AMDL Rules Quick Start Guide

Powered by Featurespace

Version: 1.0

21 June 2024

Publication number: FTMC-1.0-6/21/2024

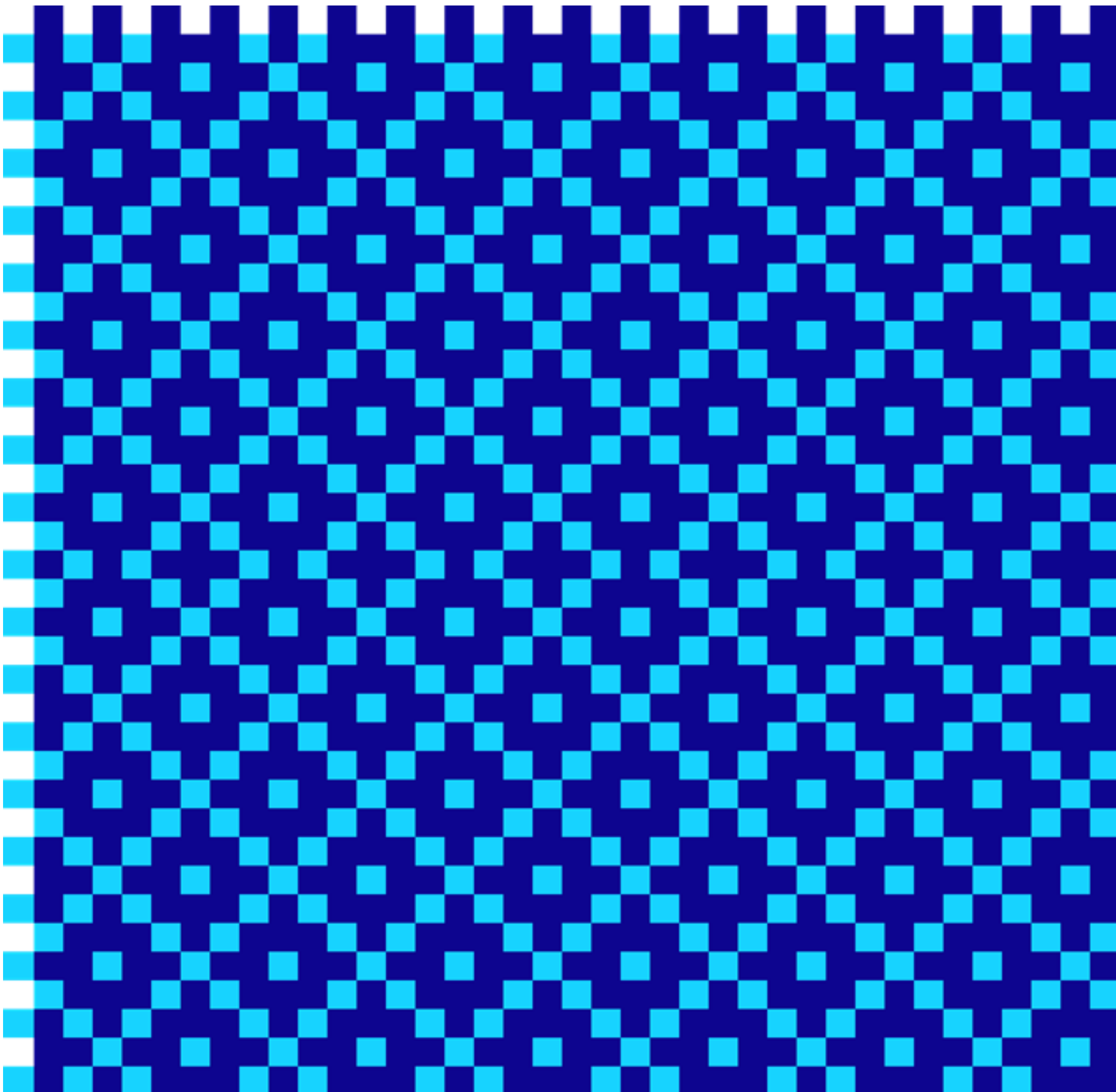
For the latest technical documentation, see the Documentation Portal.

Thredd 6th Floor, Victoria House, Bloomsbury Square, London, WC1B 4DA

Support Email: occ@thredd.com

Support Phone: +44 (0) 203 740 9682

© Thredd 2024



Copyright

© Thredd 2024

Trade Mark Notice: FEATURESPACE, ARIC, AMDL, OUTSMART RISK, and the FEATURESPACE ORB are registered trademarks and/or images in the UK, US, and EU.

The material contained in this guide is copyrighted and owned by Thredd Ltd together with any other intellectual property in such material.

Except for personal and non-commercial use, no part of this guide may be copied, republished, performed in public, broadcast, uploaded, transmitted, distributed, modified or dealt with in any manner at all, without the prior written permission of Thredd Ltd., and, then, only in such a way that the source and intellectual property rights are acknowledged.

To the maximum extent permitted by law, Thredd Ltd shall not be liable to any person or organisation, in any manner whatsoever from the use, construction or interpretation of, or the reliance upon, all or any of the information or materials contained in this guide.

The information in these materials is subject to change without notice and Thredd Ltd. assumes no responsibility for any errors.

About

This document describes how to configure the rules used by the Thredd Fraud Transaction Monitoring System, using the AMDL (Advanced Model Definition Language) Business Rules.

