read messages from a queue



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

API Client app user

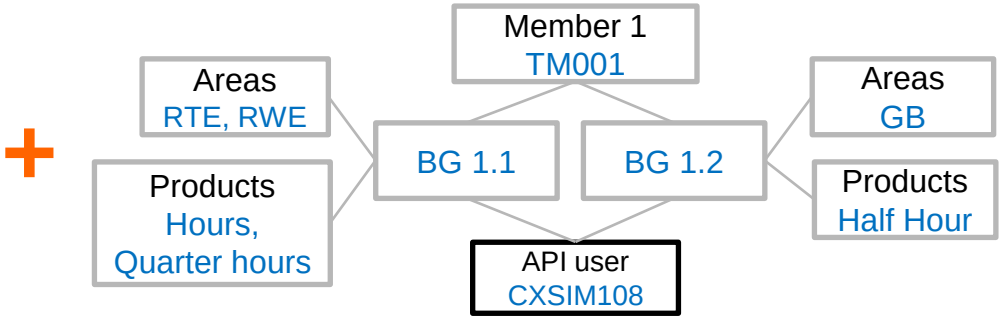
Login

User Response

Assignments information

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

Member Id
+ List of BGs (account Ids)



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**

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.

Information distribution rules: « Broadcasting rules »

Routing Keys

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

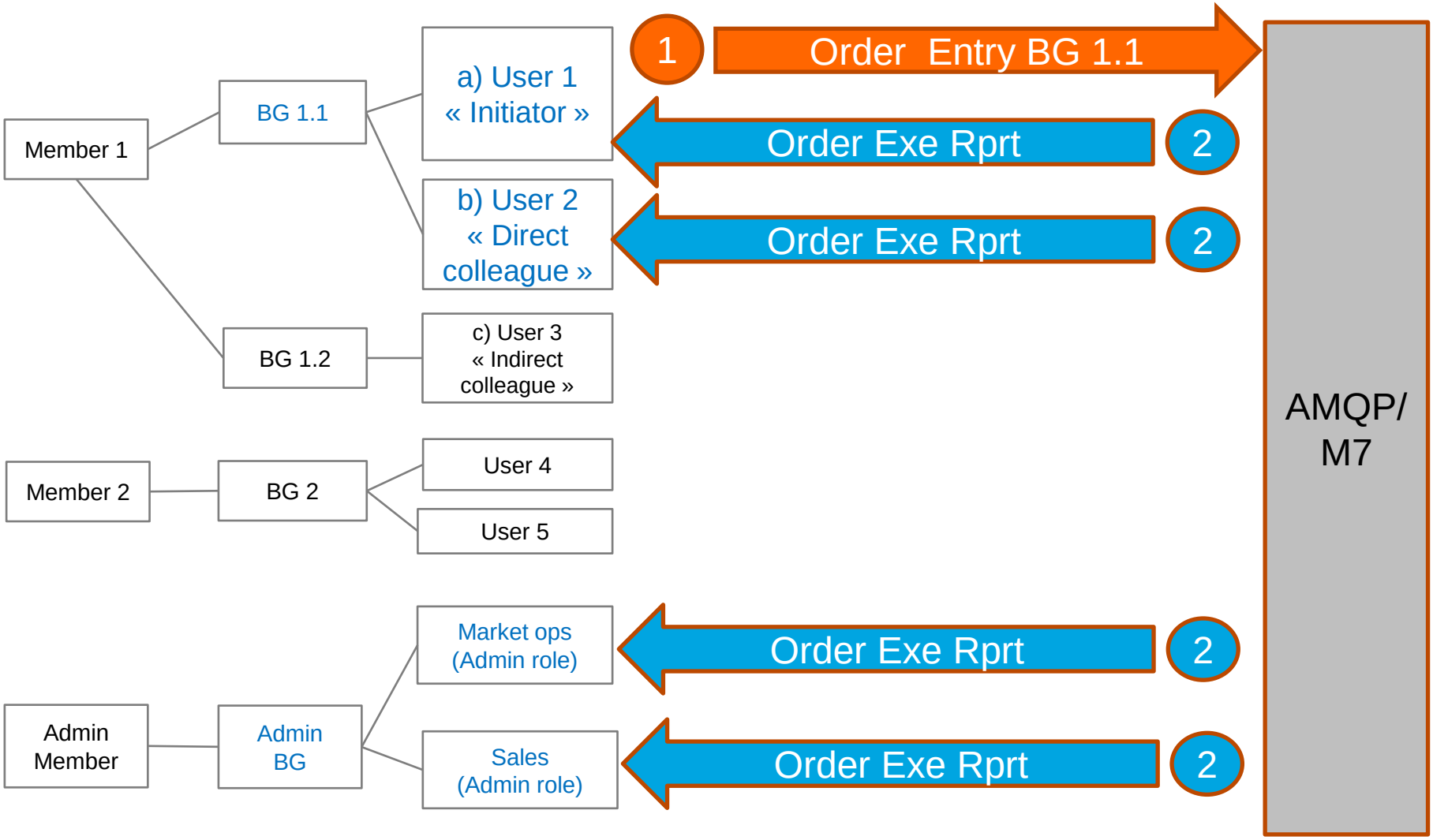
6.2.4 Order Execution Report (OrdrExeRprt)

