Pricing Rate Service

Solution design

Document version: 1.0

Document date: 27-Aug-19

Reference: #SOD

Company confidential

DRAFT



For more information on UNIMONI, visit our Web site:

<https://www.unimoni.com/>

Every reasonable effort has been made to ensure the information and procedures detailed in this document are complete and accurate at the time of writing. However, information contained in this document is subject to change.

© 2019 UX Holdings.

Document history

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Change summary | Owner |
| 1.0 | 28-Aug-19 | First draft | Rajveer Singh |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

|  |
| --- |
| Changes since last issue |
|  |

|  |  |  |  |
| --- | --- | --- | --- |
| Document control | | | |
| Owner: | Rajveer Singh | Title: | Technical Consultant |
| Approved by: | Vishal Kumar | Title: | Product Owner |
| Signature: |  | Date: | 28-Aug-19 |
| Review record ref.: |  | Distribution: | Development & Business team |
| File name: |  | | |

Contents

[1 Introduction 1](#_Toc18504219)

[1.1 Purpose 1](#_Toc18504220)

[1.2 Scope 1](#_Toc18504221)

[1.3 Intended audience 1](#_Toc18504222)

[1.4 Glossary 2](#_Toc18504223)

[1.5 References 2](#_Toc18504224)

[2 The General Description 3](#_Toc18504225)

[2.1 Product Perspective 3](#_Toc18504226)

[2.2 System Overview 4](#_Toc18504227)

[3 System Architecture 5](#_Toc18504228)

[3.1.1 Database 5](#_Toc18504229)

[3.1.2 Pricing Integration Adapter 8](#_Toc18504230)

[3.1.3 Rest API 10](#_Toc18504231)

[3.1.4 Quartz Job Scheduler 11](#_Toc18504232)

[3.1.5 T&R Elektron SDK Client 11](#_Toc18504233)

[3.1.6 Service Layer 11](#_Toc18504234)

[3.1.7 Repository Layer 12](#_Toc18504235)

[4 Admin Console 13](#_Toc18504236)

[4.1.1 Screen design 14](#_Toc18504237)

[4.1.2 Features 14](#_Toc18504238)

[4.1.3 Recovery measures 15](#_Toc18504239)

# Introduction

## Purpose

It has been observed recently that some unexpected issues can occur on Production which are hard to detect, hence noticed when it’s too late. Also there is no interface to the system where the internals of system can be monitored to take any measures required, if something goes wrong such as database goes down or connection issue, ActiveMQ goes down, T&R client issues with streams and channel, network issues, Quartz scheduler related issues etc.

Pricing Rate Service is required to be fault tolerant and should be able to detect and recover from possible faults or at least generate alert on User Interface or email etc. so that if manual intervention can happen in timely fashion. Any failures to do so may cause serious business loss.

The purpose of the document is to collect and analyze all assorted ideas that have come up to define the Monitoring and Troubleshooting of the system. It is meant to provide a detailed description of all functional and non-functional requirements, to assist the business, testing and production support team to keep a look on systems internals to detect any issues and take the required measures. It will also explain system constraints, interface and interactions with other external applications at high level.

It defines how our client, team and audience see the product and its functionality. Nonetheless, it helps any designer and developer to assist in software delivery lifecycle (SDLC) processes.

It does not specify any technical design or low level description of how the requirements are going to be fulfilled by system. This document is subjected to continuous update as per review by business team and due to evolution in requirements.

## Scope

Pricing Rate Service collects the exchange rates from rate providers such as Thomson & Reuters and Future Source, and publishes the same on configured schedule frequency to Pricing Engine Service. Current version captures following features.

* Pricing Integration adapter, responsible for managing the communication with Pricing Engine Service.
* System overview and application architecture.
* The relevant database tables along with description.
* The events data being used as inter-microservices communication between Pricing Rate Service and Pricing engine Service.
* All use cases along with their implementation details.
* The rest API and T&R client channels to receive rate changes from both Rate Providers i.e. Thomson & Reuters, Future Source.
* New requirements for Admin Console, capturing along with screen design to assist the administration of application catering to view all rate records for T&R and F&S, view the state and history of schedules, T&R client streams (where rate provider is T&R), the rates (fluctuation) etc. over a period of time.
* The implementation details of Admin console are not in scope of this document as of now.