Note: The Fraud Transaction Monitoring System is based on the Featurespace ARIC system. This guide refers to the Featurespace ARIC Risk Hub User Interface as the Fraud Transaction Monitoring Portal.

Target Audience

This audience for this document is Thredd clients (Program Managers) who are using the Thredd Fraud Transaction Monitoring System.

What's Changed?

If you want to find out what's changed since the previous release, see the Document History page.

Related Documents

Refer to the table below for other documents which you use in conjunction with this guide.

Document	Description
Fraud AMDL Rules Configuration Guide	In-depth explanations on how to configure the rules used by the Fraud Transaction Monitoring System.
Fraud Transaction Monitoring System Access Configuration Guide	Describes how to set up user access and user access roles.
Fraud Transaction Monitoring Portal Guide	Describes how to use the Fraud Transaction Monitoring Portal to review high-risk incidents flagged by the system, configure the behaviour of the system, and view, manage and understand the various reports and metrics available in the portal.
Fraud Transaction Monitoring Portal Client Landlord Guide	Describes how to use the Fraud Transaction Monitoring Portal as a Client Landlord user.

Other Guides

Refer to the table below for other relevant documents.

Document	Description
Payments Dispute Management Guide	Describes how to manage chargebacks and the disputes management process using Thredd.
Smart Client Guide	Describes how to use the Thredd Smart Client to manage your account.

Tip: For the latest Thredd technical documentation, see the Documentation Portal.

Overview

This guides describes how you can start configuring fraud rules using the Fraud Transaction Monitoring System. The document includes:

- A definition of a fraud rule
- Descriptions of what you need to consider before creating a fraud rule
- Examples of how to create fraud rules

What is a Fraud Rule?

A fraud rule is an essential component in the Fraud Transaction Monitoring System for managing and preventing fraud. As a rule builder, you configure rules for identifying fraud and reducing the number of occurrences where legitimate transactions are flagged as suspicious, or where transactions are inadvertently stopped and the cards have been locked.

You create rules using AMDL (ARIC Modelling Data Language) on card transactions through the portal interface of the Fraud Transaction Monitoring System. These rules can trigger an incident based on an event, for example, authorisations on card transactions, meeting the criteria of the rule. An Investigator can then use the portal interface to review incidents where the fraud rules have been applied.

Fraud Rule Components

When you configure a rule, there are several components that you define using AMDL expression variables. These include the:

- **Event:** – The activity that can cause a fraud rule to trigger, for example, a transaction.
- **Entity:** – An object associated with an event that allows it to take place. For example, in a card transaction event, the card and the customer are considered entities.
- **Entity State:**– Information about the entity that the system has accumulated over time. The entity states that are stored against multiple events over time, or events where the values are added up across a proportion of the population. Every event processed by the system can update an entity's state, adding more information for the Fraud Transaction Monitoring System to build a behavioural profile of, for example, a card. This allows the rule engine to decide on an outcome using in-depth analysis.
- **Alerts and Tags:** – Definition of the outputs of a rule in the form of alerts and tags.

