FlexNet Operations Solution Design - Technical Part

2024-07-26

Table of Content

[1 Introduction 3](#_Toc178778295)

[1.1. Objectives of this document 3](#_Toc178778296)

[1.2. Context 3](#_Toc178778297)

[1.3. Contributors 4](#_Toc178778298)

[1.4. Validation 4](#_Toc178778299)

[1.5. Glossary 5](#_Toc178778300)

[2 FNO Design Principles 7](#_Toc178778301)

[2.1. Key Solution Design Drivers 7](#_Toc178778302)

[2.2. Solution Design Decisions and Assumptions 8](#_Toc178778303)

[3 Data Model 16](#_Toc178778304)

[3.1. Introduction 16](#_Toc178778305)

[3.2. Global Entity Relationship Diagram 16](#_Toc178778306)

[3.3. Usage for Materialise Digital Products 17](#_Toc178778307)

[3.4. Validation Rules ? 26](#_Toc178778308)

[4 Front-End 27](#_Toc178778309)

[5 Back-End 28](#_Toc178778311)

[5.1. Introduction 28](#_Toc178778312)

[5.2. Jobs 28](#_Toc178778313)

[6 Integration 29](#_Toc178778314)

[6.1. Overall Information Flows 29](#_Toc178778315)

[6.2. Integration Scenarios for FlexFlex 29](#_Toc178778316)

[6.3. Custom API 31](#_Toc178778317)

[7 Data Security 33](#_Toc178778318)

[8 Annex 34](#_Toc178778319)

[8.1. Data Model FNO 34](#_Toc178778320)

[8.2. System Constraints and Timings 48](#_Toc178778321)

# Introduction

## Objectives of this document

The aim of a **Solution Design** is to provide a high-level visual - textual representation of the envisioned solution that outline how the company will support the maximization of the solution’s features richness and quality and at the same time minimize the total cost of the solution. In essence, a Solution Design supports **Design Choices**.

This document covers the **technical part** of Solution Design, the functional part of the Solution Design finds place in Azure DevOps. This document is not the full solution documentation, as such solution design is only a sub-set of the full solution documentation.

This document represents the solution design of the FlexNet Operations for the technical part of it at a certain moment. New elements that appear from the previous version will be highlighted and described; elements already present from the previous version published are kept in this document as well.

## Context

### Introduction

This section outlines the new element’s part that have been added to the solution design of FlexNet Operations since its previous version published.

### Scope, Objectives and Business Drivers

<Link to Release / Project charter document of associated release / project>

<Link to Scope of associated maintenance / service activities>

### Capabilities

<Link to domain architecture document – capabilities section of associated release / project>

### Applications Landscape

<Link to domain architecture document – applications landscape section of release / project>

### Information Flows

<Link to domain architecture document – information flows section or release / project>

### End to End process Flows

<Link to end-to-end process flows of associated release / project>

## Contributors

This section lists the contributors to the new elements part of this version.

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Function | Topic | Date |
| Julian Smith | Solution Architect | All | 2024-07-26 |
|  |  |  |  |
|  |  |  |  |

## Validation

This section lists the validators to the new elements part of this version.

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Function | Topic | Date |
|  | Process Owner | Logical Data Model |  |
|  | Subject Matter Expert | Logical Data Model |  |
| Yves Houtevels | Domain Architect | All |  |
| J-Fr. Bauduin | Lead Enterprise Architect | All |  |

## Glossary

### Architecture and Applications

This [glossary](https://materialisenv.sharepoint.com/sites/extranet-Tiger_implementation/Tiger/ALL%20Architecture%20Deliverables/99_TIGER%20Architecture_Glossary%20General%20Capabilities%20Applications%20OLD.xlsx?d=wea172db0a3f54f2cb006fa7a7f0d5812&csf=1&web=1) encompasses general IT acronyms, capabilities and applications.

In this document the terms FlexMat and FlexFlex will used. In order to clarify these terms, a definition is provided:

* FlexMat: FlexMat refers to the LEGACY license technology. With FlexMat the Materialise license technology (MAT) is used together with the Revenera license technology (features and activations)
* FlexFlex: FlexFlex refers to the FUTURE PROOF license technology. With FlexFlex only the license technology of Revenera is used.

### Revenera and FlexNet Operations

This glossary encompasses specific Revenera and FlexNet operations in addition to the general glossary above. Product documentation for FlexNet Operations and FlexNet Embedded can be found on the Revenera Community documentation pages [here.](https://docs.revenera.com/)

Specific Revenera product documentation is provided by the following links:

* [FlexNet Operations 2024 ALM Release Notes](https://docs.revenera.com/fno_CURRENT/rn/Default.htm)
* [FlexNet Operations User Guide](https://docs.revenera.com/fno_CURRENT/producer/Default.htm)
* [FlexNet Operations Web Services Integration Guide](https://docs.revenera.com/fno_CURRENT/webservicesintegration/Default.htm)
* [FlexNet Operations SOAP Web Services Guide](https://docs.revenera.com/fno_CURRENT/soapapi/Default.htm)
* [FlexNet Embedded License Server Administration Guide](https://docs.revenera.com/fne/2024_06/adminguide/Default.htm)
* [FlexNet Embedded License Server Manager Guide](https://docs.revenera.com/fne/2024_06/flsmguide/Default.htm)

#### Early Binding

Early binding (of licenses) refers to the practice of requiring customers to provide information about the systems where licenses will be deployed: usually, this means providing the host-id of the target systems either when the software order is placed or before licenses are generated. This approach is quite common, especially where the software producer or reseller generates licenses on behalf of their end-customers An example is SEAT in the FlexMat license solution.

#### Late Binding

Late binding (of licenses) means enabling customers to decide which systems to deploy license on at the point that the licenses are generated. This is a flexible approach to license management that allows customers to manage their license lifecycles (activation, returns, rehosting etc..) without involving the software producer: however, it requires that customers have access to a licensing portal – the Revenera End-user Portal for example – or the software enables users to activate licenses electronically. An example of the FlexFlex license solution which allow late binding.

#### Commercial and Technical Products

The term commercial product refers to a product definition in a sales system used to provide quotes and orders to customers when purchasing software. Technical products are created in FNO and define how, in conjunction with license models, licenses are constructed for the commercial products purchased. Oftentimes, commercial and technical products are related using part numbers (SKU, ERP product code, …) but this need not always be the case.

#### Part Numbers and Product Codes

In Revenera there are two types of activatable products:

1. Products that specify a fixed number of features that will be added to licenses.
2. Suites that specify a fixed number of products – fixed bundles – and the sum of all features from the specified products in the suite will be added to licenses.

Part-numbers in Revenera link together products (or suites) and a License Model: based on the part-Number, FNO generates licenses in accordance with technical specification of product (or suite) and metadata provided by the License Model.

It is the sales system’s product codes -- used as Part Numbers in Revenera -- that identify which technical products (or suites) to use when creating Entitlements and Line Items in Revenera.

# FNO Design Principles

Design principles adopted for the FlexMat and FlexFlex phase of the Tiger Project follow established best-practice. This can be summarized by the following:

## Key Solution Design Drivers

### As from release FlexMat

* The CCKey concept is required to support existing software deployments on the client side.
* Avoid customization; prefer an out-of-the-box implementation wherever possible.
* Avoid deploying or maintaining external services where equivalent functionality can be provided by prerequisite systems.
* Avoid duplicating functionality that is provided by prerequisite systems.
* Keep all technical product catalogue and licensing details in FNO; avoid any of this information being required by other systems. FNO is the master application for technical product catalogue and licensing details.
* Maintain all licensing functionality in FlexNet Operations. Avoid implementing Licensing functionality in the technical integration layer (e.g. For FlexMat Boomi is required to generate Key files).

### As from this release FlexFlex

* Use recommended best practices for implementing and managing electronic licensing with FlexNet Operations and FlexNet Embedded.
* Empower customers to choose how and where to deploy licenses.
* Specifically remove CCKey and license-file concepts and the need for early entitlement binding to endpoints: see the glossary for more information.
* Use native FlexNet Embedded license functionality, tools, and products to fulfil licensing requirements.
* Leverage FlexNet Operations’ support for electronic license delivery and management using FlexNet Embedded: do not offload this to the integration layer.
* Use recommended (best) practices off-line license management.
* Use out-of-the-box product and suite packaging for the technical product catalogue.
* Move to a date-based feature approach to remove existing complexities around products and reference-module release cadences.
* Ensure that no impact is made on the existing Flex-Mat (project R1) solution: other than configuration changes to support FlexFlex, all systems should work as is.

### FNO Specific Customizations

Within the constraints of the overall solution, avoid all customizations in FlexNet Operations. For information purposes, customization options could include (inter alia):

* Implementation and deployment of custom key generators.
* Support for custom (3rd -party) license technologies: FlexMat is an example of this.
* Specialized use of FNO entities to model real-world objects or relationships that are not supported natively in the FNO data model.
* Specialized encoding of FNO entity attributes to represent external data links, external system or data references, or external system states.

### Configuration

Avoid configuring the FNO platform or FNO entities unless there is a clear operational benefit, and where no good alternative is available. Configuration options include (inter alia):

* Custom attributes
* License model attributes
* Product line attributes
* License headers and footers
* Customer portal style sheets
* Customer portal page HTTP header and footer
* Various platform settings
* Various FNE global settings

### Simplicity

While there is no technical need to simplify product and feature structures, from a best-practice perspective it should be considered. The impact would be a reworking of the license and feature management within each product.

* The existing product structures are complex and the number of features in each product could be reduced (e.g. products for FlexMat are complicated because of the support of product and reference modules versions and release cadence).
* The existing LicGenId values should be changed to strings that relate to areas of functionality being monetized more clearly.
* Care should be taken where current products share LicGenIds; especially reference modules because sharing features introduces dependencies and potentially monetization issues.

### Transparency

The new system should promote transparency so that customers and partners can verify what licenses they have deployed, and what they have purchased, easily and accurately.

## Solution Design Decisions and Assumptions

### As from release FlexMat

#### Portals

No use of the FNO End-User portal

#### LicGenIds

These are never shared between reference modules; this enables predictable versioning to be implemented because the reference module for any LicGenId may be determined uniquely.

#### Versioning

Perpetual licenses:

Should a perpetual line item have no maintenance defined, the entitlement creation date shall serve as the effective maintenance expiration date.

Licenses for Entitlement Line Items with valid maintenance shall receive the current (latest) reference module version.

Licenses for Entitlement Line Items with expired maintenance shall receive the reference module version that was generally available on the maintenance expiration date.

Limited licenses:

* Are always treated as subscriptions
* Have no separate maintenance
* Licenses shall always receive the current (latest) reference module versions.

Beta Licenses

* Are always limited
* Licenses shall always receive the latest reference module versions that are flagged as beta.