## Intended audience

This document is intended for both the stakeholders and the developers of the system and will be proposed to the Business team for its approval.

## Glossary

1. Document glossary

|  |  |
| --- | --- |
| Item | Definition |
| T&R | Thomson and Reuters rate provider |
| F&S | Future Source rate provider |
| Treasury User |  |
| RIC ID | A Reuter’s instrument code, or RIC, is a ticker-like code used by Thomson Reuters to identify financial instruments and indices. The codes are used for looking up information on various Thomson Reuters financial information networks (like Bridge and RMDS). |
| Ask Value | The ask price represents the minimum price that a seller is willing to receive |
| Bid Value | The bid price represents the maximum price that a buyer is willing to pay for a security |
| Stakeholder | Any person or party with an interest in the project who is not a developer |

## References

The following documentation and links can help the reader understand this document:

1. IEEE. IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications. IEEE Computer Society, 1998.
2. Thomson & Reuters Elektron SDK client <https://developers.refinitiv.com/elektron/elektron-sdk-java>
3. Quartz Job Scheduler <http://www.quartz-scheduler.org/>

# The General Description

## Product Perspective

There are three types of Rate sources in Pricing Engine Service.

1. **Manual – ID is 1**: The rates need to be updated by treasury user manually. The unique identifier is 1, being used in database tables and internal logic of the system.

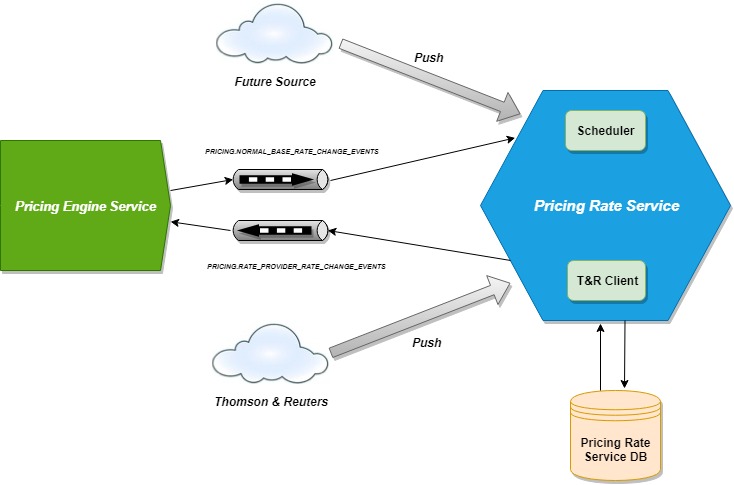
**Note**: *Manual rate provider is for use in Pricing Engine Service only, Pricing Rate Service is only concerned with following non manual Rate Providers.*

1. **Future Source (F&S) – ID is 2**: The rate changes are pushed into Pricing Rate Service from Future Source. A rest API has been exposed to provider to push the rates into the system.
2. **Thomson & Reuters (T&R) – ID is 3**: The rate changes are pushed into Pricing Rate Service from Thomson & Reuters. T&R Elektron client is integrated into the system to receive any rate changes from the provider. It’s also the push mechanism where T&R pushed the rates into the system

The Pricing Rate Service is responsible for collecting the rate changes from F&S and T&R and publishing to Pricing Engine Service periodically as per configured frequency for each currency in Pricing Engine Service.

## System Overview

Pricing Rate Service run as separate micro service and responsible for keeping the required data provided by Pricing Engine service such as Rate ID (The correlation ID of a currency to communicate messages to and forth between Pricing Rate Service and Pricing Engine Service), rate update frequency for each record. Managing the scheduled jobs to publish latest rates into Pricing Engine Service. Collecting the rate changes from both F&S, T&R rate providers by keeping the channel of communication with Rate Providers, up and running all the time. Self-recovery from any possible failures if possible but also providing an interface to view current internal state of application such as database health, ActiveMQ health, T&R client channel and streams status. History of rate changes and Schedules executed with status details over period of time. The overall system design is as follows.