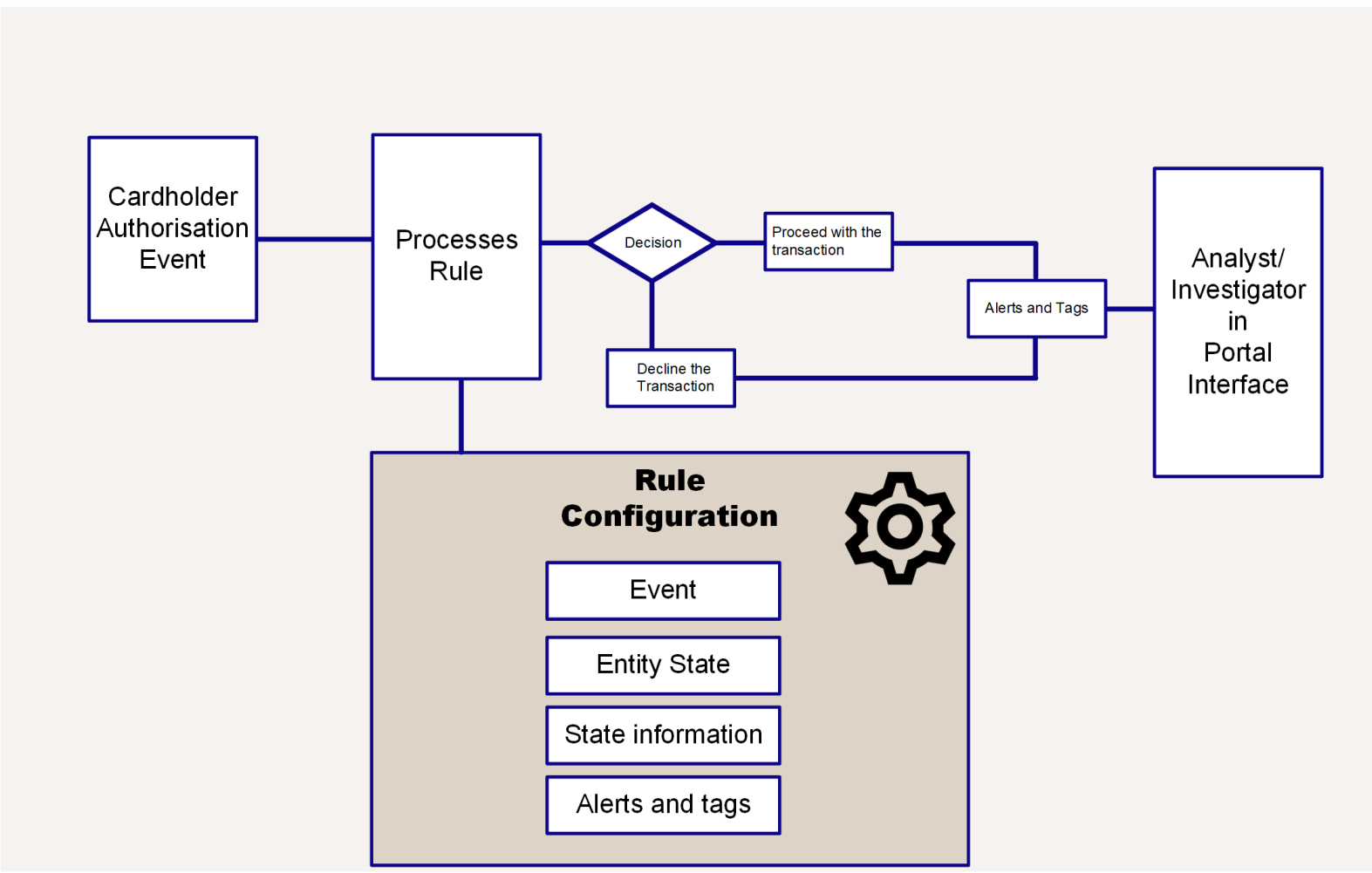


Figure 1: Rules and Rule Components in the Fraud Transaction Monitoring System

Considerations Before Creating Fraud Rules

Before configuring a rule using Fraud Transaction Monitoring, you should consider the following:

- The customer base and segment of your card programme based on factors like transaction history, risk profile, and the types of products or services that they use.
- The role that is set up for you in the Fraud Transaction Monitoring System. Creating and editing fraud rules, as well as publishing to the Live environment, requires the Risk Manager role.
- There exists a sandbox environment in the Fraud Transaction Monitoring System for testing the rules that you create. The environment allows you to adjust the rules before they are published to the Live environment.
- You should have some familiarity with the syntax and logic of declarative programming languages. AMDL is a declarative programming language.

Worked Examples

This section describes how you can create three types of rules using AMDL. You create rules from the **Rules** menu in the **Analytics** section of the portal interface.