OrdrExeRprt	
Type:	Management Response; Broadcast
Response to:	OrdrEntry; OrdrModify; OrdrReq; ModifyAllOrdrs; (sent to private response queue see 3.1)
Broadcast:	Yes
Broadcast Routing Keys:	<schema-version>.bg.<acctId>
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		
DlvryAreaInfoRprt		
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
PblcOrdrBooksDeltaRprt	**<schema-version>.prddlvr.<prodName>.<dlvryAreaId>** 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.<marketAreaId>**	All users assigned to the market area (via one of its delivery area)
MsgRprt	**<schema-version>.dlvr.<dlvryAreaId>**	All users assigned to the delivery area
MsgRprt	**<schema-version>.mbr.<mbrId>**	All users of the member
UserRprt		
CashLmtRprt		
CashLmtDeltaRprt		
OrdrExeRprt	**<schema-version>.bg.<acctId>**	All users assigned to the BG
MsgRprt	**<schema-version>.prvt.bg.msg.<acctId>**	All users assigned to the BG
TradeCaptureRprt	**<schema-version>.hlftrd.<acctId>**	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	
HubToHubNtf		

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 (family of messages sharing the same sequence nb)

- 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 <> 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>.<dlvryAreaId>:
  - 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 < [new sequence nb for a given routing key]<[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 occured.

Broadcast messages order

- The Sequence nb existence implies that messages are expected in a certain order, per routing key.
- More exactly they must be processed by API apps in an expected order, other wise data reliability gets broken.
- BUT there is no guarantee about messages reception order across several routing keys
- M7 6.6 increased the situation where the sendng 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 since M7 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

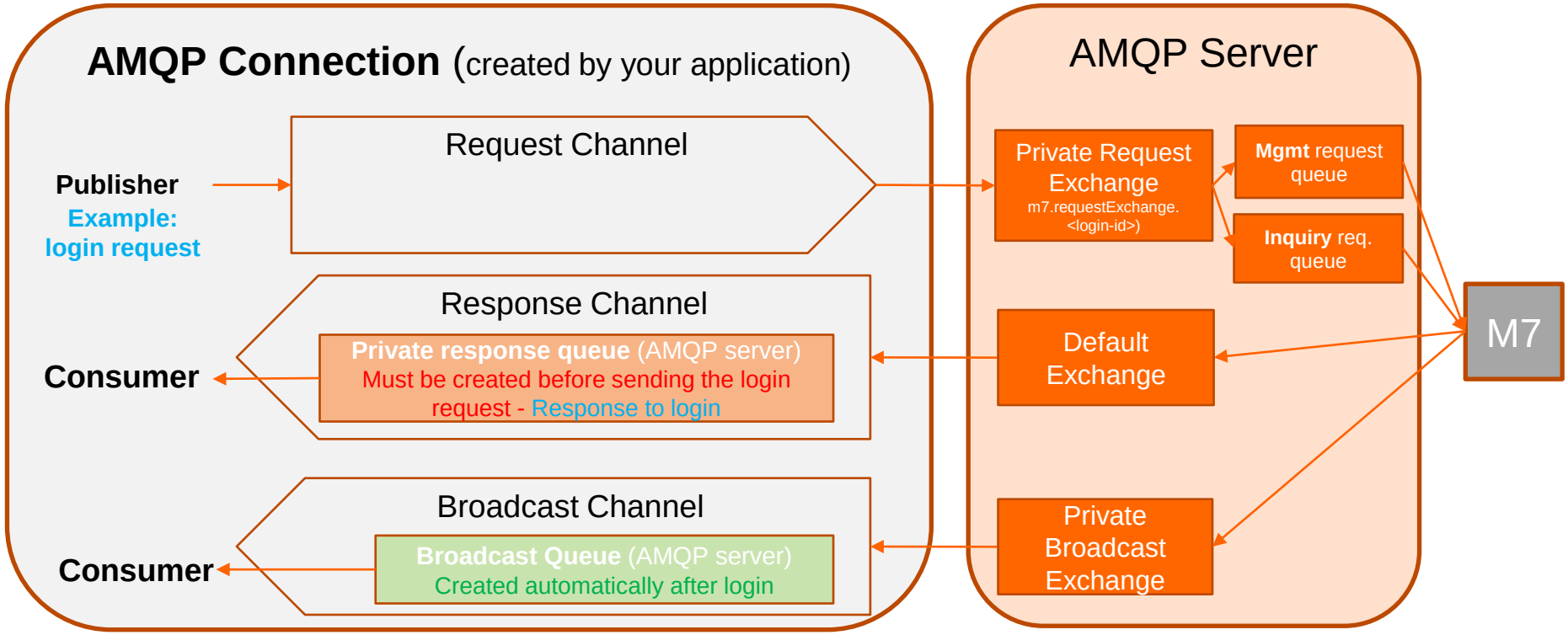
Caution: **public data may have a different correlation Id** in the rare cases where M7 would merge several (customer) order events in the same Public Order Books Delta Report: your broadcasted private data will always have the correlation Id you sent in your order management request, but this can not be the case for public broadcasts

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: URL1 and URL2
  - Rabbit User: your M7 API user
  - 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