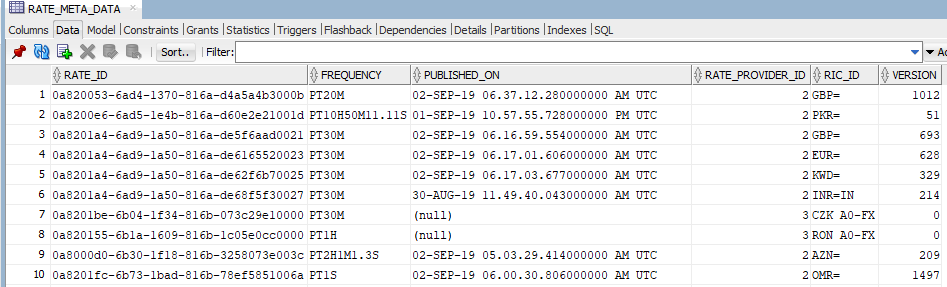
# System Architecture

Following are the main building blocks of the system.

### Database

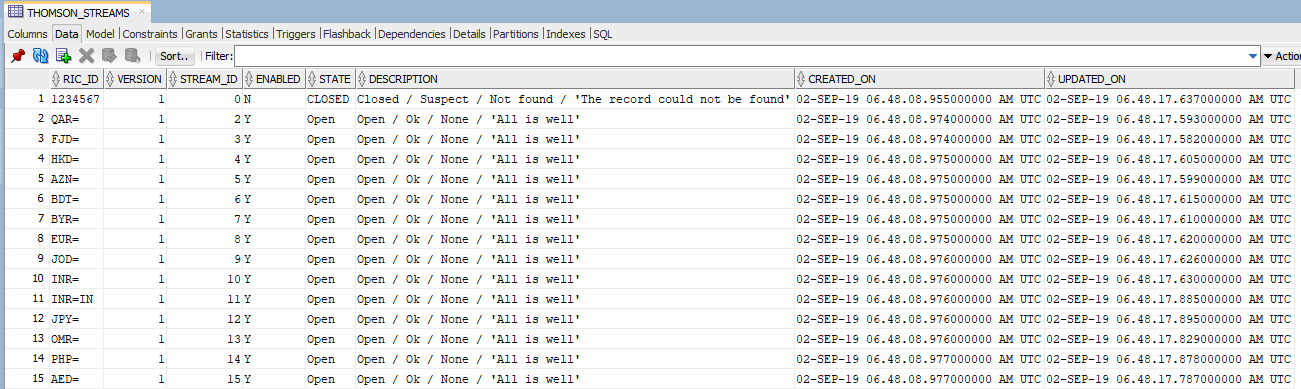
1. **RATE\_META\_DATA:** The minimal details of Rate records in Pricing Engine Service are stored in this table on receiving **ProvidedRateEnabledEvent** message from **PRICING.NORMAL\_BASE\_RATE\_CHANGE\_EVENTS.** Then as and when the **ProvidedRateFrequencyChangedEvent** is received, the respective record is updated with new frequency for respective record. When **ProvidedRateDisabledEvent** is received the respective record id deleted from this table.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data Type** | **Nullable** | **Description** |
| RATE\_ID (Primary Key) | VARCHAR2(36 CHAR) | NO | The primary key of rate record in pricing engine, to be used as correlation ID between Pricing Engine Service and Pricing Rate Service |
| RATE\_PROVIDER\_ID | NUMBER(10,0) | NO | The identifier of a rate provider |
| RIC\_ID | VARCHAR2(10 CHAR) | NO | The identifier of a currency combination understood by rate providers, such as INR= for source currency INR and target currency USD |
| FREQUENCY | VARCHAR2(15 CHAR) | NO | The frequency or fixed repeat interval after which an attempt should be made to publish rate changes into pricing engine, if there are any after last attempt |
| PUBLISHED\_ON | TIMESTAMP (6) | NO | The timestamp on which the latest rate was last published in Pricing Engine Service |