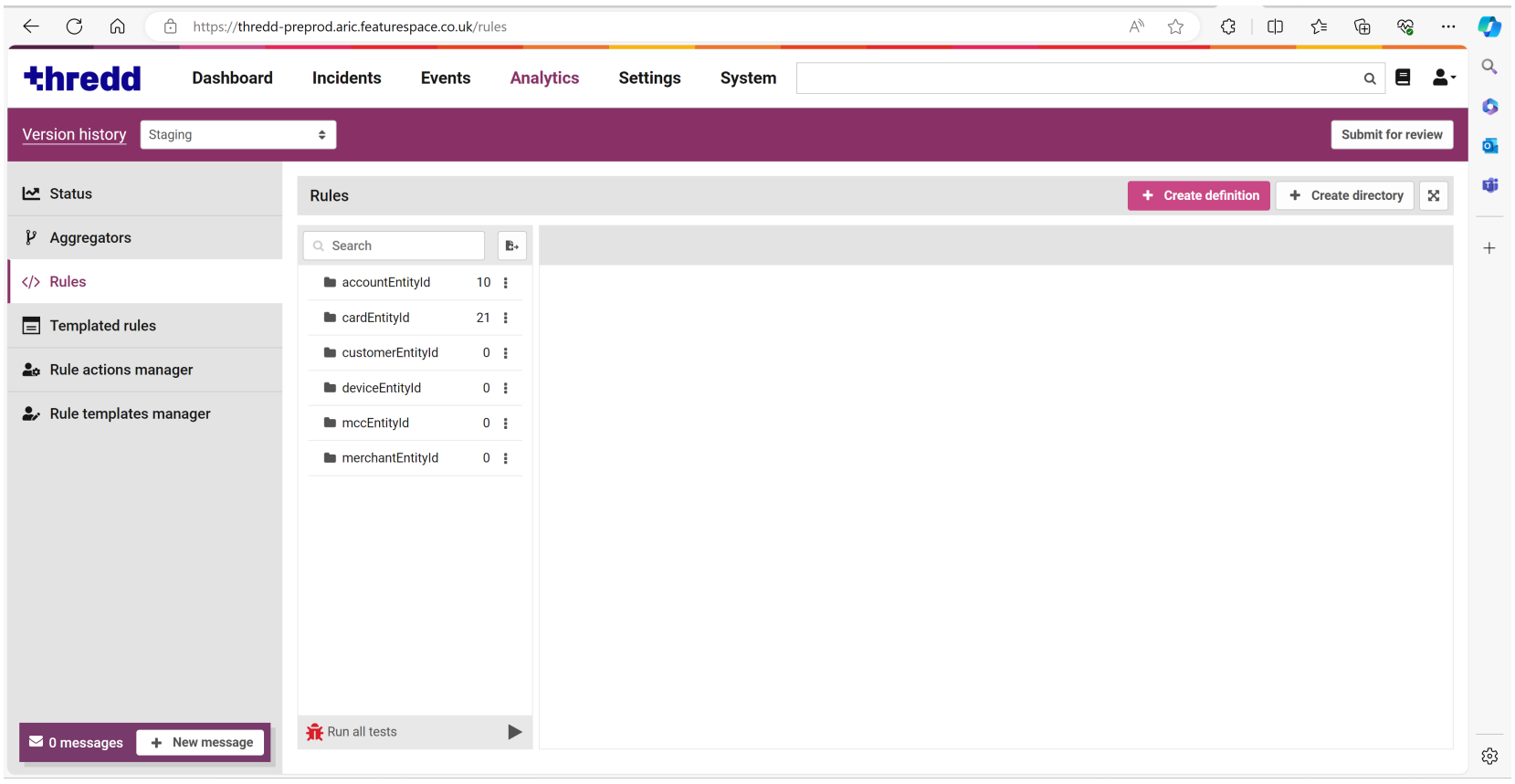


Figure 2: Portal Interface in the Fraud Transaction Monitoring System

The rules described in this section include a:

- Rule with an entity.
- Rule for all transactions in a period.
- Rule for all transactions at a merchant.

If you are new to Fraud Transaction Monitoring System, you should start by creating the first rule, and then create the second and third rules.

For each rule you create, you can run unit tests to ensure that they work correctly. The unit tests generate an event with the data appearing in JSON format.

Rule with an Entity

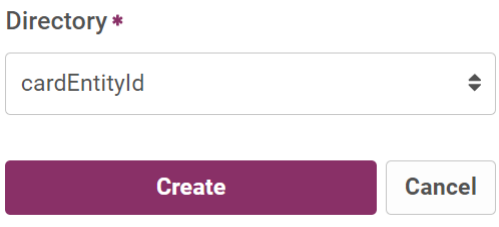
In this example, you create a new rule with a card entity. The rule includes an alert (by using the `@alert` annotation). The `eventType` is cardRT and the authorisation `baseValue` is equal to or greater than 1000. Thredd converts any currency the cardholder has on the transaction into GBP and displays it in the `baseValue` field.

Note: CardRT represents real-time transactions where the cardholder information is passed at the time of the transaction. Conversely, CardNRT indicates non-realtime transactions where the merchant passes the card information later in a working day rather than immediately.

Create a Rule

1. Log into the UI.
2. Go to **Analytics**.
3. Go to **Rules**.
4. Click **Create Definition** and select the new rules destination.

5. Select the directory for the Entity you would like the rule to be under. For the rule to trigger for a card entity, you select **cardEntityId** and click **Create**.



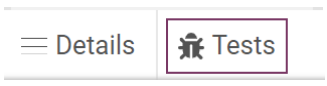
6. Type in the following in the blank rule block that appears when you have selected the directory:

```
@alert
@eventType("cardRT")
@eventType("cardNRT")
rules.RuleExample1:
event.amount.baseValue >= 1000
```

7. Click **Save** to create the rule in staging. When you have saved the rule, you can perform unit tests.

Run Unit Tests:

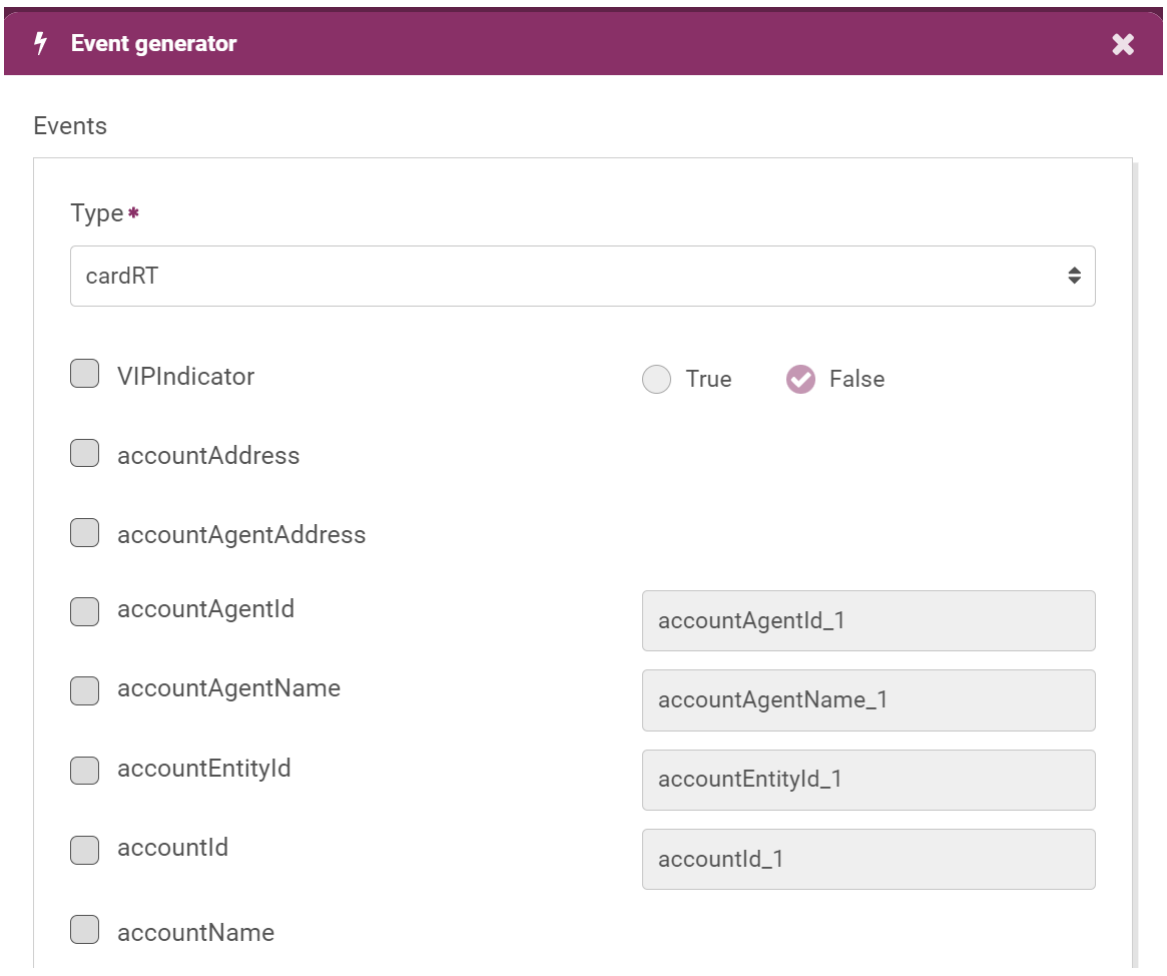
1. Open the created rule and click **Tests**.