#### Part Numbers

All part numbers are mastered in the SFO Product object.

* Modules shall have separate part numbers for Limited and Perpetual licenses; hence, also in Salesforce.
* Beta modules shall have separate part numbers.
* Beta modules shall always be sold with Limited licenses.
* Floating and Local license (deployment) models shall not be differentiated by part numbers.

#### Legacy License Metadata

Floating and Local license (deployment) models shall be indicated by flags in the asset extracts from SFO. These shall not have separate license models.

#### License Technology

When licenses for a non-Revenera License Technology are required, FNO allows bespoke License Technologies to be configured through the following means:

* A Remote SOAP Service,
* Manual or
* Hands-Free License Technology

(source-Revenera website: [https://docs.revenera.com/fno2020r1sp1/producer/Content/helplibrary/opsGS\_NonFN\_Technologies.htm#fno\_gs\_non\_fn\_555546149\_1115829](https://docs.revenera.com/fno2020r1sp1/producer/Content/helplibrary/opsGS_NonFN_Technologies.htm" \l "fno_gs_non_fn_555546149_1115829))

Comparison between Hands free and Remote Soap Service:

|  |  |  |
| --- | --- | --- |
|  | Hands Free | Remote Soap Service |
| **+** | * Can be used for MTLS legacy products. * Fewer webservice calls (only need to decode the entitlement structure once) | * Can be used for MTLS legacy products. * FNO customer portal can be used |
| **-** | * FNO customer portal cannot be used | * Needs out of the box FNO product definitions (not suitable for MTLS legacy products) * More webservice calls (1 for every line item), will have negative impact on performance. * Extra service to be created and hosted |

Advice is to use hands-free technology to implement **legacy** key-file generation (instead of Remote SOAP Service).

Legacy products will use a hands-free license technology and licenses will not be generated automatically through FNO. Boomi orchestrates the license generation and activation process.

#### Accounts

Accounts in Revenera shall be created lazily during entitlement management – i.e. only when they are needed for entitlement ownership.

There will be no separate integration with SFO, or any other system, to synchronize account information. If an account is blocked, SFO is responsible to block the active Entitlements (if desired).

#### Integration

Web services shall be provided that implement the integration functionality to create and manage entitlements. These shall be invoked from BOOMI.

#### License Model Attributes

The following License Model Attributes shall be used to provide information to the license generation process and will be provided automatically through the License Model configuration.

* Perpetual
* Limited
* Beta

An additional attribute will be used to capture the non-revenue type of the license and will be injected during the Entitlement creation through the automation based on attributes provided by SFO.

* RevenueClass (Revenue, Evaluation, Testing, …)

#### License Duration

Because license key-files and passwords cannot be revoked once in the field, for perpetual licenses, Materialise has only supplied time-limited license to its customers; this practice will be continued.

FNO will be configured so that the required license duration will be stored with each entitlement; the assumption is that SFO will supply this information at order fulfilment and update FNO whenever this value changes.

* Initial duration representing the payment terms – say 45 days from start date.
* Key file or password duration once payment has been received.

Blocked status when payment has not been received that will prevent licenses being generated; this will be a simple Boolean value with a reason.

The initial pre-payment duration for licenses will be an attribute of the Entitlement and must therefore be provided by SFO when Entitlements are created.

FNO shall be notified that payment has been received and the actual license duration provided.

FNO shall be notified that payment has not been received to prevent subsequent license activation.

#### Install Base Information

FNO can capture information when licenses are activated but these requirements have not been discussed. E.g. information captured from the forms filled in during the activation process.

What information is required?

### As from release FlexFlex

#### Activation

A key difference between the current solution (FlexFlex) and FlexMat is that there will be no middleware involvement (i.e. Boomi) in the activation process: Entitlement line items will be activated using standard, built-in mechanisms. That is, one of the following:

* A capability exchange using one of the FNE toolkits,
* Web-based REST calls using the Cloud Monetization API (CMAPI),
* Managing devices and entitlements in the FNO producer portal or the customer portal.

This allows metadata to be injected into the license, using the license model and license model attributes or custom attributes, or data to be collected against the device in the vendor dictionary but none of these mechanisms can affect key data like start date, expiration date, quantity for instance.

#### Blocking or Disabling a License

Revoking activated licenses is problematic as it requires synchronization between the device and FNO: the device needs to call home or process a capability response off-line.

1. Set the entitlement state or the line-item state to INACTIVE or OBSOLETE. Note that setting the state to OBSOLETE prevents the line item or the entitlement ever being used again; the INACTIVE state can be reset later to DEPLOYED.
2. Set the expiration date to a date in the past.

In both cases above, calling home (resynchronizing a license) will result in the license rights being removed for the client.

#### Payment Period

To limit the generated license to duration shorter that the full subscription term – for instance, to reflect the payment period -- requires modifying the line-item expiration date it may also require modifying the start-date option to have a fixed expiration date. Once payment has been received, the subscription expiration can be reset to the subscription expiration.

#### Feature Versioning

Adopt date-based versioning where the entitlement defines feature versions which are validated against product release dates. This will bring significant benefits:

* Removes the need to cater for reference model release dates and the need to include reference modules in the technical product catalogue.
* Removes the need to deploy any new versions of any products.
* Fits comfortably with a subscription approach.
* Caters for perpetual licenses with software maintenance elegantly.
* Start Dates

For direct sales, the contract start date shall define the start date for activated licenses. For indirect sales and some vouchers, the license start date will be based upon the first activation date.

#### Features

Previous licgenids will be replaced by FNO Features: these are unique strings -- not integers. This is strictly not a change from the earlier solution (FlexMat) but considering them as names allows for more meaningful product structures moving forward.

Quite often customers adopt naming conventions that avoid features being shared, for instance

{product code}.{feature name}

The following illustrates this principal for 3-Matic Medical: features have a name and some licgenids have been combined into a singe feature.

|  |  |  |
| --- | --- | --- |
| Licgenid | FNE Feature Name | Description |
| 861 | 3-matic.med.fea | Remeshing |
| 862 | 3-matic.med.base | Import Mimics |
| 863 | 3-matic.med.base | Fixing |
| 864 | 3-matic.med.design | Primitives |
| 864 | 3-matic.med.analysis | Primitives |
| 865 | 3-matic.med.design | Inspection |
| 865 | 3-matic.med.analysis | Inspection |
| 866 | 3-matic.med.design | Sketcher |
| 867 | 3-matic.med.design | Boolean |
| 868 | 3-matic.med.design | Thickness |
| 869 | 3-matic.med.design | Construct |
| 870 | 3-matic.med.design | Curves |
| 871 | 3-matic.med.design | Finishing |

#### Validity Periods

Validity periods will be required; however, there is no requirement to model a validity start date. In FNO entitlements do not have a time-to-live and line item may be activated at any point. Limiting the availability of line-items within an entitlement will require intervention from the middleware to obsolete the entitlement or line items.

#### License Technology

The solution will use FlexNet Licensing technology: this is the default for FNO when using FNE and requires no configuration.

#### Part Numbers

New part numbers will be required for the FNE enabled products: the role of part numbers will not change.

#### License Metadata

License metadata will be used to tag floating licenses so that only these will be made available from FNE license servers. Technically, a selector value will be placed into the license which, in combination with a client-side implementation in the licensing libraries to add the selector value to the feature request, will prevent local licenses being served. Unfortunately, there is no native mechanism in Revenera to prevent local licenses being provisioned on a license sever and being available to be borrowed.

No other metadata requirements have been identified to date.

#### User Portal

The FNO End-User portal will be used until it can be replaced by a specific Materialise customer facing portal. This is a significant change from the previous release (FlexMat) and exploits existing FNO functionality to be used in entitlement and license management.

Access to the FNO End-User Portal can be authenticated by using activation or entitlement Ids; neither mechanism requires a password. Specific named-user access is also possible but requires customer user management.

* Portal authentication will be restricted to entitlement and activation IDs only for the first release.
* Users will not be modeled: this does not prohibit Materialise administrative portal users accessing the End-User portal.
* Emails will not be sent from FNO.
* Transfer of license rights – from a partner to end-customer for instance – will not be supported.

Users will have the ability to manage the allocation of line-items on devices so that devices calling home will receive the appropriate licenses. This process is cumbersome and restrictive when logging into the FNO End-User Portal with entitlement-Ids or activation-Ids: most devices and entitlements will not be visible because they are not mapped to the activation-id or entitlement-id used to login. For this process to be useful to customers, some authenticated users will need to be created for customers.

#### End-user Portal Users

Creating and management of FNO portal users has not been discussed but has been identified as desirable.

#### Features

The current set of features – for FlexMat products – should not need to change; they are already date-based and can be incorporated into newly packaged FlexFlex products and suites.

#### Feature Bundles

Feature bundles will not be used.

#### Products and Suites

In line with best practice, the product structure used for FlexMat will not be followed for FlexFlex; new products should be identical in structure to the existing green products (see diagram in annex) -- this is the FlexMat products that define the licgenids required by a module -- if there is no feature simplification.

1. Standard flat products will be created for all FNE enabled orderables -- products or suites.
2. Suites will be used only where they map directly to a fixed CPQ bundle.
3. No modelling of reference modules or reference module releases.
4. No modelling of module and reference module licgenid relationships.
5. Part numbers may reference suites or products in line with commercial products in the order.

#### Attributes

Of the custom attributes and license model attributes defined for FlexMat, some of the following may be used:

* Entitlement
* PaymentState
* TargetPaymentDate
* ValidityStartDate
* ValidityEndDate
* EmergencyCount
* RehostCount
* License Model Attributes

The following new **Product** custom attribute will be created to identify which download package to use when customers select products in the Passwords Website:

|  |  |  |
| --- | --- | --- |
| Name | Type | Values |
| DOWNLOAD\_PACKAGE | TEXT | Free format |

The following new license model attributes will be created:

|  |  |  |
| --- | --- | --- |
| Name | Type | Values |
| LICENSE\_MODEL | TEXT - Single Select | FLOATING  LOCAL |

#### License Models

Two new license models will be added with the following attributes: these will work in conjunction with the new LICENSE\_MODEL license model attribute to define selector values to prevent nodelocked license being consumed from a license server.

|  |  |
| --- | --- |
| Name | Configuration |
| Flex Subscription | [FlexFlex.license.models.jar](https://dev.azure.com/MaterialiseTiger/cac9de9d-c154-4ddc-a6f8-57eb68d39996/_apis/wit/attachments/30ecd1c4-aefb-4dcb-b592-c7639ffe46ce?fileName=FlexFlex.license.models.jar&download=true) |
| Flex Permanent | [FlexFlex.license.models.jar](https://dev.azure.com/MaterialiseTiger/cac9de9d-c154-4ddc-a6f8-57eb68d39996/_apis/wit/attachments/30ecd1c4-aefb-4dcb-b592-c7639ffe46ce?fileName=FlexFlex.license.models.jar&download=true) |

#### Products

Standard technical products will be created in FNO.

|  |  |
| --- | --- |
| Name | Configuration |
| Software | TBD |
| Medical | TBD |

#### Entitlement Structure

It has been agreed that for subscriptions, a contract in Salesforce will equate to a single entitlement in FNO and subscription lines in a contract equate with line items in the same entitlement.