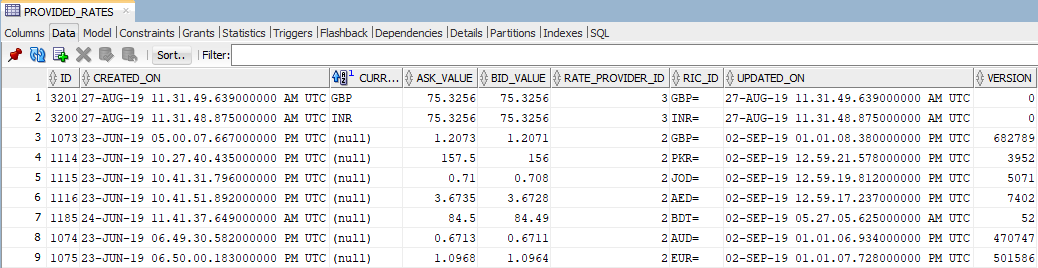
1. **THOMSON\_STREAMS:** For each unique RIC\_ID in RATE\_META\_DATA where Rate Provider is Thomson & Reuters (RATE\_PROVIDER\_ID = 2), a stream is opened on Channel in T&R Elektron client.This table maintains the stream status of same.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data Type** | **Nullable** | **Description** |
| RIC\_ID (Primary Key) | VARCHAR2(10 CHAR) | NO | The identifier of a currency combination understood by rate providers, such as INR= for source currency INR and target currency USD |
| STREAM\_ID | NUMBER(19,0) | NO | The identifier of a rate provider |
| ENABLED | CHAR(1 CHAR) | NO | Whether or not the stream is open |
| STATE | VARCHAR2(15 CHAR) | NO | The T&R client managed State of stream. Possible values are WAITING, OPEN, NON\_STREAMING, CLOSED\_RECOVER, CLOSED, CLOSED\_REDIRECTED |
| DESCRIPTION | VARCHAR2(255 CHAR) | NO | String description of current state of stream. Ex. *Open / Ok / None / 'All is well'* , *Closed / Suspect / Not found / 'The record could not be found'* or *Waiting for stream status event* |
| CREATED\_ON | TIMESTAMP (6) | NO | The timestamp on which the stream status record created the first time |
| UPDATED\_ON | TIMESTAMP (6) | NO | The timestamp on which stream status was updated |



1. **PROVIDED\_RATES:** This table contains the latest rate for individual RIC IDs.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data Type** | **Nullable** | **Description** |
| ID (Primary Key) | VARCHAR2(36 CHAR) | NO | The auto generated surrogate primary key |
| RIC\_ID | VARCHAR2(10 CHAR) | NO | The identifier of a currency combination understood by rate providers, such as INR= for source currency INR and target currency USD |
| CURRENCY | VARCHAR2(3 CHAR) | NO | The ISO currency code, specifically for the rates pushed by Future Source |
| ASK\_VALUE | NUMBER(\*,10) | NO | The Ask Price of currency |
| BID\_VALUE | NUMBER(\*,10) | NO | The Bid Price of currency |
| RATE\_PROVIDER\_ID | NUMBER(10,0) | NO | The identifier of a rate provider |
| CREATED\_ON | TIMESTAMP (6) | NO | The timestamp on which the record is created the first time |
| UPDATED\_ON | TIMESTAMP (6) | NO | The timestamp on which the latest rate was last updated. It is used in combination with the PUBLISHED\_ON timestamp from RATE\_META\_DATA to decide whether or not the rate is to be published in Pricing Engine Service |



**The other tables are of internal user of Quartz library to manage the Scheduled Jobs and associated Triggers with them.**

### Pricing Integration Adapter

PricingIntegrationAdapter.java is the adapter abstracting away the integration logic with ActiveMQ Queues which are the communication channels with Pricing Engine Service.

Following are the queues in place.

* **PRICING.NORMAL\_BASE\_RATE\_CHANGE\_EVENTS**

Transports the messages from Pricing Engine Service to Pricing Rate Service. Following are three kind of messages being transported along with the details of attributes in each type of message

* 1. **ProvidedRateEnabledEvent:** Fired when a new rate record in created or a disabled rate record is enabled on IBR Management screen where Rate Provider for the record in either T&R or F&S in Pricing Engine Service.

|  |  |  |
| --- | --- | --- |
| **Name** | **Data Type** | **Description** |
| rateProviderId | int | The identifier of a rate provider |
| rateId | String | The primary key of rate record in pricing engine, to be used as correlation ID between Pricing Engine Service and Pricing Rate Service |
| ricId | String | The identifier of a currency combination understood by rate providers, such as INR= for source currency INF and target currency USD |
| frequency | java.time.Duration | The frequency or fixed repeat interval after which an attempt should be made to publish rate changes into pricing engine, if there are any after last attempt |