2. Click **+** under the Tests options to add a new unit test.
3. Click **New test**.
4. Click **Event generator** from the Input event (JSON) section.

5. From the Event Generator window, select the event type as cardRT in **Type** from the menu.

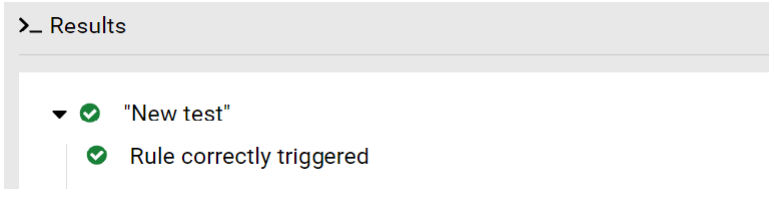
6. Keep the populated mandatory and non-mandatory values for generating an event.



For this example, ensure that the baseValue is set to a figure over 1000, for example, 2000.

7. Click **Submit** to generate an event.
8. Under the Check Rule list, select either **Check triggers**. Check triggers tests that there is a positive outcome, while **Check does not trigger**, checks for a negative outcome.
9. Click **Run tests**. If the test is successful, a tick appears in the Results pane.
10. Click **Save** to save the test.

    The ticks under the Results section indicates that the rule was correctly triggered.



Rule for All Transactions in a Period

In this example, you can extend the basic rule that you created in the first example to create an Entity State that collects data of all ATM transactions in the last 30 days for a token.

Details of the Entity State

The state includes the following conditions:

- Identifying if the transaction took place outside of the country.
- Identifying if the billing amount of the ATM transaction exceeds 30 in the last 30 days.
- Indicating if the MCC code is 6011.

The state also include a histogram for analysing the data over time. The histogram includes a bucket size in order to collect the accumulated values every 2 days.

Details of the Rule

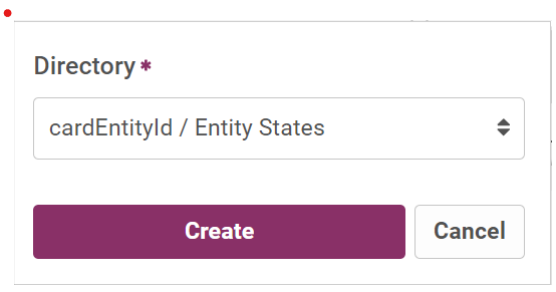
When you've specified the state, you can create a rule that uses the data in the state. In this example, the rule creates an alert (by using the `@alert` annotation). The rule contains the following information:

- `eventType` is cardRT or cardNRT.
- The MCC in the rule is 6011.
- Transactions are through an ATM where the cumulative value of all ATM transactions in the last 30 days is greater than 1000.
- The token is not listed in the `whitelistedCards` data list.

Create a Rule

You need to include details of the state in its own directory, and then create the rule.

1. If a directory for the state doesn't exist, create a directory for the state.
    1. Click **Create Directory**.
    2. In the displayed window, enter a suitable name (for example Entity State) in **Directory Name** and click **Create**.
2. When you have created the directory, click **Create Definition** and select the created directory.



3. Click **Create**.
4. Type in the following in the blank rule block.

```
@eventType("cardRT")
@eventType("cardNRT")
@histogram(historyLength=30d, bucketSize=1d)
state.ATM_History30days:
(event.cardCountryCode != event.merchantCountry) &&
event.amount.baseValue > 30 &&
(event.merchantCategoryCode == "6011") ? event.amount.baseValue
```

5. Click **Save**.

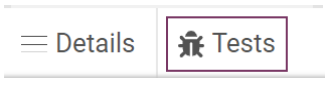
6. When you have created the state, create the rule in the **cardEntityId** directory in the blank rule block.

```
@alert
@eventType("cardRT")
@eventType("cardNRT")
rules.RuleExample2:
event.merchantCategoryCode == "6011" &&
lists.whitelistedCards !# event.cardId &&
((state.ATM_History30days.total(30d) ?? 0) + event.amount.baseValue) > 1000
```