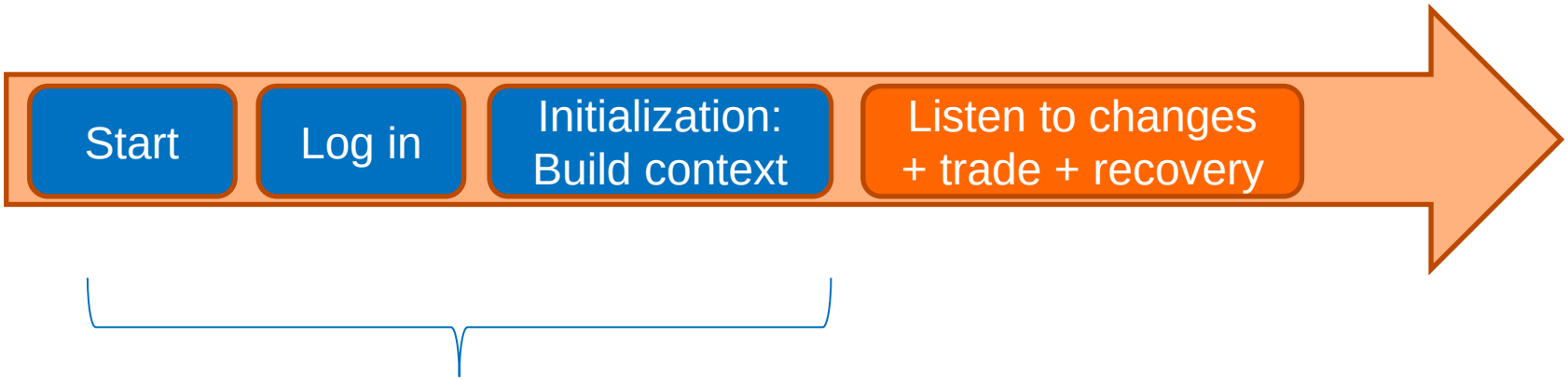
Recurrent questions from customers

- **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

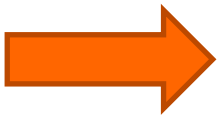


- **Proper login/start Procedure**

2. API Functionalities and messages

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-4), 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? (3/3)

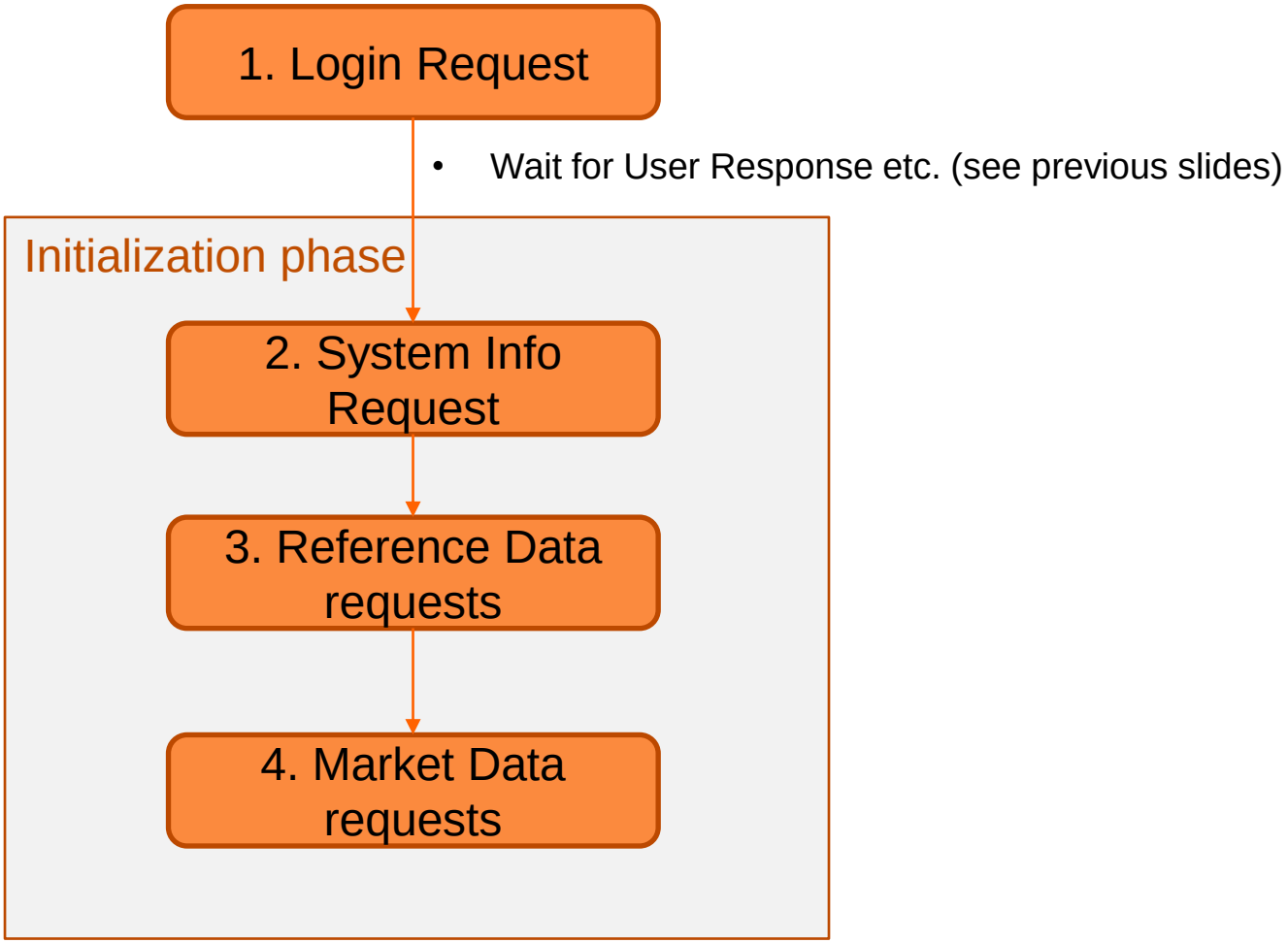
Conclusion:

- 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)
   - **DlvryAreaInfoReq: (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)

Color code: Mandatory, Recommended, optional

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)

(*): 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 and H2HAreaInfoReq	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	56	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

Best practice

Observe messages through ComTrader or our sample Test Client

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

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

Export All to CSV file | Copy All (Excel) | Clear list | Pause

Export Selection to CSV file | Copy Selection (Excel) | Search

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	
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	
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	
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	
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	
23.05.2018 15:37:25	23.05.2018 15:37:25	22	↘	6_0.mbr.CENEX	48eac199-516a-	CashLmtDeltaRprt	
23.05.2018 15:37:25	23.05.2018 15:37:25	18	↘	6_0.bg.CENEXxxxxxxxxxxx	48eac199-516a-	OrdrExeRprt	
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	
23.05.2018 15:37:25	23.05.2018 15:37:25	0	↗	m7.request.management	48eac199-516a-	OrdrEntry	

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

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 cyles.
- 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

since XBID

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

Local Linked product « Intraday_Hour_Power »

AMP: Closed | Area **tradingPhase = Closed**, Area **state = STBY** | CONT | SDAT (Germany split) | Closed

15:00 The day before delivery — Delivery -60mn — Delivery -30mn — Delivery -5mn — Delivery

AMP: Closed | CONT | Closed

XBID Remote Hourly Product « XBID_Hour_Power »

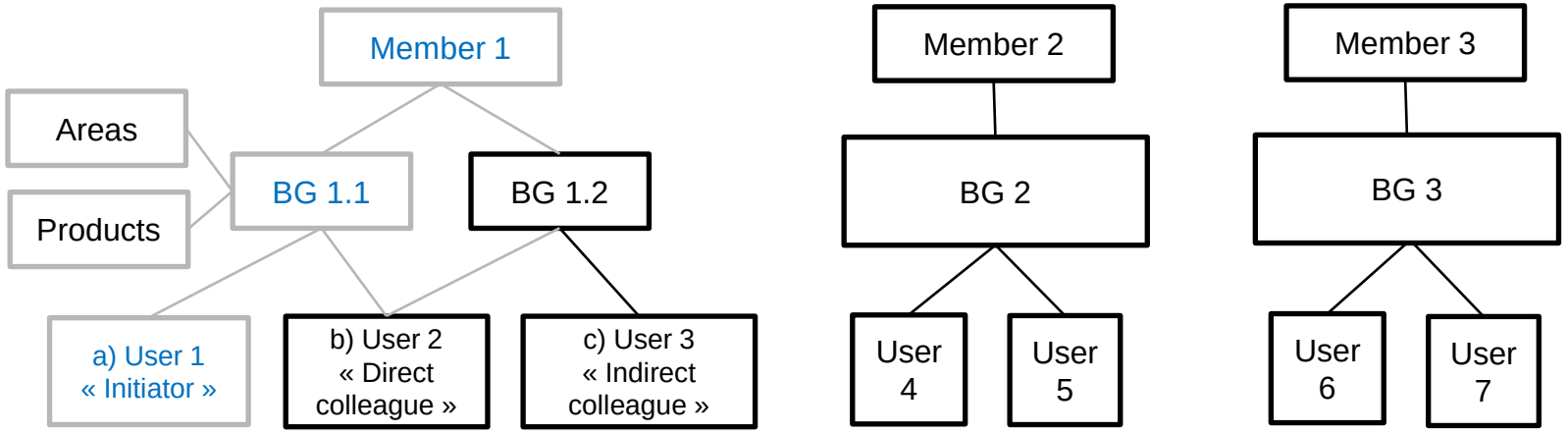
Each product has a Schedule compound of different phases