* 1. **ProvidedRateDisabledEvent:** Fired when a rate record in created or an active rate record is disabled on IBR Management screen where Rate Provider for the record in either T&R or F&S in Pricing Engine Service.

|  |  |  |
| --- | --- | --- |
| **Name** | **Data Type** | **Description** |
| rateProviderId | int | The identifier of a rate provider |
| rateId | String | The primary key of rate record in pricing engine |
| ricId | String | The identifier of a currency combination understood by rate providers, such as INR= for source currency INF and target currency USD |

* 1. **ProvidedRateFrequencyChangedEvent**: Fired when a rate record’s *Frequency* is updated on IBR Management screen where *Rate Provider* for the record in either T&R or F&S in Pricing Engine Service.
* **PRICING.RATE\_PROVIDER\_RATE\_CHANGE\_EVENTS**

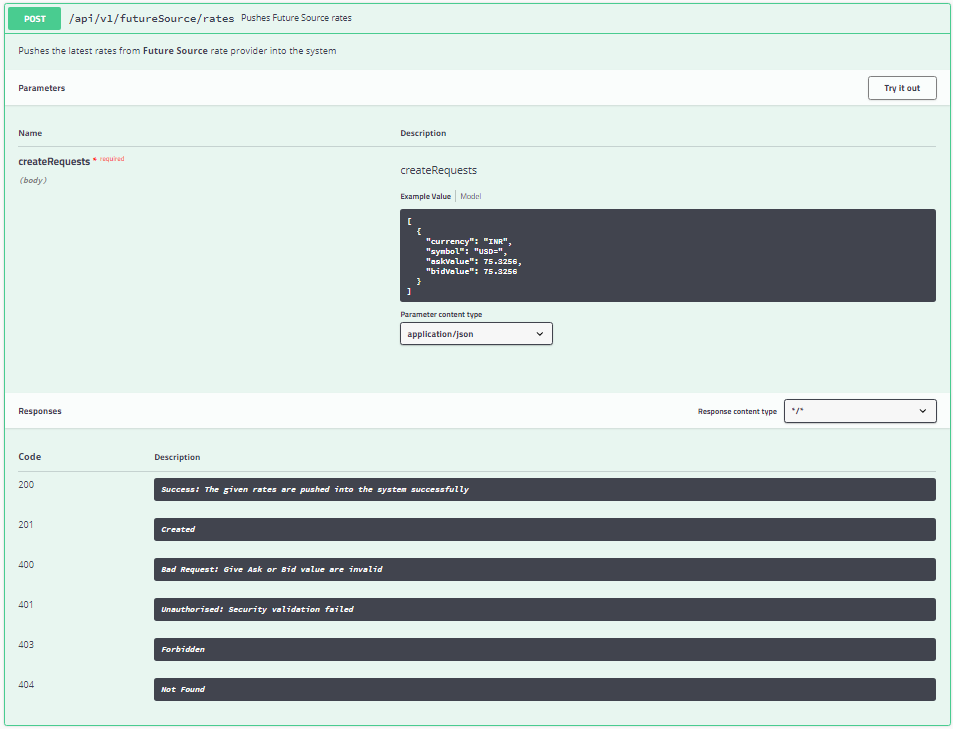
Transports the messages from Pricing Rate Service to Pricing Engine Service. Following is the message being transported with the details of attributes in each type of message

* 1. **ProvidedRateEvent**: Fired from Pricing Rate Service after Frequency or repeat interval when the rate for the RIC ID associated with a rate record is changed since the last attempt.