7. Click **Save** to create the rule in staging. When you have saved the rule, you can perform unit tests.

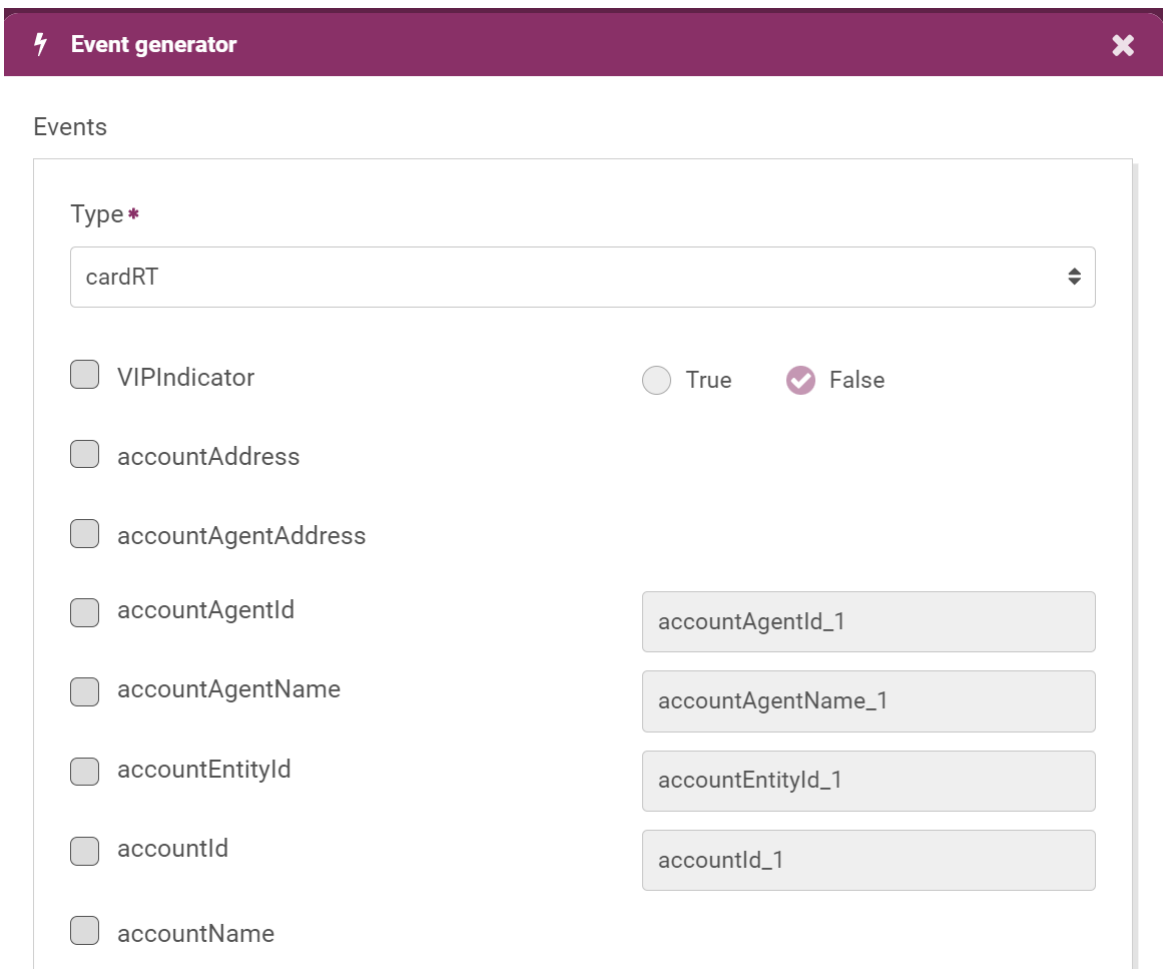
Run Unit Tests

1. Open the created rule and click **Tests**.



2. Click **+** under the Tests options to add a new unit test.
3. Click **New test**.
4. Click **Event generator** from the input event (JSON) section.

5. From the Event Generator window, select the event type as CardRT **or** CardNRT from the **Type** menu.



6. Keep the populated mandatory and non-mandatory values for generating an event.
7. In this example, enter a baseValue of over 1000. Ensure that you set the mccEntityId to 6011. You can refer to the following JSON example to ensure that the values match:

```
{
   "amount": {
        "baseCurrency": "baseCurrency_1",
        "baseValue": 2000,
        "currency": "currency_1",
        "value": 994.8
            },
```

```
            "cardEntityId": "cardEntityId_1",
            "cardId": "cardId_1",
            "cardProductType": "cardProductType_1",
            "direction": "inbound",
            "eventId": "eventId_1",
            "eventTime": "2024-05-28T18:05:01.058Z",
            "eventType": "cardRT",
            "fromId": "fromId_1",
            "localDateTime": "2024-05-28T18:05:01.058Z",
            "mccEntityId": "6011",
            "merchantCategoryCode": "merchantCategoryCode_1",
            "msgStatus": "msgStatus_1",
            "msgType": "msgType_1",
            "schemaVersion": 1,
            "tenantId": "tenantId_1",
            "toId": "toId_1",
            "transactionId": "transactionId_1",
            "transactionType": "transactionType_1"
        }
```

8. In the Initial state area, add the condition to test the state used in the rule.

```
@histogram(bucketSize = 1d, historyLength = 30d)
state. ATM_History30days: {
"data": {
"2023-03-21": { "size":0, "total":850}
}}
```

9. Under the Check Rule list, select **Check triggers**. Check triggers tests that there is a positive outcome, while **Check does not trigger**, checks for a negative outcome.
10. Click **Run tests**. If the test is successful, a tick appears in the Results pane.
11. Click **Save** to save the test.