- **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

Reference Data - 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

Order books
and
Order management

Order management : State transitions

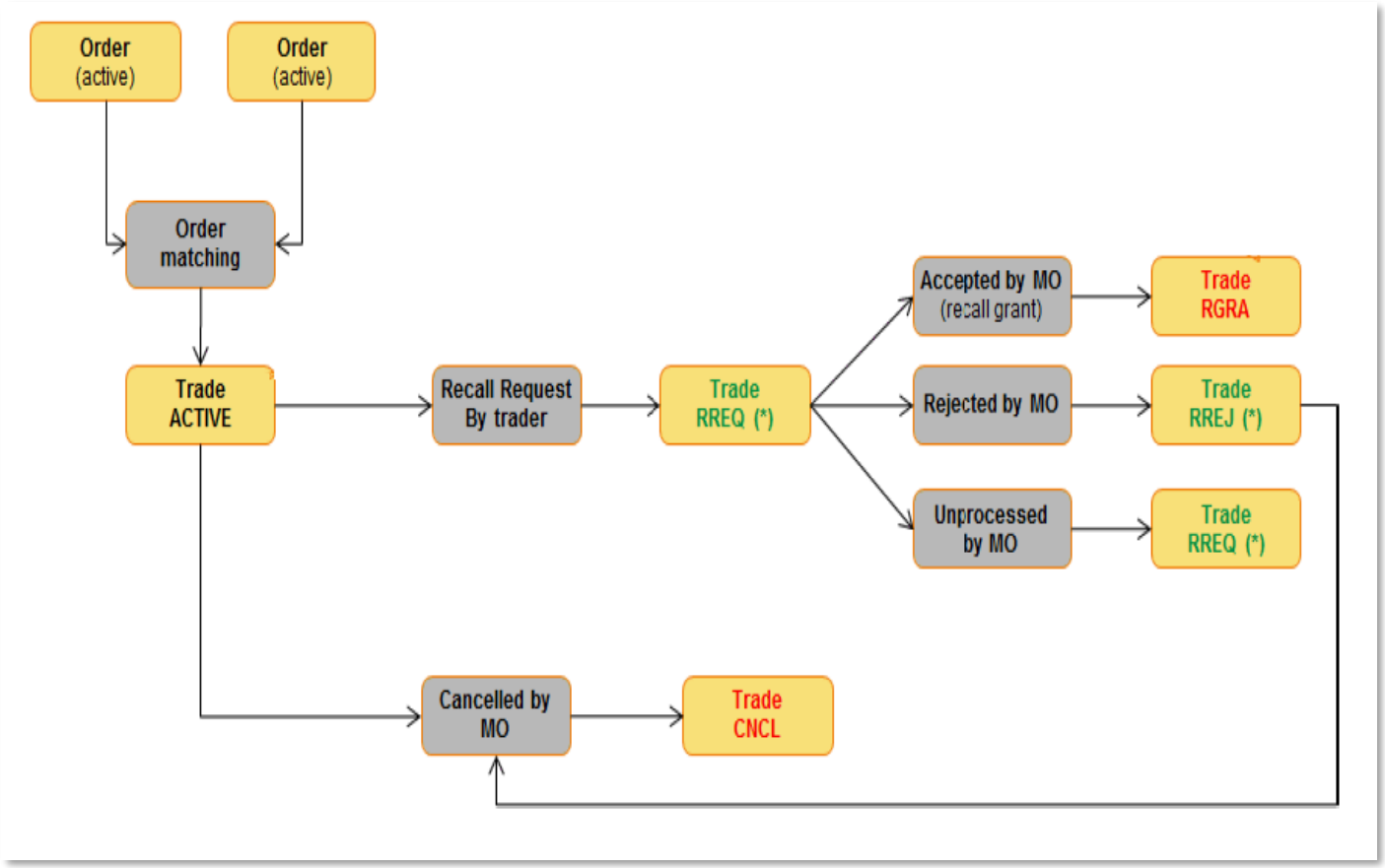
- **An order can be created active or hibernated**
- **It can be modified** (Active -> Active, Hibe -> Hibe)
- **It will get partially traded** (Active -> Active)
- **etc.**

- **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.

Trades

Trades state diagram

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

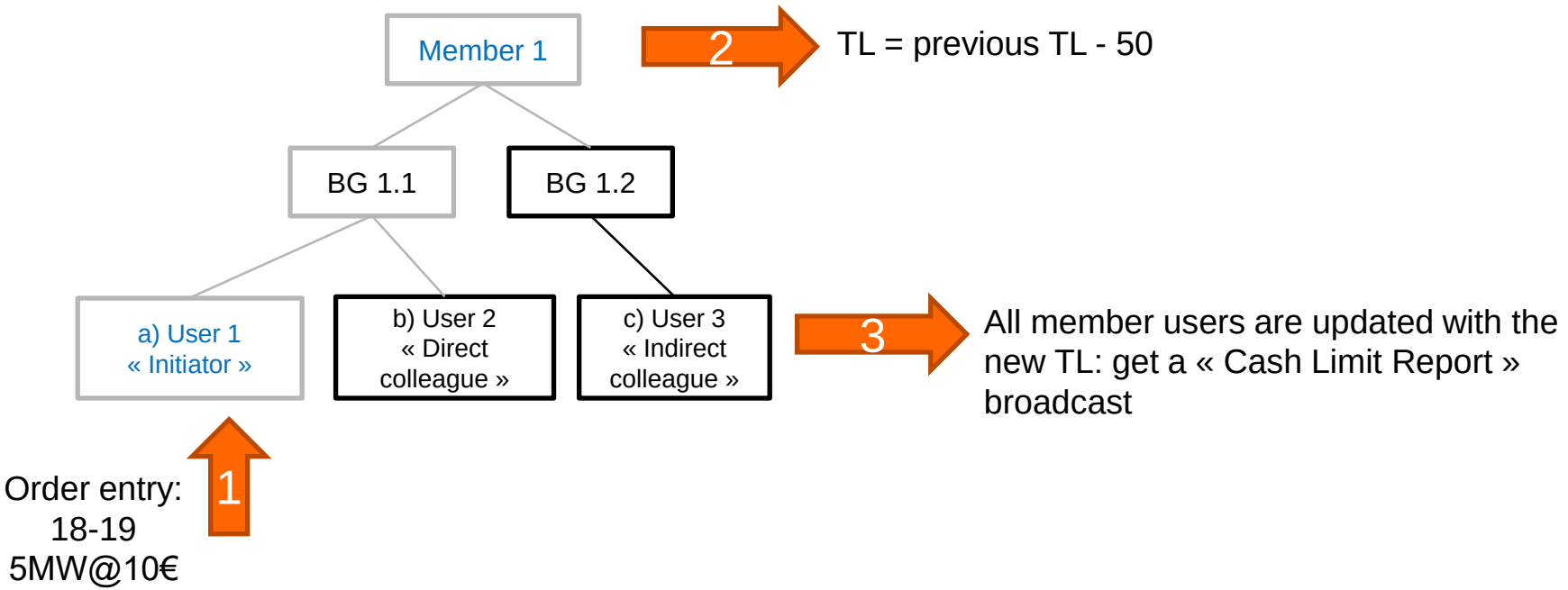


- **Any Trade new state / transition will generate a private (if own trade) and public trade broadcast**