#### General

Entitlement and line-item identifiers will be generated in FNO automatically: Salesforce identifiers will not be used in this primary role.

If there is a need to capture any Salesforce identifiers in FNO, this will be done with the addition of custom attributes; or the reuse of existing attributes if this is feasible.

#### Amendments

Additional subscription lines in a contract will be modelled as new line items in the entitlement relating to the original contract: these can be added safely for a future date.

Ante-dated contract changes will not be generated in FNO; reductions in quantity, cancellation, or replacement of subscription lines must be made in FNO on the date the change comes into effect. Whilst it is technically possible to restructure the entitlement to reflect the current and the future states following the contract amendments, it is too complicated to implement effectively through Boomi, and changes the entitlement structure in non-intuitive ways.

This imposes a requirement on the Boomi and Salesforce integration to implement ante-dated contract changes in FNO at the time they would come into effect.

#### Amendments Reducing Quantities

Reducing the number of quantity is possible in FNO but the quantity count cannot be reduced below the number of current activations. This needs to be managed carefully because it can lead to license enforcement problems where the reduction is contracted but Boomi is unable to reduce the quantity count.

#### Start Dates and Expiration Dates

In general, line items in an entitlement will be co-termed with the contract expiration date; however, this is not necessary or mandatory.

Line-item start dates can be defined at entitlement creation – for a fixed date -- or can be specified to use the first activation date and applied subsequently to all line-items in the entitlement.

#### Payment Date and License Term

The payment term mechanism used already in the earlier solution shall be continued. Licenses activated before payment has been received should be for a limited duration specified in Salesforce; once payment has been received, licenses can be reactivated for the entire subscription term.

* Validation Period attribute is not required but may be desirable for keeping the UI in FNO consistent.
* The actual license term must be specified correctly on the line item because Boomi takes no part in the activation process.
* When payment has been received, the line-item detail must be updated to represent the full term of the subscription covered by the payment: for instance, the first year of a multi-year subscription where yearly payments have been agreed.

Line-item start dates are defined at entitlement creation time; either as a fixed date, or fall on the first activation date, or each subsequent activation date.

#### License Activation

Licenses will be activated in several ways:

1. On-line using the inbuilt licensing wizard in the application
2. On-line using a simple call-home
3. Off-line using the license wizard and file transfer from FNO
4. Off-line using call home and file transfer from FNO .

The attributes of the resulting license are determined solely by the line-item configuration in FNO; specifically:

1. Product
2. License model
3. Metadata defined in custom or license model attributes.
4. Date configurations: start date, expiration date, version date.

Unlike the key-file generation in the FlexMat solution, custom attributes cannot alter key properties such as start-date, expiration-date, or license term; nor can they determine whether a license can be generated. For properties such as validity dates, blocked status, and so on, where the intention is that licenses should not be eligible for activation, the line-item state would have not to be in the DEPLOYED state; that is, OBSOLETE, INACTIVE, or DRAFT.

In these cases, the entitlement state, or individual entitlement line-item states would need to be managed and synchronized by the integration layer from data present in Salesforce or potentially other systems.

# Data Model

## Introduction

This section of the document provides information around logical data model of the FlexNet Operations and how it is used to fit to the Materialise digital products.

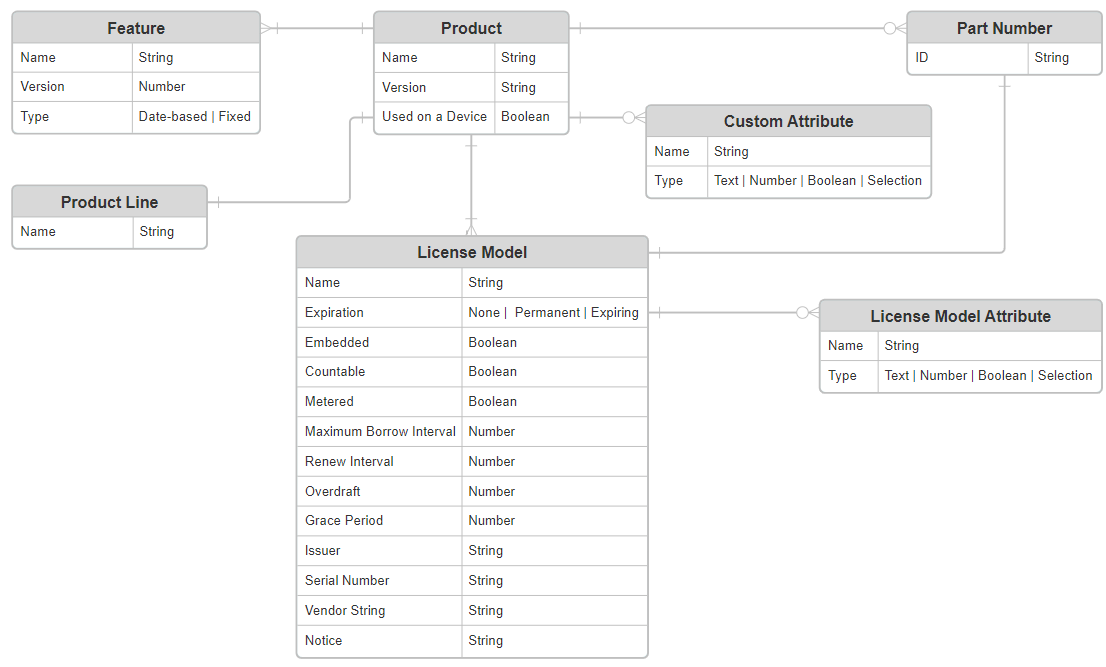
## Global Entity Relationship Diagram

The diagram below lists the main entities in the Revenera data model.

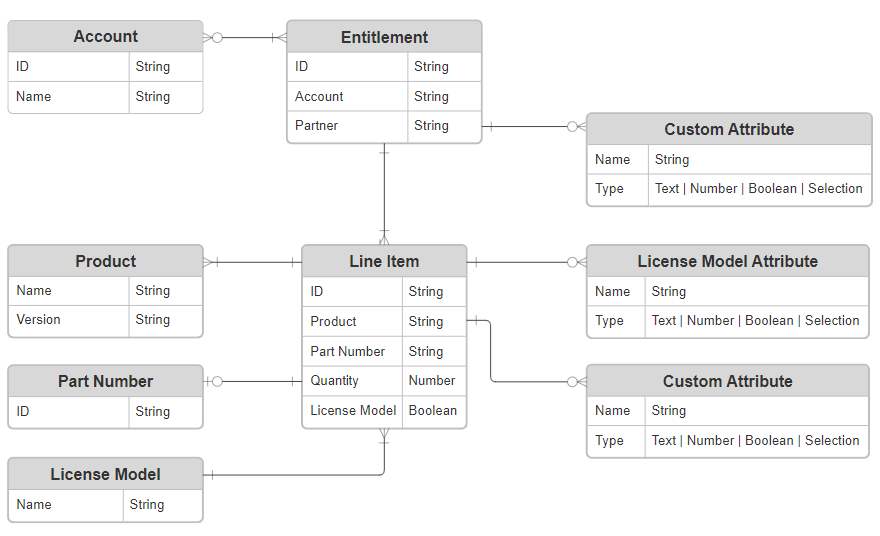
A diagram of a company

Description automatically generated

### Product and Related Entities



### Entitlement and Related Entities



## Usage for Materialise Digital Products

### Introduction

The data model in FNO is not extensible: custom objects cannot be added, nor additional relationships be created.

### As from release FlexMat

#### Product Structure

The structure of Products in FNO, to support the current Materialise License Technology, is driven by two main requirements:

1. Enable the use of CCKeys as is.
2. Emulate the product deployment mechanism used currently in SAW; specifically, the reference-module model currently employed.

Note

SFO Products do not map directly to FNO Product: SFO Products map to FNO Part Numbers that in turn reference FNO Products or Suites.

The generic structure is shown in the diagram below.

Diagram

Description automatically generated

Note that orderables for the existing Materialise license technology have no numeric version.

#### Module (Suite)

The orderable module is modelled by the Suite (orange) which holds references to the relevant reference module Products (blue) and an additional Product (green) defining the Licgen IDs needed for the module. When a customer purchases a module, the entitlement will contain a reference to the relevant Suite in FNO. All three of these entities are technically unversioned in FlexNet Operations but have textual version for ***MODULE***, ***PRODUCT*** and ***REFERENCE*** that indicates their role.

#### Licgen ID Container (Product)

In order to determine which LicGenIds are needed by the module, a separate product is defined that contains these IDs as features. This enables the license generation process to determine precisely which LicGenIds should be injected into the license file; but not the version yet.