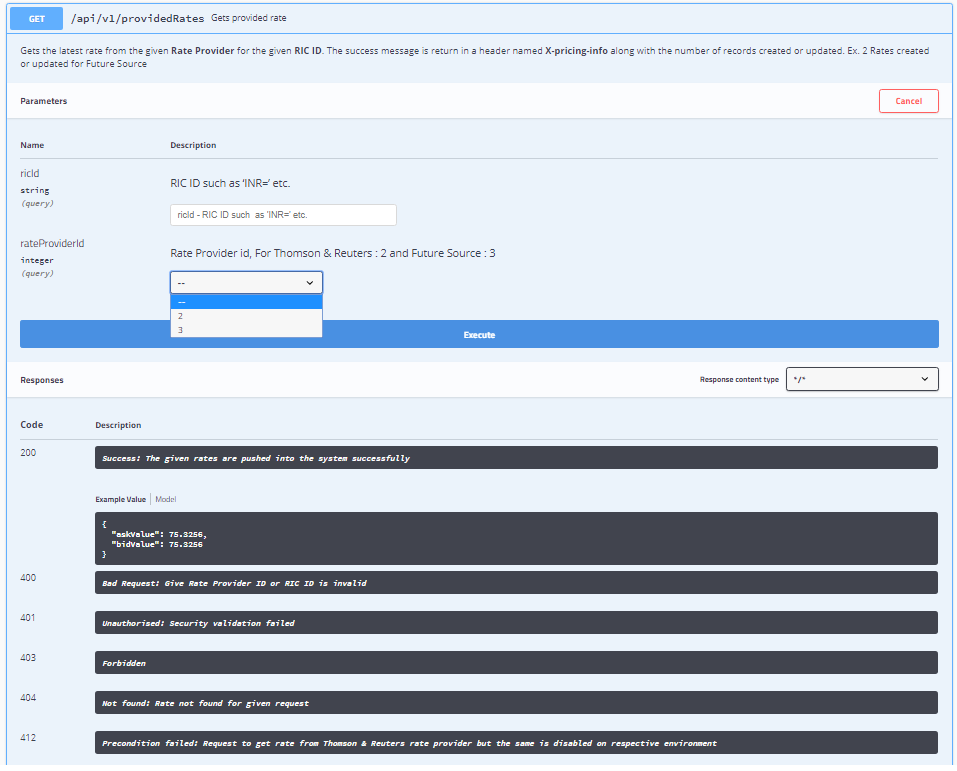
|  |  |  |
| --- | --- | --- |
| **Name** | **Data Type** | **Description** |
| rateId | String | The primary key of rate record in pricing engine |
| ricId | String | The identifier of a currency combination understood by rate providers, such as INR= for source currency INF and target currency USD |
| askValue | java.math.BigDecimal | The Ask Price of currency |
| bidValue | java.math.BigDecimal | The Bid Price of currency |

### Rest API

1. **Push Future Source rates**: This API is the channel for Future Source rate provider to push the rate changes into the system. For given input if no rate exists new rate record is created or existing rate record is updated.



1. **Get Ad hoc Provided Rate:** Gets the latest rates from either of Rate Providers T&R or F&S for a particular currency or RIC ID



### Quartz Job Scheduler

Quartz library is used to implement scheduled jobs to be executed at fixed repeat interval or Frequency configured for a Rate Record in Pricing Engine Service. The scheduled job contains the logic to check if new rates have been received for the RIC ID associated with a Rate Record from the configured Rate Provider. If yes then a **ProvidedRateEvent** is pushed into **PRICING.RATE\_PROVIDER\_RATE\_CHANGE\_EVENTS** otherwise do nothing and try again after the job’s repeat interval (Frequency of the rate record)

// More details to be added later to elaborate the internal working of the scheduler.

### T&R Elektron SDK Client

// To be done

### Service Layer

// To Be done

### Repository Layer

// To be done

# Admin Console

Currently the channel establishment with T&R and Scheduled Job execution is happening in background. So there is no concrete way to identify if something wrong happens in T&R Client channel/stream or schedule jobs starts failing which may result into inability of Pricing Rate Service to collect rate changes from rate provider and publishing the same to Pricing Engine Service.