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

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

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="600000000" 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>
```

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

M7 Load Management new functionality

New M7 API load management solution in a nutshell

1/ SCALABLE

Members API orders allowance

- ✓ Proportionate and adapted to the trading activity of each individual member
- ✓ Predictable for the member

2/ TRANSPARENT

Members can query on M7 where they stand on their current order consumption

- ✓ Transparent, accurate and real-time self-monitoring

3/ REAL TIME & GRADUAL

Members get first alerts when order limit is reached [(1)] ; order rejection happens only if the breach is prolonged or further exceeded – and back to normal as soon as the limit is respected

- ✓ Warning first so member can react
- ✓ Only API order submission is paused, ComTrader always open
- ✓ Automatic recovery to normal trading

4/ SAFE

A shorter time period monitoring is provided (10 seconds rolling) to detect short-term load burst

- ✓ Helps early detection and prevention of critical algo-incidents (bugs…)

M7 API Load Management

❑What does it do?

- ❑ What are OMTs?
- ❑ How does it work?
- ❑ What may happen if a Member breaches the limit?
- ❑ What may happen to the Member's active orders in case of OMT Restriction?
- ❑ Can a Member keep placing orders via ComTrader even in case of OMT Restriction?

Main principles

Allows to automatically control the number of orders (« OMTs ») submitted by Members in M7

- OMT = Order Management Transaction
- Load management 1 OMT = **any creation, modification, deletion, hibernation, activation** of an order triggered by an API application

- **M7 counts OMT submitted via an M7 API app** over an observation period, divided into "buckets"
- **Per M7 Member**

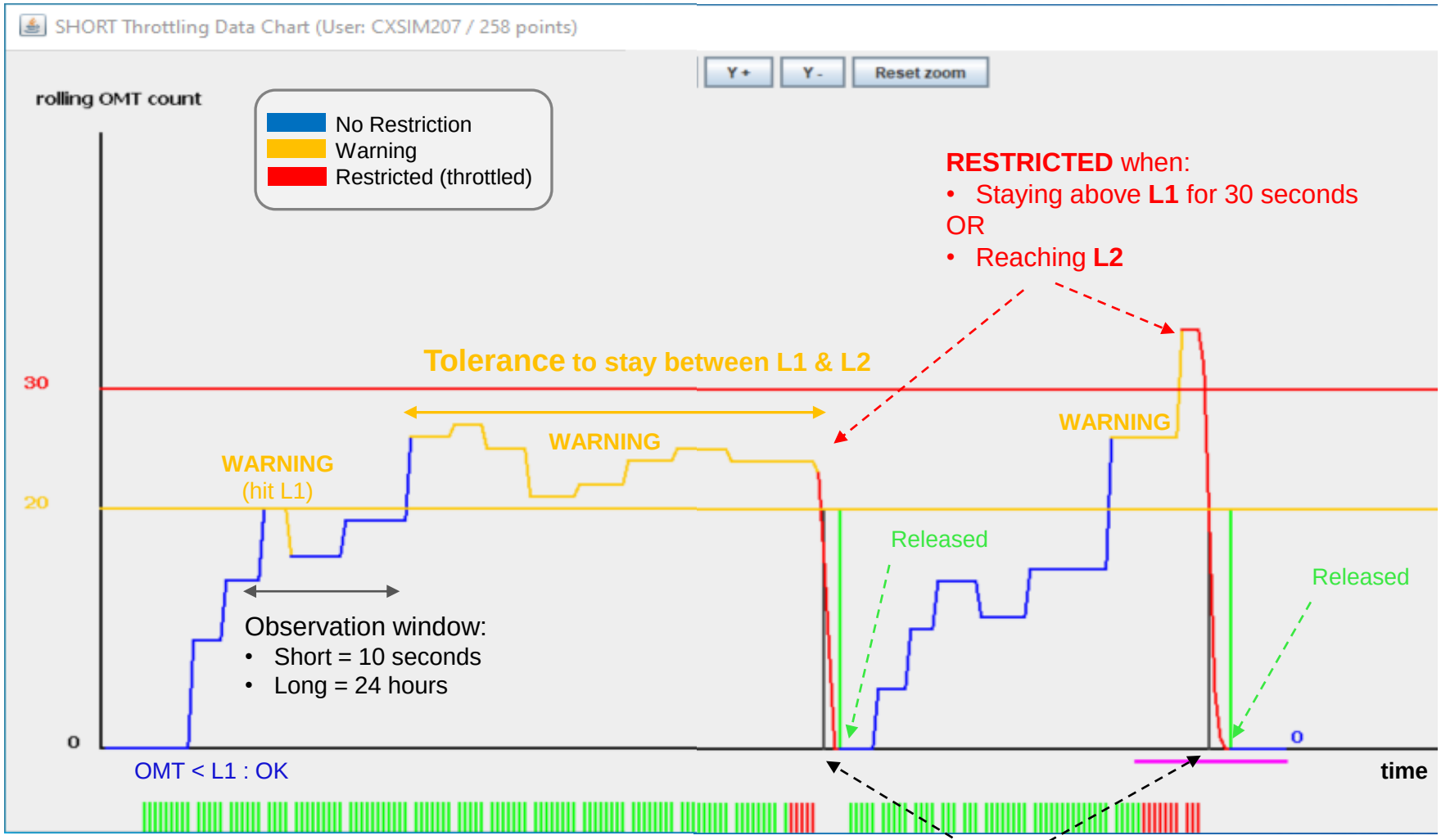
- ComTrader order actions are not considered as part of the OMT count.

M7 API Load Management – Main principles