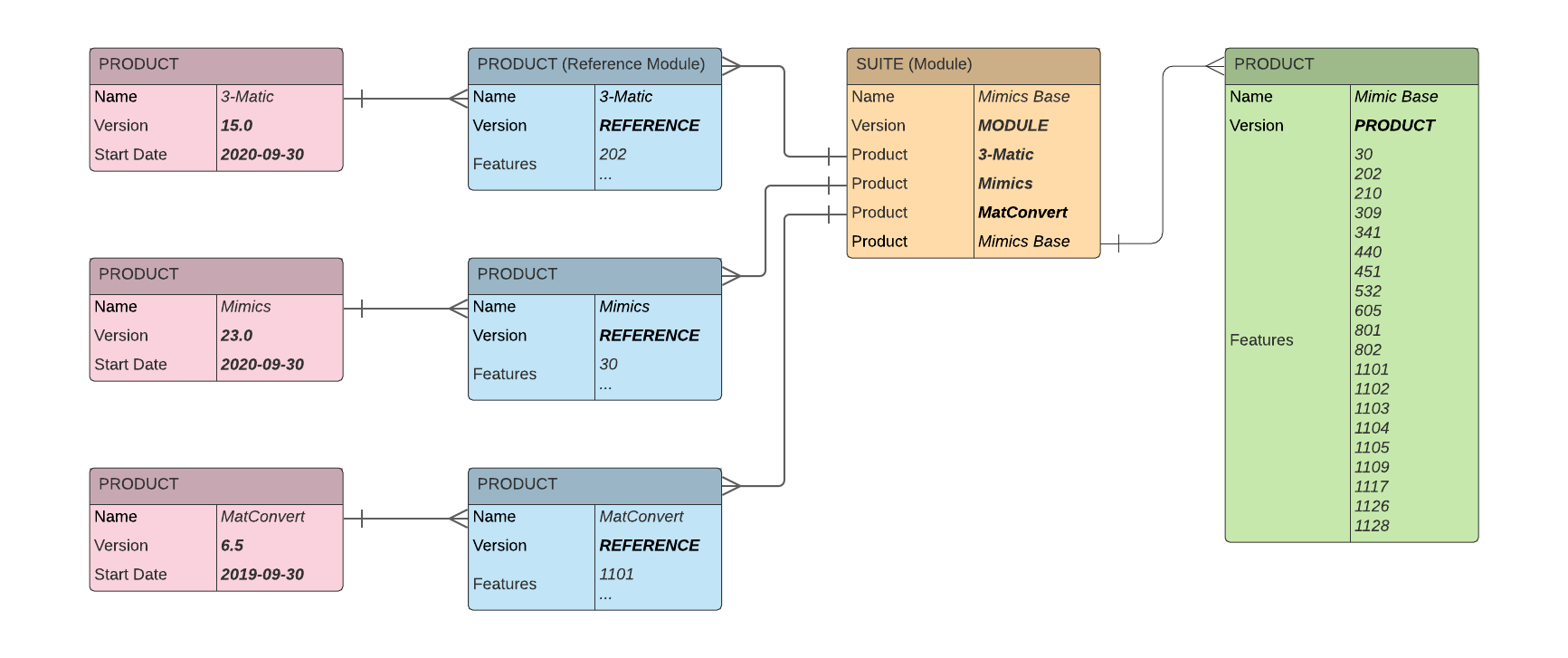
#### Reference Module (Product)

The reference module products are unversioned and contain all the LicGenIds associated with each reference module. Because reference modules do not share any LicGenIds, it is possible to determine for each required LicGenId the corresponding reference module uniquely.

#### Released Reference Module (Product)

The model allows for multiple versions of each reference module to be released with a specified version and release date. Because each perpetual line item in an entitlement (see below) will be associated with a maintenance expiration or no maintenance, and each limited line-item should get the latest released version, it is now possible to associate the correct version of each required LicGenId in the license.

The example below is for the Mimics Base module; note that the release dates and version of the (pink) products are purely illustrative.



For each of the required LicGenIds defined in the green product it’s possible to determine the correct reference module (blue) product and then all the released versions the pink products. Hence for LicGenId “1101” which is part of the MatConvert reference module, the released version is 6.5 in this example. In practice there would be multiple released versions with different release dates. The actual version selected would depend on the maintenance expiration for perpetual entitlements or the latest for limited (subscription) entitlements.

#### Beta Products

An extension to the data model will be required to support beta releases. Here we introduce a Boolean Custom Product Attribute and License Model Attribute – Beta. To enable Beta version to be provided to customers with the correct LicGenId versions FNO will be updated as follows:

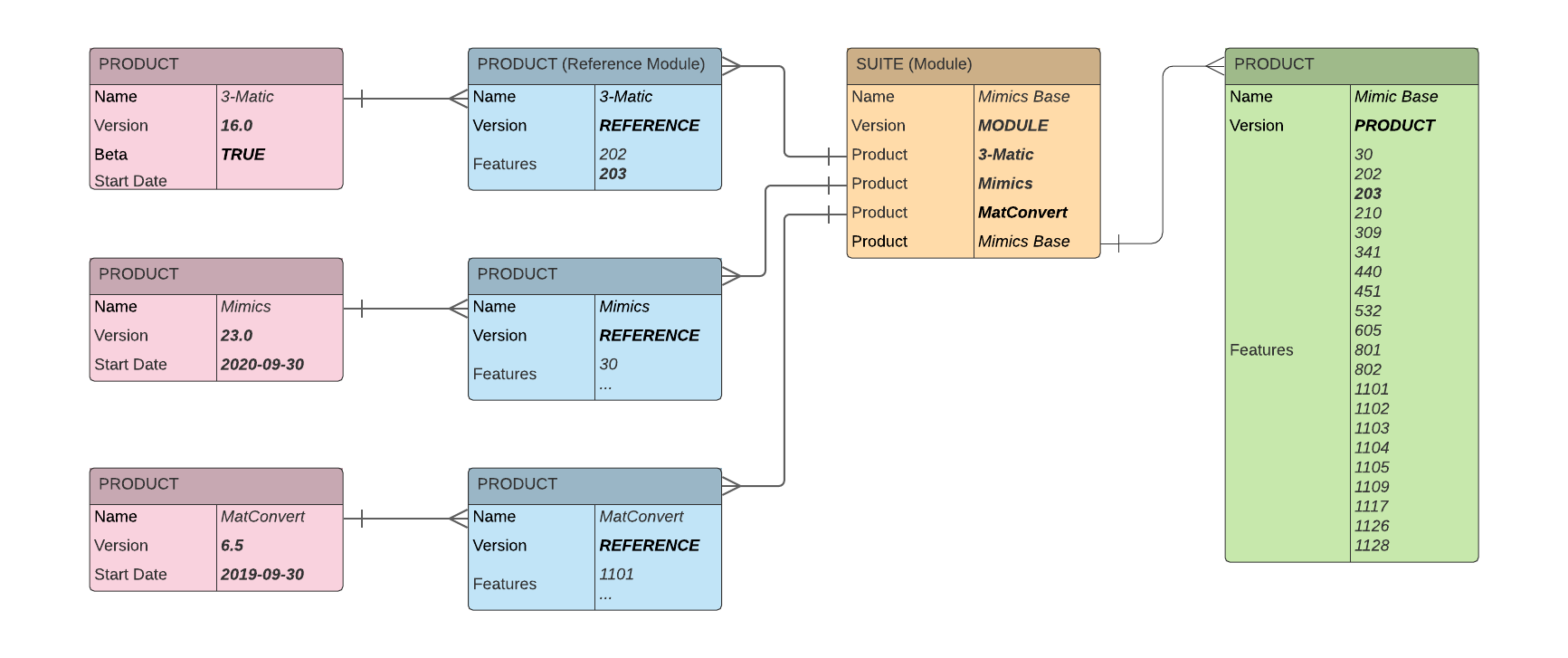
* Create a (temporary) copy of the module with the Beta attribute set to TRUE
* Create a copy of the latest released reference module with the correct beta version and the Beta attribute set to TRUE
* Create a new Part Number using the correct SFO Product id and will be created in FNO with the Beta product attribute set as TRUE. A new release reference module will be created for the correct Beta version also with the Beta attribute set True.

**Example**

The previous example has been extended to illustrate a beta release of 3-Matic. Here the release version is 16.0 but there is no release date yet; the release beta pink product has been tagged as Beta. There has been and new LicGenId included – “203” – which has been injected into the blue reference module definition and the green product definition.

If an entitlement for a Beta part-number been created, the Beta flag will have been set because the beta part-number will map the Beta license model that has this attribute set. During the versioning exercise it is now clear that Beta releases should be chosen to provide the version for reference modules. Here the only beta reference module is 3-Matic and this version 16.0 will be used in the license.

The new LicGenId “203” will also be included in the license with version 16.0.



A consequence of updating the green product definition and the blue reference module specification is that any licenses generated will include the new LicGenId “203”; this has been discussed and agreed that there would be no damaging consequences since 3-Matic version 15.0 would not look for or be able to process Licgen ID 203.

Note: in SAW if you would generate a license for 16, it would produce for the new LicGenId 203 and the 15 will see it but ignore it. Difference is that if you would generate a license specifically for 15 via SAW, it would not include 203, while in this set up it would. But it doesn’t matter, as it is there in 16 license anyway.

#### License Model

Three license models are envisaged that will be mapped to SFO part numbers (Product IDs) for each Suite (technical module):

* Perpetual
* Limited
* Beta

License model attributes will be defined so that this information can be injected into the entitlement automatically during entitlement creation; this will enable the correct LicGenId versioning to be calculated for reference modules.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Model | Description | Perpetual | Limited | Beta |
| Perpetual | Perpetual licenses with maintenance |  |  |  |
| Limited | Subscription licenses with implicit maintenance |  |  |  |
| Beta | Beta licenses |  |  |  |

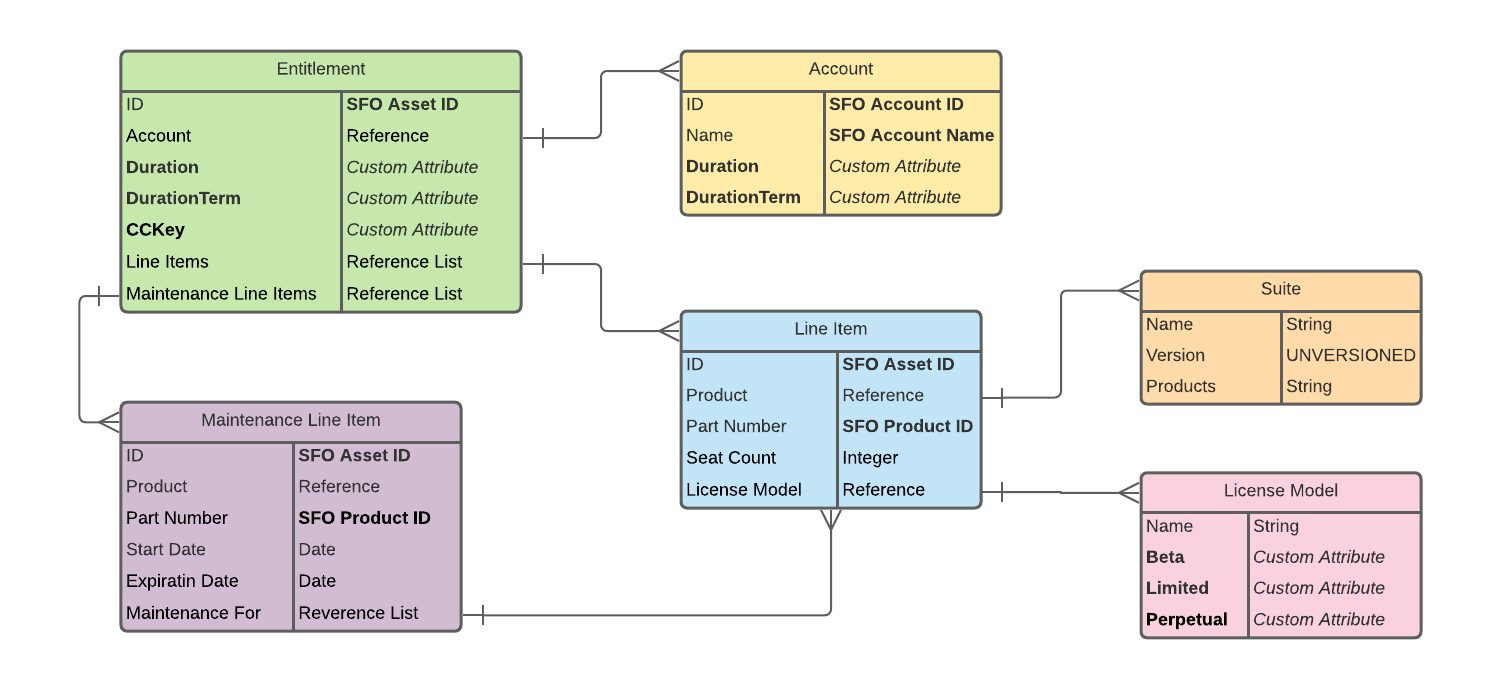
Note that the Limited model will be used for all non-revenue entitlements. The type of the entitlement will be specified in the RevenueType; the value of attribute and this will be used when generating key files if any special parameters are required.