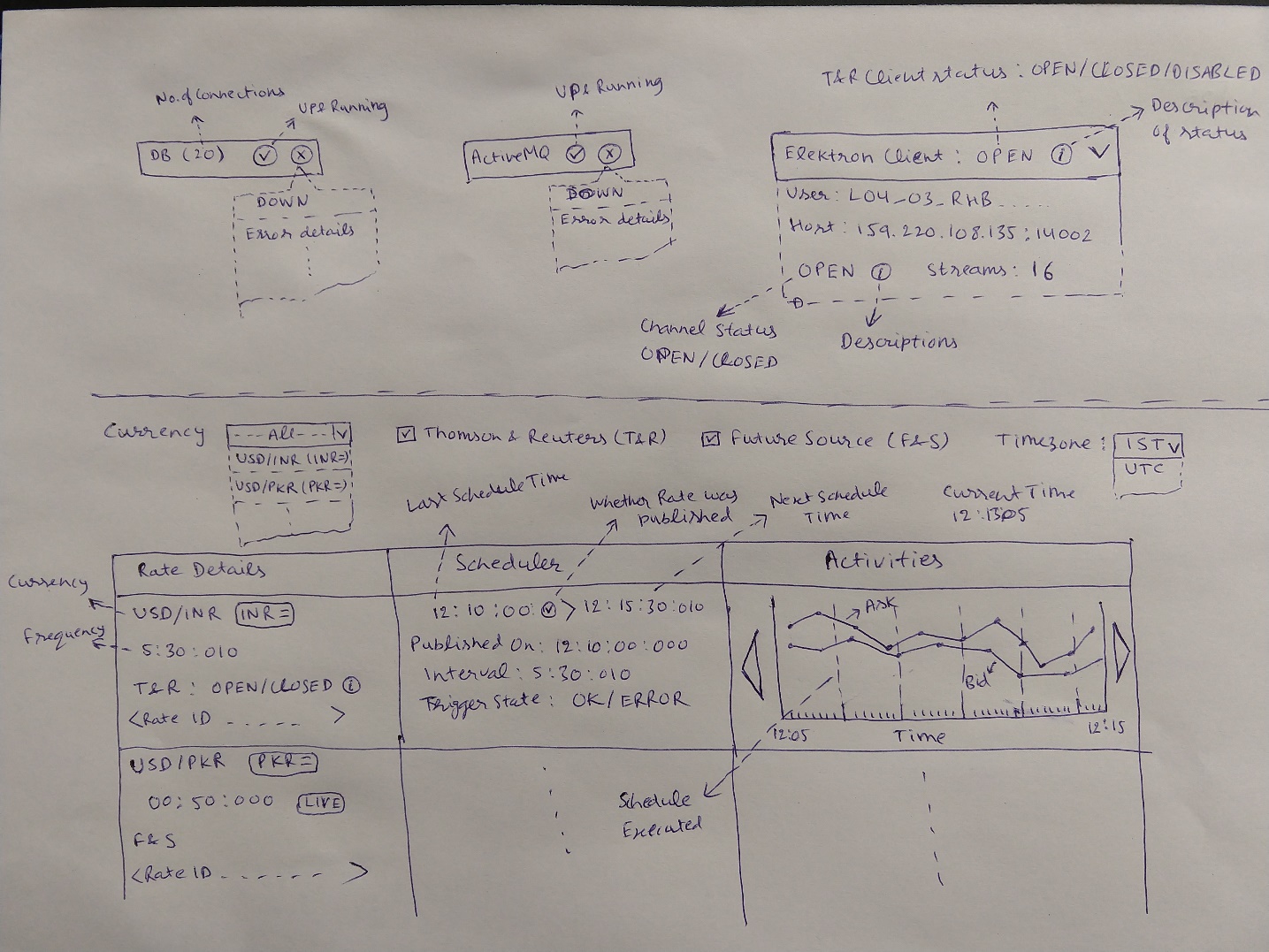
It has been observed that Oracle Database and ActiveMQ may also go down, hence system cannot function at all, without manual intervention to recover from such infrastructure failure. But currently there is no Monitoring and Alerting mechanism where such failures can be detected and alerted to outside world to take measures before it is too late, causing business loss.

Production support team is relying on keeping an eye on rate updates. If they are not happening then it could be either the case that rates are actually not being pushed from Rate Providers as Market is closed on weekends or late night, but the system is working just fine or the provider is trying to push the rate changes but system is failing to do its job because of some internal or infrastructure issues as highlighted above

So new business requirements have emerged to provide an Admin Console to meet following needs.