How does it work?	**EPEX configures parameters per Member** (per ECC code) over 2 different rolling periods of times, called « Observation Windows »: • the **Short OW** will be 10 seconds (bucket = 1 second) • the **Long OW** will be 24 hours (bucket = 15 mn) For each Observation Window: • A **lower threshold** L1 **:** x OMTs • An **upper threshold** L2 **:** y OMTs • A **Tolerance Period**: period during which a Member is allowed to **stay above L1** (but not L2) • A **Cooldown Period**: minimal period during which a Restriction applies to a Member
What may happen if a Member **breaches** the limit?	• **Warnings** (start of tolerance period), • **Order rejections** (« restricted »), • **Disconnection + user suspension in extreme cases:** Protective suspension (nb of OMT msg per User level per second)
What may happen to the Member's active orders in case of OMT Restriction?	A User can define different actions at each Login (auto-hibernation at different levels)
Can a Member keep placing orders via ComTrader even in case of OMT Restriction?	Yes, it is always possible to manage orders using ComTrader.

M7 API Load Management – Chart view – **Short rule** example

Chart built sending a Throttling Status Request every 3 seconds



Load Mgmt. API ecosystem for R/W API apps

1 **2 New Login Request options**

Note : RO apps do not need to be updated.
Just perform regression tests as usual

Keep control of your API orders, should you get restricted

« If my member gets restricted I want my/all my BG/all members orders to get hibernated »

An applicative limit will be put in place with 6.13

2 **Throttling Status Request / Response**

Know your Load situation (every 3 seconds max TBC)

Response : short L1 = xxx, OMT = yyy, … Long L1 = … End of TP = … etc.

3 **2 new Message Reports**

Be informed in real time of status changes

- *Message #168 : « warning », « restricted », #166: back to « no restriction »*
- *Text + variables (DFS200)*

4 **order rejections**: *ErrResp in response Q*

Order management requests can get rejected once restricted

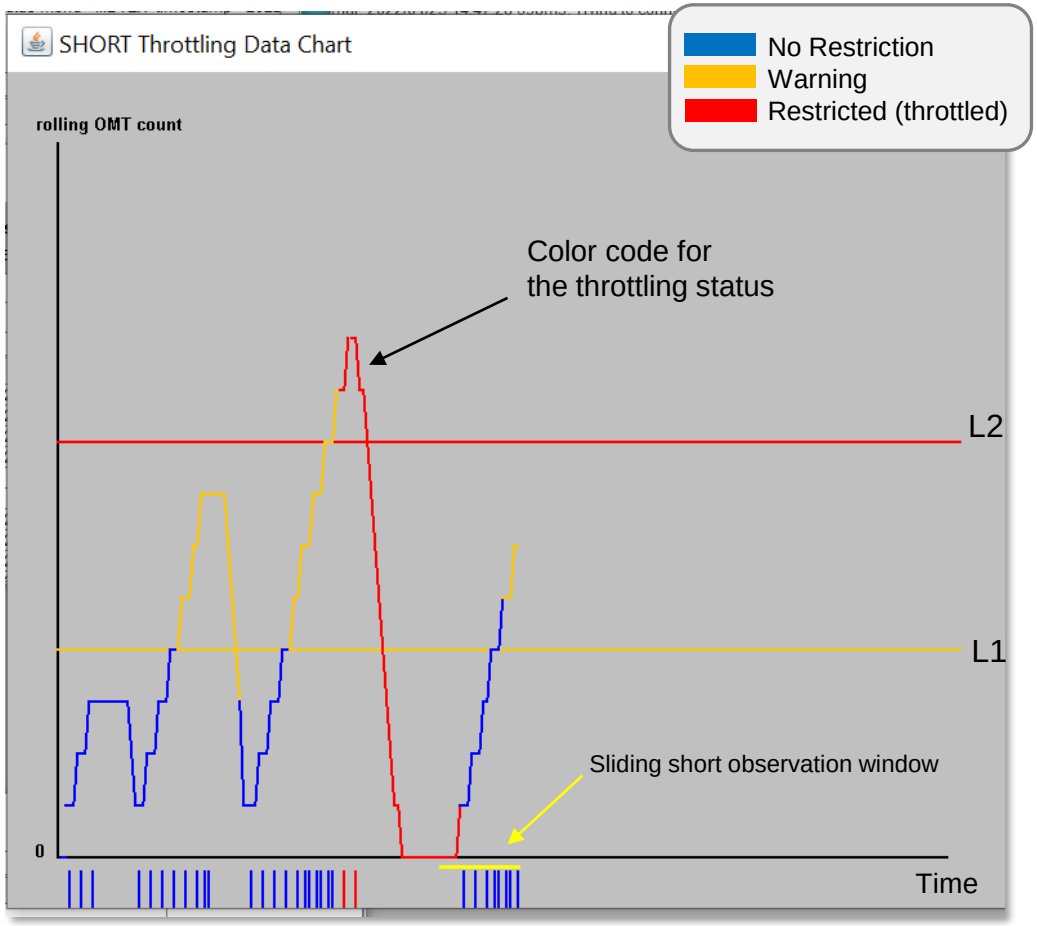
Order management request

ErrResp in response queue instead of AckResp

M7 6.12 API

Load Mgmt. API impacts

Example: Graph built off periodic Throttling Status Response data (short rule)



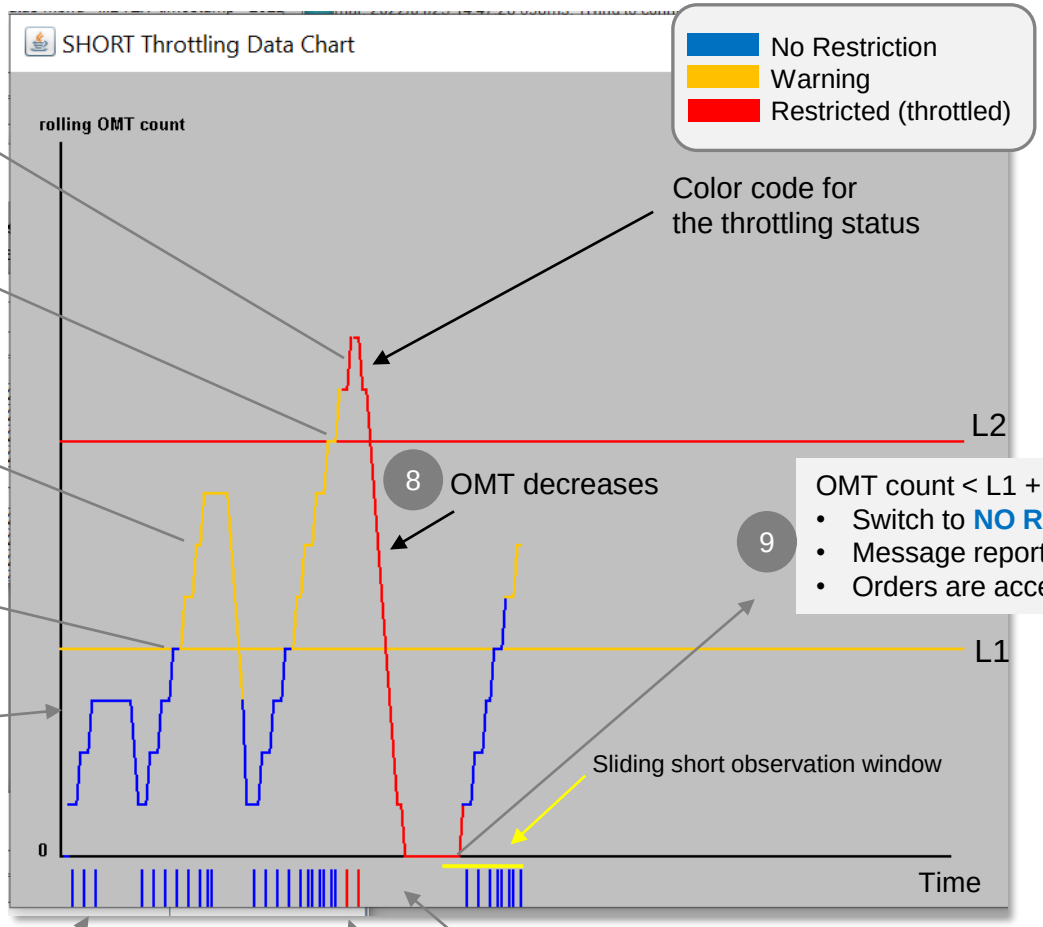
1 bar = 1 order submission attempt

- Blue: Ack Resp received
- Red: ErrResp: rejected because restricted

Note: it is still possible after an AckResp to get a « logical » Error (ErrResp as a broadcast, e.g. when submitting an order on a closed contract)