#### Entitlement Structure

The legacy entitlement structure is imposed by the need to create and use CCKeys in the way that they are used currently. In effect, a CCKey defines the content of a license file without binding to a specific endpoint. Because CCKeys need to be delivered to customers soon after an order is confirmed (so they can activate their licenses as they do now) the CCKey must be generated when the order is fulfilled and then injected in to SFO.

**Example**

The diagram below illustrates how SFO identifies and FNO Custom Attribute will be used.



The mechanism to determine reference module versions based on maintenance expiration, mandates that each perpetual line in an entitlement can be associated with a maintenance expiration date; or no maintenance, in which case the reference module version will be based on the entitlement creating date.

Table

Description automatically generated

The order data from SFO must include details of any maintenance that is in place for perpetual licenses; that is, the maintenance product part number and the start and expiration dates, and the mapping to each asset in the order. Maintenance is not mandatory, and one maintenance asset could map to multiple lines in the order.

#### Maintenance (Product)

FlexNet Operations has a maintenance model whereby Maintenance Line Items are associated with one-or-more Entitlement Line Items (product lines) in the entitlement. The Maintenance Line Item is for a maintenance product with a maintenance start date and an expiration date; it’s also possible to have perpetual maintenance if this were ever required. Multiple Maintenance Line Items may be created with a one-to-many mapping to product lines. In addition, maintenance products have relationships defined in FlexNet Operations that govern which orderable products they may be associated with. In entitlement it’s allowed to link maintenance lines only to product lines that are defined in the maintenance product definition.

Note here that the Maintenance model – specifically the maintenance expiration - is used to determine the correct reference module LicGenIds in a license illustrated in the examples given above, it does not affect the generated license term; this is taken from attributes in the entitlement.

#### License Metadata

Any metadata that is required in the legacy license generation process, whether this will be injected into the license or used for identification, can be set into custom attributes or license model attributes in FNO.

The following attributes have been defined so far:

| Attribute | Entity | Type | Value | Comments |
| --- | --- | --- | --- | --- |
| CCKey | Entitlement | Text |  | Injected during entitlement creation |
| Duration | Entitlement | Number |  | Length of license term |
| DurationUnit | Entitlement | Single Select | DAYS |  |
| MONTHS |  |
| ProductType | Product | Single Select | MODULE | Orderable module (Suite) |
| LICGEN | Licgen definitions |
| REFERENCE\_MODULE | Reference module |
| RELEASE | Release reference module |
| BETA | Beta release reference module |
| RevenueType | License Model | Single Select | REVENUE |  |
| EVALUATION | Eval license |
| TRAINING | Training license |
| LicenseType | License Model | Single Select | FLOATING |  |
| LOCAL |  |
| LicenseTerm | License Model | Single Select | PERPETUAL | The line item is perpetual |
| LIMITED | The line item is a subscription |
| BETA | The line item is for a beta module |
| FF\_CCKey | License Model | Text |  | CCKey fulfilled |
| FF\_SystemId | License Model | Text |  | System Id fulfilled |

The CCKey must be delivered back to Salesforce after the entitlement has been created.

[the model is incomplete since the metadata requirements have not yet been discussed]

#### License Generation

Licenses generation will depend on the license technology in play.

Legacy products will use a hands-free license technology and licenses will not be generated automatically through FNO.

Customers will use the existing online or offline processes to generate licenses. The gateway services will be reengineered and the backend to interface with FNO and exiting Materialise web services - as described below - to complete the process.

These licenses will be generated in the integration layer; first by querying FNO to get LicGenIds and license metadata for the supplied CCKey, then use this data to generate the license key file via existing Materialise web services. The resulting license file will be injected into FNO as part of the activation process using FNO SOAP web services and then returned to the customer through the gateway as now; the CCKey and the System ID will also be injected at fulfilment time as license model attributes.

#### Module Versioning

All legacy orderables in FNO will be unversioned; for perpetual license models the version of the Licgen IDs that goes into the key file will be evaluated based on the ‘effective’ maintenance expiration date; for limited license models, the latest released reference module version; and for beta license models, the version of the latest beta release reference module.

The legacy product structure has been created to support this process by determining

1. the reference modules in the product
2. the release dates of known versions of these reference modules – including beta releases
3. the maintenance expiration date for limited licenses

This allows the correct General Available (GA) version of the reference module to be identified and hence what version these LicGenIds should take.

Where maintenance has not expired, or the entitlement is a subscription, then the latest reference module will be used.

#### Algorithm

Based on the legacy product structure (defined above) the process to generate a key-file runs as follows.

1. Get entitlement details based on the CCKey
2. For each line item note the effective maintenance expiration date based on license model attributes:
   1. BETA: empty
   2. LIMITED: empty
   3. PERPETUAL: The maintenance expiry date if maintenance is defined for a perpetual license, otherwise the entitlement start date.
3. Get the suite details for each line item
4. For each suite
   1. Get the PRODUCT details
   2. For each REFERENCE get all products with the same name
5. Then for each unversioned LicGenId defined in the PRODUCT
   1. Determine which REFERENCE product contains this
   2. If the effective maintenance expiration date is set, determine which versioned product with the same name as the reference product has the appropriate release date based on the effective maintenance expiration as calculated above.
   3. For beta module select the highest versioned product with the BETA attribute set
6. Build a list of the versioned LicGenIds along with any other relevant metadata gathered from entitlement or line-item, level custom attributes
7. Generate the license file from the Materialise façade license generator gateway service.

### As from release FlexFlex

<mapping schema of how entities of FNO are used to support Materialise Digital Products structure> <textual explanation>

## Validation Rules ?

### New validation rules

|  |  |
| --- | --- |
| Rule Name | Reasoning and Motivation |
|  |  |
|  |  |

### Existing validation rules

Not relevant.

# Front-End

Front End means here the technical components for the presentation logic. In the case of Revenera and FlexNet Operations (FNO), there are no possibilities to create new custom components in this area. The section is thus not applicable for this technology.

# Back-End

## Introduction

Back End means here the technical components for the business logic. In the case of Revenera and FlexNet Operations, there are no possibilities to create new custom components in this area. The section is thus not applicable for this technology.

## Jobs

There are several jobs that can be executed in FlexNet Operations:

* Alerts
* Updates
* Data extracts – usage, fulfillment, devices, accounts, entitlements,
* Conditional notifications
* Snowflake ETL export
* Data analytics – renewals, fulfilment, accounts, entitlements,

### This Release

Requirements identified so far comprise the following:

|  |  |
| --- | --- |
| Name | Reasoning and Motivation |
| License Model Export | Management of UAT and PROD systems |
| License Model Import  Product Import | Manual configuration of UAT and PROD systems |

### Existing jobs

Not relevant.

# Integration

## Overall Information Flows

This section provides an overview of the information flows between applications (logical level) and highlight of the impacted existing information flows or new information flows as from this version.

We will refer therefore to the Domain Architecture deliverable :

https://materialisenv.sharepoint.com/sites/extranet-Tiger\_implementation/Tiger/ALL%20Architecture%20Deliverables/99\_TIGER%20Domain%20Architecture%20-%20Visio.vsdx?d=wfc18ad554be24c3bba650a9ad75fb042 Owner : Enterprise Architecture Team

## Integration Scenarios for FlexFlex

This section provides a complete list, description, and details of integration scenarios between applications (technical level) and highlight of the impacted or new integration scenarios as from this version.

### Direct Sales Subscription

A new subscription is contracted in Salesforce: Boomi synchronizes the contract and subscriptions with FNO to create a new entitlement. The relationship to Salesforce is as follows:

* SFO **Contract** maps to an **Entitlement**
* SFO **Subscription** maps to an **Entitlement Line Item**
* License type (**Local** or **Floating**) is captured in the **License Model Attribute**: **LICENSE\_MODEL**

The integration uses the Revenera Entitlement Order web service: see <???> for details.

#### Entitlement

|  |  |
| --- | --- |
| Field | Description |
| ID | Autogenerated |

#### Line Items

|  |  |
| --- | --- |
| Field | Description |
| ID | Autogenerated |
| Part Number | SFO product code |
| Product Name | *Derived from Revenera part number* |
| Product Version | *Derived from Revenera part number* |
| License Model | *Derived from Revenera part number* |
| Start Date | Depending on payment status in SFO:   * Unpaid: Payment Term expiration * Paid: Contract start date |
| Expiration Date | SFO Contract end date |
| Version Date | SFO Contract end date |
| LICENSE\_MODEL | SFO License Type |

#### Amendments

Contract amendments in SFO are synchronized with FNO but there are some complications owing to potentially negative subscription quantities in SFO and data model restrictions in FNO.

|  |  |
| --- | --- |
| SFO Action | Revenera |
| Existing SFO subscription quantity is reduced but remains positive | The line-Item quantity in FNO is reset to the sum of the subscription quantities in SFO. Because…   1. FNO web services allow the line-item quantity to be set below the number of activated licenses: this is something the UI does not permit 2. Revenera does not manage the total license count when devices, with previously activated licenses, call home   …the line-item will be removed from all devices: this should not include license servers where the Revenera does manage total line-item quantity.  Customers will need to reactivate the line item on devices: the new line-item quantity will be respected in this scenario. |
| Existing SFO subscription is removed | The line-item state in Revenera is set to OBSOLETE and the line item-removed from all devices where it has been activated previously. Again, Revenera does not remove activated licenses from devices when they call home, event when the line-item state is OBSOLETE. |
| Existing SFO subscription quantity is increased. | The line-Item quantity in FNO is reset to the sum of the subscription quantities in SFO |
| New SFO subscription is added | New line-item is created in Revenera. |

### Indirect Sales Subscription

TBD

### Non-revenue (Voucher)

Vouchers follow the FlexMat and are of two types:

* **FIXED** – the start and expiration dates are fixed: like normal subscriptions but without any payment period.
* **VARIABLE** – the start date will be the activation date of the line-item but activation is constrained by the validity end date.