* View all rate records in Pricing Rate service with all attributes to identify if all data propagation from Pricing Engine Service to Pricing Rate Service is happening correctly or not. If any data sync issue is found corresponding measures need to be taken to automatically recover from JMS message failure between services and recover from them.
* Keep an eye on the internals of application such as T&R Elekron client channel/streams and Scheduled jobs and take any measures if required is anything is wrong in the application internally.
* View all rate changes and scheduled jobs executed over a period of time to identify if system is working good or not.
* An alerting mechanism to notify the respected authority if any failures require human intervention to recover from.

### Screen design



### Features

1. **Database status**: Either Up and Running or Down/Connection issue. In later case the error details should also be available for further investigation
2. **ActiveMQ status**: Either Up and Running or Down/Any other ActiveMQ issue. In later case the error details should also be available for further investigation
3. **T&R Elektron client status**: Following information should be visible

* The T&R Username, if required mask it.
* Host and Port of T&R server
* Status: The current status of client channel with T&R status along with textual description of each status type
  + 1. DISABLED – Thomson & Reuters rate provider is disabled on respective environment. So there would be no rate updates for currencies whose Rate Provider is T&R. In this case, no other details related to T&R client would be visible
    2. OPEN – Channel is open and in good health
    3. CLOSED – Channel is closed most probably by user or application
    4. DOWN\_RECONNECTING – Channel is down and trying to reconnect
* Streams: Number of streams, which should be equal to unique valid RIC IDS of rate records where Rate Provider is T&R

1. **Filters**: Following filters would be there to filter the records on screen

* **Currency**: Source target currency combination, along with respective RIC ID, the dropdown option would be searchable, so that user can filter on the basis of either source currency, target currency or RIC ID
* **Rate Provider**: User should be able to select either or both of T&R or F&S to filter the records for respective Rate Providers only
* **Time zone**: User should be able to select the time zone to see the timestamps associated with dataset in required zone. Options could be UTC, Asia/Kolkata, Asia/Dubai. The **Current timestamp** should also be visible on screen.

1. **Rate record details**

* **Currency**: Source currency, target currency and RIC ID
* **Frequency**: The fixed time after which latest rates are published into Pricing Engine Service, if any rate changes have happened since they published the last time. The format of time duration is hh:mm:ss:ms along with Live or Not Live flag.
* **Rate Provider & Stream status**: The configured rate provider for this record, either T&R or F&S. If Rate Provider is T&R, the respective stream status should also be there with possible values OPEN/CLOSED accompanied by an info icon, on clicking info icon additional details such as description of status and last updated on etc. would be visible.
* **Rate ID**: The primary key of rate record in Pricing Engine Service. It is used as a correlation ID between two services to exchange messages and process accordingly.

1. **Scheduler**

* **Last Job execution time**: The timestamp on which the rate publication job was executed most recently, along with additional info such as whether job execution was successful or misfired happened and whether or not new rates were available and published into Pricing Engine Service.
* **Next Job execution time**: The timestamp the job would executes again in future.
* **Last Published On**: The timestamp on which the rate was published into Pricing Engine Service the last time, could be different than Last Job execution time.
* **Interval**: The repeat interval of Quartz job, if found to be different than rate frequency then it’s an issue and needs to be fixed.
* **Trigger state**: OK/ACQUIRED/ERROR, in case its ERROR then it’s an issue and needs to be fixed because in that case the Job would never be executed again unless state is recovered to any other state value.

1. **Activities**: The recent activities in the system in graphical representation containing following details

**Note**: *The feasibility of implementation of this feature needs to be analyzed*

* The Ask and Bid value fluctuations over a period of time
* The schedules executed over a period of time highlighting the time instants on which the job was executed. Whether the executed job actually published rates to Pricing Engine Service or not should be highlighted.
* Vertical axis represent money amount, on hover of ask/bid price line, the amount should be displayed.
* A constant horizontal line of ask and bid value would suggest that rate changes have not happened.
* User should be able to navigate the time window till a configurable time in past.

### Recovery measures

User should be given with the options to recover from whichever error scenario it’s possible. Following would be analyzed and wherever possible recovery option would be give if human intervention is required. Preferably, If possible the system should recover from any errors automatically.

* T&R electron client, channel and stream recovery options
* Recover Trigger from ERROR state
* More options could be added if required.