Rule for All Transactions at a Merchant

In this example, you create a rule that has an Entity State for collecting data of all transactions at Car Rental Agencies (MCC 7512) over the last 30 days in a token.

Details of the Entity State

The state includes the following conditions:

- the `eventType` is cardRT or cardNRT.
- the MCC code is 7512.
- There are five or more transactions for the token associated with the 7512 MCC code in the last 30 days.

The state also includes a histogram for analysing the data over time. The histogram includes a bucket size in order to collect the accumulated values every day for the 30 day period.

Details of the Rule

The rule includes all the above information of the entity state. However, the token is used 5 or more times for a Card Present transaction.

Create a Rule

The Entity State needs to be in its directory. You can use the Entity States directory that you created in the second example.

1. Click **Create Definition**.

2. Select the directory for the Entity you would like the state to be under. For example if you want the state to trigger for a card, you select **cardEntityId > Entity States** and click **Create**.
3. In the blank rule block that appears when you have selected the destination directory of the state, enter details of the state.

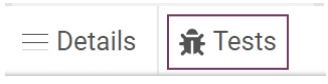
```
@eventType("cardRT")
@eventType("cardNRT")
@histogram(historyLength=30d, bucketSize=1d)
state.Txn_at_MCC7512_30d:
event.msgType.lowercase() == "authorization" &&
(event.merchantCategoryCode == "7512") ? Event.amount.baseValue
```

4. When you have created the state, create the rule in the **cardEntityId** directory in the blank rule block that appears.
5. Enter details of the rule.

```
@alert
@eventType("cardRT")
@eventType("cardNRT")
rules.RuleExample3:
event.msgType.lowercase() == "authorization" &&
event.merchantCategoryCode == "7512" &&
!event.pointOfServiceContext.cardPresent &&
((state.Txn_at_MCC7512_30d.size(30d) ?? 0) + 1 >= 5)
```

Run Unit Tests

1. Open the created rule and click **Tests**.



2. Click **+** under the Tests options to add a new unit test.
3. Click **New test**.
4. Click **Event generator** from the input event (JSON) section.

5. From the Event Generator window, select the event type as CardRT from the **Type** menu.
6. Keep the populated mandatory and non-mandatory values for generating an event. These values match the following JSON.

```
"amount": {
    "baseCurrency": "baseCurrency_1",
    "baseValue": 1001,
    "currency": "currency_1",
    "value": 0
        },
        "cardEntityId": "cardEntityId_1",
        "cardId": "cardId_1",
        "cardProductType": "cardProductType_1",
        "direction": "inbound",
        "eventId": "eventId_1",
        "eventTime": "2024-05-28T22:18:24.237Z",
        "eventType": "cardRT",
        "fromId": "fromId_1",
        "localDateTime": "2024-05-28T22:18:24.237Z",
        "mccEntityId": "6011",
        "merchantCategoryCode": "merchantCategoryCode_1",
        "msgStatus": "msgStatus_1",
        "msgType": "msgType_1",
        "schemaVersion": 1,
        "tenantId": "tenantId_1",
        "toId": "toId_1",
        "transactionId": "transactionId_1",
        "transactionType": "transactionType_1"
            }
```

7. In the Initial state area, add the conditions to test the state used in the rule.

```
@histogram(bucketSize = 1d, historyLength = 30d)
state.Txn_at_MCC7512_30d: {
"data": {
"2024-04-09": { "size":4, "total":500}
}}
```

8. Under the Check Rule list, select **Check triggers**.
9. Click **Run tests**. If the test is successful, a tick appears in the Results pane.
10. Click **Save** to save the test.

Glossary

This page provides a list of glossary terms used in this guide.

A

Aggregator

An aggregator is a type of analytic that can combine and use the outputs of multiple rules and models to generate alerts.

Alert

The Fraud Transaction Monitoring System can flag up high-risk events for alert reviews. A flagged event is said to have generated an alert. The system's analytics rules, models and aggregators) can all generate alerts.

Alert Review

This is where analysts review alerts generated by the Thredd Fraud Transaction Monioring System. They can classify alerts as ‘Risk’ or ‘No Risk’, refer them to other users, or put them aside for further monitoring or to await additional information.

AMDL

AMDL (ARIC Modelling Data Language) is a language for specifying rules and logic within the Fraud Transaction Monitoring System. It is a declarative language for specifying state updates and executions on each event that passes through the system. An example of an event is an account registration or a transaction. Every event contains a reference (for example, an ID field) to one or more entities of different types, such as a merchant and a consumer. You can use AMDL to create Business Rules for the detection of fraud.

C

Chargeback

Where a cardholder disputes a transaction on their account and is unable to resolve directly with the merchant, they can raise a chargeback with their card issuer. The chargeback must be for a legitimate reason, such as goods and services not received, faulty goods, or a fraudulent transaction. For more information, see the Payments Dispute Management Guide.

E

Entity

Events happen to entities. An entity represents a unique individual or object, and every event is associated with at least one entity. For example, if a customer makes a card transaction, that event can be associated with the customer entity, the card entity, or both.

Entity ID

Each entity is identified by a unique entity ID in the event data for example, a 16-digit token.