Because FNO entitlements to not have expiration dates, duration-based line-items may be activated at any point; Via integration (Boomi) the voucher line-item will be set to Obsolete once the validity end date has passed to prevent the voucher being activated after this date.

The integration uses the Revenera Entitlement Order web service: see <???> for details.

#### Entitlement

|  |  |
| --- | --- |
| Field | Description |
| ID | Autogenerated |

#### Fixed Voucher Line Items

|  |  |
| --- | --- |
| Field | Description |
| ID | Autogenerated |
| Part Number | SFO product code |
| Product Name | *Derived from Revenera part number* |
| Product Version | *Derived from Revenera part number* |
| License Model | *Derived from Revenera part number* |
| Start Date | Voucher start date |
| Expiration Date | Voucher end date |
| Version Date | Voucher start date |
| LICENSE\_MODEL | Voucher License Type |

#### Variable Voucher Line Items

|  |  |
| --- | --- |
| Field | Description |
| ID | Autogenerated |
| Part Number | SFO product code |
| Product Name | *Derived from Revenera part number* |
| Product Version | *Derived from Revenera part number* |
| License Model | *Derived from Revenera part number* |
| Start Date | Use the activation date |
| Term | Voucher duration |
| Term Units | Voucher duration units |
| Version Date | Voucher start date |
| LICENSE\_MODEL | Voucher License Type |

### Emergency License

TBD

## Custom API

NA

### Introduction

### New Custom API

|  |  |  |
| --- | --- | --- |
| Custom API name | Reasoning and Motivation | Link to API description |
|  |  |  |
|  |  |  |

### Existing Custom API as from previous release

|  |  |  |
| --- | --- | --- |
| Custom API Name | Reasoning and Motivation | Link to API description |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

# Data Security

Not relevant.

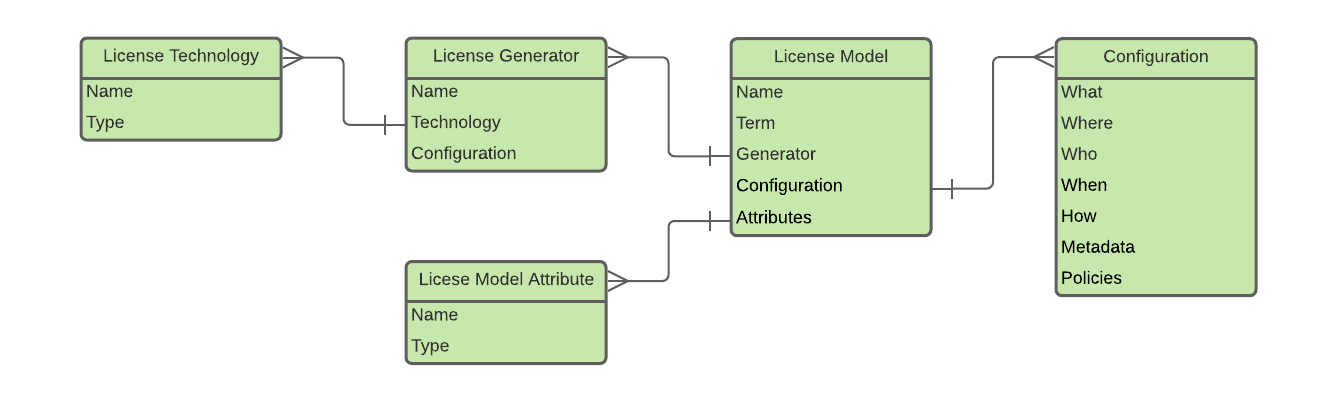
# Annex

## Data Model FNO

This chapter explain the general datamodel of FNO.

### Licensing

The main entity attributes are described in the tables below



#### License Technology

License technology is the root entity in FNO and forms a major partition for license generators, license models and products.

License Technology has the following attributes:

| Attribute | Type | Description |
| --- | --- | --- |
| Name | String | Unique name |
| Type | FlexNet Licensing | Default for use with FNP and FNE toolkits. |
| Manual | Licenses are generated using an external process and the license then uploaded to FNO |
| Hands-Free | Licenses are generated using an automated external process; the licenses are subsequently injected into FNO. E.g. Materialise internally developed license technology (MatSDK). |
| Remote SOAP Service | Uses an external SOAP web service; this is a built-in process in FNO |
| Generator Interface | Uses local Java extensions – not available in the cloud. |
| Counted | Boolean | Does this technology use counted licenses? |

Out-of-the-box, FNO provides a default License Technology - *FlexNet Licensing* - that enables FlexNet Publisher and FlexNet Embedded licenses to be generated leveraging built-in FNP and FNE license generators.

If licenses for a non-Revenera License Technology are required, FNO allows bespoke License Technologies to be configured as either a *Remote SOAP Service*, *Manual* or *Hands-Free* License Technology. In the case of Remote SOAP Service there is built-in license generation functionality FNO that calls-out to the remote SOAP service; for all other License Technologies, license generation is delegated to an external process, the licenses being injected into FNO subsequently.

#### License Generator

License Generators enable different license generation behaviours to be configured within a License Technology; for example, signature strength, cryptographic keys, other licensing parameters. It is beyond the scope of this document to describe this in any detail; please refer to publicly available Revenera product documentation for a complete overview.

#### License Model

License models supplement the license generation process in FNO, allowing some licensing policy decisions to be automated, and license metadata to be injected into the generated licenses. The elements of the standard FNO license models are beyond the scope of this documents and is presented in full in the administrative guides[[1]](#footnote-2).

| Attribute | Type | Description |
| --- | --- | --- |
| Name | String | Unique name of this model. |
| Technology | License Technology | License Technology name |
| Term | Selection | Default license term or perpetual; this can be overridden in the UI or through web services. |
| Generator | License Generator | The License Generator used by this License Model |
| Attributes | License Model Attribute | A list of License Model Attributes used by this License Model - see below for an explanation |

#### Configuration

A description of the mechanics of FNO license models is beyond the scope of this document; please refer to the FNO, FNP and FNE technical guides for a complete introduction.

#### License Model Attribute

License Model Attributes enable additional named data fields to be defined when creating entitlements or activating licenses; this process will be covered in more detail in later sections. License model attributes are defined for specific License Technologies and made available through License Models; they can be populated at various points in the entitlement lifecycle:

|  |  |
| --- | --- |
| Stage | Description |
| License Model Time | The attribute is hard-coded for the license technology; all line-items for any license model for the same technology will receive the attribute and its value. |
| Entitlement Time | The attribute value is populated when the line -item is created. |
| Fulfilment Time | The attribute value is populated when the line-item is activated. |

The License Model configuration specified which License Model Attributes are to be used and those that are ignored in an Entitlement. Unlike Custom Attributes (see later) License model attributes are not searchable using web services; however, they are available in web service queries and are also made available during license generation.

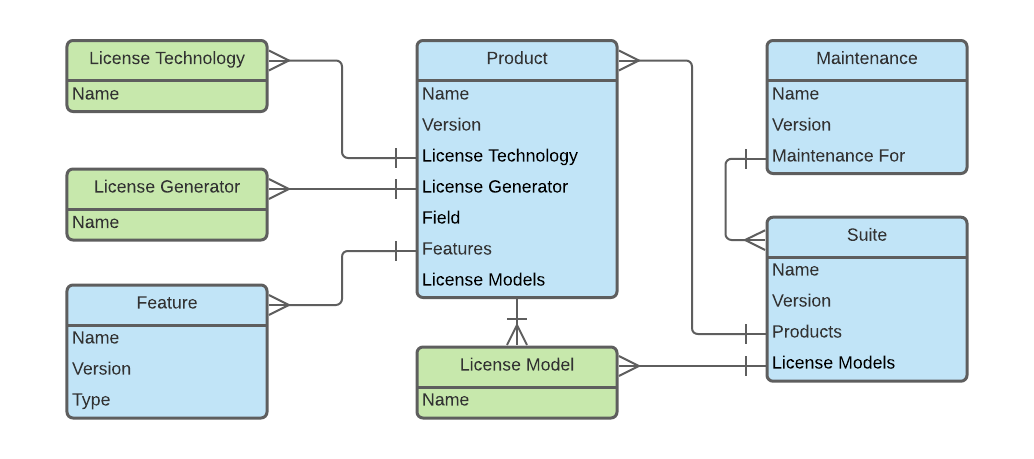
This mechanism provides a flexible means whereby licensing metadata can be captured. Data captured during Entitlement creation can be used for straightforward reporting purposes but more importantly to provide additional license data during License Activation; data captured during License Activation is typically used to persist install base information, for example, target operating system information, user names, domain names etc.

Attributes have a unique name and one of the specified types as follows:

|  |  |  |
| --- | --- | --- |
| Attribute | Type | Description |
| Name | String | Unique name of the attribute |
| StringValue | String | A free-format text value, or a single selection or multiple selections from a predefined drop-down list. |
| NumberValue | Long | Single integral value |
| DateValue | DateTime | Single date or date-time value |
| BooleanValue | Boolean | True or false. |

### Product Packaging

Product packaging entities in FNO define the technical implementation of the commercial product catalogue; these represent what customers may purchase. Orders within the ERP system will be transformed into one-or-more Entitlements containing product packaging entities corresponding to the commercial product SKUs purchased. It is the configuration of product packaging entities in FNO that allows licenses to be generated correctly for what has been purchased.



#### Feature

Features are the basic unit of licensing and comprise a name and a version; they are the elements that enable functionality in license-enabled applications. Feature objects are not related to SFO Feature objects.

| Attribute | Type | Description |
| --- | --- | --- |
| Name | String | The name of the feature (this is the unique identifier). |
| Version | String | The version of the feature. |

From a licensing perspective, the presence of a Feature (in a license) will enable the functionality associated with the Feature in the license-enabled application; very often the version of the Feature also plays a role.

##### Feature with Count

Whenever features are referenced by any other data entity they are always associated with a numeric multiplier; this defines how many of the feature is included. Typically, this 1 but may in general be any non-negative integral value.

Example:

Consider the following Feature defined in a license:

|  |
| --- |
| {  "feature": {  "name": "1117",  "version": "23.0",  "expires": "permanent",  "count": 1,  "host-id": "A803354C-53XE-0011-82218927A41A7B9C",  "start": "2021-01-01",  "issuer": "Materialise NV",  "signature": "0dabf59a5df3c8b29e2d9e22a45af079c20d876020c16fc3e6706094b472c9c2888278445bec…"  }  } |

The Feature is 1117, version 23.0 with a count of 1; how the application reacts to the presence of this specific counted Feature is part of the licensing implementation within the software.