Anyway **all « order action attempts » are counted**

Load Mgmt. API impacts

Example: Graph built off periodic Throttling Status Response data (short rule)



5 The next Throttling Status response Reflects the **RESTRICTED** status

4 Reaching L2 :
Swtich to **RESTRICTED** status:
Message Report sent

3 Orders are still accepted by M7

2 Reaching L1 :
Switch to **WARNING** status:
Message Report sent

1 OMT count < L1 :
- **NO RESTRICTION** status:

Orders are accepted by M7
- When Order magement stops or slows down, OMT count decreased

Submitted orders:
- Blue: Ack Resp received
- Red: rejected because restricted

6 **RESTRICTED** status:
Orders are rejected : Err Resp

7 Order submission is stopped
OMT decreases

FAQ

M7 API FAQ

Q: Which version of AMQP do you support?

A: The AMQP server is currently using the AMQP implementation from RabbitMQ. This version supports AMQP versions **0.9.1, 0.9 and 0.8**. See www.amqp.org and www.rabbitmq.com for more information.

Q: Which version of the TLS security protocol do you support?

A: TLS v1.2.

Q: What do I need to log in M7 via the M7 API?

A: You need a certificate and password, user and password, and application Id to connect to the AMQP server.

Once this AMQP connection is done you only need the user id to log in.

Q: How many test environments do you have?

A: 2 environments. 1 in an iso-prod version ("XSIM") and 1 in the upcoming (N+1) version ("ASIM").

Q: What is the process to go to production?

A: Your API application needs to go through a conformance test that last 24 hours. You need to book an appointment with our Market Operations 2 weeks in advance. The goal of this test is to ensure that your API app respects our Terms Of Reference (basically: respects the request/broadcast logic and does not send any request every now and then, remains logged in the whole time, goes seamlessly through a market halt/restart, an XBID disconnection, contracts opening and closing events, does not consume too many AMQP ressources).

Q: Are messages compressed

A: You can send request in a compressed or uncompressed format. M7 send indicates in the message header when replying or when broadcasting message if the message is gzipped or not. Your application must be able to dynamically decompress messages when required.

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