Entity State

Every entity has a state – a combination of information about the entity that the system has accumulated over time. This is also called a behavioral profile. Every event processed by the system has the potential to update an entity’s state, adding more information or updating information that the system can use to build a behavioral profile of a customer or card for example.

Event

The Fraud Transaction Monitoring System recognizes potential fraud and financial crime by monitoring events. An event could be a customer transaction, a new customer application, or a merchant attempting to process a payment – these are all examples of event types. Each event is associated with one or more entities and one or more solutions.

I

Incident

In the Fraud Transaction Monitoring System, alerts are grouped into incidents. Each incident contains all the unreviewed alerts related to a particular entity.

Issuer (BIN sponsor)

The card issuer, typically a financial organisation authorised to issue cards. The issuer has a direct relationship with the relevant card scheme (payment network). For more information, see the Key Concepts Guide.

L

Label Events

Label events are types of event that contain ground truth information. They are used to label other events as 'risk' (i.e. confirmed fraud, financial crime, etc.) or 'no risk' (i.e. genuine). Alert reviews are one common form of label event, but your portal may also use other kinds of label event, such as chargebacks or manual fraud reports. Labels are used by Adaptive Behavioral Analytics models to learn to better identify high-risk events. They are also used to quantify and report on the performance of models.

M

Mastercard Fraud and Loss Database

A Mastercard repository of fraud transactions submitted by issuers. It is used for reporting, monitoring, and combating card fraud. Previously know as: System to Avoid Fraud Effectively (SAFE).

MasterCom API

MasterCom API offers Mastercard customers the ability to create and manage dispute claims in MasterCom. MasterCom is a system for dispute management. All activities for any given dispute can be tracked within a single claim using Mastercom, including Retrieval Request and Fulfilment, First Chargeback, Second Presentment, Fraud reporting, Case Filing, and Fee Collection requests. All activities for any given dispute throughout its lifecycle can be tracked within a single claim.

Model

A model in the Thredd Fraud Transaction Monitoring System is a predictive model that processes events and generates a risk score for certain event types, for example, authorisations.

P

PAN

The Primary Account Number (PAN) is the card identifier found on payment cards, such as credit/debit/prepaid cards, as well as stored-value cards, gift cards and other similar cards. The card's 16-digit PAN is typically embossed on a physical card. For more information, see the Key Concepts Guide.

R

Real time/near-real time events

Every event is processed by analytics in the Featurespace fraud monitoring system engine. This processing happens in strict chronological order, so that no event is ever processed out of sequence. This is asynchronous processing, and happens to all events. However, some events, such as authorisations, require a real-time response (within a few hundredths of a second). These must be processed in a way that prioritizes low latency (such as a fast response), rather than chronological order. This kind of event is called a real-time event, and is processed by the portal synchronous response generator (as well as the portal Engine). Events that do not require a real-time response (asynchronous events), are only processed by the engine, for example, chargebacks, address or phone number updates

Rule

A rule defines some simple logic – rules take in information from events, entity states, and other data, and output a simple true/false response. Rules are written in the business logic definition language, AMDL.

Rule Set

Each Analytical Workflow is divided into a series of Rule Sets. Each Rule Set contains a number of expressions written in AMDL, and one or more Scorecards which contain conditions that determine what effects the Workflow triggers (e.g. generating an alert, adding a tag, outputting a risk score). Each Rule Set may also have a condition that determines whether or not that Rule Set is executed for an event.

S

Single Sign-On (SSO)

An identification method that enables users to log in to multiple applications and websites with one set of credentials.

Smart Client

Smart Client is Thredd's user interface for managing your account on the Thredd system. Smart Client is installed as a desktop application and requires a VPN connection to Thredd systems in order to be able to access your account. For more information, see the Smart Client Guide.

Solution

Multiple product Solutions may be configured in your portal deployment. Each Solution provides a combination of UI configurations, data enrichment and analytics for detecting a specific type of risk. For example, you may have a Solution for application fraud and another for inbound/outbound payments, subject to your programs set up with Thredd and Featurespace. The same event may trigger separate alerts in different Solutions.

Solution ID

Each Solution is uniquely identified by a Solution ID in the event data.

Solution UI

The fraud system user interface that users access when they open the relevant Solution. The Solution UI is mainly used for reviewing incidents that are specific to that Solution, and can be customized for detecting the relevant type of financial risk.

State

Every entity has a state – a combination of information about the entity that the systems has accumulated over time. This is also called a behavioral profile. Every event processed by the system has the potential to update an entity's state, adding more information or updating information that the system can use to build a behavioral profile of a customer or card for example.

T

Tag

Rules, aggregators and models can add tags to alerts, to give analysts more information or to automate a response in a downstream system, such as declining a transaction.

Token

Displays the unique token linked to the card PAN on which the transaction was made.

Document History

Version	Date	Description	Revised by
1.0	21/06/2024	First version	KD

Contact Us

Please contact us if you have queries relating to this document. Our contact details are provided below.

Thredd Ltd.

Support Email: occ@thredd.com

Support Phone: +44 (0) 203 740 9682

Our Head Office

6th Floor,

Victoria House,

Bloomsbury Square,

London,

WC1B 4DA

Telephone: +44 (0)330 088 8761

Technical Publications

If you want to contact our technical publications team directly, for queries or feedback related to this guide, you can email us at: docs@thredd.com.