Other significant elements are:

* The start date and the expiration date; the Feature is only valid between these dates.
* The host-id; the Feature is only valid if the software is running on a machine with this unique identifier.
* The signature, which prevents license tampering; if the license details are changed the digital signature will be different and the license will be invalid.

#### Feature Bundle

Features may also be grouped into a Feature Bundle; this is a convenience. Bundles are not versioned and define a list of features including a count.

Note that Feature Bundles are not related to SFO Bundles.

|  |  |  |
| --- | --- | --- |
| Attribute | Type | Description |
| Name | String | The name of the feature bundle |
| Features | List Feature | List of features with count contained in this bundle |

#### Orderable (Product or Suite)

While features are the fundamental unit in licenses, orderables are the basic unit of entitlement; there are three categories:

* Products
* Suites
* Maintenance

Products are collections of features and Suites are collections of products, both of which, from a licensing perspective, deliver a collection of features into licenses. Maintenance on the other hand, whilst being orderable, does not contribute to license generation.

##### Product

Products package features and feature bundles into an orderable entity; identified uniquely by its name and version. A license derived from a product will contain the sum of all features and all features in the feature bundles - including the counts. If there is duplication of features, this will be represented in the license.

| Attribute | Type | Description |
| --- | --- | --- |
| Name | String | Name of the product |
| Version | String | Version of the product |
| License Technology | Reference | The License Technology for this product |
| License Generator | Reference | The License Generator which must derive from the same License Technology |
| Start Date | Date | The date the product is active from |
| Features | Reference List | List of Features with counts contained in this Product |
| Feature Bundles | Reference List | List if Feature Bundles with counts contained in this Product |
| Custom Attributes | Reference List | List of Custom Attributes configured for this product |
| License Models | Reference List | List of License Models that are available for this Product; these must all derive from the same License Technology |

##### Suite

Suites package collections of products into an orderable entity; they are identified uniquely by name and version. They share many of the same attributes as products, having a list of products with counts in place of features and feature bundles.

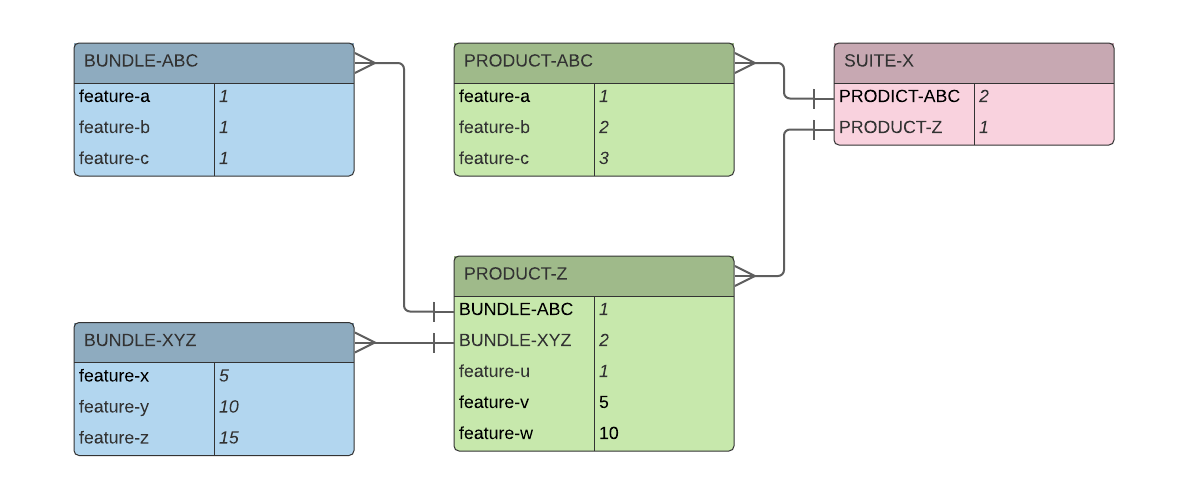
|  |  |  |
| --- | --- | --- |
| Attribute | Type | Description |
| Name | String | Name of the suite |
| Version | String | Version of the suite |
| License Technology | Reference | The License Technology for this Suite |
| License Generator | Reference | The License Generator which must derive from the same License Technology |
| Start Date | Date | The date the suite is active from |
| Products | Reference List | List of products with counts contained in the Suite; Products must derive from the same License Technology and Product Line but not necessarily the same Licence Generator |
| Attributes | Reference List | List of custom attributes defined for this suite |
| License Models | Reference List | List of license models that are available for this product when creating an entitlement |

Note that the license models are not inherited from the products defined in the suite; these are defined separately and do not have to agree with any license model defined for any of the packaged products. The product line is derived from the products; all product in the suite must share the same product line.

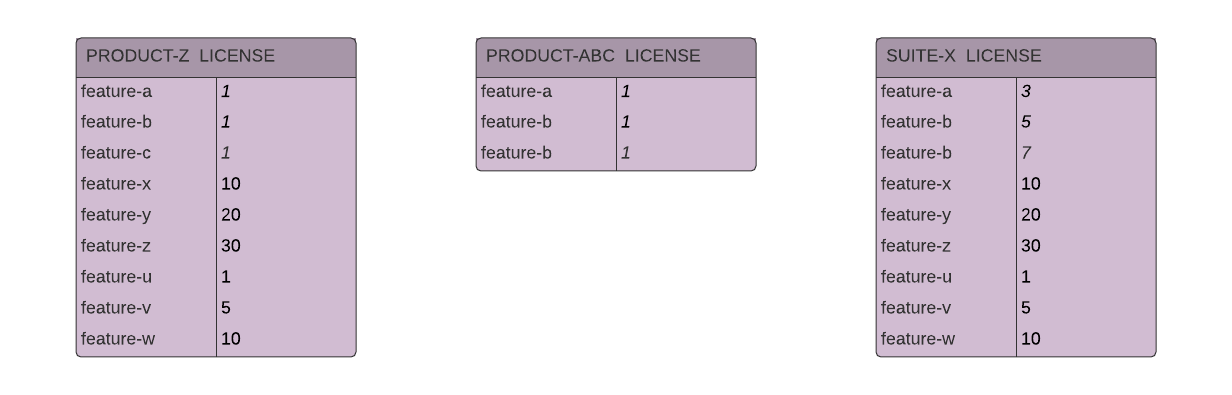
A license generated from a suite will contain the sum of all features derived from all products including the counts defined.

Example:

Consider the following Suite definition based on two Products and two Feature Bundles and nine Features:



A license generated respectively for a single count of *PRODUCT-ABC*, *PRODUCT-Z* and *SUITE-X* will be as follows:



##### Maintenance Product

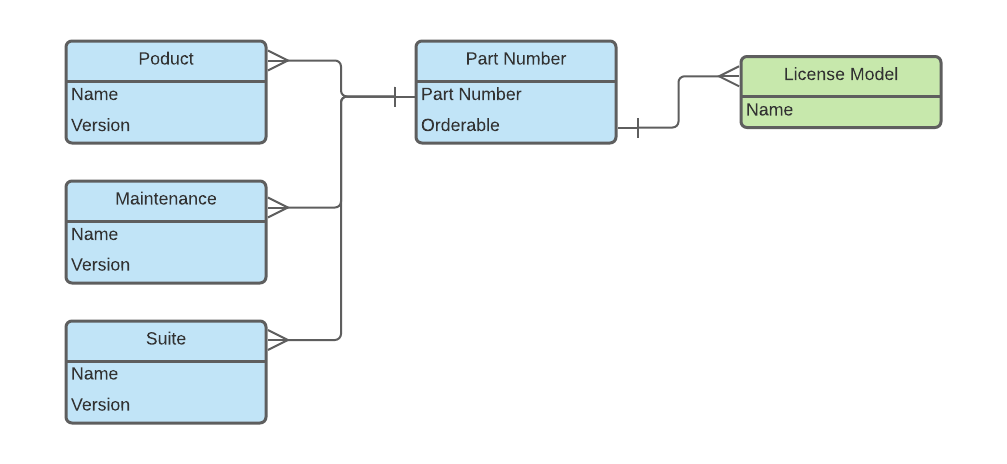
Maintenance products are orderable but do not have any features defined and are not considered when generating license for the FlexNet License Technology; for other License Technologies the license generators are at liberty to consider maintenance.

The list of referenced products or suites defines which orderables the maintenance product can provide maintenance for.

| Attribute | Type | Description |
| --- | --- | --- |
| Name | String | Name of maintenance product |
| Version | String | Product Version |
| Reference | Reference List | List of products (with versions) for which this maintenance product provides maintenance. Products can be for any License Technology |
| Attributes | Reference List | List of Custom Attributes defined for this product |

#### Part Number

Part numbers tie together products or suites, the version, and the license model and provide a convenient way to define an Entitlement Line Item, since a single data item uniquely defines the three of the mandatory elements in a line item.



The part number values are mastered in SFO and must be configured in advance.

|  |  |  |
| --- | --- | --- |
| Attribute | Type | Description |
| Part Number | String | SKU – id taken from SFO Product |
| Product | Reference | Name of product or suite |
| Version | String | Version of product or suite |
| Model | Reference | License Model |

### Account Management

#### Account

In the FNO model there are three types of Account:

* Producer
* Customer
* Partner

There is a single Producer Account (Materialise) the Users of which have access to the FNO Producer Portal; Customer Accounts, Customer Accounts own Entitlements and has Users who may have access to the FNO Customer Portal. By default, there is a single predefined producer organization (Materialise) the users for which will have access to the FNO Administrative Portal. For customer and partner accounts users would have access to the FNO Customer Portal.

An account is identified uniquely by its ID and Name; these will be taken from SFO.

| Attribute | Type | Description |
| --- | --- | --- |
| ID | String | Mandatory - taken from SFO |
| Name | String | Mandatory - taken from SFO |
| Address 1 | String | Address line 1 |
| Address 2 | String | Address line 2 |
| City | String | City |
| State | String | State or province |
| Zipcode | Select | Postal code |
| Country | String | Two-digit ISO 3166 country code |
| Region | String | Free-format text |
| Attributes | Reference List | List of custom attributes defined for this account |

#### Role

FNO provides several standard Roles for access to the FNO Producer Portal and the Customer Portal; they also extend to web services. Please refer to product documentation for a complete introduction to Roles and their use and configuration.

#### User

Users belong to one-or-more Accounts and are authenticated against FNO or an external domain. External domain and single sign-on configuration is beyond the scope of this document. Executing FNO SOAP web services requires valid User credentials,

Users belonging to the Producer Account have access to the FNO Producer Portal; users belonging to Customer or Partner Accounts, or Self-Registered Users may have access to the FNO Customer Portal and FNO Download Portal.

Users can belong to multiple accounts of the same type; the Roles they have within each Account is part of the User configuration.

| Attribute | Type | Description |
| --- | --- | --- |
| User Name | String | Mandatory |
| First Name | String | Mandatory |
| Last Name | String | Mandatory |
| Display Name | String |  |
| Email Address | String | Mandatory |
| State | String |  |
| Zipcode | Select |  |
| Country | String | Two-digit ISO 3166 country code |
| Region | String |  |
| Accounts | Reference List | The Accounts with Roles the User is in. |
| Attributes | Reference List | Custom Attributes available for the User |

### Entitlement Management

#### Entitlement line item

Entitlement Line Items (or simply Line Items) are the fundamental unit of entitlement and are always contained within an Entitlement (see below). The Line Item is identified by a unique activation ID; this is often provided to customers and can be used in activating licenses.

It is possible to define several Products with individual counts on a Line Item, but this is not best practice as it seriously complicates upgrade processes downstream; such a composite Line Item is sometimes referred to as an ad hoc Suite but bears no relationship to orderable Suites defined in FNO. The recommended model is to define a single Product or Suite per Line Item with a specified seat count.

The License Model must be selected from the list of available License Models configured in the Suite or Product. The License Model plays a role in license generation since License Model Attributes or other metadata may have been defined.

In the standard FNO Model, the start date can be defined when the Line Items is created, taken as the first activation date, or taken from each subsequent activation date if there a multiple seats defined. If the expiration date is not specified, the license term will default to whatever default value is defined in the License Model.

|  |  |  |
| --- | --- | --- |
| Attribute | Type | Description |
| Activation ID | String | The unique ID identifying this line item |
| Product | Reference List | The list of Products and/or Suites including the Version; note best practice is to have just one item in this list. |
| Quantity per Copy | Integer | The number of licenses per seat; this defaults to 1 which is almost always the case, letting the seat count determine the number of licenses available. |
| Quantity | Integer | The seat count |
| License Model | Reference | The purchased license model |
| Start Date | Date | The date the generated license id valid from |
| Expiration Date | Date | The date the generated license id valid to |
| Attribute | Reference List | List of attributes for this line item |

#### Maintenance Line Item

Maintenance Line Items are identified by an Activation ID although they cannot be activated and do not generate licenses. The Maintenance Line Item has a start date and expiration date that indicate the start and end of maintenance. Linking the Maintenance Line Item to Entitlement Line Items indicates that the Products for these Line Items are covered under maintenance; so long as maintenance has started and not expired.

Multiple Maintenance Line Items can be defined in an Entitlement, each linked to one-or-more orderable Line Items, even in different Entitlements.

|  |  |  |
| --- | --- | --- |
| Attribute | Type | Description |
| Activation ID | String | The unique ID for this line item |
| Product | Reference | The Maintenance Product including the Version |
| Start Date | Date | The maintenance start |
| Expiration Date | Date | The maintenance expiration date |
| Attribute | Reference List | List of Custom Attributes available for this Maintenance Line Item |
| Line Items | Reference List | List of Entitlement Line Items (Activation IDs) that the Maintenance applies to; the configuration of the Maintenance Product specifies which Products or Suites may be linked in this way. |

#### Entitlement

An Entitlement represents one or more elements from a customer software order; typically, a software order will equate to a single Entitlement, but this is not mandated. There need be no relation between elements of an Entitlement; the customer is free to select which elements of the Entitlement should be activated on an endpoint Device or License Server – or Cloud License Server.

Note that Entitlements are never activated, it is the Line Items that are activated to generate licenses.

| Attribute | Type | Description |
| --- | --- | --- |
| Entitlement ID | String | Unique id of this entitlement (not a CCKey) |
| Account | Reference | Reference to the Customer or Channel Partner Account that owns this entitlement. |
| Lines | List of Line Item | List of Entitlement Line Items |
| Maintenances | List of **Line Item** | List of Maintenance Line Items |
| Attributes | List of Attribute | List of Custom Attributes available for this Entitlement |

#### Fulfilment

Fulfilments are generated during license activation and relate to FlexNet Publisher and non-Flex license technologies only; this is not relevant for FlexNet embedded licensing - see Devices below.

Fulfilments records are in a many-to-one correspondence with Line Items, a Line Item may be activated multiple times; each activation will generate a license and a Fulfilment.

|  |  |  |
| --- | --- | --- |
| Attribute | Type | Description |
| Fulfilment ID | String | Auto-generate unique ID identifying the activation. |
| Fulfilment Date | Date | Date of the activation |
| License | String | License text or Base64 encoding of binary license data |

#### Device

Devices represent licensing endpoints using the FlexNet Embedded licensing; they replace the fulfilment functionality associated with FlexNet Publisher and non-Flex licensing technologies. There are four device categories:

* Devices
* License Servers
* Cloud License Servers
* Served Devices

Devices, License Servers, Cloud License Servers all activate Entitlement Line Items directly from FlexNet Operations, whereas served devices borrow features directly from License Servers or Cloud License Servers; all Devices are identified by a unique Device Identifier – often termed a Hostid. There is a predefined selection of Device Identifier Types supported by FlexNet Operations in conjunction with FlexNet Embedded license SDK; the Device Identifier can be a manufactured string, or one taken from the available Hostids identified by the licensing SDK on the local endpoint.

Devices can be created in the UI or by web services and assigned to an Account; any Entitlement Line Item can be mapped to the Device from those owned by the Account. Once the FNE enable application communicates with FlexNet Operations – called a capability exchange or call-home – the Device will receive a license for all the Features contained in the Line Items mapped to the device in FNO. A Device is an FNO entity and relates to an FNE enabled client application running in a specific context. The same application running under different user accounts for instance may generate separate Devices.

Devices can also be created automatically in FNO by the capability exchange managed by the FNE SDK; in this scenario the device requests one-or-more Activation Ids – or Rights Ids – that uniquely identify Line Items. The software will receive the appropriate license, and the Device and Account mapping will take place automatically in FNO.

Local License Servers and Cloud License Servers are all client Devices that activate Line Items directly from FNO; and serve the Features contained in the licenses they receive to FNE endpoints. Endpoints served in this fashion are termed Served Devices; these are registered and visible in FNO. Communication with License Servers is managed by the FNE SDK; the paradigm is different from direct activation in that the client requests to borrow Features not activate line items.

The capability exchange mechanism with license servers - managed by the FNE SDK - works online and offline; in the offline case this is by file transfer, but the data contracts are identical.

Discussion of the licensing options available using FNE, and best practice is beyond the scope of this document but is covered fully in the Revenera FNO and FNE toolkit documentation.

|  |  |  |
| --- | --- | --- |
| Attribute | Type | Description |
| Device ID | String | The unique identifier for this device |
| ID Type | Device Type | See below… |
| Name | String | Name of the device |
| Device Model | Device Model | See below… |
| Account | Account | The Account that owns the device |
| Notes | String | Free-format field |
| Site Name | String | Free-format field |

##### Device Models

Devices must have a device model; this defines some of the licensing behaviour and constraints for the FNE enabled device software. There are two defaults in FNO for client and servers:

* FLX\_SERVER
* FLX\_CLIENT

Generally, these are enough for most licensing scenarios; however, bespoke device models can be created easily in FNO whenever required.

###### Device ID Types (Hostid)

The device id is used to bind a license cryptographically to a machine identifier and prevent the license being used on other devices; this topic is covered in detail in the Revenera FNE documentation. There are the following built-in hostid types supported by the FNE SDKs

* Ethernet (MAC Address)
* IPV4, IPV6
* Dongle (two supported types)
* String (locally manufactured ID)
* User
* VM UUID
* Container ID

Any hostid type can be selected by the FNE enabled software provided the FNE SDK recognizes the hostid locally.

#### Trusted Storage

A key component of FNE - providing tamper resistance - is the use of trusted storage; these are local encrypted files that provide license anchoring and binding to the local endpoint. The use of trusted storage and other anti-tampering technologies render deliberate misuse of licenses very difficult; although, virtualized and containerized environments do provide challenges. A key consequence of this is the ability to return licenses back to FNO or a license server securely; the local license can be verifiably deleted and hence prevent license leakage during rehosting.

### License Model Attributes

License model attributes are defined for specific license technologies and can be populated at various points in the entitlement lifecycle:

|  |  |
| --- | --- |
| Stage | Description |
| License Model Time | The attribute is hard-coded for the license technology; all line-items for any license model for the same technology will receive the attribute and its value. |
| Entitlement Time | The attribute value is populated when the line -item is created. |
| Fulfilment Time | The attribute value is populated when the line-item is activated. |

When defining license model attributes, it is possible to select at which stages the attribute value can be populated but once populated the value becomes fixed. License model attributes can also be ignored for some license models if this is desired. License model attributes are not searchable using web services but will be returned in web service queries and are also made available during license generation, which enables metadata to be injected into the licenses.

### Custom Attributes

Attributes can be defined on the following FNO entities; these are searchable in FNO web service queries.

* Account
* User
* Product
* Entitlement
* Entitlement Line Item
* Maintenance Line Item
* Download Package
* File

Custom attributes share the same structure as License Model Attributes – see above. The main difference is they are populated when the entity they belong to is created; so only Maintenance Line Item and Entitlement Line-item Custom Attributes can contain entitlement related information. These attributes are searchable and are available to the inbuilt license generation processes in FNO. They are also returned in SOAP web service queries.

## System Constraints and Timings

This section lists recommended system throughput and capacity constraints.

### SOAP Web Services

Revenera performance testing results can be summarized by the following:

|  |  |  |
| --- | --- | --- |
| **Action** | **Comments** | **Mean Duration (secs)** |
| Create entitlement | 1 entitlement 3 line-items | 0.69 |
|  | 1 entitlement 10 line-items & custom attributes | 7.09 |
| Delete Entitlement | 1 Entitlement, 10 line-items | 15.58 |
| Search Entitlements | By Entitlement ID & State | 4.47 |
|  | By Account ID & State | 5.46 |
|  | By Entitlement Attribute & State | 4.91 |
|  | By Product Name and Version | 3.82 |
| Search Users | By Account ID | 2.51 |
|  | By Username | 1.61 |
| Search Accounts | By Account ID | 2.03 |
| Search Products | By Product name and Version | 1.95 |
|  | By Part number and State | 2.29 |

### License Activation

This covers software calling home to FNO to activate one-or-more activation Ids or simple call-home to refresh licenses.

### Access to Cloud License Server

TBC

### Access to Local License Server

TBC

1. FNO Administrative Guide:

   [https://docs.revenera.com/fno2020r2/producer/Content/helplibrary/Introduction.htm#fno\_admin\_intro\_3156978783\_1054036](https://docs.revenera.com/fno2020r2/producer/Content/helplibrary/Introduction.htm" \l "fno_admin_intro_3156978783_1054036) [↑](#footnote-ref-2)